DOCTORADO EN CIENCIA Y TECNOLOGÍA
UNIVERSIDAD DE CANTABRIA

GRUPO DE COMPUTACIÓN AVANZADA Y E-CIENCIA
INSTITUTO DE FÍSICA DE CANTABRIA
CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS (CSIC)

PRIVACY PRESERVING TECHNIQUES IN DATA SCIENCE ENVIRONMENTS

TÉCNICAS DE PRESERVACIÓN DE LA PRIVACIDAD EN ENTORNOS DE CIENCIA DE DATOS

Tesis doctoral presentada por
JUDITH SÁINZ-PARDO DÍAZ
para optar al título de
Doctora en Ciencia y Tecnología por la Universidad de Cantabria

Bajo la supervisión de Álvaro López García

Le meilleur est à venir

Agradecimientos

Hace cuatro años que comencé mi andadura en el IFCA, y estos agradecimientos no pueden empezar de otra forma que no sea agradeciendo a Álvaro por permitirme investigar en lo que me gusta. Gracias por tu apoyo en todo momento y por darme la confianza y la libertad de seguir mis ideas.

En el mismo sentido, tengo que agradecer a todos mis compañeros y compañeras del Grupo de Computación Avanzada y e-Ciencia por ser un grupo tan unido y por estar siempre los unos para los otros. Especialmente a todos con los que he compartido desde viajes hasta publicaciones, gracias por ser la mejor compañía en este camino. Este agradecimiento se extiende también a todos mis compañeros del IFCA, gracias por todo.

I'm deeply grateful to the great family of the AI4EOSC and EOSC SIESTA projects, who have made it so easy for me to work with them during this time. I look forward to continuing our collaboration.

I would also like to thank my colleagues at INRIA for their warm welcome during my doctoral stay and for making me feel truly part of the Cométe group. A special thanks to Catuscia for opening the doors of the group to me and giving me the opportunity to work with all of you.

Si he llegado hasta aquí es gracias a toda la gente que ha formado parte de este camino. Gracias por tanto a todos mis amigos y amigas: a las que son como hermanas, a los de toda la vida (y de los findes en las terrazas), a las mejores matemáticas, a los mejores emprendedores (aunque nunca emprendieron nada, de momento), a mis amigos de ECLA (que hicieron de mi estancia en París una experiencia inolvidable), y a todos los que formáis parte imprescindible de mi vida. Vosotros y vosotras sabéis quiénes sois y lo que significáis. Gracias por ser parte de esto.

A mi familia, mis abuelos, mis tíos, mi hogar. Sois mi mayor aprendizaje.

Finalmente, gracias por todo a las tres personas más importantes de mi vida. A mis padres, gracias por darme absolutamente todo, amor incondicional. A Miguel, por compartir la vida juntos, porque tras nueve años aún tenemos muchos sueños por cumplir de la mano. Siempre. Esta tesis es vuestra.

Abstract

As is widely accepted, the data *explosion* has led to a data revolution, with an increasing number of data-driven problems and approaches being developed, providing insights and solutions to existing challenges. In this line, data science arises as a multidisciplinary field that is fundamentally based on statistics and computation, and is not only concerned with the analysis of information, but involves the entire data lifecycle, including: planning, collecting, curating, analyzing, monitoring, publishing, serving and preserving.

Because of this, the study and further development of different privacy techniques in a data science and data analysis contexts is a key area for scientific research. Specifically, throughout this thesis different techniques for the secure processing, analysis and management of data, with special focus on data privacy and privacy preserving machine learning are investigated.

In particular, after introducing the basic concepts related to data science and big data, ethical, social and legal issues are addressed, including regulations such as the GDPR and the AI Act, and the importance of open science. All this to motivate and contextualize the current context and introduce the issues that lead us to say that we are living a data revolution that requires special attention to the privacy issues arising from it.

The first block of this thesis focuses on data privacy, addressing the basis of anonymization and pseudonymization techniques, as well as attacks and metrics to assess risks and balance with usability. Two tools developed in this work are presented: *pyCANON* and *anjana*, Python libraries for evaluating and applying anonymization on datasets. In addition, the impact of anonymization on the performance of machine learning models is discussed. Moreover, differential privacy, both local and global, and its mechanisms are also introduced. Applications for publishing and analyzing data using deep learning models are explored, considering approaches such as metric privacy and Rényi differential privacy, and different frameworks for their implementation.

The second major block of this thesis explores what is known as privacy preserving machine learning. Specifically, we focus on a technique known as federated learning, which allows machine and deep learning models to be developed without the need for data sharing or data leaving the storage or acquisition site, even for training purposes. Theoretical issues of this architecture and their implementation are detailed, as well as advantages, types, aggregation strategies and open

problems. In addition, issues related to drift monitoring and the inclusion of privacy enhancing technologies such as differential privacy or homomorphic encryption are discussed. Then, different frameworks for federated learning are presented and evaluated, concluding with a comparison with other existing distributed learning architectures. Within this block enters another fundamental part of the thesis, which is the transition to real world applications of what has been studied. Specifically, three applications of federated learning are presented in different sectors: medical, water quality monitoring and climate sciences.

Finally, the contributions derived from this thesis as well as the results in terms of scientific publications, congresses, software, etc., are presented, together with some potential lines of future work in the field.

Resumen

Como es ampliamente aceptado, la *explosión* de datos ha dado lugar a una revolución de datos, con un incremento del número de problemas y enfoques basados en datos que se están desarrollando, proporcionando nuevas perspectivas y soluciones a los retos existentes. En esta línea, la ciencia de datos es un campo multidisciplinar que se basa fundamentalmente en la estadística y la computación, y que no sólo se refiere al análisis de la información, sino que implica todo el ciclo de vida de los datos, incluyendo: planificación, recopilación, curado, análisis, monitorización, publicación, puesta en producción y preservación.

Por ello, el estudio y posterior desarrollo de diferentes técnicas de privacidad en un contexto de ciencia y análisis de datos es un área clave para la investigación científica. En concreto, a lo largo de esta tesis se investigan diferentes técnicas para el procesamiento, análisis y gestión seguros de datos, con especial atención a la privacidad de los datos y a la preservación de privacidad en el contexto de aprendizaje automático.

En concreto, tras introducir los conceptos básicos relacionados con la ciencia de datos y el big data, se abordan cuestiones éticas, sociales y legales, incluyendo normativas como el RGPD y la Ley de Inteligencia Artificial, y la importancia de la ciencia abierta. Todo ello para motivar y contextualizar el momento actual e introducir las cuestiones que nos llevan a afirmar que estamos viviendo una revolución de los datos que requiere especial atención a las cuestiones de privacidad derivadas de la misma.

El primer bloque de esta tesis se centra en la privacidad de los datos, abordando las bases de las técnicas de anonimización y pseudonimización, así como los ataques y métricas para evaluar los riesgos y el equilibrio con la utilidad. Se presentan dos herramientas desarrolladas en este trabajo: *pyCANON* y *anjana*, librerías Python para evaluar y aplicar técnicas de anonimización sobre conjuntos de datos. Además, se analiza el impacto de la anonimización en el rendimiento de los modelos de aprendizaje automático. Por otra parte, también se introduce la privacidad diferencial, tanto local como global, y sus mecanismos. Se exploran las aplicaciones para publicar y analizar datos utilizando modelos de aprendizaje profundo, considerando enfoques como la privacidad métrica y la privacidad diferencial Rényi, así como diferentes librerías para su implementación.

El segundo gran bloque de esta tesis explora lo que se conoce como aprendizaje automático que preserva la privacidad. En concreto, nos centramos en una técnica

conocida como aprendizaje federado, que permite desarrollar modelos de aprendizaje automático y profundo sin necesidad de compartir datos ni que estos salgan del lugar de almacenamiento o adquisición, ni siquiera para el entrenamiento. Se detallan cuestiones teóricas de esta arquitectura y su implementación, así como ventajas, tipos, estrategias de agregación y problemas abiertos. Además, se discuten cuestiones relacionadas con la monitorización de la deriva y con la inclusión de tecnologías de mejora de la privacidad, como la privacidad diferencial o el cifrado homomórfico. A continuación, se presentan y evalúan diferentes librerías para aprendizaje federado, concluyendo con una comparativa de otras arquitecturas de aprendizaje distribuido existentes. Dentro de este bloque entra otra parte fundamental de la tesis, que es el paso a aplicaciones en el ámbito real de lo estudiado. En concreto, se presentan tres aplicaciones del aprendizaje federado en diferentes sectores: médico, monitorización de la calidad del agua y ciencias climáticas.

Finalmente, se presentan las aportaciones derivadas de esta tesis así como los resultados en términos de publicaciones científicas, congresos, software, etc., junto con algunas potenciales líneas de trabajo futuro en el campo.

CONTENTS

List of Tables **VI**

List of Figures **X**

List of Acronyms **XV**

Thesis Statement **XXI**

Resumen en Español **XXV**

I Data Revolution 1

1 Introduction and Motivation **3**

1.1 Big Data and Data Science 7

1.1.1 Big Data 8

1.1.2 Data Science and Artificial Intelligence 9

1.2 Ethical, Social and Legal Considerations 14

1.2.1 Gender Perspective: Bias in Data 15

1.2.2 Ethical and Social Issues Related to Data 17

1.2.3 General Data Protection Regulation 18

1.2.4 Data Governance Act 20

1.2.5 Data Act 21

1.2.6 European Union Artificial Intelligence Act 22

1.3 Open Science 24

1.3.1 Open Data 25

1.3.2 Open Access 27
1.3.3 Open Source and Software 28
1.3.4 European Open Science Cloud 30
1.4 Motivation 30
1.4.1 Why Data Revolution? 31
1.4.2 Thesis Road-Map: from Data Privacy to Privacy Preserving Machine Learning 32

II Data Privacy **35**

2 Anonymization and Pseudonymization **37**
2.1 Pseudonymization Techniques 45
2.1.1 Classic Methods 47
2.1.1.1 Counters and Random Numbers 47
2.1.1.2 Hash Functions 48
2.1.1.3 Hash Function with Authentication Code 49
2.1.1.4 Encryption 50
2.2 Anonymization Techniques 52
2.2.1 k-anonymity 54
2.2.2 (α,k)-anonymity 56
2.2.3 ℓ-diversity 57
2.2.4 Entropy ℓ-diversity 59
2.2.5 Recursive (c,ℓ)-diversity 60
2.2.6 t-closeness 61
2.2.7 Basic β-likeness and enhanced β-likeness 62
2.2.8 δ-disclosure privacy 63
2.3 Common Attacks on Anonymized Databases 64
2.4 Privacy-Utility Trade-Off Metrics 68
2.5 Disclosure and Re-Identification Risk Control 70
2.6 *pyCANON*: a Python Library to Check the Level of Anonymity of a Dataset 73
2.6.1 Main Functionalities 73
2.6.2 Usage Examples 75
2.6.3 Further Functionalities 77
2.6.4 Summary 78
2.7 Impact of Anonymization Techniques in Machine Learning Models Training 78

2.7.1 Data Analyzed and Hierarchies 79
2.7.2 Machine Learning Models Under Study 80
2.7.3 Results and Analysis 82
2.7.3.1 Varying the Value of k for *k-anonymity* 82
2.7.3.2 Applying Further Anonymity Techniques 84
2.7.4 Summary and Wrap Up 86
2.8 *Anjana*: a Python Library for Anonymizing Sensitive data 86
2.8.1 Algorithms to Achieve *k-anonymity* 87
2.8.2 Methodology 90
2.8.3 Usage Examples 91
2.8.4 Further Functionalities 93
2.8.5 Summary 98
2.9 Conclusions 98

3 Differential Privacy 101
3.1 Theoretical Basis 103
3.1.1 Introducing Differential Privacy 104
3.1.2 Input Privacy and Output Privacy 106
3.2 Global Differential Privacy 108
3.2.1 Laplace Mechanism 108
3.2.2 Gaussian Mechanism 111
3.2.3 Exponential Mechanism 112
3.3 Local Differential Privacy 114
3.3.1 Randomized Response 114
3.3.2 Histogram Representation 116
3.4 Rényi Differential Privacy 118
3.5 Metric Differential Privacy 120
3.6 Differential Privacy for Data Publishing 122
3.7 Differential Privacy for Data Analysis 123
3.7.1 Deep Learning Meets Differential Privacy 123
3.8 Review of Differential Privacy Frameworks 127
3.9 Summary and Conclusions 130

III Privacy Preserving Machine Learning 133

4 Federated Learning 135
4.1 Machine Learning turns Distributed 137
4.1.1 Computing Resources-Aware Distributed Learning 138

4.1.2 Privacy-Aware Distributed Learning 141
4.2 Introduction to Federated Learning 142
4.2.1 Motivation 142
4.2.2 Basis of Federated Learning 145
4.2.3 Basic Implementation 148
4.2.4 Types of Federated Learning 151
4.2.4.1 Cross Silo and Cross Device Federated Learning . . . 151
4.2.4.2 Horizontal and Vertical Federated Learning 152
4.2.5 Aggregation Functions 153
4.2.5.1 Federated Average (FedAvg) 154
4.2.5.2 Federated Average with Momemtum (FedAvgM) . . 154
4.2.5.3 Federated Median (FedMedian) 154
4.2.5.4 FedProx 154
4.2.5.5 Federated Optimization (FedOpt, FedAdam, FedYogi and FedAdagrad) 155
4.2.5.6 Scaffold 155
4.2.5.7 Ditto 155
4.2.5.8 A Novel Aggregation Strategy: *FedAvgOpt* 156
4.3 Open Challenges in Federated Learning 157
4.4 Federated Learning in Production: Drift Detection 160
4.5 Differential Privacy in Federated Learning 164
4.6 Secure Aggregation via Homomorphic Encryption 166
4.6.1 Homomorphic Encryption and Federated Learning 167
4.6.1.1 Multi Key Homomorphic Encryption 169
4.7 Python Libraries for Federated Learning 171
4.8 Other Distributed Learning Architectures 173
4.8.1 All-Reduce Architecture 173
4.8.2 Ring-All-Reduce Architecture 174
4.8.3 Neighbor Architecture 174
4.8.4 Gossip Learning 174
4.8.4.1 Consensus Algorithms 175
4.9 Summary and Conclusions 178

5 Federated Learning Applications 181
5.1 Medical Imaging: Classification of Chest X-Ray Images 185
5.1.1 Data Availability and Preprocessing 186
5.1.2 Deep Learning Model 186
5.1.3 Federated Learning Approach (3 Clients) 187

5.1.3.1 Data Distribution 192
5.1.4 Federated Learning Approach (10 Clients) 193
5.1.5 Comparison: Centralized vs Federated 196
5.1.6 Federated Learning with Intermittent Clients 197
5.1.7 Federated Learning Meets Differential Privacy 201
5.1.8 Comparing Different Aggregation Strategies 205
5.1.9 Conclusions 206
5.2 Water Quality: Data Based Estimation of High Frequency Nutrient Concentrations 206
5.2.1 Data Availability and Processing 208
5.2.2 Deep Learning Model Analyzed 214
5.2.3 Methodology 215
5.2.4 Results and Discussion 216
5.2.5 Conclusions 228
5.3 Climate Sciences: Vertical Integrated Liquid Nowcasting using Weather Radar Data 229
5.3.1 Data Availability and Preprocessing 231
5.3.2 Methodology 233
5.3.3 Benchmark Model: COTREC 233
5.3.4 Deep Learning and Federated Learning Models 234
5.3.5 Personalized Federated Learning 236
5.3.6 Convolutional Neural Architecture 239
5.3.7 Results and Analysis 240
5.3.8 Conclusions 247
5.4 Summary and Wrap Up 248

IV Conclusions 251

6 Conclusions and Future Work 253
6.1 Main Contributions 255
6.2 Publications and Other Achievements 257
6.2.1 Scientific Journals 257
6.2.2 Peer Reviewed Conference Papers 259
6.2.3 International Stay as Visiting Researcher at INRIA 259
6.2.4 Invited Talks 260
6.2.5 Scientific Posters 261
6.2.6 Software 262

6.3 Future Work and Perspective 262

V Appendix 265

A Basic Algebra for Cryptography 267
A.1 Algebraic Structures 267
A.2 Homomorphisms and Isomorphisms 269
A.3 Introduction to Cryptosystems 269
A.4 Essential (and Beautiful) Results from Algebra for Cryptography . . . 271
A.5 Ring Learning With Errors 272

B FedAvgOpt Implementation 273

C Federated learning in AI4EOSC 277
C.1 AI4EOSC Platform 277
C.2 Federated Learning in AI4EOSC 278
C.2.1 Deploying and Starting the Federated Learning Server 279
C.2.2 Creating and Running the Clients 281
C.2.3 Extensions and Modifications to the Flower Library 281
C.2.3.1 Running the Server Behind a Proxy 282
C.2.3.2 Client Authentication 282
C.2.3.3 Integrating Metric Privacy 284
C.2.3.4 Carbon Footprint Monitoring 286
C.3 Summary and Conclusions 289

D Computing Facilities 291

Bibliography 319

List of Tables

Chapter 1: Introduction and Motivation

1.1 Example of gender bias when translating from English to Spanish using AI-based translators. 16
1.2 Open data ideal characteristics. Adapted from [1]. 26

Chapter 2: Anonymization and Pseudonymization

2.1 Original database (extracted from the first 10 rows of drug classification dataset [2], available in Kaggle). 44
2.2 Database obtained generalizing the QI *Age* and *Na to K*. 44
2.3 Database obtained generalizing the QI *Age* and *Na to K*, and suppressing *BP* and *Cholesterol*. 45
2.4 Example of database to pseudonymize for the identifier *name*. 51
2.5 Pseudonymized version of Table 2.4 applying MD5 to the field *name*. 51
2.6 Mapping of the identifier (*name*) using five pseudonymization methods. 52
2.7 Example of a k-anonymous database with $k = 2$. 56
2.8 Database given in Table 2.7 with a new sensitive attribute: *Hospitalization date*. 58
2.9 Simulated database used as example. Five equivalence classes are separated. Set of QIs: *Age*, *Sex* and *Native country*, and SA: *Study level*[1] and *Salary*. 64
2.10 Anonymization techniques and principal attacks that prevent (among others). Extracted from [3]. 66

2.11 Accuracy and AUC obtained for each machine learning model when varying the value of k for k-anonymity. 83
2.12 Accuracy obtained with each machine learning model according to the anonymization technique applied. 84
2.13 AUC obtained with each machine learning model according to the anonymization technique applied. 84
2.14 Example database. Quasi-identifiers: *age*, *ZIP code* and *sex*. Sensitive attribute: *disease*. Adapted de-anonymized version of Table 2.7. . . . 88
2.15 Example database anonymized for *k-anonymity* with $k = 2$. 89

Chapter 3: Differential Privacy

3.1 Example: number of data from each class in the train and test datasets.126
3.2 Example: results obtained when performing the binary prediction in the test set in terms of different metrics with SGD and DP-SGD as optimizers for the gradient descent process. 127
3.3 Review of differential privacy frameworks (1/2). 128
3.4 Review of differential privacy frameworks (2/2). 129

Chapter 4: Federated Learning

4.1 Summary: decentralized learning architectures explored. Adapted from [4]. 177

Chapter 5: Federated Learning Applications

5.1 Distribution of the X-Ray images in train, test and validation sets. . . 186
5.2 Number of train and test data of each client and average training time per epoch. Case: 3 clients. 189
5.3 Centralized approach vs decentralized approach. Case: 3 clients. . . 190
5.4 Decentralized approach. Metrics obtained for the test data varying N_r and with $N_e = 1$. Case: 3 clients. 190
5.5 Metrics obtained for the test set of each client varying N_r with $N_e = 1$ fixed. Case: 3 clients. 191
5.6 Data distribution for the three clients (without distinguishing train and test sets). 192
5.7 Number of train and test data of each client and average training time per epoch. Case: 10 clients. 193
5.8 Centralized approach vs decentralized approach. Case: 10 clients. . . 194

5.9 Decentralized approach. Metrics obtained for the test data varying N_r with $N_e = 1$ fixed. Case: 10 clients. 194
5.10 Decentralized approach. Optimal values for each client's test set. Case: 10 clients. 195
5.11 Comparison of the centralized approach and the two FL cases: 3 and 10 clients. Best performance in terms of loss, accuracy and AUC. The time reduction column corresponds with the reduction obtained with the FL architectures regarding the centralized approach. 196
5.12 Results obtained by eliminating one client and adding a new one with the two approaches described above. Case: 3 clients. 199
5.13 Results obtained by eliminating one client and adding a new one with the two approaches described above. Case: 10 clients. 199
5.14 Results obtained for the test set considering the last repetition of the FL schema and approaches A and B. Client i is the intermittent client considered in this example. Case: 3 clients. 201
5.15 Results obtained for the test set considering the last repetition of the FL schema and approaches A and B. Client i is the intermittent client considered in this example. Case: 10 clients. 202
5.16 Mean AUC and standard deviation for each client test set when varying the noise multiplier (γ). 203
5.17 Mean accuracy and standard deviation for each client test set when varying the noise multiplier (γ). 203
5.18 Mean and standard deviation of the accuracy and AUC obtained in round 10 of federated training with each noise multiplier value analyzed. 204
5.19 Chest X-Ray use case with 3 clients. Accuracy, loss and AUC in the test set of each client with each aggregation strategy. 205
5.20 Statistics associated to each parameter for the rivers Enborne and The Cut. 213
5.21 Summary of results in terms of MRE for train and test with the three approaches and using three features. 216
5.22 Summary of results in terms of MRE for train and test with the three approaches and using six features. 220
5.23 Summary of results in terms of MRE for train and test with the three approaches and using six features in a scenario with the data reduced to coincident dates in both sites. 224

5.24 Summary of results in terms of MRE for train and test with the three approaches and using six features in a scenario with data reduced to a periodicity of 24 hour for training, and testing every one hour. . . . 225
5.25 Summary of results in terms of MRE for train and test with the three approaches and using six features in a scenario with data reduced to a periodicity of one week. Prediction: every one hour. 227
5.26 Statistics regarding the mean VIL in each of the images to be predicted in the test set of each of the four zones under study. In the case of the minimum and the maximum, the average of the min and max values for each of the images are shown. 233
5.27 Number of data after processing in each zone for train and test. . . . 237
5.28 MSE (kg/m^2) obtained in the test set with the different approaches in each zone. 241
5.29 MAE (kg/m^2) obtained in the test set with the different approaches in each zone. 241
5.30 Skill score obtained with each approach in the test set of each zone. . 242
5.31 Comparison of the MSE (kg/m^2), MAE (kg/m^2) and skill score obtained in the train and test sets of the central zone with the different approaches analyzed. 245
5.32 $d_{i,j}$ calculated for each combination of zones. 247

List of Figures

Chapter 1: Introduction and Motivation

1.1 Knowledge Pyramid, adapted from [1]. 6
1.2 Diagram of the simplified life cycle of a data science project. 10
1.3 Summarized AI advances timeline. 12
1.4 Summarized Venn diagram: open science, data, access and source. . 25

Chapter 2: Anonymization and Pseudonymization

2.1 Guidelines for selecting the anonymization technique to be applied according to the data privacy objective to be achieved, the attacks to be prevented and the level of strictness. Extracted from [5]. 67
2.2 Example: evolution of t and β (in logarithmic scale) when varying k (for *t-closeness*, *basic* β*-likeness* and *k-anonymity* respectively). Extracted from [3]. 74
2.3 Example: part of the report obtained when executing the code given in Example Code 2.2. 77
2.4 Anonymizing Table 2.14 using *mondrian* for the QIs *age* and *ZIP code*. 88
2.5 Workflow: anonymizing a tabular dataset for a given value of k for *k-anonymity*. 90
2.6 Summary: harmonizing the transformation to be applied to multiple databases $\xi_j \ \forall j \in \{1,\ldots,N\}$. 97

Chapter 3: Differential Privacy

3.1 Schematic idea of the local and global differential privacy approaches.107

3.2 Probability density function (PDF) of the Laplace distribution. 108
3.3 Example: mean BMI of all persons in the database who suffered a stroke. Results applying DP with the Laplace mechanism for different values of the privacy budget vs the actual value. 110
3.4 Example: mean BMI of all persons in the database who suffered a stroke. Results applying DP with the Gaussian mechanism for different values of the privacy budget vs the actual value. 112
3.5 Example: original and DP randomized histograms for the average glucose level of the Stroke Dataset. 118
3.6 Wrap up: global, local, Rényi and metric differential privacy. 122
3.7 Example: samples extracted from the training dataset. 126

Chapter 4: Federated Learning

4.1 Model and data parallelism: intuitive view from the node or worker's perspective. 139
4.2 Data parallelism scheme: distributing the training across multiple processors. 140
4.3 Classic scheme of centralized machine learning. 142
4.4 Edge computing scheme. (1) The IoT or edge devices capture the data; (2) data are processed and analyzed including the training of ML/DL models at the edge of the network if there is sufficient computing and storage capacity, allowing them to provide real-time response (3) finally, data can be stored in data lakes, data centers and cloud servers for further research, analysis and preservation. 144
4.5 Federated learning scheme. Extracted from [6]. 145
4.6 Schematic idea of the distribution of the data in the different clients of a VFL architecture. Adapted from [6]. 153
4.7 Schematic idea of the basic integration of HE in an FL architecture in which all clients use the same public and private key pair. 168
4.8 Visual overview: server independent distributed learning architectures analyzed. 176

Chapter 5: Federated Learning Application

5.1 Example of the two categories of chest X-ray images under study (pneumonia and normal), obtained from each client train set. 188
5.2 ROC curves for each of the cases exposed in Table 5.11 197
5.3 Outliers detection of the Chl concentrations for the *river Enborne*. . . 211

5.4 Outliers detection of the Chl concentrations for the *river the Cut*. . . . 211
5.5 Violin plots with the features distribution for the river *Enborne* and *The Cut* in the train and test splits. 212
5.6 Pearson correlations between the six features (Enborne and The Cut). Triangle heat-map. 213
5.7 Prediction obtained for the test set of each river with the three learning approaches using the three physico-chemical features. 217
5.8 Mean of the absolute value of the SHAP values obtained for each feature with the three different approaches exposed in Table 5.21 for each river (3 features). 219
5.9 Mean of the absolute value of the SHAP values obtained for each feature with the four learning approaches presented in Table 5.22 for both Enborne and The Cut. 221
5.10 Prediction obtained for the test set of both rivers with each learning approach using six features. 222
5.11 Prediction obtained for the test set of *rivers Enborne and The Cut* with each model analyzed. Data reduction: same dates. 224
5.12 Prediction obtained for the test set of *rivers Enborne and The Cut* with each model analyzed. Data reduction: 24 hours. 226
5.13 Prediction obtained for the test set of *rivers Enborne and The Cut* with each model analyzed. Data reduction: 24 hours. 228
5.14 Radar images examples (regarding VIL). Images obtained after rescaling to 100×100 resolution. 232
5.15 Example of an image where the four quadrants into which it has been divided to create the four zones are shown in blue. 235
5.16 Average of all the radar images available in April, May, June and July 2016 by zone once processed. 236
5.17 Schema of the three learning paradigms implemented: individual, federated and adaptive federated learning. 239
5.18 Example of predictions for the zones 1 and 2 with *adapFL*. 242
5.19 Example of predictions for the zones 3 and 4 with *adapFL*. 243
5.20 Example of images showing the four quadrants into which it the initial images have been divided to create the four zones (black) and the central area (marked in red). 244
5.21 Example of predictions of the test set of the central zone. Federated learning approach with $N_r = 10$ and $N_e = 10$ and *adapFL* method with the configuration (9,10). 246

List of Acronyms

A

AI

Artificial Intelligence .... 5, 11–13, 15, 17, 22, 23, 30, 31, 33, 39, 131, 137, 172, 173, 178, 207, 208, 255, 256, 263, 264, 277, 278, XI

AI4EOSC

Artificial Intelligence for the European Open Science Cloud . 256, 277–280, 282–284, 287, 289, 291, XXIX

ANN

Artificial Neural Network 13, 123, 214, 234

AUC

Area Under the ROC Curve . 82–86, 187, 190, 191, 194–197, 202–206, VIII, IX

C

CNN

Convolutional Neural Network 13, 14, 125, 126

COTREC

Continuity Tracking Radar Echoes by Correlation 185, 231, 233–235, 240–245, 247–249

CPU

Central Processing Unit 279, 291

D

DGA

Data Governance Act 20–22

DL

Deep Learning .... 11–14, 27, 30, 32, 33, 39, 65, 78, 95, 98, 107, 121, 123, 125, 127, 128, 130, 131, 137, 139, 140, 144, 146, 151, 157, 163, 171, 172, 206, 207, 209, 216, 228, 230, 233–236, 238, 240, 241, 243, 244, 247, 249, 255, 256, 277, 278, 286, 287, XII

DML

Distributed Machine Learning 138, 142

DP

Differential Privacy .. 44, 104, 106–108, 110–112, 114, 115, 117, 119–131, 137, 158–160, 164–166, 170, 171, 175, 178, 184–186, 201, 202, 204, 206, 248, 256, 259, 263, 284, 285, 288–290, VIII, XII

E

EC

Equivalence class 42, 59, 61, 63

EOSC

European Open Science Cloud ..... 30, 256, 277, 290, XXVIII, XXIX, XXXVI

F

FAIR

Findable, Accessible, Interoperable, Reusable 24, 26, 27, 30

FHE

Fully Homomorphic Encryption 167

FL

Federated Learning . 141, 142, 145–147, 149, 151–154, 157–159, 163–165, 167, 169–173, 175, 177, 178, 184–187, 189, 191, 196, 197, 200–202, 205, 206, 216, 218–223, 225–230, 234, 236–249, 256, 257, 263, 276, 278, 280, 282, 284–290, IX, XXIX

G

GDP

Global Differential Privacy 107, 122, 123, 129, 131

GDPR

General Data Protection Regulation 18–21, 23, 24, 41, 46, 47

GL

Gossip Learning 175, 177, 179

GPU

Graphics Processing Unit 31, 139, 140, 172, 278, 279, 289, 291

H

HE

Homomorphic Encryption 106, 137, 158–160, 166–170, 175, 178, 263, 267, 271

HFL

Horizontal Federated Learning 152

I

IoT

Internet of Things ..... 5, 10, 22, 31, 138, 143, 144, 263, XII, XXXV

L

LDP

Local Differential Privacy 107, 114, 116, 122, 123, 131

LLM

Large Language Model ..... 13, 22, 31, 172, 264, XXXVI

LSTM

Long Short-Term Memory 230

M

MKHE

Multi Key Homomorphic Encryption 169, 170, 178, 263, 267

ML

Machine Learning ..... 11–14, 27, 30, 32, 33, 39, 44, 65, 78, 81, 82, 85, 95, 98, 99, 107, 123, 127, 128, 130, 137–140, 142, 144, 146, 151, 157, 171, 172, 207, 208, 215, 228, 238, 255, 256, 277, 278, 286, 287, XII

MLOps

Machine Learning Operations 11, 278

P

PET

Privacy Enhancing Technology ..... 159, 175, 263, 290, XXXV

PFL

Personalized Federated Learning 185, 230, 234, 236, 237, 240, 245, 247–249, 257

PHE

Partial Homomorphic Encryption 167

PPML

Privacy Preserving Machine Learning 32, 33, 159

Q

QI

Quasi-Identifier ... 42, 44, 54, 64, 70, 71, 74, 75, 77, 87–89, 91, 93–97, VII

R

RDP

Rényi Differential Privacy 118–120, 129

ROC

Receiver Operating Characteristic ..... 196, 197, XII

S

SA

Sensitive Attribute ..... 42, 62, 64, 70, 74, 88, 91, 93, VII

SEE

Secure Execution Environment 106, 264

SIESTA

Secure Interactive Environments for Sensitive daTa Analytics ... 256, XXVIII

SMPC

Secure Multi-Party Computation ..... 106, 160, 263, XXXV

SWHE

Somewhat Homomorphic Encryption 167

T

TEE

Trusted Execution Environment 106, 264

V

VFL

Vertical Federated Learning ..... 152, 153, XII

VIL

Vertical Integrated Liquid ..... 230–234, 245, 247, 249, X, XIII

THESIS STATEMENT

Objectives and Description of the Work

This manuscript is structured in four parts as follows:

- **Part I: Data Revolution**

  This first part contains an introduction and motivation for the present doctoral thesis, starting by establishing the basics about *big data*, *data science* and *artificial intelligence*. Next, different ethical, social and legal considerations to be taken into account when working in the field of data science and artificial intelligence are presented, with special attention to the normative regulation that applies in the European Union. Subsequently, the principles for open science are discussed, including open data, access and source, as well as presenting the European Open Science Cloud. Finally, it motivates the work presented in this thesis by focusing on why we consider that we are facing a data revolution and what kind of problems we encounter in the field of data science in relation to privacy.

- **Part II: Data privacy**

  This second part develops two technical chapters related to privacy techniques that can be applied to data during data processing to preserve data privacy. First, Chapter 2 shows the basics of anonymization and pseudonymization, as well as different methods applied in this field. Two software implementations for anonymizing sensitive tabular data and for checking the anonymity level of a dataset are also presented. Additionally, an analysis of the performance of different machine learning models when applied to

anonymized data with different levels of privacy depending on the applied technique is presented, as well as privacy utility trade off and disclosure risk metrics

In Chapter 3, the theoretical basis of differential privacy are presented along with different mechanisms that can be applied to ensure such a definition in the case of local and global differential privacy. Different differential privacy approaches, such as Rényi or metric privacy, are presented, as well as different approaches depending on whether they are intended to be applied for data publication or for data analysis. In relation to analysis, a review on the application of differential privacy in deep learning models, especially during the gradient descent process, is included. Finally, different openly available differential privacy software products are reviewed.

- **Part III: Privacy Preserving Machine Learning**

  This part contains the core of the work related to privacy preserving machine and deep learning. Specifically, Chapter 4 presents the idea of distributed machine learning and introduces the federated learning architecture. This architecture is the main focus of this entire chapter, including an extensive review of the theoretical basis and the advantages from a privacy point of view, as well as open problems. Next, a basic implementation example is shown and different types of federated learning are presented. Then, different aggregation functions are explored and a new strategy that seeks to improve the convergence of the state-of-the-art functions is introduced. This is followed by an analysis of different challenges that arise during the implementation of this architecture, from communication problems to privacy issues and data heterogeneity, among others. Subsequently, issues related to the implementation of these systems and the changes in client distributions are explored, as well as the possibility of integrating differential privacy at different stages of the federated learning architecture. In addition, aspects related to secure aggregation in federated learning using homomorphic encryption are explored. Before concluding this chapter, different open software products for the use of this architecture are presented and, finally, different distributed architectures for training deep and machine learning models without dependence on a central server are analyzed.

  Next, in Chapter 5 different use cases are presented, covering different fields: medical imaging, water quality and climate science. The first case shows the implementation of this architecture in a problem of binary classification of

chest X-ray images by varying the number of clients and analyzing different scenarios of intermittent clients. Additionally, the incorporation of differential privacy in the training of the models under the federated learning architecture is studied and finally, different aggregation strategies are compared, including a novel approach presented in the Chapter 4. Regarding the second use case, federated learning is applied to a regression problem to estimate chlorophyll concentration in different rivers. Centralized, individual and federated approaches are compared and three data reduction scenarios are analyzed, simulating the case where sampling is less frequent. Finally, the third case consists of the nowcasting prediction of the vertical integrated liquid based on radar images. Thus, the aim is to predict this parameter in the next five minutes based on the previous radar images. In this case, the application of a personalized federated learning architecture is proposed to predict as accurately as possible in each area and it is compared with the classical federated approach as well as with the individual training in each area and with a classical model used in the field. In addition, an analysis of the divergence between the different areas studied is carried out. Finally, a detailed summary of three of the different use cases studied is presented.

- **Part IV: Conclusions and Future Work**

  This last part seeks to outline the main contributions made in the present doctoral thesis, as well as the articles published in high impact journals and in high level international conferences. Invited lectures, contributions during the pre-doctoral stay in an international center, posters and different software products implemented are also presented. Finally, the future work that emerges from this research is presented in relation to the perspective that is foreseen in the field.

Resumen en Español

Objetivos y descripción del trabajo

La presente tesis se estructura en cuatro partes como sigue:

- **Parte I: Revolución de los datos**

  Esta primera parte contiene una introducción y motivación para la presente tesis doctoral, comenzando por sentar las bases sobre el *big data*, la *ciencia de datos* y la *inteligencia artificial*. A continuación, se exponen distintas consideraciones éticas, sociales y legales a tener cuenta cuando se trabaja en el campo de la ciencia de datos y la inteligencia artificial, con especial atención a la regulación normativa que aplica en la Unión Europea. Posteriormente, se analizan los principios para la ciencia abierta, incluyendo datos, acceso y código abierto, así como presentando la Nube Europea de Ciencia Abierta. Finalmente, se motiva el trabajo presentado en esta tesis recayendo en por qué consideramos que estamos ante una revolución de los datos y que tipo de problemas enfrentamos en el campo de la ciencia de datos en relación con la privacidad.

- **Parte II: Privacidad de datos**

  En esta segunda parte se desarrollan dos capítulos técnicos relacionados con técnicas de privacidad que pueden aplicarse sobre los datos durante su procesado para preservar la privacidad de los mismos. En primer lugar, en el Capítulo 2 se presentan los fundamentos de las técnicas de anonimización y pseudonimización, así como distintos métodos que aplican en este campo. Se exponen además dos software implementados para anonimizar datos sensibles tabulares y para comprobar el nivel de anonimato de un conjunto de

datos. Adicionalmente, se presenta un análisis del rendimiento de distintos modelos de aprendizaje automático cuando se aplican sobre datos anonimizados con distintos niveles de privacidad en función de la técnica aplicada, así como métricas para lograr equilibrio entre la utilidad y la privacidad de los datos y para medir el riesgo de divulgación de información.

En el Capítulo 3 se presentan las bases teóricas de la privacidad diferencial junto con distintos mecanismo que puedan aplicarse para garantizar dicha definición en el caso de privacidad diferencial local o global. Se presentan distintos enfoques de privacidad diferencial, como Rényi o métrica, así como distintas variantes dependiendo de si se desea aplicar para la publicación de los datos o para su análisis. En relación con el análisis, se incluye una revisión sobre la aplicación de privacidad diferencial en modelos de aprendizaje profundo, especialmente durante el proceso de descenso de gradiente. Finalmente, se revisan distintos productos de software de privacidad diferencial disponibles en abierto.

- **Parte III: Preservación de la privacidad en aprendizaje automático**

  Esta parte contiene el grueso del trabajo en relación con preservación de la privacidad en aprendizaje automático y profundo. En concreto, en el Capítulo 4 se presenta la idea de aprendizaje automático distribuido y se introduce la arquitectura de aprendizaje federado. En dicha arquitectura se centra el desarrollo de todo este capítulo, incluyendo una amplia revisión de las bases teóricas y las ventajas desde el punto de vista de la privacidad, así como problemas abiertos. A continuación, se muestra un ejemplo de implementación básica y se presentan distintos tipos de aprendizaje federado. A continuación se exploran distintas funciones de agregación y se introduce una nueva estrategia que busca mejorar la convergencia de las funciones del estado del arte. Seguidamente se analizan distintos retos que surgen durante la implementación de esta arquitectura, desde los problemas de comunicación hasta cuestiones de privacidad y heterogeneidad de los datos entre otros. Posteriormente se presentan exploran las cuestiones relacionadas con la puesta en producción de estos sistemas y los cambios en las distribuciones de los clientes, así como la posibilidad de integrar privacidad diferencial en distintas etapas de la arquitecture de aprendizaje federado. Además, se exploran aspectos relacionados con la agregación segura en aprendizaje federado usando cifrado homomórfico. Antes de concluir esta capítulo se presentan distintos productos de software abiertos para el uso de esta arquitectura y, finalmente, se analizan distintas arquitecturas distribuidas para el entrenamiento de mo-

delos de aprendizaje automático y profundo sin dependencia en un servidor central.

A continuación en el Capítulo 5 se presentan distintos casos de uso de aplicación de aprendizaje federado, cubriendo distintos campos: imagen médica, calidad de aguas y ciencias climáticas. En el primer caso se muestra la implementación de esta arquitectura en un problema de clasificación binaria de imágenes de radiografías de tórax variando el número de clientes y con el objetivo de analizar distintos escenarios de clientes intermitentes. Adicionalmente se estudia la incorporación de privacidad diferencial en el entrenamiento del modelos bajo la arquitectura de aprendizaje federado y por último se comparan distintas estrategias de agregación, incluida un enfoque novedoso presentado en el Capítulo 4. Respecto al segundo caso de uso, se aplica aprendizaje federado a un problema de regresión para estimar la concentración de clorofila en distintos ríos. Se comparan los enfoques centralizado, individual y federado y se analizan tres escenarios de reducción de datos, simulando el casos en que el muestro es menos frecuente. Por último, el tercer caso consiste en la predicción a corto plazo del flujo vertical integrado a partir de imágenes de radar. Así, se busca predecir dicha variable en los próximos cinco minutos basándose en las imágenes de radar previas. En este caso se propone la aplicación de una arquitectura de aprendizaje federado personalizado para predecir de la forma más precisa posible en cada zona y se compara con el enfoque federado clásico así como con el entrenamiento individual en cada área y con el modelo clásico empleado en el área. Además en este caso se realiza un análisis de la divergencia entre las distintas zonas estudiadas. Finalmente, se expone un resumen detallado de tres los distintos casos de uso estudiados.

- **Parte IV: Conclusiones y trabajo futuro**

  Esta última parte busca exponer las principales contribuciones realizadas en la presente tesis doctoral, así como los artículos publicados en revistas de alto impacto y en conferencias internacionales de alto nivel. Se presentan también las charlas invitadas, las contribuciones durante la estancia pre-doctoral en un centro internacional, los pósteres realizados y los distintos productos de software implementados. Finalmente, se presenta el trabajo futuro que se abre tras esta investigación en relación con la perspectiva que se prevee en el campo.

Principales resultados y conclusiones

A continuación se detallan las aportaciones realizadas en esta tesis doctoral divididas en dos partes: las relativas a los aspectos de privacidad relacionados con el procesamiento y la curación de datos, y las relacionadas con el la preservación de la privacidad en el contexto de aprendizaje automático y profundo, utilizando adicionalmente tecnologías de mejora de la privacidad

En este sentido, en este documento se detallan distintas líneas de acción dependiendo de si el foco se pone en el procesado de los datos en si mismo o en el los modelos y algoritmos desarrollados y aplicados. El objetivo es cubrir el ciclo de vida de los datos completo con especial atención a los aspectos de privacidad y seguridad que surgen desde el procesado hasta el análisis y desarrollo y despliegue de modelos (incluidos explorando las ventajas que algunas arquitecturas de preservación de la privacidad pueden aportar en términos de eficiencia de los modelos). Así, las conclusiones derivadas de esta tesis en forma de las contribuciones realizadas, distinguidas por área, se resumen de los siguientes puntos:

- Con relación a aspectos de privacidad en contextos de procesamiento de datos:
  - He desarrollado e implementado una librería de software abierto escrita en Python para comprobar el nivel de anonimato de un conjunto de datos: *pyCANON*.
  - He desarrollado e implementado una librería de software abierto escrita en Python para anonimizar datos sensibles tabulares: *anjana*.
  - He estudiado y revisado distintas técnicas de pseudonimización así como mejores prácticas para su aplicación en entornos donde se manejan datos sensibles, especialmente en el contexto del proyecto EOSC SIESTA.
  - He analizado el estado del arte relativo a la privacidad diferencial, incluyendo las versiones local, global, Rényi y métrica. He introducido su uso y despliegue en el desarrollado de modelos de aprendizaje automático y profundo diferencialmente privado así como para la publicación y análisis de datos bajo privacidad diferencial.

  Parte del trabajo expuesto anteriormente se enmarca en el contexto del proyecto del programa Horizonte Europa EOSC SIESTA, que busca proveer a los usuarios con una plataforma en la nube compuesta por un conjunto de herramientas, servicios y metodologías para el intercambio seguro y efectivo

de datos sensibles en el contexto del EOSC. Así, un prototipo de integración de estas herramientas (anonimización, pseudonimización y privacidad diferencial) se esta desarrollando actualmente en el contexto de dicho proyecto junto con su aplicación a distintos casos de uso.

- Con relación a aspectos de privacidad en el desarrollo de modelos de aprendizaje automático y profundo basados en datos:
  - He implementado una arquitectura de aprendizaje federado (FL) completa, siguiendo los conceptos introducidos en el Capítulo 4, como parte del proyecto Europeo AI4EOSC, ofreciendo esta funcionalidad dentro de la plataforma en la nube asociada al mismo. Adicionalmente, se ha integrado un sistema de gestión de secretos que permite la participación de solo los clientes que se autentiquen con un cierto token. En este sentido, se ha publicado un artículo de conferencia que detalla este trabajo. En el Anexo C se dan más detalles sobre esta integración.
  - He integrado distintas funcionalidad para el entrenamiento de modelos de aprendizaje federado en AI4EOSC, incluyendo soporte para privacidad diferencial métrica y para la monitorización de la huella de carbono. Respecto a la primera, se ha propuesto integrar privacidad diferencial métrica en arquitecturas de aprendizaje federado con el objetivo adicional de prevenir ataques de inferencia de clientes (además de las preservación de privacidad derivada de la inclusión de privacidad diferencial clásica). Para más detalles sobre esto ver el Anexo C
  - He llevado a cabo un análisis detallado de la aplicación de aprendizaje federado a distintos casos de uso: imagen médica, calidad de aguas y predicción meteorológica a corto plazo. En concreto se han llevado a cabo tres publicaciones en relación cada uno de estas aplicaciones en revistas de alto impacto.
  - He estudiado el impacto de incluir privacidad diferencial en esquemas de aprendizaje federado aplicado a casos de uso de imagen médica.
  - He diseñado un nuevo método de agregación en aprendizaje federado (*FedAvgOpt*) que mejora la convergencia de las estrategias de agregación clásicas en los casos de uso estudiados.En esta línea, he comparado la eficiencia en términos de precisión de distintas estrategias de agregación en dos casos de uso de imagen médica.
  - He evaluado el impacto de la aplicación de una arquitectura de aprendizaje federado en lugar de entrenamiento individual en un caso de

uso de calidad de monitorización de calidad de aguas en el caso de dos clientes heterogéneos. He analizado el impacto en la precisión del modelo en distintos escenarios de reducción de datos y la importancia de cada variable predictiva según la arquitectura aplicada.

– He propuesto una nueva arquitectura de aprendizaje federado personalizado (*adapFL*) basada en la idea de transferencia del aprendizaje, aplicado a la predicción climática a corto plazo basada en imágenes de radar. La idea aquí era analizar como la arquitectura propuesta consigue mejorar las predicción en distintos áreas del radar con respecto al enfoque de aprendizaje centralizado.

Publicaciones y contribuciones

En este apartado se destacan los diferentes resultados obtenidos en el transcurso de la tesis doctoral relacionados con publicaciones en revistas científicas de alto impacto, congresos internacionales, conferencias y charlas invitadas.

Publicaciones en revistas científicas

Los siguientes son artículos que he publicado como autora principal en revistas de alto impacto en el curso de la presente tesis doctoral:

2025 Nguyen, G.*, **Sáinz-Pardo Díaz, J.***, Calatrava, A., Berberi, L., Lytvyn, O., Kozlov, V., Tran, V., Moltó, G., & López García, Á. (2025). Landscape of machine learning evolution: privacy-preserving federated learning frameworks and tools. *Artificial Intelligence Review*, 58 (2), 51.
https://doi.org/10.1007/s10462-024-11036-2.
*Ambas autoras han contribuido a partes iguales a este trabajo.

2024 **Sáinz-Pardo Díaz, J.** & López García, Á. (2024). An Open Source Python Library for Anonymizing Sensitive Data. *Scientific Data*, 1289.
https://doi.org/10.1038/s41597-024-04019-z.

2024 **Sáinz-Pardo Díaz, J.**, Castrillo, M., Bartok, J., Heredia Cachá, I., Malkin Ondík, I., Martynovskyi, I., Alibabaei, K., Berberi, L., Kozlov, V. & López García, Á. (2024). Personalized Federated Learning for improving radar based precipitation nowcasting on heterogeneous areas. *Earth Science Informatics*, 17, 5561–5584.
https://doi.org/10.1007/s12145-024-01438-9.

2023 **Sáinz-Pardo Díaz, J.**, Castrillo, M., & López García, Á. (2023). Deep learning based soft-sensor for continuous chlorophyll estimation on decentralized data. *Water Research*, 120726.
https://doi.org/10.1016/j.watres.2023.120726.

2023 **Sáinz-Pardo Díaz, J.** & López García, Á. (2023). Study of the performance and scalability of federated learning for medical imaging with intermittent clients. *Neurocomputing*, 518, 142-154.
https://doi.org/10.1016/j.neucom.2022.11.011.

2022 **Sáinz-Pardo Díaz, J.** & López García, Á. (2022). A Python library to check the level of anonymity of a dataset. *Scientific Data*, 9(1), 785.
https://doi.org/10.1038/s41597-022-01894-2.

Otros artículos publicados en el transcurso de esta tesis a los que he contribuido de forma significativa y que están relacionados con aprendizaje automático (aplicaciones prácticas y monitorización de los modelos) y el análisis y procesamiento de datos, son los siguientes:

2025 Berberi, L., Kozlov, V., Nguyen, G., **Sáinz-Pardo Díaz, J.**, Calatrava, A., Moltó, G., Tran, V. & López García, Á. (2025). Machine learning operations landscape: platforms and tools. *Artificial Intelligence Review* 58, 167.
https://doi.org/10.1007/s10462-025-11164-3.

2023 Heredia Cacha, I., **Sáinz-Pardo Díaz, J.**, Castrillo, M., & López García, Á. (2023). Forecasting COVID-19 spreading through an ensemble of classical and machine learning models: Spain's case study. *Scientific Reports* 13, 6750.
https://doi.org/10.1038/s41598-023-33795-8.

Artículos que se encuentran actualmente bajo revisión (preprints):

2025 **Sáinz-Pardo Díaz, J.** & López García, Á. (2025). Enhancing the Convergence of Federated Learning Aggregation Strategies with Limited Data. arXiv preprint arXiv:2501.15949.
https://arxiv.org/abs/2502.01352.
Aceptado para su presentación y publicación en los actas de la "3rd IEEE International Conference on Federated Learning Technologies and Applications (FLTA25)".

2025 **Sáinz-Pardo Díaz, J.**, Athanasiou, A., Jung, K., Palamidessi, C., & López García, Á. (2025). Metric Privacy in Federated Learning for Medical Imaging: Improving Convergence and Preventing Client Inference Attacks. arXiv preprint

arXiv:2502.01352.
https://arxiv.org/abs/2501.15949.

Artículos publicados en actas de conferencias

En cuanto a la presentación y publicación de trabajos en congresos internacionales revisados por pares destacan los siguientes:

2024 **Sáinz-Pardo Díaz, J.**, Heredia Canales, A. Heredia Cachá, I., Tran, V., Nguyen, G., Alibabaei, K., Obregón Ruiz, M., Rebolledo Ruiz, S., López García, Á. (2024). Making Federated Learning Accessible to Scientists: The AI4EOSC Approach. *In Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '24)*. Association for Computing Machinery, New York, NY, USA, 253–264.
https://doi.org/10.1145/3658664.3659642.

2023 **Sáinz-Pardo Díaz, J.** & López García, Á. (2023). Comparison of machine learning models applied on anonymized data with different techniques. *IEEE International Conference on Cyber Security and Resilience (CSR), Venice, Italy, 2023,* pp. 618-623.
https://doi.org/10.1109/CSR57506.2023.10224917.

Asistencia a congresos, conferencias y charlas invitadas

A continuación se detallan los principales congresos, conferencias, seminarios o reuniones nacionales e internacionales a los que he asistido y contribuido:

2025 **Sáinz-Pardo Díaz, J.** (2025). Evento: NeuroAI, desde las neuronas y conectomas a los engramas. Curso de verano de la Universidad Internacional Menéndez Pelayo (UIMP). Charla: *Aprendizaje federado*. Santander, España. 25-29 de Agosto de 2025.

2025 **Sáinz-Pardo Díaz, J.** (2025). Evento: Accelerating Research with AI4EOSC: Real Use Cases Exploiting the Platform. Charla: *Introducing the AI4EOSC platform*. Online, 9 de Mayo de 2025.

Agenda del evento: https://indico.ifca.es/event/3441/.

2025 **Sáinz-Pardo Díaz, J.** (2025). Evento: Exploring AI4EOSC: AI and LLMs from Theory to Practice. Charla: *Why AI4EOSC? Take advantage of the platform*. Online, 7 de Marzo de 2025.

Agenda del evento: https://indico.ifca.es/event/3390/.

2025 **Sáinz-Pardo Díaz, J.** (2025). Evento: EOSC SIESTA All Hands Meeting. Charla: *Data privacy tools in EOSC SIESTA, anjana, pyCANON and differential privacy*. Instituto de Física de Cantabria (IFCA), Santander, España, 4-5 de Febrero de 2025.

Agenda del evento: https://indico.ifca.es/event/3384/.

2025 **Sáinz-Pardo Díaz, J.**, Aguilar Gómez , F. & Lloret Iglesias, L. (2025). Evento: Artificial Intelligence and Software Engineering Winter School. Workshop: *the eyes of AI. Topic: Federated learning applications*. University of Doha for Science and Technology (UDST), Doha, Qatar, 19-22 de Enero de 2025.

Página web del evento: https://www.udst.edu.qa/AiSeschool.

2024 **Sáinz-Pardo Díaz, J.** (2024). Evento: Workshop conjunto de la PTI Ciencia Digital y la PTI Horizonte verde. Charla: *AI and development of AI-based solutions*. Online, 17 de Septiembre de 2024.

Página web del evento: https://pti-cienciadigital.csic.es/evento/ jornada-pti-horizonteverde-cienciadigital/.

2024 **Sáinz-Pardo Díaz, J.** (2024). Evento: Los datos en investigación: retos y oportunidades. Curso de verano de la Universidad Internacional Menéndez Pelayo (UIMP). Charla: *Aprendizaje distribuido: ventajas y riesgos*. Sesión práctica: *Demostración sobre aprendizaje federado*. Santander, España. 26-28 de Agosto de 2024.

Programa del evento: https://wapps001.uimp.es/uxxiconsultas/ficheros/5/6842965SI.pdf.

2024 Asistencia a la conferencia: 12th ACM Workshop on Information Hiding and Multimedia Security. Presentación del artículo: *"Making Federated Learning Accessible to Scientists: The AI4EOSC Approach"*. 24-26 de Junio de 2024. Baiona, España.

DOI del artículo: https://doi.org/10.1145/3658664.3659642.

2024 **Sáinz-Pardo Díaz, J.** & Heredia Cachá, I. (2024). Evento: Zero-code tools & BMZ Community partners workshop. Charla: *Advanced AI for scientists: the AI4EOSC platform approach*. Madrid, España. 10-11 de Junio de 2024.

Página web del evento: https://ai4life.eurobioimaging.eu/event/workshop-hackathon-uploathon-madrid/.

2024 **Sáinz-Pardo Díaz, J.** (2024). Evento: AI4EOSC webinars (3): Introduction to Federated Learning. Charla: *Demo: Federated Learning in AI4EOSC*. Online, 22 de Abril de 2024.

Agenda del evento: https://indico.ifca.es/event/3134/

2024 **Sáinz-Pardo Díaz, J.** (2024). Evento: Flower AI Summit 2024. Talk: *Federating AI in the European Open Science Cloud*. Londres, Reino Unido. 14-15 de Marzo de 2024.

Página web del evento: https://flower.ai/conf/flower-ai-summit-2024/

2024 López García, Á. & **Sáinz-Pardo Díaz, J.** (2024). Evento: Flower monthly (January 2024). Charla: *Federated Learning in the European Open Science Cloud*. Online, 3 de Enero de 2024.

Video de la charla: https://www.youtube.com/watch?v=4OC2y0kCvPY.

2023 **Sáinz-Pardo Díaz, J.** (2023). Evento: AI4EOSC user's workshop. Charla: *Federated Learning with Flower*. Institute of Informatics of the Slovak Academy of Sciences (IISAS), Bratislava, Eslovaquia. 15-16 de Noviembre de 2023.

Agenda del evento: https://indico.scc.kit.edu/event/3845.

2023 Asistencia a la conferencia: IEEE International Conference on Cyber Security and Resilience 2023. Presentación del artículo: *"Comparison of machine learning models applied on anonymized data with different techniques"*. Venecia, Italia. 31 de Julio - 2 de Agosto de 2023.

DOI del artículo: https://doi.org/10.1109/CSR57506.2023.10224917.

2023 Asistencia a la escuela de verano: AIHUB CSIC Summer School 2023. Presentación del póster: *"Application of federated learning to medical imaging scenarios"* [7]. Barcelona, España. 4-8 de Julio de 2023.

2023 Asistencia a las X Jornadas Doctorales y V Jornadas de Divulgación del Grupo de Universidades del G9, Universidad de Oviedo. Presentación del póster: *"Privacy preserving techniques for data science"* [8]. Oviedo, España. 31 de Mayo - 2 de Junio de 2023.

Trabajo futuro

El campo de la ciencia de datos, especialmente potenciado por el desarrollo de soluciones basadas en modelos de inteligencia artificial, es un área en constante crecimiento con aplicaciones en múltiples sectores. Esto, acompañado por la creciente

generación de datos e información, a menudo de carácter sensible por estar asociados a personas identificadas o identificables, o por contener datos confidenciales de entidades o instituciones, pone de manifiesto la importancia de preservar la privacidad en estos entornos. Garantizar la protección de la información a lo largo de todo el ciclo de vida de los datos, desde su adquisición hasta la implantación, despliegue y puesta en producción de modelos, resulta cada vez más esencial. El futuro de la inteligencia artificial y la ciencia de datos también está estrechamente ligado a la adecuación al marco normativo. Así, es esencial que el progreso tecnológico y el desarrollo de soluciones avanzadas vayan de la mano de la evolución de la legislación y de las cuestiones éticas.

Además, la disponibilidad de procesadores cada vez más potentes y asequibles fomenta el procesamiento en el propio dispositivo, lo que impulsa el desarrollo de modelos de inteligencia artificial de forma distribuida. La aplicación de arquitecturas de aprendizaje federado y otros métodos descentralizados sin dependencia de un servidor central se vuelve clave, ampliando el alcance de las posibles aplicaciones relativas a IoT y dispositivos en el borde. Asimismo, deben explorarse medidas de privacidad complementarias como la privacidad diferencial, la privacidad diferencial métrica, el cifrado homomórfico y su variante multiclave, cuya aplicabilidad dependerá de condiciones computacionales adecuadas. Desde una perspectiva más general, la privacidad en escenarios de procesado y análisis de datos en dichos dispositivos es cada vez más relevante, garantizando la seguridad en sistemas distribuidos que procesan datos localmente en lugar de en servidores centralizados.

Respecto al procesado y análisis de datos, el trabajo futuro pasa por continuar con el desarrollo y la actualización de las dos librerías presentadas y desarrolladas en el marco de esta tesis, *pyCANON* y *anjana*, añadiendo funcionalidades adicionales para analizar la utilidad de los datos así como la resistencia a distintos ataques. También es importante continuar explorando la escalabilidad y el impacto de integrar privacidad diferencial en distintos escenarios, en concreto en lo que respecta a aprendizaje federado y su uso combinado con otras técnicas de mejora de la privacidad (PETs), como el cifrado homomórfico. Además, en lo que respecta a privacidad en el intercambio de datos entre organizaciones, se requiere la integración de tecnologías de preservación de la privacidad que permitan el intercambio de dicha información sin comprometer la confidencialidad, incluyendo técnicas avanzadas basadas en computación segura de múltiples partes (SMPC).

En cuanto a la adquisición de los datos, la generación de datos sintéticos se posiciona como una solución clave para el entrenamiento de modelos sin los riesgos asociados a la normativa de protección de datos, garantizando al mismo tiempo

una adecuada representatividad estadística. Además, es clave seguir investigando diferentes tipos de ataques a la privacidad, como ataques de inferencia, pertenencia, inferencia e inversión de modelos, y desarrollar estrategias de mitigación dentro de arquitecturas distribuidas, especialmente en lo que respecta al aprendizaje federado así como otras soluciones descentralizadas.

En la misma línea, el trabajo futuro (ya en curso) sobre aprendizaje federado es extenso. Además de los tecnologías de mejora de la privacidad y la prevención de ataques, la investigación futura se centra en el desarrollo de funciones de agregación para tratar con datos heterogéneos, el diseño de medidas para la detección, monitorización y prevención de deriva en los datos y/o clientes, y la optimización de herramientas criptográficas como el cifrado homomórfico para reducir los costes computacionales manteniendo la integridad. Todo este desarrollo también es relevante para las arquitecturas descentralizadas sin dependencia de un servidor central, donde otras cuestiones de implementación práctica surgen de forma evidente al eliminar dicha dependencia.

La privacidad en entornos de inteligencia artificial generativa y modelos de lenguaje a gran escala (LLMs) es otro reto emergente. El entrenamiento de estos modelos requiere enormes recursos computacionales, y el aprendizaje federado surge como una alternativa adecuada para optimizar su eficiencia. En concreto, aunque esto se plantea como trabajo futuro, ya se ha avanzado en la aplicación de aprendizaje federado y LLMs especializados, especialmente en el ámbito médico y la investigación en el contexto del EOSC. El desarrollo de este tipo de modelos es un área clave de la investigación científica en cuanto a inteligencia artificial y potenciar su desarrollo con recursos distribuidos (y teniendo en cuenta aspectos de privacidad asociados) será clave para ahorrar recursos computacionales y permitir su aprovechamiento para mejorar el acceso a resultados de investigación, por ejemplo mediante el entrenamiento e integración de agentes especializados.

Otro aspecto clave es el desarrollo de algoritmos eficientes para el intercambio seguro de datos, así como la investigación en relación con entornos de ejecución seguros. En la misma línea, desarrollar métodos criptográficos resistentes a los retos derivados de de la computación cuántica es un reto actual que no puede perderse de vista al trabajar en entornos de ciencia de datos en que se manejan datos sensibles.

Las herramientas y metodologías presentadas a lo largo de esta tesis deben estar disponibles (y de hecho lo están) tanto para el público en general como, en particular, para la comunidad investigadora, ya que cada día los investigadores trabajan con volúmenes de datos cada vez mayores. El método científico abarca desde la observación sistemática y la recogida de datos hasta la experimentación, el análisis, la

formulación de hipótesis y su posterior revisión o modificación. Dado que los datos son un componente esencial del proceso científico, es esencial tener en cuenta los retos que surgen al tratar con información sensible, ya sea por su naturaleza o por estar protegida en el contexto de la investigación. En estos casos, las cuestiones de privacidad adquieren un papel central, y las soluciones propuestas en esta tesis y en relación con el trabajo futuro son de especial relevancia. Esto adquiere aún más importancia cuando se pretende explotar el potencial de los datos mediante su análisis, así como en el desarrollo y despliegue de modelos de inteligencia artificial. Por ello es crucial que todas las herramientas presentadas en esta tesis sean accesibles a la comunidad investigadora, para que puedan aprovechar su potencial y llevar a cabo sus investigaciones con especial atención a la privacidad y seguridad de la información manejada.

Part I: Data Revolution

CHAPTER 1

Introduction and Motivation

Las puertas abiertas
dan siempre a una sima
mucho más profunda
si la casa es vieja.
...
¡Qué trabajo nos cuesta
traspasar los umbrales
de todas las puertas!

Federico García Lorca,
Puerta abierta

Contents

1.1 Big Data and Data Science 7
1.1.1 Big Data 8
1.1.2 Data Science and Artificial Intelligence 9
1.2 Ethical, Social and Legal Considerations 14
1.2.1 Gender Perspective: Bias in Data 15
1.2.2 Ethical and Social Issues Related to Data 17
1.2.3 General Data Protection Regulation 18
1.2.4 Data Governance Act 20
1.2.5 Data Act 21
1.2.6 European Union Artificial Intelligence Act 22
1.3 Open Science 24
1.3.1 Open Data 25
1.3.2 Open Access 27
1.3.3 Open Source and Software 28
1.3.4 European Open Science Cloud 30
1.4 Motivation 30
1.4.1 Why Data Revolution? 31
1.4.2 Thesis Road-Map: from Data Privacy to Privacy Preserving Machine Learning 32

Abstract

This first chapter presents the basic background that lays the foundation for this work. Specifically, the concerns related to data privacy in the era of big data and data science are discussed, exposing the basis of these two concepts. Ethical and social issues, different precedents and applicable standards on legal aspects related to data protection are also presented, with a particular focus on the EU regulation. Furthermore, the ideas of open science, open data, open source and software are stated, together with a brief introduction to the European Open Science Cloud. Finally, the motivation for this research work is presented, explaining why we can talk about a data revolution and some privacy concerns in data science processes.

It may be redundant to highlight the benefits of *artificial intelligence* (AI), *big data* and *data science* techniques in today's society, but it is necessary to lay the groundwork that motivates this work. In order to do so, let us expose the fundamentals of *big data* and *data science*, both concerning ethical and legal aspects related, in addition to the foundations of open science and open data.

Huge amounts of data[1] are created, stored and collected on a daily basis. We can think of the large amount of data that are generated due to the use of smart devices such as smartphones, smartwatches or voice assistants, but also due to the Internet searches and the use of Internet of Things (IoT) devices. And not only this, but data are collected in every sector, from the medical field to finance, education, etc. However, in addition to these, there is other information about us, the users, that accompany us almost from the very beginning of our lives: our name, identification number or date of birth are just simple examples, in addition, of course, to our biometric data: fingerprint, voice and facial patterns.

In Figure 1.1 the knowledge pyramid is presented [1]. This scheme establishes that without *data* there is no *information*, without information there is no *knowledge* and without knowledge there is no *wisdom*. Then, it can be seen that *data* constitute the base of the pyramid, hence its great importance (and because of its value, it is essential to pay attention to its privacy). Moreover, it can be observed that the path is composed of different steps, the first one being to process and organize the data.

[1] The term *data* derives from Latin as the plural form of the word *datum*. Although in common usage in modern English the term *data* is often used as a singular mass noun (e.g. *the data is*), in academic and scientific contexts it is generally treated in the plural form (e.g. *the data are*). Throughout this thesis we will try to maintain linguistic coherence by using the plural form and concordance of this term.

The initial data can be of several types: images, sounds, tables, etc. In particular, we can distinguish four essential types of data according to how they are stored:

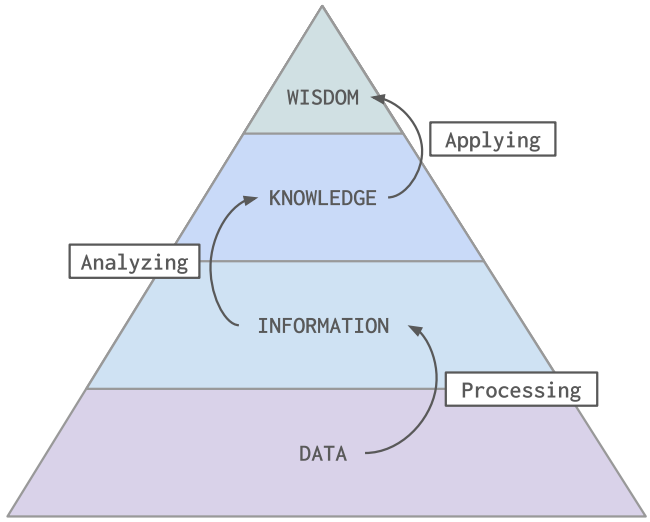


Figure 1.1: Knowledge Pyramid, adapted from [1].

- **Non-structured:** These are data that do not follow a predefined structure, schema or model. They are not (or even cannot be) displayed in tables formed by rows and columns. The most intuitive examples are photos, videos, text messages, posts on social networks or PDFs, and therefore form the vast majority of online data. In general, they are more difficult to store and manage than other types of data such as those discussed below.

- **Quasi-structured:** In this case, these data consist of textual content but stored in imprecise and irregular formats. An example could be the registration data to a site or the search history of a user.

- **Semi-structured:** This kind of data have a certain level of structure, hierarchy and organization, i.e. they have organizational properties that make their analysis easier than that of unstructured or quasi-structured data. They contain metadata with information on how they are stored and organized. An example is data in XML format or other markup languages.

- **Structured:** They are those data that can be stored in tables consisting of rows and columns, i.e. they can be stored in relational databases (e.g. SQL) and follow predefined standards. They require less storage capacity than

unstructured data, and are easier to manage and protect. Examples include databases such as excel tables or CSV files.

Moreover, as far as data storage solutions are concerned, we can mainly distinguish between data lakes and data warehouses.

On the one hand, data lakes are particularly used in the case of non-structured quasi-structured or semi-structured data (images, videos, logs). On the other hand, data warehouses are particularly suited to store structured data, requiring a predefined schema before storage (while data lakes do not require the definition of such a structure, but follow the schema on-reading). The latter have an optimized search process which makes them a good fit for fast queries and searches, while data lakes are especially useful when large volumes of data are available from different sources.

Once the data have been processed and organized, we can consider that they have been transformed into valuable information from which knowledge can be extracted. In fact, this knowledge is reached through analysis, conceptualization, synthesis or visualization among other steps. Finally, applying knowledge, we arrive to wisdom. It is important to highlight that seemingly insignificant pieces of information, combined together, can lead to knowledge which reveal important information (in some cases related to individuals). Here again, privacy protection comes into play, not only in the first phase of work on the data, but also during analysis and knowledge extraction. We will further elaborate on this in the course of this thesis.

1.1 Big Data and Data Science

As it is widely accepted, the data deluge has led to a data revolution, with an increasing number of data-driven problems and approaches being developed, providing insights and solutions to existing challenges, outperforming more traditional methods in some cases. The disruptive potential (in a positive sense) of these techniques is high.

In April 2025, a web search for the term *"big data"* produced 6.720.000.000 results, and for the term *"data science"*, 3.600.000.000 results were found on the Internet. Both concepts are widely cited in the media and in different spheres: social, economic, research and education among others. Therefore, in this section we will focus on briefly outlining some characteristics of both big data and data science, as well as the relations and differences between these two terms.

1.1.1 Big Data

The concept of *big data* was first spread and put into concept in the mid-1990s by John Mashey, in order to talk about the need of handling and analysis massive datasets [1].

The term big data is usually presented through a series of characteristics, among which the 5V's are particularly relevant: *volume*, *variety*, *velocity*, *value* and *variability*. In short, these characteristics can be defined as follows (see [1] and [9]):

- **Volume:** extremely high amount of data, even petabytes, that need to be stored, processed and analyzed in order to extract information and knowledge. This is the most associated (and most significant) characteristic of big data, as can be directly deduced from the term itself.

- **Variety:** refers to the great diversity of data according to their type. For example, data can be structured, semi-structured or unstructured, as well as it can came from different areas: business, health, banking, climate data, social data, etc. Fortunately, this is not a particular problem due to advances in, among others, distributed computing, databases, and techniques for information processing and analysis in general.

- **Velocity:** refers to the rate at which data are generated, processed, analyzed and visualized. It is important to note that in many cases data are taken on the fly, for example from observations made continuously, as may be the case with weather stations, sensors, and other types of monitored devices.

- **Value:** In terms of value, here again it is important to emphasize the power of transforming data into information and information into knowledge. In particular, this knowledge can be used for decision making or for future predictions, especially when there is a large volume of data, as is the case with big data.

- **Variability:** refers to the changes that can constantly occur in the data. It is to consider that data can change, for example when dealing with real-time sensor data. It can also be seen as the dispersion that may be present in the data. In addition, in the context of big data, variability also refers to the number of inconsistencies in data.

In addition to these five, we can consider other features in order to define the concept of big data, such as:

- **Exhaustibility:** It should be noted that in most disciplines the total number of data is limited, both for analysis and storage reasons. However, big data has the characteristic of being exhaustive precisely because it concerns to huge amounts of data. This may be the case of data from high-energy physics or climate data. For example, in the latter case, exhaustibility is achieved by having a large volume of data captured in different ways and with a large historical record.

- **Scalability:** refers to possible changes in the data in terms of data volume, either by considering new users, by increasing the number of sensors collecting data, or simply by adding new sources.

Although there is not a standard definition for the term of big data, there are many popular definitions, given by different authors. The most relevant and accepted ones are collected in Table 1 of [9] and also in Table 1 of [10]. In order to conclude this brief introduction to the concept of *big data*, our own formal definition of the term is given:

Definition 1.1.1. Big data. It refers to extremely high-volume data sets that require complex computational techniques to extract information, trends and patterns from them.

1.1.2 Data Science and Artificial Intelligence

Interestingly, in many areas the terms *big data* and *data science* are used interchangeably, but in reality they are different concepts. In fact, the concept of *data science* appeared around the 1970s (see, for instance, the book "Exploratory data analysis" by J. W. Tukey from 1977 [11]), to refer to the methods and techniques used for data processing and analysis, but it was not until well into the 21st century (specifically, in 2001), when it was separated from the concept of big data as a stand-alone discipline. In addition, while when referring to big data, as discussed above, it covers those highly voluminous sets of data which are difficult to process and store using traditional techniques and databases, data science is in charge of collecting, curating, analyzing these data sets, extracting information and driving decision making. A formal definition of *data science* in given in the following:

Definition 1.1.2. Data science. It is an interdisciplinary field which combines statistics, data analysis, scientific computing, mathematical methods, algorithms, and data engineering tools in order to extract insights and knowledge from structured and unstructured data.

Because data are ubiquitous, data science is broadly applicable to a large variety of fields. A wide range of applications may pose technical, ethical, legal or privacy issues (e.g. medical, financial, telecommunications, IoT, etc), and currently there are more and more concerns in the way data are being handled according to these aspects.

In general, the life cycle process of a data science project including the development of data based models, can be summarized as given in the simplified schema of Figure 1.2 and explained below.

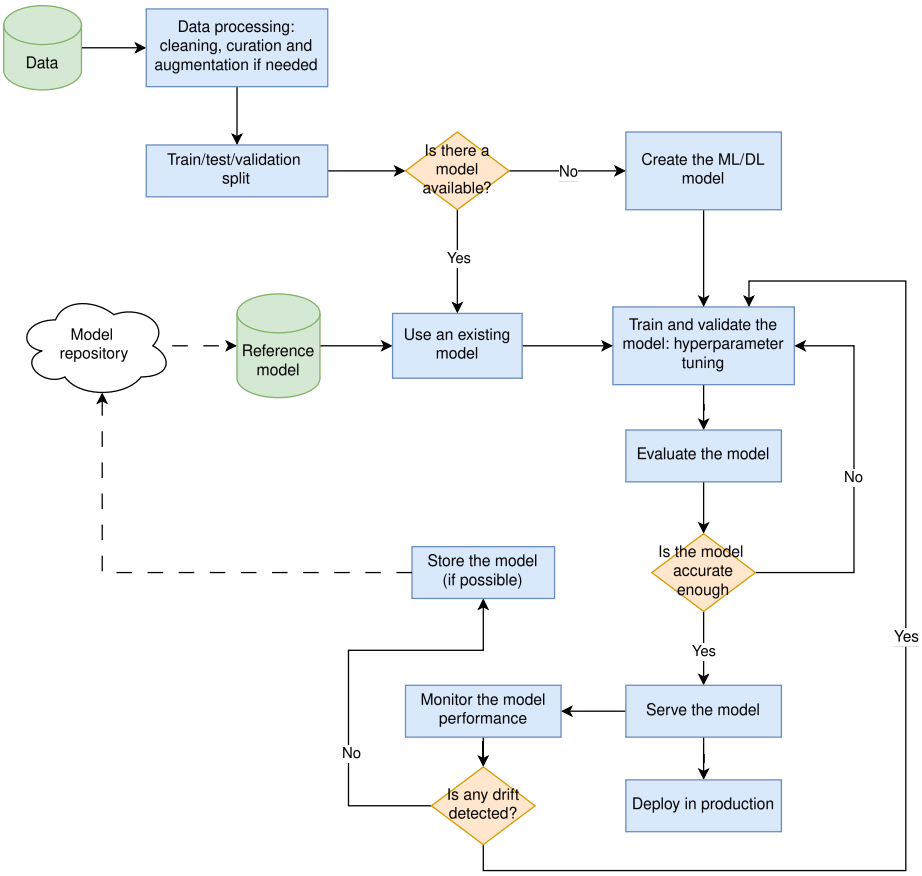


Figure 1.2: Diagram of the simplified life cycle of a data science project.

- The first steps concerns the data: obtaining, collecting or extracting.
- Once the data are available, perform data processing including data cleaning, curation, and data augmentation if need (in case of small data settings).

Other processing methods can be applied such as data scaling, de-noising, normalization, etc.

- Split into train, validation and test for applying the predictive data-based models.
- Check if there is any model available to be used (e.g. in a model repository).
- If a model is available, take it as a reference for use. Otherwise the ML/DL model is created from scratch.
- Train the model using the train set, check the performance in the validation set and optimize the hyperparameters.
- Evaluate the model using the test set and different metrics. Graphical visualization and analysis of the results can be performed in this step.
- If the model is not sufficiently accurate, it shall be retrained and a fine tuning of the hyperparameters of the model should be performed. Otherwise it can be served and deployed in production.
- When serving the model, it has to be monitored using MLOps techniques (see [12]) and analyze concept and data drift. If drift is detected, again the model shall be re-trained and fine-tuned. Otherwise, the model can be stored in a model repository (if possible).

In view of the above, it should be noted that although in some cases the most appealing part of a data science project can be identified with the application of algorithms for data analysis, all the steps mentioned above are equally important. Without data collection, we have no asset to analyze and draw conclusions from. Moreover, without its curation and cleaning, such analysis would be meaningless, or much more complex. Visualization is key to understanding the results obtained and facilitating decision making. Finally, monitoring the model to analyze its adaptability to new data, as well as the process of serving and publishing the model and performing inference, are essential steps.

In terms of analysis, statistical techniques can be used, such as hypothesis testing, Bayesian inference and others. However, in this thesis we are going to focus on two branches or subgroups framed in the concept of *artificial intelligence (AI)*, namely *machine learning* (ML) and *deep learning* (DL) models.

Multiple definitions have been given for the concept of artificial intelligence (AI). Specifically, as collected by S. Russell and P. Norvig in their book *"Artificial*

Intelligence: a Modern Approach" [13], there are different definitions for this concept that can be classified into four main categories: (1) systems that think like humans, (2) systems that think rationally, (3) systems that act like humans and (4) systems that act rationally. Regarding the first one, we can highlight the Turing test, proposed by Alan Turing in 1950 with the idea of designing a system to obtain an operational definition of intelligence. In relation to systems that think like humans and in order to pass the Turing test, it will be necessary for the system to have the capacity for natural language processing, knowledge representation, automatic reasoning, machine learning to adapt to new circumstances and to detect and extrapolate patterns, as well as computer vision and robotics to be able to manipulate and interact with objects and the environment.

In relation to the other three approaches, with respect to thinking rationally, the development of logic stands out with the idea of being able to solve any problem in logical notation and thus be able to build intelligent systems. Regarding thinking like humans, cognitive sciences aim to combine computational models and experimental techniques of psychology to build theories that allow us to understand how the human mind works and to replicate it through computational models. Finally, regarding acting rationally, we can highlight the use of computational agents, as well as rational agents being those that act to achieve the best possible outcome in a given situation, interacting with their environment provided with sensors and actuators.

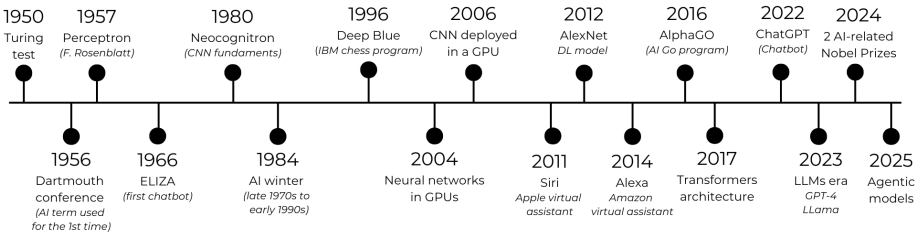


Figure 1.3: Summarized AI advances timeline.

We can state that the term artificial intelligence was originally introduced by J. McCarthy in 1956 during the Dartmouth conference. AI became established as a science in 1987, following a period known as the *"AI winter"*) that covered the late 1970s and early 1990s. Since then, AI has continued to advance, with a particular focus on machine learning and deep learning (ML/DL), as will be discussed in the following. This progress has been driven by the availability of hardware suitable for successful application, as well as recent advances in convolutional neural networks

(CNNs), transformers and Large Language Models (LLMs), among others.

On the one hand, with regard to ML, it is a branch of AI that seeks to provide machines or computers with learning capabilities. On the other hand, DL similarly seeks to provide learning capacity by replicating human thinking. For this purpose, artificial neural networks (ANNs) and convolutional neural networks (CNNs), among other architectures, are widely used. Applications range from image classification to voice recognition or making predictions.

Two types of problems can be classically addressed when referring to ML algorithms: classification and regression. Classification is probably the most common use of ML, and can be used to solve a wide variety of problems, such as document or image classification, spam filtering, etc. On the other hand, while in classification the response variable is discrete, in regression it is continuous. Some common problems where regression is applied are, for example, temperature prediction, trend analysis or time series forecasting. In addition, ML algorithms can be essentially divided into supervised and unsupervised learning [14]:

- **Supervised learning:** an algorithm must generalize knowledge from the available data with their respective labels, so that it can be used to make predictions when new unlabeled data are available. Some models that stand out are, among others: k-Nearest Neighbors (kNN), Support Vector Machines (SVM), Kernel Ridge Regression (KRR), Decision Trees, Random Forest, Gradient Boosting, etc.

- **Unsupervised learning:** attempts to group data into clusters using automated methods or algorithms on data that have not been categorized. In this case, some methods which can be highlighted are: k-means, Gaussian Mixture Models (GMM), Singular Value Decomposition (SVD), or Autoencoders (which are a kind or ANN).

Concerning DL, this concept encompasses algorithms involving the use of multilayer (deep) artificial neural networks (ANN). Specifically, in relation to ANN, the goal of the perceptron or artificial neuron is to mathematically simulate the behavior of a biological neuron. This means that it follows an analogous scheme: a stimulus/input is received, which is processed (an activation function is used for this purpose), and it produces a response/output. In neural networks, the output of some neurons will be the input of others (e.g. dense layers). In other words, ANN are composed of multiple hidden layers. Depending on the number of hidden layers, it will be a simple or a deep neural network.

The main difference between a dense network and a convolutional neural network (CNN) is that while dense networks learn global patterns (e.g. in the case of images, they use all the pixels of the image), CNNs learn local patterns (in the case of images, small windows or filters are used). Therefore, CNNs are well suited for image processing, among other tasks.

As have been already exposed, while *big data* deals with extracting useful information found in huge data sources, *data science* uses intelligent models that learn from themselves using the data, such as ML or DL models, together with statistical methods, in order to train different models based on the data available and thus extract knowledge, perform inference on new data and even allow to make informed decisions.

1.2 Ethical, Social and Legal Considerations

Big data and data science techniques are commonly used to predict needs or extract patterns, among others. It helps us and is present in our daily lives, for example by improving early disease detection techniques thanks to the analysis of medical data, or by detecting possible attacks or frauds in our bank accounts. It also allows us, through personalized recommendations, to discover new books or songs thanks to the knowledge of our tastes. Nevertheless, the great value of information also implies the imperative need to guarantee its security and privacy, as well as to analyze the ethical aspects of its processing. What information can we allow to be used as a basis for prediction? Can we classify people according to algorithm predictions based on their personal information: gender, religious beliefs, race, etc? Would this be perpetuating bias? A simple example can be found in gender bias: the data with which we are feeding our algorithms may be biased by the past, where women were underrepresented for example in terms of higher education or high responsibility jobs. It is therefore important that data, like society, can reflect the social progress.

Individual personal information is very valuable in the field of data science, as it allows to extract more precise insights about people's behavior, preferences and needs. In this line, as everything we consider valuable, it must be protected accordingly, paying special attention to the collection, access and treatment of data before, during and after data analysis and the application of learning models. Both big data and data science can open a new paradigm concerning research, innovation and decision making in many fields, as well as new opportunities, but it is important to always take into account the social issues that this may involve. In

this sense, in this section we will review the perspective of some aspects related to data bias and its impact from the point of view of the gender perspective. Then, we will review some ethical aspects derived from the social impact of this technology, to conclude with an overview of the legal aspects by reviewing the legislative regulations in terms of data protection and data management as well as in relation to the development of AI methods, tools and solutions in the context of the European Union.

1.2.1 Gender Perspective: Bias in Data

As exposed when presenting the characteristics of big data, large volumes of data are generated every minute, many of them in real time, and can cover a wide variety of fields. Also, data can be collected or generated in different ways, either by new technologies such as smartphones, or from data generated by the citizens themselves (texts, social networks, news in the media, etc). Therefore, it is important to keep in mind that the data used for the different analyses may contain gender or racial biases, and to try to eliminate (or at least reduce) these biases so as not to perpetuate them in future analyses or applications of such data.

In the following, to reflect this problem, we present the results of the analysis conducted in [15] on gender bias in big data. In that work, a corpus extracted from the 2006 Spanish Wikipedia, courtesy of the Polytechnic University of Catalonia was taken, and a neural network that generated a collection of embeddings representing each of the vocabulary words in the corpus, was used. This neural network ordered the words in such a way that the distances between them indicate the proximity of their use in the language with which the network was trained. The results would be something similar to *"one is to two as three is to..."*. Listed below are three examples of sentences obtained:

"Man is to faithfulness as woman is to obedience."
"Man is to work as woman is to mother."
"Man is to intelligence as woman is to show off."

This type of phrase reveals a clear bias in the corpus used, re-validating that these biases that are often present in common language can be reflected in the data used for several analyses, in the most unsuspected way. Specifically, as the authors of the study themselves state, in the corpus used "there is a great omission and familiarization of women in the published articles", while "men usually appear as substantive entities in their individuality" (see [15]).

This is a bias due to the language, which shows us that it is necessary to be critical of the data used, review it, process it and work with it to prevent this type of problem from occurring. That is because data depends on the ideas of the people who generated it, and also on the social-historical context. However, this is a continuous work as a society, and shows once more the need to adapt our language and our actions to the social reality.

Another example of gender bias in data could be found in the online translators. In Spanish, when we refer to professions, in most cases these are associated with their gender, e.g. scientist —in Spanish, *el científico* (M), *la científica* (F))— doctor —in Spanish, *el doctor* (M), *la doctora* (F)—, teacher —in Spanish, *el profesor* (M), *la profesora* (F)—, etc. We would expect that if we translate these professions from English to Spanish, we would get them always in masculine or always in feminine gender, but however, we can see that this varies according to the adjective accompanying the noun. Some examples are shown in Table 1.1.

Phrase introduced in English	**Literal translation obtained in Spanish**
The scientist is pretty	La científica es guapa
The scientist is clever	El científico es listo
The doctor is pretty	La doctora es guapa
The doctor is clever	El doctor es listo
The teacher is pretty	La profesora es guapa
The teacher is clever	El profesor es listo

Table 1.1: Example of gender bias when translating from English to Spanish using AI-based translators.

We can observe how in the three cases analyzed in Table 1.1, despite the fact that the six sentences were introduced in a row, whenever the profession was followed by the adjective *pretty*, it was assigned with a woman, while if the adjective was *clever*, it was assigned with a man. This reflects a clear gender bias by assigning certain attributes to women or men according to their social implications. The same was true if we translated these sentences from English to French (except for doctor, which was always translated in the masculine gender). Again this is also a collective task for society, and in particular it is an extremely important issue: remember that data are the stepping stone to knowledge and wisdom.

1.2.2 Ethical and Social Issues Related to Data

We have previously seen clearly reflected the social aspect of gender bias related to the data. Now let us discuss other ethical aspects that arise from data processing and analysis.

When we talk about *ethics*, we are talking about a philosophical discipline, discussed since ancient Greece. Specifically, ethics comes from the Greek word *ethos*: custom. Both as a society as a whole and as individuals, we continuously address questions concerning what is right and what is wrong, questions about justice, fairness, honesty, rights, etc. Specifically, ethics deals with the foundation of the moral standards that govern our actions.

It is therefore extremely important to be aware of the potential risks of big data and data science methods when used without ethical considerations, from the implicit bias discussed above, to the social impact of automation and the protection of sensitive data. All of this is driven by the increasing digitization of human activity in almost every conceivable field, as well as a growing dependence on technology, which blurs the ability to discern the traditional concepts of privacy, fairness and trust.

The interaction of individuals in social communities makes it possible to reevaluate attitudes, ideologies or very specific thoughts, being a highly powerful weapon for those who possess such information. In fact, it is important to be aware of the dangers that arise, for example, from the application of data science algorithms to create personalized targeted advertising, such as those made by the algorithms of social media. This ability to personalize the stimuli that an individual perceives online is a powerful tool in our current society, specifically in the social sphere. In fact, feeding users according to what they usually watch or read isolates them in a bubble, always putting them information they agree with, something like "telling them what they want to hear", so that they keep consuming these media. This will result in a lack of critical thinking on the part of individuals, as they will be in their own bubble (which they may have not decided to enter), in which all the information will be biased beforehand by previous ideas, with hardly any possibility of including new ways of thinking. A clear example can be seen in political ideologies: someone who always reads positive news in relation to a certain ideology or political group, or negative news in relation to another, will always receive this type of news due to the personalization of the information (based, among others, on the use of data science and AI and data based methods), making it almost impossible to receive news that follows a contrary viewpoint. This can isolate users in a pervasive blind box that prevents them from seeing beyond and broadening their horizons of

thought. This trend is increasingly enhanced by the use of data-driven models that allows to predict what the users will want to see or read. In these contexts, it is worth asking whether the privacy of users is respected.

One can even state that technology is not neutral, because as we have already mentioned, its ethics begins with the ethics of those who program an algorithm and of those who create, collect and include the data with which the algorithm will learn. Indeed, decision making by these algorithms, which has a great responsibility, can be biased beforehand.

Finally, to conclude this section, we will briefly comment on some legal aspects related to data protection, which are essential to define a clear legal framework in an increasingly digitized and data-empowered world.

1.2.3 General Data Protection Regulation

Let us briefly focus on the legal issues related to and/or arising from the use of big data and data science. First, it is important to note that since the Universal Declaration of Human Rights (1948) [16], the right to privacy is included (article 12):

> *No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honor or reputation. Everyone has the right to the protection of the law against such interference or attacks.*

The protection of natural persons in relation to the processing of personal data is a fundamental right. Article 8(1) of the Charter of Fundamental Rights of the European Union [17] and Article 16(1) of the Treaty on the Functioning of the European Union (TFEU) [18] provide that everyone has the right to the protection of personal data concerning him or her.

Before briefly exposing the basis of the GDPR, we are going to comment its predecessor: Directive 95/46/EC. This European Union directive, adopted in 1995, was responsible for regulating the protection of natural persons with regard to the processing of personal data and the free movement of such data. However, this directive does not fully take into account how to protect the information in the era of big data and cloud applications.

The General Data Protection Regulation is not a directive, but a law of the European Union. It was approved in April 2016, but it came into force on May 25th, 2018 and obliges organizations to safeguard personal data and uphold the privacy rights of any individual on EU territory. The aim is to harmonize data protection

issues for all EU states, in addition to strengthening the rights of data subjects. Although it applies to all member states, they may introduce national provisions or clarify specific rules. In addition, all international companies providing services in the EU must comply with its rules. With the introduction of the GDPR, fines for data protection violations amount to up to 20 million euros, or 4% of the company's annual worldwide revenue, whichever is higher [19].

Finally, let us summarize the seven principles about personal data privacy exposed in the Article 5 of the GDPR (see [20, 21]):

- **Lawfulness, fairness and transparency:** The processing of personal data must be lawful and fair. This means that one shall not process the data in a way that is inappropriately, detrimental, unexpected or misleading to the individuals concerned. In addition, fairness implies that personal data should only be processed in ways that individuals would reasonably expect and not be used in ways that have unreasonable adverse effects on them. Furthermore, the way in which data are collected, used and processed must be transparent to the data subject. Transparent treatment is about being open, honest and clear with individuals from how and why their personal data are used.

- **Purpose limitation:** Data must be collected for specified, explicit and legitimate purposes and not further processed in a way incompatible with those stated purposes. It should be made clear at the beginning why personal data are collected and what it intends to do with it; comply with documentation obligations to specify its purposes; comply with transparency obligations to inform individuals about its purposes; and ensure that, if it intends to use or disclose personal data for any further or different purpose from the originally specified one, the new use is fair, lawful and transparent.

- **Data minimization:** Personal data must be adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed. In short, you should identify the minimal amount of personal data needed to fulfill the purpose for which it was collected. No more than that amount of information should be retained.

- **Accuracy:** Personal data must be accurate and must be kept up to date on a regular basis as necessary. In addition, all reasonable steps should be taken to ensure that personal data that is inaccurate should be deleted or rectified as soon as possible. The source and status of personal data must be clear.

- **Storage limitation:** it is necessary to ensure that personal data are deleted once they are no longer required to be stored, as this will reduce the risk of it becoming useless, inaccurate, excessive, irrelevant or obsolete. Therefore, even personal data are collected and used fairly and lawfully, it should not be stored longer than strictly necessary.
- **Integrity and confidentiality:** Appropriate security measures must be taken to protect the security of sensitive data. Data must be processed in a secure way including adequate protection against unauthorized or unlawful treatment and against unintentional loss, damage or corruption, using technical and/or suitable organizational measures.
- **Accountability:** Article 5 of the GDPR states that the data controller shall be responsible for, and shall be able to demonstrate compliance with the above exposed aspects.

Specifically, the general data protection regulation tells us that the principles of data protection must apply to all information relating to an identified or identifiable individual. In this sense, pseudonymized data should be considered as sensitive information about an identifiable natural person. In determining whether a natural person is identifiable, all the means that the controller or any other person can reasonably use to directly or indirectly identify that person should be taken into account. Similarly, in determining whether there is a reasonable likelihood that identification methods can be used to identify a natural person, all objective resources must be taken into account, both in terms of cost and time required for identification. In addition, both current technology and future technological developments must be taken into consideration. Consequently, if the information has been anonymized, data protection principles should not apply, as the information does not relate to an identified or identifiable person. Therefore, it is very important to note that the Regulation does not affect the processing of such anonymized information.

1.2.4 Data Governance Act

The Data Governance Act (DGA) [22] is an European Union regulation effective from June 23rd, 2022 and applicable from September 2023. The aim of the DGA is to enhance the protection derived from the GDPR in the field of data protection regarding data sharing (the GDPR providing the necessary safeguards in the context of personal data) with a focus on creating a framework to increase trust in the voluntary exchange of data for the benefit of businesses and citizens.

As stated in its Article 1, this regulation lays down:

(a) *conditions for the re-use, within the Union, of certain categories of data held by public sector bodies;*

(b) *a notification and supervisory framework for the provision of data intermediation services;*

(c) *a framework for voluntary registration of entities which collect and process data made available for altruistic purposes; and*

(d) *a framework for the establishment of a European Data Innovation Board.*

This regulation, which focuses on data sharing with the aim of contributing to building trust in international data flows, also states that a re-user in a third country must ensure the same level of protection with respect to the data in question as the level of protection guaranteed by the EU law, as well as the acceptance of the EU jurisdiction.

Furthermore, in its Article 2, paragraph (10), the DGA defines data sharing as "the provision of data by a data subject or a data holder to a data user for the purpose of the joint or individual use of such data, based on voluntary agreements or Union or national law, directly or through an intermediary, for example under open or commercial licenses subject to a fee or free of charge".

Moreover, Article 5 states the following on the re-use of data: "conditions for re-use shall be non-discriminatory, transparent, proportionate and objectively justified with regard to the categories of data and the purposes of re-use and the nature of the data for which re-use is allowed".

In summary, the DGA aims to regulate the re-use of public or protected data, both with regard to personal and non-personal data, with special safeguards to increase trust in the sharing and re-use of data in addition to the guarantees of the GDPR.

1.2.5 Data Act

The Data Act [23] is an European Union regulation which entered into force on January 11th, 2024 and that will be applicable from September 2025. The objective of this regulation is to complement the DGA by clarifying who can create value from data and under what conditions.

This regulation seeks to achieve an equitable distribution of the value of data by establishing clear and fair rules for the access and use of data within the European

data economy, which is of vital importance in a context of increased presence of IoT systems. Thus, connected products should be designed and manufactured in such a way that users can securely access, use and share the data generated.

As more concrete measures, this regulation seeks to increase legal certainty, mitigate the abuse of contractual imbalances and establish rules that allow public sector agencies to access and use data held by the private sector for specific purposes in the public interest, among others.

Specifically, article 3 of this regulation deals with the "obligation to make product data and related service data accessible to the user", and article 4 establishes "the rights and obligations of users and data holders with regard to access, use and making available product data and related service data". Specifically, with respect to the latter, the first point of article 4 establishes the following in relation to access to data:

> *Where data cannot be directly accessed by the user from the connected product or related service, data holders shall make readily available data, as well as the relevant metadata necessary to interpret and use those data, accessible to the user without undue delay, of the same quality as is available to the data holder, easily, securely, free of charge, in a comprehensive, structured, commonly used and machine-readable format and, where relevant and technically feasible, continuously and in real-time. This shall be done on the basis of a simple request through electronic means where technically feasible.*

In addition, Article 5 presents "the right of the user to share data with third parties", and Article 6 "the obligations of third parties receiving data at the request of the user".

Following the Data Governance Act (DGA), the Data Act is the second main legislative initiative resulting from the European strategy for data (from February 2020).

1.2.6 European Union Artificial Intelligence Act

The first regulation on the use of artificial intelligence, the EU AI Act [24], brought a breakthrough in the field, which was living the accelerated advances of AI with the rise of generative AI-based chatbots such as ChatGPT, and the increasing appearance of such technologies based on Large Language Models (LLMs).

As early as April 25th, 2018 the European Commission established a strategy on AI. However, the EU AI Act was a proposal of the Commission from April 2021, and

it was on December 9th, 2023 that the Parliament reached a provisional agreement with the Council. Its main objective is to provide developers, providers and AI-users with clear guidelines on the requirements and obligations in relation to specific uses of AI. Thus, it follows a risk-based scheme so that requirements are set for providers based on the estimated level of risk. Some examples of each type of risk are given below:

- **Unacceptable risk:** biometric categorization systems that infer sensitive attributes (e.g. race), systems that evaluate or classify individuals or groups based on social behaviors or personal features, systems that evaluate the risk of an individual committing a criminal or penal action based solely on personality profiles, etc. AI systems with this level of risk are prohibited by this regulation (with certain exceptions).

- **High risk:** biometrics, education evaluation performance, safety components in critical infrastructure, credit scoring, risk assessing, migration and border control management, etc.

- **Limited or transparency risk:** this category involves risks related to lack of transparency in the use of AI systems.

- **Minimal risk:** AI systems which doesn't belong to the previous categories. The AI EU Act does not establish any obligation or requirement on these. The vast majority of AI systems currently used in the EU fall into this category.

It is important to note that the EU AI Act is a complementary legislation to the GDPR, so it seeks to reinforce it by adding the conditions and requirements for the development and implementation of AI systems. This is the first law in the world on artificial intelligence, which opens a broad paradigm in relation to regulation in this field, and aims to mitigate the risks it poses to European security, and the fundamental rights of citizens of the Union.

Finally, regarding the impact of this regulation on research, Article 2 of the EU AI Act establishes the sectors to which it applies, and point 6 specifies that "this Regulation does not apply to AI systems or AI models, including their output, specifically developed and put into service for the sole purpose of scientific research and development".

1.3 Open Science

The *open science* paradigm has emerged strongly recently, driven by the needs of scientists themselves, but also by the agencies responsible for funding and managing public research all around the world. It goes hand in hand with the goal of complying with the growing demands for greater transparency and accountability of public governance systems and with the blossoming of a knowledge and data-intensive economy.

Open science extends the idea of openness to all aspects and processes of the research cycle. It ranges from open and online laboratory notebooks to the open evaluation, citation or metrics of research results and *Research Open Data*, as well as the availability of documents and publications describing scientific findings without legal, economical or technological barriers (Open Access).

However, science cannot be made completely open in all cases. Specifically, there are three specific types of limits, restrictions or constraints in open science. This means that the main general regulatory principle established by the Horizon 2020 Program Guidelines on FAIR Data is *"as open as possible, as closed as necessary"*.

- **Security:** the protection of information integrity is a condition that may restrict the accessibility or openness of research data and contents.
- **Privacy:** special attention in the case of handling personal and sensitive data rights, as stated in the European Union General Data Protection Regulation (GDPR).
- **Ownership:** the generated information and knowledge may be subject of intellectual and industrial property, where the owners have the right to limit accessibility or exploitation.

In terms of regulation, in Spain the Spanish Strategy for Science, Technology and Innovation 2021-2027 [25] —in Spanish *Estrategia Española de Ciencia, Tecnología e Innovación (EECTI)*— makes a commitment to open science in its objective 4 "Generation of knowledge and scientific leadership" and in line of action 14 "Science and innovation in society". In addition, on May 3, 2023 was approved the first National Strategy for Open Science 2023-2027 [26] —in Spanish *Estrategia Nacional de Ciencia Abierta (ENCA)*— which aims to promote open access to research results, and consists of four main dimensions: digital infrastructures for open science; management of research data according to FAIR Principles[2]; open

[2]The FAIR principles are a set of guidelines aimed at making data Findable, Accessible, Interoperable and Reusable. More details on this will be given in Section 1.3.1.

access to scientific publications; incentives, recognition and training. In the same line, in France two national plans for open science stand out for implementing a comprehensive regulatory framework to promote open science. Specifically, the Second French Plan for Open Science 2021-2024 [27] —in French *Deuxième Plan national pour la science ouverte*— has as main objective the generalization of open science practices in France. This plan includes the continue development of the HAL national open archive for promoting open access. A second path includes the structuring, sharing and opening up research data.

In the following, three key aspects of open science are presented: open data, open access and open source.

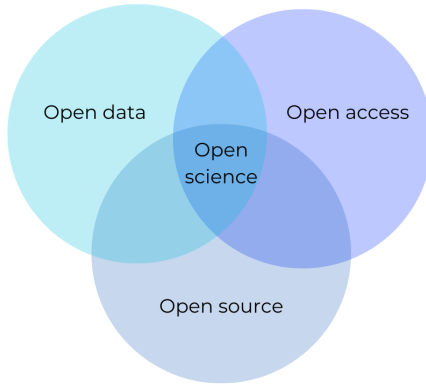


Figure 1.4: Summarized Venn diagram: open science, data, access and source.

1.3.1 Open Data

In 2007 the Organization for Economic Co-operation and Development (OECD) began to promote Research Open Data, but it was from 2017 when the European Union's Horizon 2020 Programme [28] began to require that all research projects met a series of requirements in this area, which are maintained in the current Horizon Europe Programme. Research data, according to the Higher Education Funding Council for England[3] [29] (HEFCE 2008) are the basis of the evidence on which academic researchers build their work.

The open data philosophy consists of making different types of data accessible to society as a whole, to be used and exploited for different purposes. To this

[3]Currently replaced by the UK Research and Innovation national funding agency and the Office for Students.

end, this type of data are licensed under open licenses, such as Creative Commons, among others. Open data must be easily accessible, reliable, structured, properly documented, and free of charge for access and use. The most significant characteristics related with the definition of open data can be summarized (view from the *ideal* approach) as in Table 1.2.

Principle	Description
Access	Data are available in its entirety, in a convenient and modifiable form, and the cost of reproduction is reasonable.
Redistribution	The license does not prevent its free sale or distribution. Moreover, no royalty or fee will be required for it.
Reuse	The license must allow derivative works under the same terms as the original data.
No technological restriction	There are no technological obstacles to distribution.
Attribution	The license may require attribution to creators and contributors for distribution.
Integrity	The license may require the modifications to have a different name from the original.
Distribution	The rights of the work must also apply to its redistribution and to those who perform it.
No discrimination	The license cannot restrict who may make use of the data or the work.

Table 1.2: Open data ideal characteristics. Adapted from [1].

In 2016, M. D. Wilkinson et al. presented the FAIR Guiding Data Principles for scientific data management in [30]. In addition, the European Union and its Horizon programs in particular, have popularized the use of the so-called FAIR principles of open access to research data, which are summarized in the following four key terms:

- **Findable:** data are uniquely and persistently identifiable. They must have

basic machine-readable descriptive metadata.

- **Accessible:** data are accessible to humans and machines using standard formats and protocols.
- **Interoperable:** data are machine-readable and annotated with resolvable vocabularies and ontologies[4].
- **Reusable:** data are sufficiently well described to allow (semi-)automated integration with other compatible data sources.

FAIR data are of paramount importance for scientific communities as they help to improve reproducibility, transparency and even collaboration. However, it is important to note that FAIR data does not imply open data. Table 1.2 summarizes the key characteristics of open data.

In addition, since January 2017, Horizon 2020 funded projects include research data in open access and it is mandatory to store them in a data repository and to provide a Data Management Plan (DMP)[5] during the first six months of the project.

As we have already pointed out, data are a key element in the construction of knowledge, which has led to the huge potential of open data. However, as discussed above, there are cases in which the analysis of certain data may breach privacy barriers and violate users' rights, so it is key to evaluate the ethical and social aspects related to the publication of open data. For this reason, it is important to pay attention to both the data anonymization process and the application of additional layers of privacy protection to prevent possible attacks during secure protocols for information transmission and during data publishing. In addition, as will be further explored in this thesis, privacy preserving techniques should be considered during data analysis, especially when applying ML/DL models.

1.3.2 Open Access

As it is widely known, in the academic field the form of disseminating research results is essentially reflected in journal articles or presentations at conferences. In the first case, a few years ago, access to these articles was limited to the academic community, generally on a subscription basis. However, the beginning of the 21st

[4]Ontology: denomination and formal definition of the types, properties and interrelationships of the entities that actually exist in a given domain.

[5]According to the Horizon 2020 online manual, a Data Management Plan (DMP) describes the data management lifecycle for the data that will be collected, processed and/or generated by an Horizon 2020 project [31].

century brought with it a new movement that sought to make these results, and therefore scientific research, accessible to the general public: *open access*. This aligns with the concept of open science, whose main objective pursued is to make scientific knowledge accessible and made by and for the whole of society.

The open access revolution was initially driven by the known as 3 B's: Budapest Open Access Initiative (2002), Bethesda Statement on Open Access Publishing (2003) and Berlin Declaration on Open Access to Knowledge in the Sciences and Humanities (2003). The first was an adoption arising from a convention organized in December 2001 by the Open Society Institute, and which was made public in 2002. The content developed in that statement constituted the first and most widely used definition of open access, given as follows:

> *By "open access" to this literature, we mean its free availability on the public internet, permitting any users to read, download, copy, distribute, print, search, or link to the full texts of these articles, crawl them for indexing, pass them as data to software, or use them for any other lawful purpose, without financial, legal, or technical barriers other than those inseparable from gaining access to the internet itself. The only constraint on reproduction and distribution, and the only role for copyright in this domain, should be to give authors control over the integrity of their work and the right to be properly acknowledged and cited.*

Open access allows unrestricted access without authentication. The main forms are self-publication in open repositories, or publication in open access journals. Regarding the former, researchers can archive their work, research results, data or even software products in thematic or institutional repositories, where they are deposited for free consultation. Regarding the second, certain types of journals or publishers provide the possibility to publish in open access, following business models such as public subsidies, payment by the author's institutions (for which article processing charges apply), financial contributions by university libraries or academic institutions, etc. In both cases, publications have a DOI associated with them for identification purposes, which is a persistent identifier that allows the content to be uniquely and unequivocally identified in a stable manner [32].

1.3.3 Open Source and Software

The source code is the basis for any software, so to execute it, it can either be compiled or interpreted. As far as compiling is concerned, it is a matter of converting the code to a functionally equivalent object code that can be executed. Interpreting

tries to translate it on the fly with a dedicated interpreter (e.g. a Python interpreter) [33].

Software products can evolve rapidly through changes made by the original authors or by external contributions, which makes it essential to label the versions and their corresponding modifications.

From a legal point of view, software is considered an intellectual creation that is therefore protected by copyright. These rights range from the economic point of view, related to the exploitation of the software, and moral rights, which go hand in hand with the protection of authorship. That is to say, the authorship must always remain with the natural persons who wrote the source code.

When distributing software or code, a license must be associated with it in order to make clear the rights and duties of those who use it, so that everything that is not explicitly authorized in the license will be prohibited. Thus, software is protected by copyright and licenses, which are a legal agreement on those copying rights. In any case, ownership is retained by whoever publishes the software [33]. In relation to software licenses, we can distinguish three types:

- **Proprietary:** The developer or distributor reserves all rights and freedoms, as is the case with most commercial software.
- **Free and open source (FOSS):** Gives the user considerable rights and freedoms, such as the right to redistribute, study and modify free software thanks to royalty-free access to the code.
- **Hybrids:** Are more flexible as they are a combination of FOSS with a proprietary license.

On the other hand, with regard to Open Source Software (OSS) licenses, we can essentially distinguish between permissive and copyleft licenses:

- **Permissive:** Basic license that guarantees the freedom of use, modification and redistribution of the software, including attribution to the original author. This type of license does not require redistribution of the original source code or its modifications and are easier to use in a commercial environment. Examples: MIT, Apache 2.0, BSD.
- **Copyleft:** In this type of license, both the binaries and the source code must be made available to users. They include further conditions for redistribution and modifications are also covered by the copyleft license. Examples: GNU General Public License (GPL), GNU Affero General Public License (AGPL).

Open source dissemination increases the visibility of the work among the scientific community, fostering collaboration and the development of new ideas and solutions, which make it fundamental to properly archive, reference and describe the software developed.

1.3.4 European Open Science Cloud

The European Open Science Cloud (EOSC) [34] is a process of transformation of research conducted in the European Union. This initiative is dedicated to furthering open science practices among researchers by delivering a comprehensive computing environment for the storage, processing, and analysis of research data. Specifically, the EOSC is an European Commission initiative that aims to promote the creation of a *"Web of FAIR Data and Services"* for Science in Europe, in order to accelerate scientific advances and innovation, bringing out the full potential of research activities. EOSC aims to foster innovation by making it easier for researchers to share, collaborate and build on each other's work.

In the same line, another key objective is to improve scientific research, which can be achieved again by facilitating access to a large amount of data and research results from diverse fields and institutions. With the focus on improving transparency and reproducibility, EOSC seeks to make data and methods more open and accessible to the scientific community and the general society, which can also help to reduce data silos by promoting data sharing and reuse. This initiative encompasses pan-European research infrastructures and services supported by the various EU stakeholders (including research communities) to support scientific efforts.

In the same line, aligned with the EOSC, the EOSC Federation [35] has as main objective to strengthen open science as a collaborative, transparent and accessible research model that seeks to enhance access to scientific resources. It aims to be a *"system of systems"* of research data and services, providing users with the capability of storing, processing, analyzing and reusing research outputs from a wide range of disciplines.

1.4 Motivation

This section presents the motivation for the current research, exposing the concerns and problems encountered in relation to open data and the privacy issues arising from the massive use of data in data science ecosystems. This also applies to the use of AI techniques such as ML or DL. This motivation is driven by keeping the focus

on the raw material we work with in data science: *data*, and the revolution entailed by its availability, access and use in numerous AI applications across different fields.

1.4.1 Why Data Revolution?

As we showed and discussed in view of the pyramid in Figure 1.1, data represent a major opportunity for the development of an increasing number of applications. In this sense, it is an extremely valuable raw resource. Of particular interest is the intrinsic relationship between today's hyper-connected society and data generation.

In particular, it is of paramount importance to highlight the rise of artificial intelligence, especially boosted by the development of specialized hardware, such as graphics processing units (GPUs), which can accelerate general-purpose computation. In addition, in recent years the rise of LLMs, such as GPT-3 (2020) or GPT-4 (2023), has brought data-driven applications closer to society, making people aware of the importance of the privacy related to the data used to train these models, including ethical aspects derived from their use. However, these models are not bias-free and have put the focus on security and privacy derived from the use of large volumes of data to power these types of models.

However, not only LLMs are revolutionizing the scene, but numerous AI and data-driven applications are currently on the rise. From IoT systems, smart assistants or smart healthcare devices, all of these technologies collect data, often sensitive and associated with individuals, and it is necessary to be aware of the risks and potential issues that this may entail.

Data-based AI applications in sectors as important as the medical one are also of great interest. In this case, this technology plays a relevant role in bringing society closer to the 4P's medicine: predictive, personalized, preventive and participative. This has the potential to improve diagnoses, personalize them according to the specific case study, and facilitate the task of healthcare professionals, even accelerating certain processes. For this task, large volumes of data are a fundamental requirement, but several privacy restrictions apply. In particular, in this sector the sensitivity of the data is clear, and numerous data protection measures must be taken into account from its acquisition and processing to its use as a basis for AI models and during their development and deployment.

The scientific method begins with observation, to then allow us to pose the problem, propose a hypothesis, conduct the experiments, analyze the data obtained during them and draw conclusions. In every scientific research we develop, data are a fundamental part, and with the large volumes of data that are collected daily, and the powerful data processing centers that provide us with not only processing

capacity, but also storage capacity, we are facing a real revolution. However, this revolution also entails its own problems, and in this thesis we will focus on those arising from privacy issues both in relation to the raw data and during its processing, analysis and use as a basis for ML/DL models.

Data are the essential foundation on which machine and deep learning models widely used under the umbrella of the artificial intelligence context are built. Without data, we cannot train models that allow us to predict tomorrow's weather, the likelihood of an individual suffering from a certain disease based on his history or medical test results, or estimate how long an airplane might be delayed, among many other applications. However, there are also numerous privacy restrictions that apply to such data for processing and analysis and use in the training of such models.

1.4.2 Thesis Road-Map: from Data Privacy to Privacy Preserving Machine Learning

In this thesis, we will start by exploring data privacy methods (Part II) and continue with the study concerning Privacy-Preserving Machine Learning (PPML) techniques, dedicated to privacy-preserving models and analytics without sharing data among different collaborating entities (Part III).

Regarding the first, if we think about data privacy, it is important to classify data according to their level of sensitivity. From the first level, which corresponds to open data, which do not require the use of a Trusted Research Environment (TRE), to data with low risk (e.g. pseudonymized with low risk of re-association or low sensitivity of the information), low (pseudonymized with some indirect identifiers), medium (pseudonymized containing confidential information), high (sensitive personal or commercial data without de-identification), very high (special category data, personal, very sensitive or confidential information of organizations, institutions and/or governments).

In this regard, Chapter 2 outlines a battery of anonymization and pseudonymization techniques, common attacks on anonymized databases, metrics for evaluating privacy-utility trade-offs, two software products implemented for data anonymization and checking, and the impact of anonymization on ML/DL model development. This is followed in Chapter 3 by an explanation of the theory and associated mechanisms concerning *differential privacy*. Variations such as local differential privacy and metric and Rényi differential privacy are addressed. Direct application of these kind of mechanisms on the data before use and/or publication are addressed. However, data privacy issues are not limited to the processing and acquisition of the data

itself. The problem extends if we think about machine learning models, since, even if the data are not exposed, the results that the models return in predictions, as well as the parameters that define them, can give us additional information about the data with which they were trained. In this sense, we explore to impact of incorporating differential privacy mechanisms during the training phase in order to mitigate the loss of information derived from the trained model and its predictions.

The rise of data-driven technologies, together with the great value of information, implies the imperative need to guarantee its security and privacy, which motivates the development of privacy-preserving techniques within machine learning systems (what we know as PPML). In many cases, the problem arises when we want to train models with large volumes of data that cannot be centralized. To obtain robust models of ML/DL, it is essential that they have been trained with a significant amount of data. However, in many cases, data cannot be centralized on a single central server to train these models due to privacy, legal and ownership issues. In this sense, the application of techniques that allow us to perform the training in a decentralized way, as is the case of federated learning, becomes increasingly valuable. This kind of distributed architectures, theoretical basis, advantages, and implementations details among other insights will be discussed, together with different practical use cases, in Chapters 4 and 5 respectively. Regarding the use cases, and with the aim of highlighting the multidisciplinary nature of this training methodology for learning models with privacy-preserving measures, three particularly diverse applications will be discussed in detail: medical imaging, water quality monitoring and climate sciences.

Having large volumes of open data available to scientists and society in general, from different fields and institutions, is key to empower scientific development and innovation. However, it is critical to focus on the privacy issues that arise from the acquisition of the data, its processing, its analysis and its use as a basis of AI models, aspects that will be carefully analyzed within this thesis.

Part II: Data Privacy

CHAPTER 2

Anonymization and Pseudonymization

En los ojos se abren
infinitos senderos.
Son dos encrucijadas
de la sombra.

Federico García Lorca,
Los ojos,
Suite de los espejos

Contents

2.1 Pseudonymization Techniques 45
2.1.1 Classic Methods 47
2.2 Anonymization Techniques 52
2.2.1 k-anonymity 54
2.2.2 (α,k)-anonymity 56
2.2.3 ℓ-diversity 57
2.2.4 Entropy ℓ-diversity 59
2.2.5 Recursive (c,ℓ)-diversity 60
2.2.6 t-closeness 61
2.2.7 Basic β-likeness and enhanced β-likeness 62
2.2.8 δ-disclosure privacy 63
2.3 Common Attacks on Anonymized Databases 64
2.4 Privacy-Utility Trade-Off Metrics 68
2.5 Disclosure and Re-Identification Risk Control 70
2.6 *pyCANON*: a Python Library to Check the Level of Anonymity of a Dataset 73
2.6.1 Main Functionalities 73
2.6.2 Usage Examples 75
2.6.3 Further Functionalities 77
2.6.4 Summary 78
2.7 Impact of Anonymization Techniques in Machine Learning Models Training 78
2.7.1 Data Analyzed and Hierarchies 79
2.7.2 Machine Learning Models Under Study 80
2.7.3 Results and Analysis 82
2.7.4 Summary and Wrap Up 86
2.8 *Anjana*: a Python Library for Anonymizing Sensitive data . . . 86
2.8.1 Algorithms to Achieve *k-anonymity* 87
2.8.2 Methodology 90
2.8.3 Usage Examples 91
2.8.4 Further Functionalities 93
2.8.5 Summary 98
2.9 Conclusions 98

Abstract

This second chapter introduces the topics of anonymization and pseudonymization, as well as different insights regarding of both techniques, with a special focus on tabular data and their fundamental concepts. A battery of methods to anonymize tabular data are analyzed, from the most classical ones such as *k-anonymity* and *ℓ-diversity*, to other less complex ones, such as *entropy ℓ-diversity*, *β-likeness* and *δ-privacy disclosure* among others. Then, common attacks on anonymized databases, a set of privacy utility trade-off metrics and some methods for preventing disclosure and re-identification attacks are analyzed. Subsequently, the implementation of a Python library to check the anonymity level of a given dataset is presented. Next, a comparative study of the performance of a battery of machine learning models after applying different anonymization techniques on a given dataset, and with different levels, is conducted. Finally, the implementation of a Python library to anonymize tabular data with the techniques explored in this chapter is presented.

Advances in the field of data science, artificial intelligence (AI), machine and deep learning (ML/DL), especially with applications in data analysis and processing, and their use for decision making, put the focus on the privacy of the data that is being exploited. In particular, the regulatory requirements, on the one hand, to publish data in open access, and on the other hand to promote collaboration between different entities, make it necessary to study and develop technologies and methods focused on guaranteeing the privacy of the data before they are published or shared. There are certain types of data that are extremely important to protect before they are shared or published (as in the case of medical records) since it is critical to prevent the extraction of sensitive patient information. The same can be extrapolated to records related to the economic or banking field, among many others. However, as will be explained below, data that are a priori less compromising can also allow an attacker to extract sensitive information about an individual.

The motivation for introducing the concepts of anonymization and pseudonymization is clear: to protect the information of individuals, and that they cannot be identified from it. But first of all, let us start by further introducing the problem. There are numerous examples that show the need to continue studying and working on techniques that allow us to achieve a good balance between privacy and utility for data analysis. For example, suppose that we want to analyze the census

of Spain, in order to extract knowledge about the population. We know that it is not enough to remove the name and ID of individuals, because with data such as date of birth, zip code, gender and marital status, we can obtain a lot of information about an individual. In fact, a study conducted in 2000 using the 1990 United States of America census revealed that in 87% of the time, three pieces of information (zip code, gender and date of birth) are enough to identify someone in a database [36]. Therefore, hiding the most sensitive information, such as name, phone number or ID number, may not be enough if the rest of the attributes that seem harmless (date of birth, gender, city, etc.) remain intact, because when combined with other publicly available data (linkage attacks), there can lead to revealing the identity of the individuals represented in the data or sensitive information about them.

An example that is classically presented when studying anonymization techniques is that of the known as the *Netflix Prize*. The goal of this contest was for participants to create an algorithm to recommend to users the movies that best fit their preferences. To do this, they had "anonymized" user data provided by Netflix, replacing the user's name with a numeric ID, as well as the movie name, but showing the rate and the grade rate. Researchers from from the University of Texas at Austin [37] demonstrated that they could identify users by their movie ratings by combining this information with the public *IMDb* database. Although apparently this information may not seem to be of great importance, the same researchers revealed that an individual's political orientation and religious opinion can be revealed by his or her opinions on different movies [38]. As discussed in the introduction, being able to extract this type of data on a specific individual due to lack of anonymity is a very serious violation of privacy, due to social risks that an individual may suffer when this private information is exposed.

Examples as simple as the above show the need to properly apply different anonymization techniques when dealing with personal data. However, problems are encountered when trying to maintain this balance between privacy and preserving as much information as possible, so that data remains usable. If in the case of the *Netflix Prize* the information on the ratings given by users had been hidden, the objective would simply not have made sense, but as we will see below, techniques could have been applied to make re-identification more complex.

Another example can be seen in the case of the Taxis of New York City. In 2014, the New York City Taxi and Limousine Commission (TLC) made public data recorded by cab GPS systems, comprised of more than 173 million individual taxi trips taken in New York City during 2013. The fact of publishing this data openly was considered useful for urban transportation research and planning, however,

the controversy arises from the privacy risks derived from this publication. But, is it really important to anonymize cab trip data? Well, actually, knowing where a certain person takes a taxi, how much he/she pays for it, the tip left to the taxi driver, or where is going, can reveal private information. The information published in this dataset: exact location, time, tip given, price, etc combined with a dataset with images of celebrities getting into cabs, allowed to reveal much of the sensitive information mentioned for certain celebrities. In addition, the data on the medallion and driver license IDs had been pseudonymized in the original dataset using the cryptographic algorithm known as MD5, so it could be de-anonymized quite easily [39], for example via a brute force attack, generating all the possible combinations of the licenses and applying such algorithm.

For now, we have used the terms of anonymization and pseudonymization. What are the differences between these two terms? When do we use one or the other technique? To address these questions, let us take a look to the following definitions:

Definition 2.0.1. Pseudonymization. The General Data Protection Regulation (GDPR) defines the concept of **pseudonymization** as "the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organizational measures to ensure that the personal data are not attributed to an identified or identifiable natural person" [40].

Definition 2.0.2. Anonymization. The GDPR defines the concept of **anonymous data** as "information which does not relate to an identified or identifiable natural person or to personal data rendered anonymous in such a manner that the data subject is not or no longer identifiable" [40].

According to definitions 2.0.1 and 2.0.2, it is clear that the concepts of pseudonymization and anonymization are not interchangeable, in fact, anonymous data cannot be associated to specific individuals, unlike pseudonymized data. For example, note that encrypting data is not an anonymization technique, but a pseudo-anonymization technique. In fact, according to the GDPR, pseudonymized data are personal data, but not anonymized ones. When to use one technique or another depends on the type of records available and the purpose of the database, as well as on the columns of the dataset in the case of tabular data: for example, if one of them is the ID number, we will probably be interested in either suppressing that information, or applying a pseudo-anonymization technique such as encrypting it or even transforming it using, for example, an approach based on random numbers.

In this chapter we focus in particular on the anonymization of tabular data and/or metadata or information that can be converted to this format for the privatization process. Therefore, in the following chart we introduce some important concepts related to the characteristics of a tabular database.

BASIC CONCEPTS - TABULAR DATABASE

- **Identifiers**: are those variables in a database that make it possible to unequivocally identify an individual. The most common examples are the name and surname, identification number, email, phone number, etc.
- **Quasi-identifiers (QI)**: these are the variables in a database that when combined can allow to identify an individual or a group, and which are accessible to an attacker. Although individually they may seem to be irrelevant information, as we have already mentioned combined with other information, they can reveal the identity of the individual. Some classic examples include age, date of birth, city of residence, gender, etc.
- **Sensitive Attributes (SA)**: are database properties that should not be associated with individuals because they are considered sensitive information. For example, in the case of medical databases, the sensitive attribute can be the illness of the patient or the duration and reason for an hospital admission.
- **Insensitive attributes**: are the properties or information of an individual, which are present in the database, but do not involve any risk. Examples of such attributes depend on the database in question and the information available. For example, a person's preferences, such as favorite series or movies, religion or political views in some cases may be QIs when combined with other information, in others they may be insensitive and in other cases it may be the sensitive attribute to protect.
- **Equivalence Class (EC)**: is a partition of the dataset in which all quasi-identifiers have the same value within the equivalence class. That is, users that are in the same equivalence class are indistinguishable with respect to QIs. Note that the values of the sensitive attributes do not have to be the same for all equivalence class entries.

In this sense, identifiers should be removed from the database. A priori, if no transformation is applied to a tabular data base, it is very likely that each row or

record in the data base will form an equivalence class. Then, in order to obtain different equivalence classes, the quasi-identifiers can be processed using the technique known as generalization based in hierarchies which can be defined as follows (see, among others, [41]):

Definition 2.0.3. Database generalization (without records suppression). Be ξ a database and QI the set of quasi-identifiers of the database. ξ' is a **generalization (without records suppression)** of ξ iff $|\xi| = |\xi'|$ and ξ' can be ordered verifying that $\xi'_{i,j} \in \psi(\xi_{i,j})$ (where $\psi(x)$ is a **generalization function** which returns the set of generalizations of the value x) for every row i and $\forall j \in cols(QI)$, where $cols(QI)$ are the columns of the database which correspond with the quasi-identifiers. If known values are generalized into unknown values, setting it, for example, to $*$, this is called **cell suppression**.

Note that in some cases we can allow record suppression in the generalized database for a achieving certain privacy constraint. In that case, in the previous Definition 2.0.3 we have that $|\xi| \geq |\xi'|$.

Now, let us introduce the concept of hierarchy:

Definition 2.0.4. Hierarchy. Let QI be the set of quasi-identifiers of a database ξ, we define a **hierarchy** h_1 over a certain quasi-identifier q_i ($i \in \{1, \dots |Q|\}$) as a mapping from $\xi(q_i)$ to $\xi(q_i^{(1)})$, $h_1 : \xi(q_i) \longrightarrow \xi(q_i^{(1)})$, with $\xi(q_i)$ the initial set of values of q_i in ξ, and $\xi(q_i^{(1)})$ being the new set of new values of q_i once generalized (first transformation).

Thus, let $\{x_1, \dots, x_m\} \in \xi(q_i)$ be the set of unique values taken by the attribute q_i in ξ, the function h_1 assigns $h_1(x_i) = y_j \ \forall i \in \{1, \dots, m\}$. Note that different values of x_i can take the same value y_j $j \in \{1, \dots, p\}$, $p \leq m$. Also, note that h_1 is supposed to operate upon all the values of $\xi(q_i)$.

Thus, we can define n mapping h_i $\forall i \in \{1, \dots, n\}$ to generalize the attribute q_i as much as necessary by applying a new hierarchy over each state of the database: $h_1 : \xi(q_i) \longrightarrow \xi(q_i^{(1)})$, $h_2 : \xi(q_i^{(1)}) \longrightarrow \xi(q_i^{(2)})$, ..., $h_n : \xi(q_i^{(n-1)}) \longrightarrow \xi(q_i^{(n)})$. Then, for the quasi-identifier q_i we define a set of hierarchies $H_{q_i} = \{h_1, \dots, h_n\}$, with n the number of possible transformations for q_i.

Example 2.0.1. Let us consider the database given in Table 2.1 in which certain characteristics of some patients are shown, such as the *age*, *sex* and some *medical conditions*. Then, be the quasi-identifiers *age*, *sex*, *blood Pressure levels (BP)*, *cholesterol* and *Na to potassium ration (Na to K)*, and *drug* the sensitive attribute (the drug that should be administered according to each patient condition). This example is composed of the first 10 rows of the drug classification dataset (available in Kaggle

[2]) and its main purpose is to apply ML/DP techniques to predict the appropriate drug for each patient based on his/her *age*, *sex* and *medical conditions*.

Age	**Sex**	**BP**	**Cholesterol**	**Na to K**	**Drug**
23	F	HIGH	HIGH	25.355	DrugY
47	M	LOW	HIGH	13.093	DrugC
47	M	LOW	HIGH	10.114	DrugC
28	F	NORMAL	HIGH	7.798	DrugX
61	F	LOW	HIGH	18.043	DrugY
22	F	NORMAL	HIGH	8.607	DrugX
49	F	NORMAL	HIGH	16.275	DrugY
41	M	LOW	HIGH	11.037	DrugC
60	M	NORMAL	HIGH	15.171	DrugY
43	M	LOW	NORMAL	19.368	DrugY

Table 2.1: Original database (extracted from the first 10 rows of drug classification dataset [2], available in Kaggle).

It can be observed that no row is equal to another with respect to the QIs. Therefore, in Table 2.2, a generalization of this dataset with respect to the numerical QIs *Age* and *Na to K* is shown using interval based hierarchies.

Age	**Sex**	**BP**	**Cholesterol**	**Na to K**	**Drug**
[20, 25)	F	HIGH	HIGH	[25, 30)	DrugY
[45, 50)	M	LOW	HIGH	[10, 15)	DrugC
[45, 50)	M	LOW	HIGH	[10, 15)	DrugC
[25, 30)	F	NORMAL	HIGH	[5, 10)	DrugX
[60, 65)	F	LOW	HIGH	[15, 20)	DrugY
[20, 25)	F	NORMAL	HIGH	[5, 10)	DrugX
[45, 50)	F	NORMAL	HIGH	[15, 20)	DrugY
[40, 45)	M	LOW	HIGH	[10, 15)	DrugC
[60, 65)	M	NORMAL	HIGH	[15, 20)	DrugY
[40, 45)	M	LOW	NORMAL	[15, 20)	DrugY

Table 2.2: Database obtained generalizing the QI *Age* and *Na to K*.

In the previous table it can be seen that with such a generalization, there is only

one equivalence class that has more than one row, the one formed by rows 2 and 3 (highlighted in yellow). With an even greater generalization on the attributes *Age* and *Na to K*, and applying suppression on the quasi-identifiers *BP* and *Cholesterol*, as done in Table 2.3, a new equivalence class formed by two rows, 4 and 6, is obtained, and the one formed by rows 2 and 3, now also contains row 8.

Age	Sex	BP	Cholesterol	Na to K	Drug
[20,30)	F	*	*	[15,30)	DrugY
[40,50)	M	*	*	[0,15)	DrugC
[40,50)	M	*	*	[0,15)	DrugC
[20,30)	F	*	*	[0,15)	DrugX
[60,70)	F	*	*	[15,30)	DrugY
[20,30)	F	*	*	[0,15)	DrugX
[40,50)	F	*	*	[15,30)	DrugY
[40,50)	M	*	*	[0,15)	DrugC
[60,70)	M	*	*	[15,30)	DrugY
[40,50)	M	*	*	[15,30)	DrugY

Table 2.3: Database obtained generalizing the QI *Age* and *Na to K*, and suppressing *BP* and *Cholesterol*.

Despite the deletion of information from two quasi-identifiers, there are five equivalence classes that consist of a single row (note that there are only ten rows), which again indicates the need to develop techniques that allow us to achieve the desired balance between privacy and data utility (since there will be information that we do not want to delete or overgeneralize).

It is important to note that the initial information can be very sensitive because it contains information about individuals, but also very significant due to the value it can provide to different pharmaceutical companies in their production strategies and marketing campaigns for the design and sale of drugs.

2.1 Pseudonymization Techniques

The information exchange and communication of sensitive data is essential in many sectors, such as banking, economics, education, medicine and research. It is therefore critical to implement preventive measures to protect the privacy of such data, which raises numerous challenges as discussed throughout this thesis. In section

we will present a technique that opens up a different paradigm beyong anonymization by facilitating the processing of personal data while providing solid guarantees of their security: pseudonymization.

Essentially and concisely, pseudonymization is the masking of individual identifiers in a database by pseudonyms. This technique is explicitly mentioned in Article 25 of the GDPR, hence its special interest and attention. Pseudonymization is a deidentification technique, which aims to remove the potential association between the identifiers in a database and the individual. To do this, we must establish a link between the original data and the pseudonyms applied to them, called an association table. This correspondence or association table can only be accessed by the entity that carried out the pseudonymization process, which we call the pseudonymization entity.

In the recital number 28 of the GDPR (April 27th 2016), is stated the following assumption concerning why and when apply pseudonymization techniques to data:

> *The application of pseudonymisation to personal data can reduce the risks to the data subjects concerned and help controllers and processors to meet their data-protection obligations.*

In addition, the Article 32(1) of GDPR states the following concerning security of processing:

> *Taking into account the state of the art, the costs of implementation and the nature, scope, context and purposes of processing as well as the risk of varying likelihood and severity for the rights and freedoms of natural persons, the controller and the processor shall implement appropriate technical and organisational measures to ensure a level of security appropriate to the risk, including inter alia as appropriate:*
>
> *(a) the pseudonymisation and encryption of personal data;*
>
> *(b) the ability to ensure the ongoing confidentiality, integrity, availability and resilience of processing systems and services;*
>
> *(c) the ability to restore the availability and access to personal data in a timely manner in the event of a physical or technical incident;*
>
> *(d) a process for regularly testing, assessing and evaluating the effectiveness of technical and organisational measures for ensuring the security of the processing.*

This means that in view of the GDPR, pseudonymization can be an suitable technical and organizational measure in order to provide an appropriate level of security.

2.1.1 Classic Methods

We can essentially distinguish three types of methods depending on how pseudonymization techniques are applied:

- **Deterministic**: the same pseudonym will always be used for the same data. For example, if the pseudonym *"abcd"* is always used for the name *"Juan"* in database *A*, the same pseudonym will be used in database *B*.
- **Document randomization**: within the same document or database, the same pseudonyms will always be used for the same data, but only at the local application level (that is, within the same document). In this case, if in database *A* the pseudonym *"abcd"* is used for the name *"Juan"*, in database *B* another pseudonym may be used (e.g. *"efgh"*), but always the same one within the database.
- **Random**: for the same data, a different pseudonym is always used. In this case, given a database, the pseudonym used for the name *"Juan"* may change in different records, e.g. it may be in one case *"abcd"*, in another *"efgh"*, in another *"ijkl"*, etc.

Some of the most classic techniques used to pseudonimize data are presented in the following, according with the best practices stated by the European Union Agency for Cybersecurity (ENISA) in the report *"Pseudonymisation techniques and best practices"* [42].

2.1.1.1 Counters and Random Numbers

The first idea that usually arises for obfuscating an identifier is to replace it with a numerical value. The intuitive idea of using a counter is a first approximation, so that each identifier is substituted in an orderly manner by an integer. That is, the identifiers are substituted by a number chosen by a monotonic counter. However, the sequential character of the counter can still provide information on the order of the data within a dataset.

A more viable option that also arises intuitively is to perform this value-number assignment randomly, without using a counter. Thus, each value of the identifier

will be associated with a random number. If this is done deterministically within the dataset itself, for the same identifier value the same random number will always be applied, whereas this will not be the case if the association is done randomly in the dataset. It is important to note that in the second case a collision may occur in the case of two identifiers being associated to the same pseudonym. In this line, an random number generator (RNG) can be used for this purpose. More precisely, as a general rule, a pseudorandom number generator (PRNG) will be used, since the sequence will be generated from an initial value (seed).

In both cases, once a pseudonym is associated to each value of the identifier to be pseudonymized, the relationship between the two (real value and pseudonym) must be stored in a mapping table.

Using random numbers as pseudonyms gives a better guarantee of security than simply using a counter because if the mapping table is not compromised, an attacker will find it difficult to extract information, since the pseudonyms were generated randomly.

2.1.1.2 Hash Functions

We now turn to cryptographic systems, which are based on three fundamental types of algorithms: symmetric key, asymmetric key and hash functions. If we think of pseudonymization as an approach in which personal data is protected by replacing direct identifiers with pseudonyms in a reversible way, hash functions are of special relevance. An introduction to cryptographic systems and to the symmetric and asymmetric key methods is given Section A.3 from Appendix A as a reference for the reader.

A hash function is cryptographic system based on a computational function that maps an input of an arbitrary size to values of a fixed size. The formal definition is given as follows:

Definition 2.1.1. Hash function. A h**ash function** $\mathscr{H} : D \longrightarrow R$ verifies:

- $\forall x \in D$, $\mathscr{H}(x) \in R$ always has the same length, regardless of the length of x.
- Given $x \in D$ and $\mathscr{H}(x) \in R$, it must be computationally complex to find $y \in D$, $y \neq x$, such as $\mathscr{H}(x) = \mathscr{H}(y)$.
- Given $r \in R$ it must be computationally complex to find $x \in D$ with $\mathscr{H}(x) \in R$ such as $\mathscr{H}(x) = r$.

The digest is the output obtained from a hash function. The main property of the hash functions is that for two different inputs, the digest generated by the hash

function must always be different (so that no collision occurs), or at least it must be computationally complex to find a collision. In this line, we have two kind of properties concerning resistance to collisions:

- **Weak resistance:** if it is computationally easy to find a pair $x, y \in D$ such as $\mathscr{H}(x) = \mathscr{H}(y)$.
- **Strong resistance:** if it is computationally complex to find a pair $x, y \in D$ such as $\mathscr{H}(x) = \mathscr{H}(y)$.

The list of cryptographic hash function is wide and the most widely known are the two following families:

- **Message digest functions (MD)**. It combines a compression function f with an output function g. Some examples are: MD2, MD4, MD5, MD6. The most widely used is MD5, which generates a 128-bit hash. This hash function was widely used in software development, but collisions have been found. Typically, 128-bits (16-bytes) MD5 hashes are represented as a sequence of 32 hexadecimal digits.
- **Secure hash algorithms (SHA)**: It is a family of six algorithms: SHA-i with $i = 0, 1, 224, 256, 384, 512$. For $i = 0, 1$ the output block length is 160 bits (40 digits in hexadecimal format). However, SHA-1 is not collision resistant so its use is discouraged. For $i = 224, 256, 384, 512$, the output block length is i bits. SHA-256 is more secure than MD5 or SHA-1. However it is 20-30% slower.

2.1.1.3 Hash Function with Authentication Code

Hash-based Message Authentication Code (HMAC) methods aim to use together a hash function and a secret key. Specifically, the idea is that with HMAC we need to use a secret shared key for calculating and verifying the message authentication values [43]. In addition, this protocol uses generic cryptographic hash functions such as the ones mentioned above. For example, we will denote as HMAC-MD5 the case in which we use a hash function MD5 with a secret key under the scheme of the HMAC method. The key used for HMAC can be of any length but if L is the byte-length of hash outputs, then is recommended to use keys with more that L bytes in order to increase the security [43]. Some implementation notes on this approach can be found in [43] and in [44], the last using SHA-256 as hash function.

2.1.1.4 Encryption

Intuitively, encryption consists of applying a bidirectional (reversible) cryptographic function that transforms the input identifier data into values that can be transformed back into their original format using a key.

As explained when presenting the hash functions, we can think about symmetric and asymmetric encryption, according to the pair of keys used. For example, concerning public-key cryptosystem (see , the RSA algorithm (September 1983) is widely used, which is a deterministic encryption algorithm verifying the multiplicative property. The scheme of the RSA protocol is given below (note that some assumptions for this protocol and theoretical concepts are reviewed in Appendix A):

RSA cryptosystem scheme

1 Select two primes p and q.

2 Compute $n = p \cdot q$.

3 Compute the Euler function $\phi(n)$. As p and q are two prime numbers, $\phi(n) = (p-1) \cdot (q-1)$.

4 Select e verifying $1 < e < \phi(n)$, with $gcd(e, \phi(n)) = 1$.

5 Calculate d verifying $e \cdot d = 1 \ (mod\ \phi(n))$.

6 The public key is (e, n) (modulus and encryption exponent), and the private key is (d, n) (modulus and decryption exponent).

Example 2.1.1. In this example we show how to calculate the public and private keys for the RSA cryptosystem and how to encrypt and decrypt a plaintext:

Be $p = 31$, $q = 37$, then $n = 31 \cdot 37 = 1147$ and $\phi(n) = 1080$. We have to select e verifying $1 < e < \phi(n)$, with $gcd(e, \phi(n)) = 1$, for example, $e = 23$. Now, we calculate d verifying $e \cdot d = 1 \ (mod\ \phi(n))$, that is $d \cdot 23 = 1 \ (mod\ 1080)$. For doing this we can use de extended Euclidean algorithm and we get that $d = 47$ as follows:

$$\begin{array}{c|c} 1080 = 23 \cdot 46 + 22 & 1 = 23 - 22 \cdot 1 \\ 23 = 22 \cdot 1 + 1 & 1 = 23 - (1080 - 23 \cdot 46) \cdot 1 \\ 22 = 1 \cdot 22 + 0 & 1 = 23 \cdot \mathbf{47} - 1080 \end{array}$$

Then, the public key is $(e, n) = (23, 1147)$ and the private key is $(d, n) = (47, 1147)$. Now suppose that we want to encrypt $c = 5$. First, we get that $enc(c) =$

$c^e \ (mod\ n) = c^{23} \ (mod\ 1147) = 242$. Then, the decryption of $enc(c) = 242$ is $c' = enc(c)^d \ (mod\ n) = 242^{47} \ (mod\ 1147) = 5$.

The above techniques (counters, RNG, hash functions, HMAC and encryption) can be applied to define pseudonyms for identifiers in a tabular database. Usually, these pseudonyms are stored in a mapping table, in case it is necessary to reverse the process. The following is an example of how to apply each of these techniques to a specific case.

Example 2.1.2. Let us consider the database given in Table 2.4 with the identifier filed *name*, *age* as quasi-identifier and *temperament* as sensitive attribute.

Name	***Age***	***Temperament***
Bernarda	60	Authoritarian
Angustias	39	Distressed
Magdalena	30	Depressive
Amelia	27	Submissive
Martirio	24	Jealous
Adela	20	Rebel

Table 2.4: Example of database to pseudonymize for the identifier *name*.

Table 2.5 presents a pseudonymized version applying the hash function MD5 to the identifier.

Name	***Age***	***Temperament***
0538701bb2679f0e2a927352ae852f30	60	Authoritarian
5d2dab4f125ce4b655fa5bde98523295	39	Distressed
a9a217a75f0848afa3a801eafbf61b8b	30	Depressive
93ea6597c3cbd06e93a46b9f5368732d	27	Submissive
21884a11d48d01a58f9976c4d55088b8	24	Jealous
652c32e3acec34c19d209ebca0a0dcd1	20	Rebel

Table 2.5: Pseudonymized version of Table 2.4 applying MD5 to the field *name*.

In addition, Table 2.6 shows the resulting pseudonyms based on the 5 techniques presented above: counter, RNG, hash function, HMAC and encryption. Specifically, for the case of the hash function MD5 has been used, and for HMAC MD5 with

the key *"lcdba"* (HMAC-MD5). For encryption, the RSA protocol has been applied with 2^8 as the number of bits required to store n.

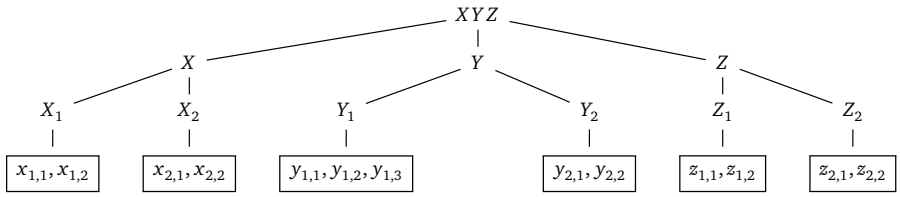
Identifier	***Method***	***Pseudonym***
Bernarda	*Counter*	1
	RNG	18
	Hash (MD5)	0538701bb2679f0e2a927352ae852f30
	HMAC (MD5)	257c358d04158a2b3fe1a1e9b36ac6f0
	Encryption	1aea0c8ab176125286aba79615fbcc39ffa7c3f85887abf37333e7cbcde3f043
Angustias	*Counter*	2
	RNG	73
	Hash (MD5)	5d2dab4f125ce4b655fa5bde98523295
	HMAC (MD5)	dd82da91e1deddfb197955edee23e8f8
	Encryption	7894977b4fd9e85b18b67a339cd30208ce657cecb82d36bfea96950e197a9e3d
Magdalena	*Counter*	3
	RNG	98
	Hash (MD5)	a9a217a75f0848afa3a801eafbf61b8b
	HMAC (MD5)	dd82da91e1deddfb197955edee23e8f8
	Encryption	61d94b914922744577dcffb4c26c6c6f4504dbf88c348da1b0c3d124fa966f7f
Amelia	*Counter*	4
	RNG	9
	Hash (MD5)	93ea6597c3cbd06e93a46b9f5368732d
	HMAC (MD5)	c585f3e8057a6f4ac8c177b036b51bd9
	Encryption	9a02b98b84c2eb1e34e5863aefe8dcf04de589d4cdb819c112f26950dbd58811
Martirio	*Counter*	5
	RNG	33
	Hash (MD5)	21884a11d48d01a58f9976c4d55088b8
	HMAC (MD5)	63418906ef1fbf429692047bc2688ae5
	Encryption	2a202164e189d796777b6ad22fc050eb9678eb268174b2322121b26b5bcb6d5c
Adela	*Counter*	6
	RNG	16
	Hash (MD5)	652c32e3acec34c19d209ebca0a0dcd1
	HMAC (MD5)	5d87eaae826e1c66ed5b7817b6267c7e
	Encryption	9aac19f8a138d5d3e570007f53afe43b59f01474d7ae5c74313672a846282a08

Table 2.6: Mapping of the identifier (*name*) using five pseudonymization methods.

2.2 Anonymization Techniques

In this section, a battery of techniques used in order to achieve data anonymization, is presented. In all of them, a parameter is defined according to the level of anonymization on the basis of the type of attack to be avoided and the desired level of anonymization.

First, note that an example of generalization scheme can be given by the following diagram, with different levels of hierarchy, where XYZ are all possible values of a sensitive attribute:



Be $\mathscr{S} = \{x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}, y_{1,1}, y_{1,2}, y_{1,3}, y_{2,1}, y_{2,2}, z_{1,1}, z_{1,2}, z_{2,1}, z_{2,2}\}$ the set of possible values for a given sensitive attribute of a database, let us look at some of the possible different levels of generalization:

a) No generalization: do not change the values for the sensitive attribute.

b) Transform $[x_{1,1}, x_{1,2}]$ into X_1 and $[x_{2,1}, x_{2,2}]$ into X_2. All other values remain unchanged.

c) Transform $[x_{1,1}, x_{1,2}]$ into X_1, $[x_{2,1}, x_{2,2}]$ into X_2, $[y_{1,1}, y_{1,2}, y_{1,3}]$ into Y_1, $[y_{2,1}, y_{2,2}]$ into Y_2, $[z_{1,1}, z_{1,2}]$ into Z_1 and $[z_{2,1}, z_{2,2}]$ into Z_2.

d) Transform $[x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}]$ into X, $[y_{1,1}, y_{1,2}, y_{1,3}]$ into Y_1, $[y_{2,1}, y_{2,2}]$ into Y_2, $[z_{1,1}, z_{1,2}]$ into Z_1 and $[z_{2,1}, z_{2,2}]$ into Z_2.

e) Transform $[x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}]$ into X, $[y_{1,1}, y_{1,2}, y_{1,3}, y_{2,1}, y_{2,2}]$ into Y and $[z_{1,1}, z_{1,2}, z_{2,1}, z_{2,2}]$ into Z.

f) Transform all the values to XYZ.

Before going on to study each of some of the most popular anonymization techniques, it should be pointed out that essentially two approaches can be considered to anonymize by means of generalization: *local recording* and *global recording* [45].

- *Local recording:* values can be generalized to different domain levels of the hierarchy. Some examples are *b)* and *d)* from the previous list.

- *Global recording:* all values must be generalized to the same domain level of the hierarchy. Some examples are *c)*, *e)* and *f)* from the previous list.

2.2.1 *k*-anonymity

The method of *k-anonymity* is one of the most classic and well-known anonymization techniques. As will be exposed bellow, it is a very simple method that allows us to reduce the risk of re-identification considerably. This technique was first proposed in 2002 by L. Sweeney in [46] as a formal protection to re-identification attacks. It is a mathematical definition acting on the equivalence classes of a database as follows:

Definition 2.2.1. k-anonymity. Given a database ξ, it verifies **k-anonymity** if each equivalence class of ξ has at least k rows.

Proposition 2.2.1. *Any database verifying k-anonymity for a given value k' fulfills that for each row in the database, there are at least $k'-1$ records that are indistinguishable regarding the set of quasi-identifiers.*

Proof. Trivial from definition. □

Proposition 2.2.2. *Any database verifies k-anonymity with $k = 1$.*

Proof. Trivial from definition. □

Proposition 2.2.3. *Be ξ a database with $|\xi| = N$. If there is a QI that takes N different values in the database, then $k = 1$ for k-anonymity.*

Proof. If $\exists qi$ in the set of *QI* of ξ that takes N different values ($|\xi(qi)| = N$), then there will be N equivalence classes in the database with only one row each of them. □

In view of the definition, if we have a database verifying k-anonymity for $k > 1$, the risk of re-identification is reduced, since there will be at least two rows of the database identical with respect to the QI, i.e. two individuals indistinguishable (with respect to the quasi-identifiers). However, this does not give us any guarantee on the sensitive attributes. Hence, the need to introduce other techniques such as those that will be presented in the following sections.

In order to calculate k for *k-anonymity* we need to know the rows which compose each equivalence classes. To do that, the pseudocode exposed in Algorithm 1 can be followed:

Algorithm 1 Gets the equivalence classes of a database

INPUT: *data*, dataset under study
INPUT: *QI*, list with the quasi-identifiers.
OUTPUT: list of equivalence classes

```
1: function GetEquivClasses(data, QI)
2:     index ← [ ]                                          ▷ empty list
3:     for q ∈ QI do
4:         D ← domain(q[data])        ▷ domain of the quasi-identifier in the data
5:         tmp ← [ ]                                        ▷ empty list
6:         for x ∈ D do
7:             tmp ← tmp + id(data[q = x])   ▷ add to tmp the index of the rows
   with value x for the quasi-identifier q
8:         end for
9:         index ← index + tmp                              ▷ add to the list
10:    end for
11:    Sort index in decreasing order of length
12:    while length of index > 1 do
13:        index ← intersect(index)        ▷ intersect is an auxiliary function that
   intersects the values in the first and second position of the index list
14:        Sort index in decreasing order of length
15:    end while
16:    ec ← index
17:    return ec
18: end function
```

In view of Algorithm 1, we have a list of length the number of equivalence classes, where at position i we have a list with the rows that compose the equivalence class i. Once we have calculated the equivalence classes, to obtain the value of k for *k-anonymity* is immediate, as it is enough to take the minimum length of all the equivalence classes.

Finally, different methods for achieving *k-anonymity* constitute the state of the art, such as *mondrian*, *incognito* and *datafly*. These algorithms will be reviewed in Section 2.8.1. It is important to note that the main idea behind these methods can be used as basis for anonymizing a database regarding the other anonymity techniques that will be explained in the reminder of this chapter. In fact, this will be detailed in Section 2.8.

2.2.2 (α,k)-anonymity

While *k-anonymity* reduces the risk of re-identification, it does not protect sensitive attributes. In Table 2.7 a simple example of a 2-anonymous database is presented. In this case, the quasi-identifiers are *Age*, *Sex* and *City*, and *Disease* (the disease the individual has), is the sensitive attribute. Although the database is 2-anonymous, it is simple to note that if we know a woman with 23 year, living in Madrid, and which is in the database, immediately we will know that the disease she has is pneumonia. As will be explained in Section 2.3, this is known as an *homogeneity attack*.

Age	Sex	City	Disease
$[20,25)$	F	Madrid	Pneumonia
$[20,25)$	F	Madrid	Pneumonia
$[25,30)$	M	Barcelona	Appendicitis
$[25,30)$	M	Barcelona	Coronary heart disease
$[25,30)$	M	Barcelona	Pneumonia

Table 2.7: Example of a k-anonymous database with $k = 2$.

To take into account the risk of sensitive attribute extraction, we introduce a new condition in addition to *k-anonymity*: for all equivalence classes, the frequency of each sensitive attribute value must not be greater than α (see [47]).

Definition 2.2.2. (α,k)-anonymity (only one sensitive attribute). Given a database ξ with only one sensitive attribute, it verifies **(α,k)-anonymity** if it verifies *k-anonymity* and the frequency of each value of the sensitive attribute is not greater than α in each equivalence class.

For Table 2.7, the minimum value we can take for α is 1 (which is also the maximum). However, if we eliminate the first equivalence class (rows 1 and 2), and keep only the second equivalence class (rows 3, 4 and 5), the minimum value of α would now be $1/3$.

Proposition 2.2.4. *Be ξ a k-anonymous database, then the value of α for (α, k)-anonymity must be in the interval* $[1/k, 1]$.

Proof. Trivial from definition. □

2.2.3 ℓ-diversity

The idea behind *ℓ-diversity* was introduced in 2007 by A. Machanavajjhala et al. in [48]. It is actually similar to the purpose of the α parameter in *(α,k)-anonymity*.

Definition 2.2.3. ℓ-diversity (only one sensitive attribute). Given a k-anonymous database ξ, if there is only one sensitive attribute, ξ verifies **ℓ-diversity** if each equivalence class has at least ℓ different values for the sensitive attribute.

In this case, instead of using the frequency of each value of the sensitive attribute as in the case of *(α,k)-anonymity*, the objective is to have a large number of different values for the sensitive attribute. In other words, the parameter ℓ measures the level of diversity of the sensitive attribute, the higher the value of ℓ, the more different values, and therefore, more protection from extraction of sensitive attributes.

Proposition 2.2.5. *Any database verifies ℓ-diversity for $\ell = 1$.*

Proof. Trivial from definition. □

Remark 2.2.1. Note that we have talked about both *ℓ-diversity* and *(α, k)-anonymity* only for the case where the database has a single sensitive attribute. However, this does not always have to be the case. Then, in case of multiple sensitive attributes, we can take two options:

1. Calculate the different properties for each of the sensitive attributes individually, and take the maximum or minimum as appropriate. For example, in the case of *ℓ-diversity*, the minimum value obtained for ℓ in each case, but for *(α,k)-anonymity*, the maximum value for α should be taken.

2. Calculate the different properties for each of the sensitive attributes individually **but modifying the quasi-identifiers**. That is, let Q be original the set of quasi-identifiers, and S the set of sensitive attributes. If we want to apply or check these properties (for example, calculate the value of ℓ for *ℓ-diversity*) for each sensitive attribute $S_i \in S$, the set of quasi-identifiers considered in each case would change (and with it the different equivalence classes), so that now it will be $Q \cup (S \setminus S_i)$.

Note that this two approaches will be further explained in Section 2.6. In order to motivate the introduction of the second approach in Remark 2.2.1, Table 2.8 is presented.

Age	Sex	City	Disease	Hospitalization date
[20, 25)	F	Madrid	Pneumonia	April 2022
[20, 25)	F	Madrid	Pneumonia	March 2022
[25, 30)	M	Barcelona	Appendicitis	March 2022
[25, 30)	M	Barcelona	Coronary heart disease	February 2022
[25, 30)	M	Barcelona	Pneumonia	March 2022

Table 2.8: Database given in Table 2.7 with a new sensitive attribute: *Hospitalization date*.

In the previous table, as in Table 2.7, the quasi-identifiers are *Age*, *Sex* and *City*, but know, in addition to the sensitive attribute *Disease*, we introduce another one sensitive attribute: *Hospitalization Date*. Then, is attacker know someone who is in this database, it would be simple (for example if it is a neighbor) to know his or her sex, age and city. Let us suppose it is the case of a man, 27 years old, from Barcelona. In view of the three quasi-identifiers, the attacker could know that the disease for which he was admitted could be appendicitis, pneumonia or a coronary heart disease. However, if the attacker also knows the date of admission, for example, if he/she knew that the hospitalization was in February 2022, then with a 100% certainty he/she would know the cause of hospitalization: coronary heart disease.

Therefore, it is important to consider approach 2 given in Remark 2.2.1 to avoid that information can be extracted from some of the sensitive attributes. It is easy to see that this is a much more restrictive approach, and moreover, much more computationally complex, because the equivalence classes will have to be recalculated for each sensitive attribute under analysis.

Definition 2.2.4. ℓ-diversity (generalizing to multiple sensitive attributes). Be ξ a k-anonymous database and SA the set of sensitive attributes. Then, ξ verifies **ℓ-diversity (by generalization)** if each equivalence class has at least l different values for each sensitive attribute S, $\forall S \in SA$.

Definition 2.2.5. ℓ-diversity (with multiple sensitive attributes). Be ξ a database with the set of quasi-identifiers $Q = \{Q_1, Q_2, \ldots, Q_n\}$, and of sensitive attributes $S = \{S_{n+1}, S_{n+2}, S_{n+m}\}$. Then, ξ is ℓ-diverse if $\forall i \in \{n+1, \ldots, n+m\}$, ξ is **ℓ-diverse** when S_i is treated as the only sensitive attribute and considering the set of quasi-identifiers: $Q \cup (S \setminus S_i)$.

From now on, as we can take either of the two ways to apply the different

anonymization techniques when there is more than one sensitive attribute (i.e. generalizing or updating the set of quasi-identifiers), in the following we will give only the definition of the different properties for a single sensitive attribute (the definition for the case of multiple attributes comes trivially from Remark 2.2.1 and the definition for one attribute in each case).

2.2.4 Entropy ℓ-diversity

The concept of entropy ℓ-diversity was first introduced by [49] in order to defend against the problem of homogeneity (Section 2.3).

Definition 2.2.6. Entropy ℓ-diversity (one sensitive attribute). Be ξ a database with only one sensitive attribute S and D the domain of S. Be EC an equivalence class of ξ, and $p(EC,s)$ the fraction of records in EC that have as sensitive attribute S the value s. Then, the entropy of an equivalence class EC with respect the sensitive attribute S is defined as follows:

$$Entropy(EC,S) = -\sum_{s\in D} p(EC,s)\log(p(EC,s))$$

The database ξ has **entropy ℓ-diversity** if for every equivalence class EC of ξ it is verified:

$$Entropy(EC,S) > \log(\ell).$$

Note: *D should be taken as the domain of S in each EC in order to be $p(EC,s) > 0$ and $\log(p(EC,s))$ be defined.*

The motivation for using this method instead of *ℓ-diversity*, comes from the fact that the entropy increases when frequencies become more uniform, allowing to capture the notion of well represented groups [48].

Proposition 2.2.6. *Be ξ a database verifying entropy ℓ-diversity for $\ell = \ell_e$. Then, ξ verifies ℓ-diversity for $\ell \geq \ell_e$.*

Proof. For simplicity, we will proof this proposition by assuming that ξ has only one single sensible attribute S (generalization follows naturally).

If ξ verifies *entropy ℓ-diversity* for $\ell = \ell_e$, then for every equivalence class EC of ξ, with D the domain of the sensitive attribute S in the equivalence class:

$$H(EC) = -\sum_{s\in D} p(EC,s)\log(p(EC,s)) > \log(\ell_e)$$

Note that as $p(EC,s)$ is the fraction of records in EC that have the value s as sensitive attribute for S, then $p(EC,s) \in (0,1]$, $\log(p(EC,s)) < 0$ and $H(EC) \geq 0$.

Let us proof the proposition by distinguishing different cases:

- If $H(EC) = 0$, then $\ell_e < 1$ and by Proposition 2.2.5, ξ verifies ℓ-*diversity* for $\ell \geq 1$, so $\ell > \ell_e$. Also, this case can only be obtained if there is an equivalence class with $|D| = 1$, and then $\ell = 1$ for ℓ-*diversity*.
- If $\ell_e = 1$, using again Proposition 2.2.5 $\ell \geq 1 = \ell_e$.
- If $\ell_e = 2$, suppose by reduction to the absurd that $\ell < 2 = \ell_e$ for ℓ-*diversity*, so $\ell = 1$. But if $\ell = 1$, that implies that $\exists EC$ an equivalence class with $|D| = 1 \Longrightarrow H(EC) = 0 > \log(\ell_e) = \log(2)$ $\perp$.
- If $\ell_e = n$, suppose by reduction to the absurd that $\ell < n = \ell_e$ for ℓ-*diversity*. Then $\exists EC$ an equivalence class with $|D| \leq n-1$. Suppose $|D| = n-1$, then, using that $H(EC)$ is monotonically decreasing[1] in $p(EC,s)$, the maximum value which can be reached for $H(EC)$ is achieved for the minimum value of $p(EC,s)$. Then $\log(n-1) \geq H(EC) > \log(\ell_e) = \log(n)$ $\perp$.

  Note that for $|D| < n-1$, the contradiction is reached in the same way.

□

In view of the previous Proposition, we can conclude (as expected) that *entropy* ℓ-*diversity* is more restrictive than ℓ-*diversity*.

2.2.5 Recursive (c,ℓ)-diversity

An extension of the ℓ-*diversity* is now presented: *recursive (c,ℓ)-diversity*. This technique aims to ensure that the most common value of a sensitive attribute does not appear excessively frequently, while for the less common values of that sensitive attribute, it is intended that they do not appear too infrequently.

Definition 2.2.7. Recursive (c,ℓ)-diversity (one sensitive attribute). Be ξ a database with only one sensitive attribute S verifying ℓ-diversity. Be n the number of values of the sensitive attribute S in an equivalence class, and r_j ($j \in \{1,\ldots,n\}$) the number of times that the j-th most frequent value of S appears in the equivalence class EC. Then, EC has recursive **(c,ℓ)-diversity** for the sensitive attribute S if $r_1 < c(r_l + r_{l+1} + \ldots + r_n)$ [48].

A database has recursive ***(c,ℓ)-diversity*** if every EC has recursive ***(c,ℓ)-diversity***.

Remark 2.2.2. Some points that can be extracted from the definition:

[1] For a proof of the monotonicity of the entropy, see for instance [50].

- If $\ell = 1$, then $c > 1$.
- If $\ell = n$ then $c > r_1/r_n$.
- We are interested in the smallest value of c for which the inequality is satisfied to be large.
- The main strength of this technique is that if a value s' of the sensitive attribute S is removed from an equivalence class which verifies *(c,ℓ)-diversity*, it is preserved *(c,ℓ-1)-diversity*.

2.2.6 *t*-closeness

As an alternative to *ℓ-diversity*, in order to protect the extraction of information on sensitive attributes, the concept of *t-closeness* is proposed. In fact, it can be seen as an improved version of the *ℓ-diversity* model.

Definition 2.2.8. t-closeness (one sensitive attribute). Given a database ξ with one sensitive attribute S, an equivalence class of ξ has ***t-closeness*** if the distribution of the values $s \in S$ are no more than a distance t apart from the distribution of the sensitive attribute in the original database. Then, ξ has **t-closeness** iff all the equivalence classes verify t-closeness.

Note that in order to find the distinction between two distributions, many techniques can be applied. However, we propose to use the following two, which are very common in the literature (see [51]), according to the type of attribute:

- **Numerical attributes.** Calculate the *Earth Mover's distance (EMD)* between the two distributions using the ordered distance. Be n the number of different values for the sensitive attribute S. As we are considering numerical attributes, be $\mathscr{D} = \{d_1, \dots, d_n\}$ the domain of S with v_1 the lower value and v_n the bigger one. Be $\mathscr{P} = (p_1, \dots, p_n)$ the overall distribution of each value of the sensitive attribute in the database ξ and $\mathscr{Q} = (q_1, \dots, q_n)$ the distribution in an equivalence class EC of ξ. Then, be $r_i = q_i - p_i \ \forall i \in \{1, \dots, n\}$. The distance between $\mathscr{Q}$ and $\mathscr{P}$ can be defined as:
$$D(\mathscr{Q}, \mathscr{P}) = \frac{1}{n-1} \sum_{i=1}^{n} \left| \sum_{j=1}^{i} r_j \right|.$$
- **Categorical attributes.** In this case, as a total order often does not exist, the *Equal Distance* can be applied, that is:
$$D(\mathscr{Q}, \mathscr{P}) = \frac{1}{2} \sum_{i=1}^{n} |r_i|,$$

with $\mathscr{Q}$, $\mathscr{P}$ and r_i defined as in the case of numerical attributes.

With this method, we want the distribution of a sensitive attribute over the entire database to be close to the distribution in any equivalence class. That is, the distance between these two distributions is required to be less than a threshold t. This allows to limit the amount of information about a specific individual that an attacker can infer [51].

The motivation for including this technique instead of using *ℓ-diversity* (or one of its two versions) comes from the need to avoid attribute disclosure and to prevent similarity and skewness attacks (Section 2.3). That is possible because difference between the distributions of the sensitive attribute values in each equivalence class, and in the database in general, is controlled with the parameter t. In addition, in the case of numerical attributes, by using the ordered distance, similarity attacks can also be controlled. This does not occur when the *ℓ-diversity* technique is used, as in that case only the frequency of a SA value in each equivalence class is measured, without comparing it with the complete table.

2.2.7 Basic β-likeness and enhanced β-likeness

Now, two similar techniques are proposed (introduced in 2012 by J. Cao and P. Karras in [52]), both again with the aim of protecting against skewness and similarity attacks (Section 2.3): *basic β-likeness* and its stronger version, *enhanced β-likeness*.

Definition 2.2.9. $\boldsymbol{\beta}$-likeness (one sensitive attribute). Given a database ξ with only one a sensitive attribute S, be $\mathscr{D} = \{d_1, \ldots, d_n\}$ the domain of S, and $\mathscr{P} = \{p_1, \ldots, p_n\}$ the distribution of S in the database ξ. An equivalence class EC with distribution of S given by $\mathscr{Q} = \{q_1, \ldots, q_n\}$ satisfies **$\boldsymbol{\beta}$-likeness** iff $max\{D(p_i, q_i) : p_i \in P \wedge p_i < q_i\} \leq \beta$, with $\beta > 0$, and being $D(p_i, q_i)$ the distance between p_i and q_i. Then, ξ satisfies **$\boldsymbol{\beta}$-likeness** if all the equivalence classes of ξ satisfy it.

Note that, among others, the relative distance function can be considered, that is: $D(p_i, q_i) = \frac{q_i - p_i}{p_i}$.

Remark 2.2.3. Be ξ a database with n different values for a sensitive value S. Note that if the previous definition is fulfilled considering the relative distance function, then $q_i < (\beta + 1)p_i \ \forall i \in \{1, \ldots, n\}$. As, by definition $q_i \leq 1$, we are only interested in considering this bound when $(\beta + 1)p_i < 1$, that is $p_i < \frac{1}{\beta+1}$. For those cases in which the previous condition is not verified, is possible to infer information about the values of S (those which are more frequent).

That is the motivation for introducing the definition of *enhanced β-likeness*, in order to provide more robust privacy.

Definition 2.2.10. Enhanced β-likeness (one sensitive attribute). Given a database ξ with only one sensitive attribute S, be $\mathscr{D} = \{d_1, \dots, d_n\}$ the domain of S, and $\mathscr{P} = \{p_1, \dots, p_n\}$ the distribution of S in the database ξ. An equivalence class EC with distribution of S given by $\mathscr{Q} = \{q_1, \dots, q_n\}$ satisfies **enhanced β-likeness** iff $D(p_i, q_i) = \frac{q_i - p_i}{p_i} \leq min\{\beta, -\log(p_i)\}$, $\forall q_i \in \mathscr{Q}$, with $\beta > 0$ [52]. Then, ξ satisfies **enhanced β-likeness** if all the equivalence classes of ξ satisfy it.

Considering the previous definition, note that if $p_i \leq e^{\beta}$, $q_i \leq p_i(1+\beta)$ (we are paying attention to those values less frequent in the database), and if $p_i \geq e^{\beta}$, then we are paying attention to the most frequent values in the database, because $q_i \leq p_i(1 - \log(p_i)) < p_i(1+\beta)$.

As in the case of *t-closeness*, this two methods are used to control the distance between the distribution of a sensitive attribute in an equivalence class and in the whole database, so can be specially usefully to avoid skewness attacks.

2.2.8 δ-disclosure privacy

To conclude this battery of techniques, let us mention *δ-disclosure privacy* (defined in 2008 by J. Brickell and V. Shmatikov in [53]), which, as we can guess from its name, tries to prevent the disclosure of sensitive attributes.

Definition 2.2.11. δ-disclosure privacy (one sensitive attribute). Given a database ξ with only one sensitive attribute S, an equivalence class EC of ξ is said to be **δ-disclosure private** if, $\forall s \in S$:

$$\left| \log\left(\frac{p(EC, s)}{p(\xi, s)} \right) \right| < \delta,$$

being $p(EC, s)$ the frequency of s in EC, and $p(\xi, s)$ the frequency of s in ξ. The database ξ is **δ-disclosure private** if every equivalence class in ξ is **δ-disclosure private**.

Remark 2.2.4. From definition, it can be noted that the is an upper and a lower bound for the distribution of each value s of the sensitive attribute S in each equivalence class EC: $e^{-\delta} p(\xi, s) < p(EC, s) < e^{\delta} p(\xi, s)$. This is an improvement over the characteristics of *β-likeness* methods, where only an upper bound was available.

As in the two previous subsections, this techniques allows to control the difference between the frequency of a values of a sensitive attribute in an equivalence class and in the whole database, so it is really interesting its use in order to prevent from skewness or even inference attacks (Section 2.3).

2.3 Common Attacks on Anonymized Databases

The following are some of the most common attacks on databases that have been anonymized. As the simplest way to understand these attacks is through examples, Table 2.9 is presented for reference for the examples presented in this section:

Age	Sex	Native country	Study level	Salary (€)
$(20,30]$	F	Italy	3	$\leq 30k$
$(20,30]$	F	Italy	6	$\geq 30k$
$(30,40]$	M	Spain	4	$\leq 30k$
$(30,40]$	M	Spain	6	$\leq 30k$
$(30,40]$	M	Spain	6	$\leq 30k$
$(40,50]$	F	Poland	7	$\geq 30k$
$(40,50]$	F	Poland	6	$\geq 30k$
$(40,50]$	F	Germany	6	$\geq 30k$
$(40,50]$	F	Germany	6	$\geq 30k$
$(50,60]$	M	Slovakia	4	$\leq 30k$
$(50,60]$	M	Slovakia	6	$\geq 30k$

Table 2.9: Simulated database used as example. Five equivalence classes are separated. Set of QIs: *Age*, *Sex* and *Native country*, and SA: *Study level*[2]and *Salary*.

- **Re-identification attacks:** this kind of attack may occur when the anonymization process is reversed. This is one of the simplest attacks on anonymized databases

  Example 2.3.1. Let us see a simple example of a re-identification attack. If an attackers knows a woman in the database (given in Table 2.9) who is 25 years old, it is immediate to infer that she must be the first or second person in the table (and we will know at least her native country).

- **Linkage attacks:** it consists of combining at least two databases that have previously been anonymized in order to reveal the identity of one of the individuals in the database.

[2]See the International Standard Classification of Education (ISCED) for details on the meaning of this attribute.

Example 2.3.2. The most simple example of a linkage attack is the one presented in the introduction about the *Netflix prize*, where the information given by the *Netflix* database, was combined with that of the *IMDb* to extract information about the users represented in each database.

- **Background Knowledge Attack:** in this kind of attack, the adversary has some pieces of auxiliary information (about the quasi-identifiers and the sensitive attributes), with can be used to discover sensitive information.

  Example 2.3.3. Suppose an attacker knows Alice, who is a woman from Italy, aged 25 years old. Furthermore, let us assume that although the attacker does not know the exact level of education of Alice, knows that it at least encompasses post-secondary non-tertiary education (level 4). Then, with this auxiliary information, looking at Table 2.9 he/she can conclude that Alice's education level is 6 and that her salary is above 30000€.

- **Homogeneity Attack:** in this attack, the value of the sensitive attributes can be known only by using the quasi-identifiers. For example, it occurs when all the values for a sensitive attribute in an equivalence class are identical.

  Example 2.3.4. If an attacker knows Bob, a man from Spain, aged 35, and who is in the database, he/she can conclude that his salary is under 30000€. Moreover, if he knows Charlotte, a woman from Germany who is also in the database, he/she can infer that her educational level is 6 and that her salary is above 30000€.

- **Inference attacks:** consists of applying data mining techniques to obtain knowledge about a database.

  Example 2.3.5. As an example of inference attack on Table 2.9, statistical and data mining techniques by means of ML/DL models could be used to infer information about salaries once the quasi-identifiers are known (and additionally the level of education) and make assumptions about different individuals who are not in the database.

- **Similarity attack:** may occur when the values of a sensitive attribute in an equivalence class are different but semantically similar, then an adversary can learn important information.

  Example 2.3.6. Suppose an attacker knows Mary, a woman aged 42. Then he/she can infer, that the level of studies of Mary are of Bachelor or Master, which provides a similar amount of information about her background.

- **Skewness attack:** this kind of attack can occur when a sensitive attribute is not really frequent in the whole database, but it is extremely frequent in a concrete equivalence class. Also, when a value is frequent in the database but infrequent in the whole population.

  Example 2.3.7. Note that in Table 2.9, the value 6 (Bachelor's or equivalent level) for the sensitive attribute *Study level* has a frequency of 63.64%, and suppose that the frequency of this attribute for the whole population is only about 25% [3]. This allows us to deduce that it is very common for someone in this database to have a value of 6 for the sensitive attribute *Study level*.

Table 2.10 summarizes the anonimziation techniques introduced previously and the kind of attacks that they can prevent. This illustrates the need to apply different techniques and not just the most common ones (such as *k-anonymity* or *ℓ-diversity*), as each is suitable for dealing with different types of attacks. Although one technique can prevent from several attacks, Table 2.10 shows the main ones. In addition, some techniques can provide stricter measures of the privacy than others regarding the same kind of attacks, as is the case for *entropy ℓ-diversity* and *recursive (c,ℓ)-diversity* with respect to *ℓ-diversity*.

	Principal attack which prevents						
Technique	**Linkage**	**Re-identification**	**Homogeneity**	**Background**	**Skewness**	**Similarity**	**Inference**
k-anonymity	✓	✓					
(α,k)-anonymity	✓	✓	✓				
ℓ-diversity			✓	✓			
Entropy ℓ-diversity			✓	✓			
Recursive (c,ℓ)-diversity			✓	✓			
t-closeness					✓	✓	
Basic β-likeness					✓		
Enhanced β-likeness					✓		
δ-disclosure privacy					✓		✓

Table 2.10: Anonymization techniques and principal attacks that prevent (among others). Extracted from [3].

Finally, Figure 2.1 shows the workflow to be followed in order to choose the proper anonymity technique based on the objective to be achieved in relation to data privacy, the attacks to be prevented and the level of strictness of the privacy approach. This scheme can be used as reference for selecting the techniques to be applied taking into account the objective, the desired level of protection and the potential attacks to be protected against.

[3] According to https://data.worldbank.org/indicator/SE.TER.CUAT.BA.ZS, the percentage of people in Spain verifying this condition, is about 23.58%.

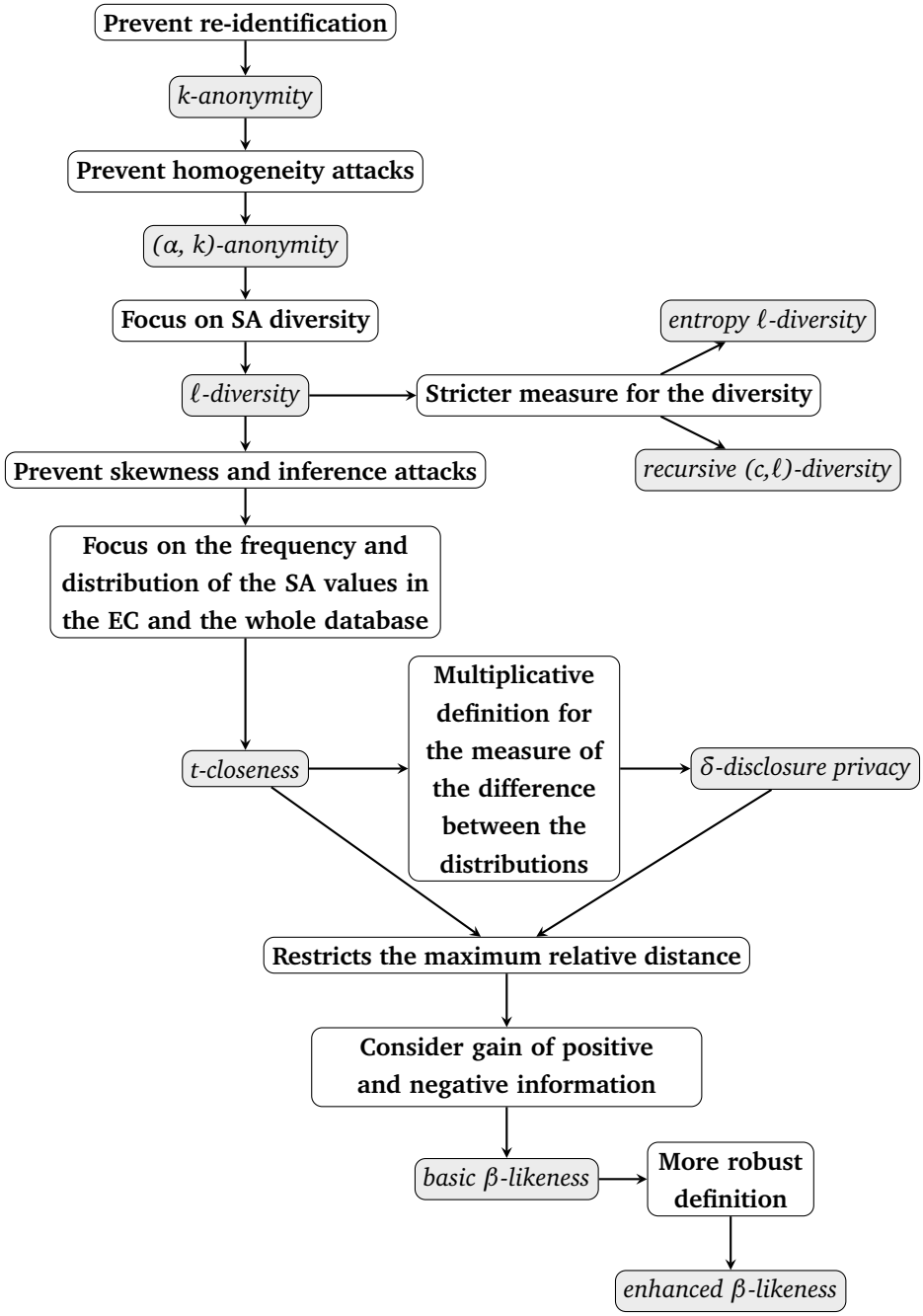


Figure 2.1: Guidelines for selecting the anonymization technique to be applied according to the data privacy objective to be achieved, the attacks to be prevented and the level of strictness. Extracted from [5].

2.4 Privacy-Utility Trade-Off Metrics

As stated in the previous section, it is important to apply different anonymization techniques to prevent different types of attacks and thus ensure data privacy. However, it is equally critical to maintain a balance between data utility and privacy. If we apply some privacy levels via anonymization techniques that are too strict, we will be reducing the usefulness of the data for further analysis, for example by means of data mining and machine learning techniques. Therefore, it is key to develop metrics in order to quantify the loss of information derived from anonymization.

In the following some metrics frequently used in the literature for evaluating the balance between privacy and utility are presented:

- **Average equivalence class size**: Be ξ the original database, and ξ_k the one anonymized applying a certain value of k for *k-anonymity*. Be *ECs* the resulting set of equivalence classes. This metric is defined as given in the following equation.

$$C_{AVG_k}(\xi_k) = \frac{|\xi|}{k|ECs|}. \tag{2.1}$$

  In the case in which record suppression is allowed with a limit of 100% instead of analyzing the number of equivalence classes as a function of the number of initial records (as exposed previously [54]), the number of records resulting from anonymization in each case will be considered. This is shown in Equation 2.2, in which *ECs* is the set of equivalence classes.

$$C^*_{AVG_k}(\xi_k) = \frac{|\xi_k|}{k|ECs|}. \tag{2.2}$$

- **Classification metric (CM):** be N the number of rows of the original dataset. This metric is defined as follows (see [55]): be ξ the original (raw) database and ξ^* the anonymized one:

$$CM(\xi, \xi^*) = \frac{1}{N}\sum_{i=1}^{N} penalty(r_i), \tag{2.3}$$

  where $penalty(r_i) = 1$ if the row r_i has been deleted or if its associated label (i.e. SA) takes a value other than the majority value in the equivalence class to which it belongs, and 0 otherwise, $\forall i \in \{1, \ldots, N\}$ with N the number of records in the original database.

- **Discernibly Metric (DM):** with this metric we aim to measure the level of indistinguishability of a record regarding others. In this way, we can assign a penalty to each record equal to the size of the equivalence class to which it belongs. If the row has been deleted, we assign a penalty equal to the size of the table. The definition of the classical equation for this metrics is given as follows [54]: let $|\xi_k|$ be the anonymized database for a value k of *k-anonymity* and ξ be the original database. Let $|EC|$ be the size of the equivalence class EC and $|\xi|$ the size of the original database.

$$DM(|\xi_k|) = \sum_{\forall EC s.t. |EC| \geq k} |EC|^2 + \sum_{\forall EC s.t. |EC| < k} |\xi||EC|. \tag{2.4}$$

  A version of the classical discernibility metric can be considered in order to penalize the deletion of records (DM^*). As can be seen from the above equation, all equivalence classes will belong to the first addend of the Equation 2.4 except for the deleted entries (by the definition of *k-anonymity*). In this sense, we define DM^* as follows: let EC_i be the i-th equivalence class, and let N_{ec} be the number of equivalence classes of the ξ_k database:

$$DM^*(|\xi_k|) = \sum_{i=1}^{N_{ec}} |EC_i|^2 + |\xi|^2 - |\xi||\xi_k|. \tag{2.5}$$

- **Generalized Information Loss (GILoss):** finally, this metric seeks to capture the loss of information derived from generalizing a specific attribute of the database. In particular, we present below a version of this metric given in [54]. For each attribute or quasi-identifier subject to generalization qi_i (assume n_{qi} the number of available quasi-identifiers/attributes), let L_i and U_i be the lower and upper bounds respectively. Each attribute qi_i generalizes to an interval ij with lower and upper bounds L_{ij} and U_{ij}. Thus, let ξ be the original database and ξ^* be the anonymity database:

$$GILoss(\xi^*) = \frac{1}{n_{qi}|\xi|} \sum_{i=1}^{n_{qi}} \sum_{j=1}^{|\xi|} \frac{U_{ij} - L_{ij}}{U_i - U_j}. \tag{2.6}$$

  In this case we assume that the attributes that undergo generalization are only the quasi-identifiers, but there may be cases in which other variables such as the sensitive attributes are also generalized.

2.5 Disclosure and Re-Identification Risk Control

In this Section we are going to review some metrics to evaluate and measure the disclosure and re-identification risks that may arise when working with tabular data. Then, let us start by formalizing the definitions of disclosure and re-identification risk:

Definition 2.5.1. Disclosure risk. According to J. Domingo-Ferrer in [56], when dealing with statistical disclosure risk, the **disclosure risk** of an attacker being able to use the anonymized or sanitized dataset $\mathscr{D}'$ to obtain sensitive information about an individual from those included in the original dataset $\mathscr{D}$.

Definition 2.5.2. Re-identification risk. The **re-identification** consists of using the anonymized dataset $\mathscr{D}'$ to find properties that could increase the risk of individuals within $\mathscr{D}$ from being identified.

Firstly, concerning disclosure risk evaluation, we can consider the following metrics:

- **Entropy-Based Metrics for disclosure risk**: These kind of metrics can be used to quantify the uncertainty or unpredictability associated with the identification of individuals or disclosure of sensitive information in a dataset. We can also consider the risk of disclosing information about a QI or a SA given certain attribute in the database. Specifically, we can consider the following three metrics:

  - **Shannon Entropy**. This metric is widely used in the field of information theory, and is employed to measure the uncertainty of an information source. Given a SA X, S_X the set of possible values that X can take in a database ξ and $x = (x_1, \dots, x_{|S_X|})$ the possible values, each of them with an associated probability $\mathbb{P}(x_i)\ \forall i \in \{1, \dots, |S_X|\}$. The Shannon entropy [57] of X $(H(X))$ is given by:

$$H(X) = -\sum_{i=1}^{|S_X|} \mathbb{P}(x_i) \log(\mathbb{P}(x_i)) \tag{2.7}$$

  - **Conditional entropy**. Again, it is a metric widely used in information theory. Specifically, in this case we have two variables and the objective is to quantify the amount of information needed to describe the outcome of a random variable X given that the value of another random variable Y is known. We can consider X a SA and Y a quasi-identifier

in a database, and this measure of entropy will allow us to infer the risk of disclosing information about a sensitive attribute X by relying on a quasi-identifier Y.

Be X a QI in ξ and S_X as defined for the Shannon Entropy. Be a QI Y, S_Y the set of possible values that Y can take in a database ξ and $y = (y_1, \dots, y_{|S_Y|})$ the possible values, each of them with an associated probability $\mathbb{P}(y_i)$ $\forall i \in \{1, \dots, |S_Y|\}$. The conditional entropy [57] of X given Y (denoted as $H(X|Y)$) is given by:

$$H(X|Y) = -\sum_{i=1}^{|S_X|}\sum_{j=1}^{|S_Y|}\mathbb{P}(x_i, y_j)\log(\mathbb{P}(x_i|y_j)) \tag{2.8}$$

– **Mutual information**. It measures the mutual dependence of the two random variables. We can use it to know the dependence between two quasi-identifiers or between a sensitive attribute and a quasi-identifier and thus analyze the information that can be derived from one based on the other.

Given the Shannon entropy and the conditional entropy defined above, the sensitive attribute X and the quasi-identifier Y, the mutual information $I(X,Y)$ [57] is given by:

$$I(X,Y) = H(X) + H(Y) - H(X|Y) \geq 0 \tag{2.9}$$

- **Sensitivity rules for a priori risk assessment**. As stated in [58], the assessment of disclosure risk in tabular data is generally performed a priori, i.e. before the tables are protected or even anonymized. The standard approach is to use a sensitivity rule to determine whether certain cell of the table is sensitive to a disclosure attacks and should be protected. Two rules can be considered to decide whether a given cell is sensitive and may pose a disclosure risk: *(n, k)-dominance* and *pq-rule*. The former (now in disuse [58]) was used in the case of magnitude tables with records associated to individuals (e.g. for economic databases, and can be used to identify cells with high contributions), while the latter in cases of tabular data in general (to assess the risk of individual inference).

  Specifically, the latter (prior-post rule) [58] is based on the concept of uncertainty before and after the publication of the data. It assesses whether the publication of a cell significantly reduces the uncertainty about an individual's contribution to the database, which could lead to the disclosure of

sensitive information. A cell is considered sensitive if once the table is public one can estimate the contribution of an individual with a margin of error of less than p, while q is the margin of error that an individual may have before the publication of the table when estimating the contribution of other participants.

Secondly, two metrics of interest to track in relation to re-identification risk control together with some considerations on the equivalence classes generated once the data are anonymized, are presented in the following:

- ***k*-map**. This method is similar to the idea of *k-anonymity* but it uses the assumption that an attacker can have an identification database ($\overline{\xi}$) of a subset S of the population represented in the original ξ database that we want to protect [59]. This assumption makes it less used in practice.

  Specifically, it is assumed that the one in charge of anonymizing the database to be protected ξ, has access to this identification table ($\overline{\xi}$) and can anonymize it using *k-anonymity* in order to get $\overline{\xi}^*$. With *k-map* it is established that each record in the anonymized version of the database ξ (ξ^*) is similar to at least k records of $\overline{\xi}^*$.

- **δ-presence**. In this case it is again necessary to have an auxiliary population or identification table that allows to calculate the probability that an individual from this identification table is contained in the database to be protected. Thus, if this probability is in the interval $[\delta_{min}, \delta_{max}]$, it is said that the database ξ is $(\delta_{min}, \delta_{max})$-present [60]. That is, given an anonymized version of the initial database ξ (ξ^*) and an auxiliary identification table $\overline{\xi}$, we have that ξ is $(\delta_{min}, \delta_{max})$-present if and only if:

$$\delta_{min} \leq \mathbb{P}(x \in \xi | \xi^*) \leq \delta_{max}, \quad \forall x \in \overline{\xi}. \tag{2.10}$$

- **Size of the equivalence classes**. It is important to note that the size of the equivalence classes generated during the anonymization process is also an indicator that allows to quantify the risk of re-identification.

  Be N_{EC} be the number of equivalence classes. Each equivalence class will have $|EC_i|$ records with $i \in \{1, \ldots, N_{EC}\}$, being those with the lower value of EC_i the ones that will have more risk of re-identification information. Thus, the maximum probability of an attacker identifying an individual in the anonymized database ξ^* (p_{id}) will be given by:

$$p_{id} = \frac{1}{\max_{i \in \{1,\dots,N_{EC}\}}(|EC_i|)}. \tag{2.11}$$

Keeping track of these metrics helps us to monitor the possible risks that the data may have once anonymized (or before anonymizing them). Likewise, as presented in Table 2.10, it is important to have a control of the techniques exposed in Section 2.2, because each one of them can contribute to protect against different types of attacks.

2.6 *pyCANON*: a Python Library to Check the Level of Anonymity of a Dataset

The Python library pyCANON was presented in the following paper published in the journal Scientific Data from the Nature Publishing Group: ***Sáinz-Pardo Díaz, J.*** *& López García, Á. (2022). A Python library to check the level of anonymity of a dataset. Scientific Data, 9(1), 785. [3].*

In this section we present the implementation of *pyCANON* [61] [62], a Python library and command line interface (CLI) to check the level of anonymity of a dataset through the nine anonymization techniques exposed previously (see [3]). The main idea behind it is to provide researchers, and in general anyone who wants to publish a dataset in open access or to share it with others, with a prior knowledge of the level of anonymization of their data. This will provide insights about the possible risks to which these information would be exposed, allowing to verify the impact and their resistance to different attacks.

2.6.1 Main Functionalities

Given a dataset, a list of quasi-identifiers and a list of sensitive-attributes, with *pyCANON* the users can check for which parameters the different anonymity techniques are satisfied. The nine techniques implemented in version 1.0.5 are: *k-anonymity*, *(α,k)-anonymity*, *ℓ-diversity*, *recursive (c,ℓ)-diversity*, *entropy ℓ-diversity*, *t-closeness*, *basic β-likeness*, *enhanced β-likeness* and *δ-disclosure privacy*. Knowing the level of anonymity of a dataset according to these techniques can enable data to be published or shared with certain privacy and security guarantees, as each technique can help to prevent a certain type of attack.

With this library the users are provided with a set of functions to calculate the anonymity level of a dataset, both for the case of one sensitive attribute and for mul-

tiple sensitive attributes. In the second case, as briefly introduced in Section 2.2.3 the results can be obtained through two approaches:

- **Harmonization of quasi-identifiers:** for each SA each of the properties is checked, and the parameter that is satisfied for all of them is kept (e.g. for *ℓ-diversity* ℓ will be the smallest value obtained once computed for each SA, while the value of α for *(α,k)-anonymity* will be the largest value of α of those obtained for each SA).
- **Updating the quasi-identifiers:** the set of QIs is updated according to the SA to be analyzed. Be Q initial the set of QIs, and S the set of SAs, $\forall S_i \in S$, the set of QIs considered in each case would change so that it will be $Q \cup (S \setminus S_i)$. The idea of introducing this second approach is that in certain cases an attacker could know some of the SAs, thus acting as QIs that would allow inferring information about the rest of the sensitive information.

In addition, *pyCANON* allows to know how different properties scale as a function of others. For example, in Figure 2.2 we can see for an specific example the evolution of t and $log(\beta)$ when varying k (for *t-closeness*, *basic β-likeness* and *k-anonymity* respectively).

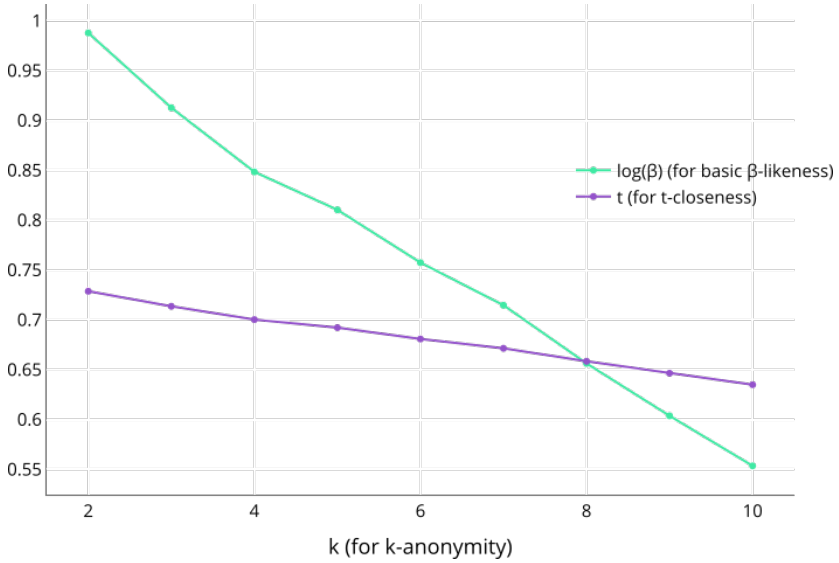


Figure 2.2: Example: evolution of t and β (in logarithmic scale) when varying k (for *t-closeness*, *basic β-likeness* and *k-anonymity* respectively). Extracted from [3].

This can help to give users an idea of which values of k are more convenient

to apply to ensure that we are already protecting the data according to other techniques, and thus against other attacks. Likewise, *pyCANON* allows to know in advance the verified parameter for some methods without the need to apply them previously. That is, once anonymized for *k-anonymity*, the value for another method (e.g. t for *t-closeness*) can be checked, and if it is satisfactory for the intended use of the anonymized data, it will not be necessary to anonymize it with respect to that method.

Furthermore, *pyCANON* can also be used to detect failures in the anonymization process. Suppose that the user want to anonymize some data with respect to a set of QI verifying *k-anonymity* with $k = 3$. An error may have occurred because the user forgot to enter one of the QIs. With *pyCANON*, he/she can check the value of *k-anonymity* that the data meet, and thus detect a possible error.

2.6.2 Usage Examples

Due to the simplicity of use, no previous knowledge of Python or anonymity techniques is required for checking the level of anonymity for a given dataset using this tool. The following code shows an example of how to calculate the parameters related to the nine anonymization techniques implemented in *pyCANON* and mentioned above for an anonymized version of the adult dataset (more details about the dataset in Section 2.7.1), simply by passing the data as a *pandas* dataframe, a list with the set of quasi-identifiers, and another with the set of sensitive attributes (in this case there is only one single sensitive attribute). Note that in all the functions there is a field `gen`, which by default is true, and that manages the case of multiple sensitive attributed (it applies the idea of harmonization of QIs by default), but in this case we do not need to include it.

```
file_name = "adult_anonymized_10.csv"
data = pd.read_csv(file_name)
quasi_ident = ["age", "education", "occupation", "relationship",
               "sex", "native-country"]
sens_att = ["salary-class"]

k_anon = anonymity.k_anonymity(
    data, quasi_ident
)
alpha, _ = anonymity.alpha_k_anonymity(
    data, quasi_ident, sens_att
)
l_div = anonymity.l_diversity(
    data, quasi_ident, sens_att
)
entropy_l = anonymity.entropy_l_diversity(
    data, quasi_ident, sens_att
```

```
)
c_div, _ = anonymity.recursive_c_l_diversity(
    data, quasi_ident, sens_att
)
t_clos = anonymity.t_closeness(
    data, quasi_ident, sens_att
)

bas_beta = anonymity.basic_beta_likeness(
    data, quasi_ident, sens_att
)
en_beta = anonymity.enhanced_beta_likeness(
    data, quasi_ident, sens_att)

delta_disc = anonymity.delta_disclosure(
    data, quasi_ident, sens_att
)
```

Example Code 2.1: Use example: calculating the parameters for the nine anonymity techniques for an anonymized version of the *adult dataset* using the Python library *pyCANON*.

In addition to obtaining the parameters of the different methods as indicated above, custom reports can be obtained in JSON and PDF formats for simplicity and ease for the user. In this sense, the `pycanon.report` package is a key functionality of this library. The purpose of this package is to generate a report with the anonymization level of the data file entered, checking all the techniques mentioned previously. This report can be generated in a JSON file, in a PDF file, or just by displaying it on the screen. An example for an anonymized version of the adult dataset (as in Code Example 2.1) of how to generate the report in PDF format is shown below.

```
import pandas as pd
from pycanon.report import pdf

file_name = "adult_anonymized_10.csv"
data = pd.read_csv(file_name)
quasi_ident = ["age", "education", "occupation", "relationship",
               "sex", "native-country"]
sens_att = ["salary-class"]
file_pdf = "report_adult.pdf"
pdf.get_pdf_report(
    data, quasi_ident, sens_att, file_pdf = file_pdf
)
```

Example Code 2.2: Use example: generating an anonymity report in a PDF file using the Python library *pyCANON*.

An screenshot of the report obtained when executing the Example Code 2.2 is shown in Figure 2.3. Note that the value of c for *recursive (c,ℓ)-diversity* is NaN as it can not be calculated for this example.

Jan 14 2025 10:18:09

Quasi-identifiers: ['age', 'education', 'occupation', 'relationship', 'sex', 'native-country']
Sensitive attribute(s): ['salary-class']

Anonymity technique	Value(s)
k-anonymity	k = 10
(α,k)-anonymity	α = 1.0 and k = 10
l-diversity	l = 1
Entropy l-diversity	l = 1
(c,l)-diversity	c = nan and l = 1
Basic β-likeness	β = 1.739309827456864
Enhanced β-likeness	β = 1.0077060008814558
t-closeness	t = 0.634944543338354
δ-disclosure privacy	δ = 2.803878920792833

Figure 2.3: Example: part of the report obtained when executing the code given in Example Code 2.2.

2.6.3 Further Functionalities

Although the main use of the library is to check the level of anonymity of a dataset, a functionality has been incorporated to analyze the utility of the data entered. The `pycanon.utility` package is available for this purpose.

First, the functions `sizes_ec` and `stats_quasi_ident` from the package `pycanon.utility` allow to calculate the statistics associated to the equivalence classes (e.g. minimum and maximum length of each equivalence class, etc.) and to calculate statistics associated to a given quasi-identifier (e.g. frequencies of each value of the QI) respectively.

Additionally, three metrics have been implemented to measure the usefulness of the data. Specifically these are the average equivalence class size, the classification metric and the discernibly metric, presented in Section 2.4. It is important to highlight the following: in the first one the two proposed versions are considered depending on whether or not records have been suppressed to obtain the anonymized dataset (the user should know this information), and the metric is calculated accordingly as shown in Equations 2.1 and 2.2. Regarding the classification

metric and the discernibly metric, in both cases it will be necessary to enter both the original and the anonymized dataset, and in the version of DM^* proposed in Equation 2.5 has been considered in order to penalize the suppression of records.

2.6.4 Summary

In this section we have presented the implementation of *pyCANON*, an open source Python library that allows to check the level of anonymity of a dataset given a list of quasi-identifiers and a list of sensitive attributes, allowing also to obtain visual reports in the form of PDF, JSON or displayed on the screen. This library can also be used to see how different properties scale as a function of others, thus allowing to apply an anonymization technique and to analyze which others are checked and with respect to which parameters.

Furthermore, in addition to knowing the level of privacy of the database before publication or before using them in an inference process (for example through the use of machine learning techniques) it is of critical importance to know the usefulness of the data after anonymization. For this purpose, different functions have been implemented to know the utility of the anonymized data, as well as to analyze statistics related to the generated equivalence classes, as detailed in the previous section.

The development of the library is fully completed and it is maintained in a GitHub repository [61] (synchronized with Zenodo for preservation purposes [62]), but the integration of new anonymization and utility methods and techniques will be pursued.

2.7 Impact of Anonymization Techniques in Machine Learning Models Training

Part of the study presented in this section has been published in the Proceedings of the 2023 IEEE International Conference on Cyber Security and Resilience (CSR) as: ***Sáinz-Pardo Díaz, J.*** *& López García, Á. (2023) "Comparison of machine learning models applied on anonymized data with different techniques", 2023 IEEE International Conference on Cyber Security and Resilience (CSR), Venice, Italy, 2023, pp. 618-623 [63].*

As we have explored through this chapter, when working with sensitive data, especially when these data will serve as the basis for ML or DL models, it is important to process it appropriately with a focus on protecting the privacy of the users repre-

sented in the data. Equally important is also to keep in mind that a too strict level of anonymity can compromise data usefulness for processing and inference. This may occur either because it has been necessary to remove a large number of records to reach the desired level of anonymization, or because the hierarchies levels applied on the quasi-identifiers dilute their initial information. Then, as exposed in Section 2.4 it is necessary to achieve a balance between the level of data anonymization and the utility of the data for analysis. Note that there is no point in obtaining data with a level of anonymity that makes them insusceptible to any attack (which is an optimal scenario from a privacy point of view), if they are of no practical use at all. In this sense, it is essential to analyze the usefulness of the data after anonymization in order to select the level that best achieves this trade-off. To this end, in [64] different *k-anonymization* algorithms are studied together with the performance of four machine learning models by varying the value of k, concluding that with an increasingly strong *k-anonymity* constraint, the classification performance generally degrades. Also, in this work the authors note that "for very large k of up to 100 the performance losses remain within acceptable limits". Following this same line, in [65] different artificial intelligence models (i.e. neural networks, logistic regression, decision trees and Bayesian classifier) are considered together with three datasets, in order to analyze the performance of the different models before and after applying *k-anonymity* to the data with a value of $k = 2$. The authors highlight that, according to their results, certain machine learning algorithms are more suited to be used with privacy-preserving data mining techniques than others. Finally, in [66] several examples applying both *k-anonymity* and *differential privacy* are presented in order to mitigate inference attacks.

In this sense, given a classical dataset in the field of anonymization (*the adult dataset*), we have applied different anonymization techniques with different levels, and then we have analyzed the performance of a battery of classical machine learning models on them, in order to study the impact of anonymization on the efficiency (measured with different metrics) of the machine learning models.

2.7.1 Data Analyzed and Hierarchies

The *adult dataset* (available in [67]) is an extraction of information from the 1994 U.S. Census database. A sub-sample containing 32561 records has been used in this analysis, and the objective is to predict whether or not the income of an individual exceeds $50000 by year, based on census data. For this analysis, we have considered six quasi-identifiers, namely: *age*, *sex*, *education*, *relationship*, *occupation* and *native-country*, together with one sensitive attribute *salary class*, which is a binary

attribute that can take as values $> 50K$ or $\leq 50K$. The hierarchies levels presented below have been applied to each quasi-identifier (they have been extracted from those presented in the ARX Software GitHub repository [68], which is the one used to carry out the anonymization process:

- *Age:* five levels of hierarchies are applied, starting with 5-year intervals: $[15,20)$, $[20,25)$,..., $[75,80)$ and finally the case where the age is greater than 80 (+80). The next level is formed by 10-year intervals: $[10,20)$, $[20,30)$,...,$[70,80)$ and finally +80. In the third level are the intervals of 20 years, 40 years in the fourth, and the case $[0,80)$ or +80 in the last one.
- *Sex:* no hierarchies are applied.
- *Education:* two levels of hierarchies are applied for each of the 16 possible values. In the first level there are five values: *Primary School*, *High School*, *Undergraduate*, *Professional Education* and *Graduate*. The second level only takes three possible alternatives: *Primary education*, *Secondary education* and *Higher education*.
- *Relationship:* the possible values are *husband*, *wife*, *not in family*, *other relative*, *own child* and *unmarried*, and no hierarchy is applied.
- *Occupation:* a hierarchy level is included that encompasses in three categories the different options, namely: *technical*, *non-technical* and *other*.
- *Native country:* a hierarchy level is introduced that generalizes the country by continent. Thus, the possible options according to the countries present in the original database are: *Africa*, *Asia*, *Europe*, *North America*, *South America* and *unknown* (if the value of the native country field was ?). It should be noted that these are the possible options according to the countries present in the original database.

2.7.2 Machine Learning Models Under Study

In this analysis we are presenting a machine learning classification problem. That is, given a set of inputs, $\mathbf{X} \in \mathrm{D}^{nxm}$ with n the number of records and m the number of features, and the corresponding labels (outputs) $\mathbf{y}$, with $y_i \in \{1,\ldots,n_C\}\ \forall i \in \{1,\ldots,n\}$, being n_C the number of different classes, our objective is to estimate a function $\hat{f}$ which approximates $\mathbf{y} = \hat{f}(\mathbf{x})$ [14]. Specifically, we have applied the following four supervised machine learning models, with the quasi-identifiers as features and the sensitive attribute as the label:

- ***k*-Nearest Neighbors (kNN):** is a non-parametric method which finds the k nearest point in the training split to the test input. Thus, it computes the posterior probability that an element belongs to certain class. Its main drawback is that its performance is poor in cases of high dimensional inputs [14]. While large values of k reduce the effect of noise in classification, a value $k = 1$ actually induces a Voronoi tessellation of the point $\mathbf{X}_i, i \in \{1, \dots, n\}$.

- **Random Forest (RF):** with this model, N decision trees can be trained and their ensemble be computed. The main motivation for introducing this method comes from trying to reduce the variance of an estimator by aggregating several of them. In order to avoid obtaining highly correlated predictors, the RF method learns trees based on randomly chosen subsets of the input data and of the features [14]. Specifically, a new training subset is constructed using bootstrapping, then decision trees are trained on it using a subset of the predictor variables. This process is repeated to finally obtain a unified prediction.

- **Adaptive Boosting (AB):** boosting is a ML approach based on the idea of creating a highly accurate prediction rule by combining many relatively weak rules [69]. Thus, with adaptive boosting we first train a classifier (by giving equal weight to all data), calculate the error associated with it, and compute a new distribution to weight the data based on whether or not they were correctly classified. This process is repeated M times and finally a weighted average of the classifiers is performed. The details of this method can be found in pseudocode form in Algorithm 16.2 of [14].

- **Gradient Tree Boosting (GB):** is ensemble classifier that trains several individual predictors sequentially. As in the case of AB, again the fundamental idea lies in combining weak predictors (decision trees) to create a robust one. Typically, with this method the first predictor learns to predict the data mean, then the second one explains the errors of the first one, the third one explains the errors of the second one, and so on. A detailed formulation can be found in [70].

In order no train and test the models object of analysis we have used the Python library *scikit-learn* under the version 1.2.0, together with the *GridSeachCV* function. This function allows us to find the optimal hyper-parameters for each ML model. In order to obtain such hyper-parameters, in each case a 5-fold cross-validated grid-search has been performed over the hyper-parameter grid presented bellow and

some fixed parameters. Once we get the optimum parameters in each the model is retrained in order to use as much data as possible (using both train and validation sets).

- ***kNN.*** Number of neighbors: [3, 4, ... , 50]. Metric to calculate the distance: *Minkowski*.
- ***RF.*** Maximum depth of the tree: [2, 3, ... , 9]. Number of trees: 100. Criterion: gini impurity.
- ***AB.*** Number of estimators: [50, 100, 150]. Learning rate: [0.01, 0.1, 0.5, 1].
- ***GB.*** Number of estimators: [50, 100, 150]. Learning rate: [0.01, 0.1, 0.5, 1]. Maximum depth of the individual estimators: [2, 4, 6, 8, 10].

As for the error metrics used for evaluating the performance of each of the four ML models proposed above, as we are dealing with a binary classification problem, the accuracy and the Area Under the Receiver Operating Characteristic Curve (AUC-ROC) are analyzed. It is important to note that although the anonymization process has been performed on the full initial database (to simulate the real case study), a stratified random training-testing split (75%-25%) has been carried out for training and testing the models.

2.7.3 Results and Analysis

2.7.3.1 Varying the Value of *k* for *k-anonymity*

Let us start by analyzing the scenario in which the only anonymization technique applied is *k-anonymity* for different values of k, with $k \in \{2, 5, 10, 25, 50, 75, 100\}$. In addition, we allow a record suppression limit of 100%, which means that there is no limit on the number of records that can be deleted to carry out the anonymity process.

For this purpose, once the anonymization process has been performed using the hierarchies exposed in Section 2.7.1 and the *ARX Software*, we evaluate the different ML models regarding the test set. Table 2.11 shows the results obtained both for the accuracy and the AUC compared also to those obtained by applying the models on the data without anonymizing.

Overall, starting first to analyze the results in terms of accuracy, we can observe that, as expected, a higher value of k goes together with a reduction in accuracy in most of the cases. This is clearly seen in the three ensemble methods (RF, AB and GB) in which for both accuracy and AUC the best value is achieved with the raw

	kNN		*RF*		*AB*		*GB*	
k	*Acc.*	*AUC*	*Acc.*	*AUC*	*Acc.*	*AUC*	*Acc.*	*AUC*
Raw	0.8056	0.6779	**0.8334**	**0.7351**	**0.8350**	0.7377	**0.8352**	**0.7437**
2	0.8176	0.7143	0.8210	0.7215	0.8266	0.7372	0.8249	0.7346
5	**0.8199**	0.7234	0.8208	0.7209	0.8197	0.7342	0.8221	0.7258
10	0.8099	0.6932	0.8139	0.6913	0.8146	0.7171	0.8168	0.7157
25	0.8170	0.7194	0.8163	0.6987	0.8164	0.7167	0.8177	0.7139
50	0.8124	**0.7268**	0.8114	0.7285	0.8051	**0.7399**	0.8114	0.7285
75	0.8129	0.7064	0.8129	0.7064	0.8091	0.7195	0.8129	0.7030
100	0.8107	0.7254	0.8083	0.6981	0.8047	0.7117	0.8103	0.7274

Table 2.11: Accuracy and AUC obtained for each machine learning model when varying the value of k for k-anonymity.

data, and the worst with $k = 100$ except for AUC with GB, which is obtained when $k = 75$. However, with *kNN* the minimum for the accuracy and the AUC is reached when using the raw data. This peculiarity can be attributed to the characteristics of the data, the dimensionality of the problem and the hierarchies applied. With respect to the AUC again, it can be observed that there is a lot of variability in the results, which start clearly decreasing for the first values of k and the ensemble methods. It is especially striking that the optimal value for the AUC with *AB* method is achieved when $k = 50$ (although it is close to that obtained by using the raw data).

Besides, we are also interested in analyzing for each of the previously exposed values of k, how close the resulting database (ξ_k) is to being optimal for the stated level of anonymization. For this purpose the *average equivalence class size metric*, which measures how well the equivalence classes are created to fit the best case [54], is studied. Specifically, the optimal value for this metric would be one, indicating that all equivalence classes are of length k. Since in our example record suppression has been allowed with a limit of 100% the metric defined en Equation 2.1 has been used ($C^*_{AVG_k}$). As intuitively expected, the lowest value for this metric has been reached for the highest value of k analyzed ($k = 100$), obtaining in particular the following values: $C^*_{AVG_k}$ =7.88, 10.94, 10.75, 7.68, 6.90, 5.13, 4.17 $\forall k \in \{2, 5, 10, 25, 50, 75, 100\}$ respectively.

2.7.3.2 Applying Further Anonymity Techniques

Once a value of $k = 5$ for *k-anonymity* has been fixed, three other techniques has been applied: ℓ*-diversity*, *t-closeness*, δ*-disclosure privacy*. In particular, a value $\ell = 2$, $t = 0.7$, and $\delta = 1.5$ have been fixed in each case. Note that $\ell = 2$ is the only possible value other than one, and for the other two cases, the values have been chosen after testing different values in order to achieve a balance with the number of deleted records and the utility. In addition, the values of δ has been chosen in order to be the most restrictive scenario of the five under study (this will be discussed further below in terms of the other parameters fulfilled). The results as a function of the accuracy are shown in Table 2.12, and the ones concerning the AUC are shown in Table 2.13.

ML model	*Raw*	$k = 5$	$k = 5$, $\ell = 2$	$k = 5$, $t = 0.7$	$k = 5$, $\delta = 1.5$
kNN	0.8056	0.8199	0.7894	0.8164	0.8061
RF	0.8334	0.8208	**0.8025**	**0.8218**	0.8070
AB	0.8350	0.8197	0.8017	**0.8218**	0.8085
GB	**0.8352**	**0.8221**	**0.8025**	0.8210	**0.8098**

Table 2.12: Accuracy obtained with each machine learning model according to the anonymization technique applied.

ML model	*Raw*	$k = 5$	$k = 5$, $\ell = 2$	$k = 5$, $t = 0.7$	$k = 5$, $\delta = 1.5$
kNN	0.6779	0.7234	**0.7145**	0.7104	0.6231
RF	0.7351	0.7209	0.7066	0.7208	0.6156
AB	0.7377	**0.7342**	0.7044	**0.7369**	0.6356
GB	**0.7437**	0.7258	0.7040	0.7259	**0.6477**

Table 2.13: AUC obtained with each machine learning model according to the anonymization technique applied.

Here again the case of *kNN* stands out for its counter-intuitiveness: both accuracy and AUC obtained by using the raw data are the second worst values of the five scenarios considered. However, with the ensemble methods the optimum is reached when using the models with the raw data (AUC), i.e. without applying

any anonymization technique. In the same line, both with *kNN* and the ensemble models the worst results in terms of accuracy are obtained when the classification is performed with the data resulting after applying ℓ-*diversity* with $\ell = 2$. This is really intuitive from the meaning of this property: for the same set of quasi-identifiers, the label will take in all cases two different values.

Similarly, the worst performance in terms of AUC clearly occurs with all four models in the case where δ-*disclosure* is applied with $\delta = 1.5$. Again this is really intuitive if we analyze the parameters that are satisfied for each technique in that scenario, because although ℓ-*diversity* is only verified for $\ell = 1$, the value of t for *t-closeness* is the most stringent of all the cases analyzed, being $t = 0.47$. This has been verified by means of the Python library *pyCANON* [3] (version 1.0.0.), and we can also observe that although in the case in which the value of δ has been set to 1.5, this property is verified for $\delta = 1.16$ (more restrictive). In the case where we set $\ell = 2$, we obtain values $t = 0.64$ and $\delta = 4.88$, while in the case where we set $t = 0.7$, $\ell = 1$ and $\delta = 4.10$

As expected, the three ensemble methods perform quite similarly in all scenarios, as shown in Tables 2.12 and 2.13. In particular, the best value for the AUC is obtained with *GB* and the raw data, and in the worst scenario ($\delta = 1.5$) the best value is also obtained with this method for both accuracy (0.8098) and AUC (0.6477). Note that when $\delta = 1.5$, although the prediction does not seem to be bad in terms of accuracy it is actually poor in view of the AUC with the four models. This reflects the need to test different error metrics in order to select an optimal anonymization technique and ML model.

In order to carry out a classification task, it should be noted that the optimum scenario would be for all the records that constitute an equivalence class to have the same label (i.e. the same sensitive attribute). However, this is in contrast to the three anonymization techniques analyzed that focus on sensitive attributes, since it would make homogeneity attacks feasible, among others. Let us therefore look at the *classification metric (CM)* obtained in each of the four cases analyzed, presented in Section 2.4 and defined as shown in Equation 2.3.

The results obtained for each of the four cases and the CM metric are as follows: 0.2569 ($k = 5$), 0.3089 ($k = 5$, $\ell = 2$), 0.2575 ($k = 5$, $t = 0.7$), 0.4589 ($k = 5$, $\delta = 1.5$). These values contrast with those obtained in Table 2.12 for the accuracy, where the results for $\delta = 1.5$ are better than those for $t = 0.7$ in 3 of the 4 cases, while they agree with those obtained for the AUC (see Table 2.13). Note that for the AUC the worst results are obtained when $\delta = 1.5$, as would be expected based on the values calculated for *CM*.

2.7.4 Summary and Wrap Up

Throughout this section we have analyzed the performance of four classical machine learning models in a classification task after subjecting the *adult dataset* to different levels of anonymity. In this line, different privacy models have been applied to the original dataset with the overall idea of providing data privacy and anonymity guarantees. As already discussed, as a too strict level of anonymity may compromise data usefulness for processing and inference, it is necessary to reach a balance between the level of data anonymization and the utility of the data for analysis.

In this line, we have analyzed two settings concerning the anonymization level applied: varying the values of k for *k-anonymity* and applying ℓ*-diversity*, *t-closeness* and δ*-disclosure privacy* in addition to *5-anonymity*. In the first scenario, we found that in the case of *kNN* model the best results in terms of both metrics are not obtained either with the raw data or with the lowest value of k, while for the ensemble methods the best performance concerning accuracy is obtained when training with the raw data. In the second scenario it is observed that in agreement with the classification metric (CM) the worst results in terms of AUC are reached with $\delta = 1.5$ (which in this case comprises the strictest level of privacy analyzed).

2.8 *Anjana*: a Python Library for Anonymizing Sensitive data

The Python library anjana was presented in the following paper published in the journal Scientific Data from the Nature Publishing Group: ***Sáinz-Pardo Díaz, J.*** *& López García, Á. (2024). An Open Source Python Library for Anonymizing Sensitive Data, Sci Data, 11, 1289 [5].*

In this section we present the implementation of *anjana* [71, 72], a Python library for anonymizing sensitive tabular data using the techniques presented in Section 2.2, that are also available in *pyCANON* library for checking the privacy level according to such parameters. Firsts, let us start by motivating why is important to provide users with this tool and reviewing the state of the art concerning other frameworks with similar objective.

The number of tools available to perform tabular data anonymization processes is limited, even more if we rely on open source software. Specifically, we can highlight a few tools: ARX [73], written in Java, is a comprehensive open source soft-

ware for sensitive data anonymization which supports several privacy models (such as the ones exposed in this work), together with risk and quality models. In addition, *ARXaaS* [74] claims to be an "anonymization as a service" project built on top of the ARX library, which uses HTTP by default. Additionally, *Amnesia* [75] is a framework deployed in Java for data anonymization that includes an online version.

The development which has taken place in recent years in terms of frameworks for data-driven model development built on top of Python motivates the need of implementing a Python library for sensitive data anonymization, so that it can be seamlessly included in data analytics pipelines. This need is compounded by the lack of comprehensive tools for tabular data anonymization written in Python.

In this line, the Python library *anjana* provides users with a catalog of anonymity tools that can be used within a Python environment and that allows them to carry out the anonymization process in a few lines of code and that can be integrated in a data science pipeline during the data processing and preparation phases.

It is important to note that the main difference between *pyCANON* and *anjana* is that the former can be used on a known dataset to find out its anonymity level and thus to be aware of the possible risks derived from its sharing or publication, while *anjana* can be used to anonymize the data with different methods, analyzing which are the most appropriate ones for each case and application, allowing users to carry out the whole anonymization process.

2.8.1 Algorithms to Achieve *k-anonymity*

In order to achieve that a database verifies *k-anonymity* for a given value of k, different types of algorithms can be implemented. In this section, three methods typically used are presented along with the description on how to use them (*mondrian*, *data-fly* and *incognito*).

Mondrian. It is a method widely used in the literature [76, 77], which applies generalization in a multidimensional way. Specifically, it works by recursively partitioning the data based on a QI, creating equivalence classes that satisfy the desired value of k for *k-anonymity*. In particular, greedy algorithms can be used, so that in the first step, multidimensional regions are defined that cover the domain space with each box created containing at least k elements. In the second step, recoding functions are generated using summary statistics for each box.

Example 2.8.1 illustrates how to anonymize a database with *k-anonymity* using *mondrian*. This example is inspired by the one presented in [77].

Example 2.8.1. Suppose that it is desired to apply *k-anonymity* for $k = 2$ for the database given in Table 2.14, being the *age*, *ZIP code* and *sex* the QIs, and the attribute *disease* the SA.

Assume further that the hierarchies allowed for *age* are 5-years intervals as follows for the first level: $[15, 20)$, $[20, 25)$, $[25, 30)$, $[30, 35)$,...,$[90, 95)$,$[95, 100)$, 10-years intervals for the second level and 20-years intervals for the third one. The hierarchies for the QI *ZIP code* will be given by removing digits by substituting with an asterisk starting from left to right (e.g. the first level to generalize 28001 will be 2800*, the second 280**, the third 28*** and so on up to deletion, allowing 5 levels of generalization including suppression). Finally, for the QI *sex* only suppression is allowed.

Age	Sex	Zip code	Disease
20	F	28005	Pneumonia
21	F	28001	Pneumonia
27	M	08019	Appendicitis
29	M	08011	Coronary heart disease
25	M	08014	Pneumonia

Table 2.14: Example database. Quasi-identifiers: *age*, *ZIP code* and *sex*. Sensitive attribute: *disease*. Adapted de-anonymized version of Table 2.7.

Let us start by taking the pair of QIs *age* and *ZIP code* and sort them. We can apply *mondrian* on these two QIs recursively as shown in Figure 2.4.

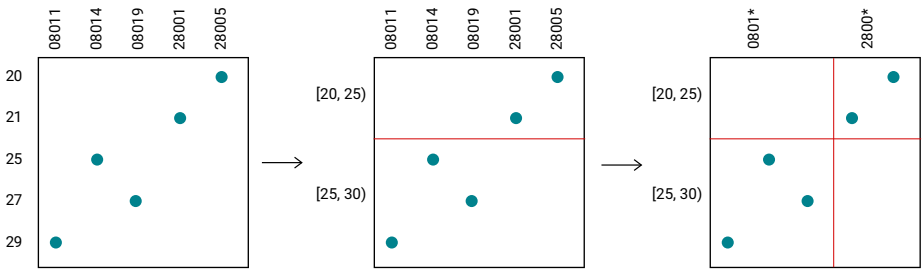


Figure 2.4: Anonymizing Table 2.14 using *mondrian* for the QIs *age* and *ZIP code*.

Finally, we can observe that if we add the QI *sex*, the database keeps *k-anonymity* for $k = 2$, so no further transformation is necessary. The result of anonymizing Table 2.14 is shown in Table 2.15.

Age	Sex	Zip code	Disease
[20, 25)	F	2800*	Pneumonia
[20, 25)	F	2800*	Pneumonia
[25, 30)	M	0801*	Appendicitis
[25, 30)	M	0801*	Coronary heart disease
[25, 30)	M	0801*	Pneumonia

Table 2.15: Example database anonymized for *k-anonymity* with $k = 2$.

Data-fly. The idea of this algorithm was first proposed in 1998 by L. Sweeney [78] and is the simplest of the ones presented in this section. Specifically, it simply involves, given a list of QIs, analyzing the frequency of the values of each QI in the database, and starting by generalizing by the one with the largest number of distinct values. Then, this method seeks to generalize the attribute with the most number of distinct values and suppressing no more than the allowed number of tuples. This is an iterative process in which different levels of generalization are applied to the QIs sorted in decreasing order of number of distinct values (updating after applying each generalization to the QIs).

In Example 2.8.1, as the number of unique records for *ZIP code* and *age* is the same, we would start by generalizing one of them randomly chosen to the first level. If for example we start with age, we would now have 2 unique values for *age*, 2 for *sex* and 5 for *ZIP code*. We would now apply the first level of generalization to the *ZIP code* and then we will have that the database verifies *k-anonymity* for $k = 2$.

Incognito. Finally, the incognito algorithm is a more efficient approach than data-fly, since it seeks to construct a lattice with the possible transformations that can be applied in each case to the QIs. Thus, the intuitive approach to incognito is to generate a graph and apply pruning to search for the transformation that produces the minimal full-domain generalization.

Concerning Example 2.8.1, in this case, since a set of only three QIs is available, it would be enough to simply find all possible generalizations of the data and select the minimum generalization that satisfies *k-anonymity* for $k = 2$, which again is to apply the first level to *age* (5-years interval), and the first level to *ZIP code* (remove the last digit).

2.8.2 Methodology

The scheme followed for implementing *k-anonymity* in *anjana* uses a modified version of the main idea of the data-fly algorithm. The method followed is presented in the workflow exposed in Figure 2.5.

Regarding the remaining techniques, we first apply *k-anonymity* and then carry out an analogy of the scheme presented in Figure 2.5, checking in this case that the desired technique is fulfilled with respect to the established parameter (i.e. ℓ for *ℓ-diversity*, t for *t-closeness*, δ for *δ-disclosure privacy*, etc.).

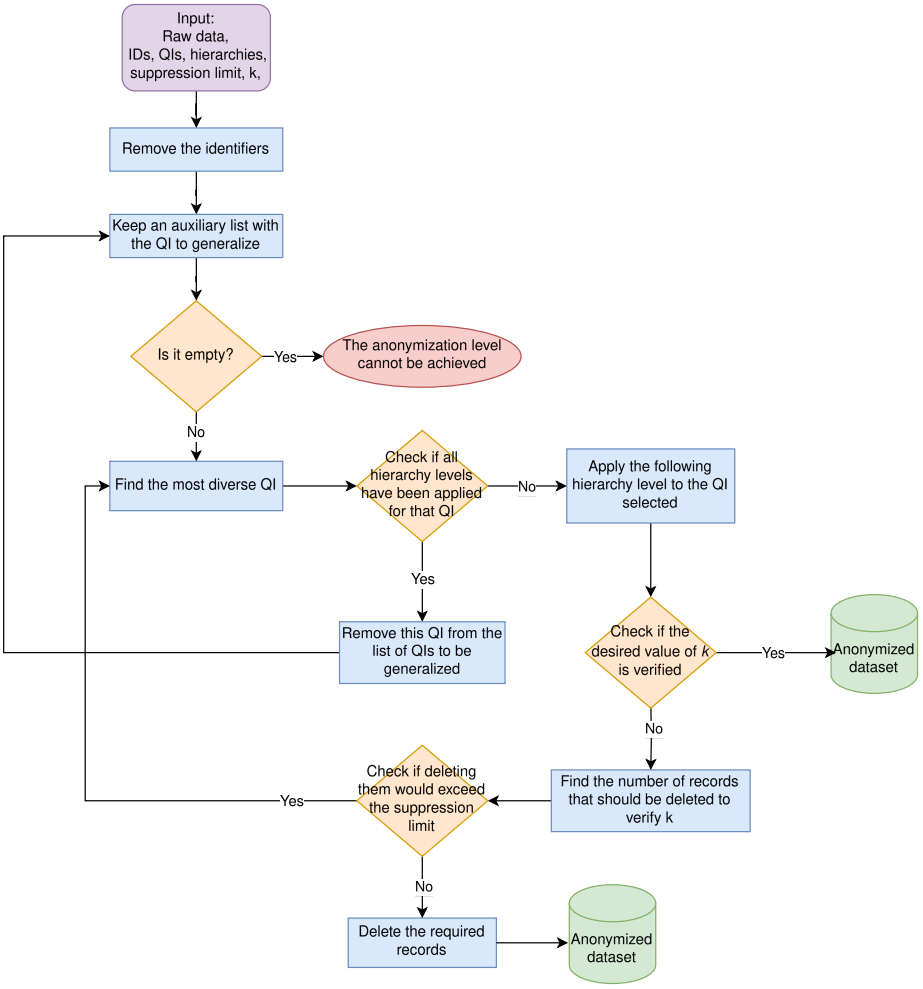


Figure 2.5: Workflow: anonymizing a tabular dataset for a given value of k for *k-anonymity*.

2.8.3 Usage Examples

In order to test the functionality of the library, we have used the adult dataset detailed in Section 2.7.1. Specifically, the *salary-class* column has been taken as a sensitive attribute. In this case, as quasi-identifiers we have taken: *workclass*, *education*, *marital-status*, *occupation*, *sex*, *native-country*, with the sets of hierarchies exposed in [63], taken from in the *ARX Software* GitHub repository [68].

As for the functionality tests and the different examples carried out, both the adult and the hospital dataset (available in the GitHub repository) have been used. On the other hand, a series of unit tests have also been implemented. These tests have been used to check the code coverage, higher than 92% for the versions 1.0.0 and 1.1.0 of the library.

In the following we show a use example of different anonymization tools implemented in *anjana*:

Example 2.8.2. The Code Example 2.3 shows how to apply *k-anonymity*, *l-diversity* and *t-closeness* for the QIs and SA already mentioned for the adult dataset and with a suppression limit of 50%.

Note that all these techniques have been successfully applied in this dataset containing 32561 rows, and it has been performed in less than one second (regarding the computational resources available during the test phase[4]).

```
import pandas as pd
import anjana
from anjana.anonymity import k_anonymity, l_diversity, t_closeness

# Read and process the data
data = pd.read_csv("adult.csv")
data.columns = data.columns.str.strip()
cols = [
    "workclass",
    "education",
    "marital-status",
    "occupation",
    "sex",
    "native-country",
]
for col in cols:
        data[col] = data[col].str.strip()

# Define the identifiers, quasi-identifiers and the sensitive attribute
quasi_ident = [
    "age",
    "education",
```

[4]For testing and developing the *anjana* Python library, a machine provided with 12th Gen Intel(R) Core(TM) i5-1 and with 16 GB RAM running under Ubuntu 22.04.4 LTS was used.

```
    "marital-status",
    "occupation",
    "sex",
    "native-country",
]
ident = ["race"]
sens_att = "salary-class"

# Select the desired level of k, l and t
k = 10
l = 2
t = 0.5

# Select the suppression limit allowed
supp_level = 50

# Import the hierarchies for each quasi-identifier.
#Define a dictionary containing them
hierarchies = {
    "age": dict(pd.read_csv("hierarchies/age.csv",
                            header=None)),
    "education": dict(pd.read_csv("hierarchies/education.csv",
                                  header=None)),
    "marital-status": dict(pd.read_csv("hierarchies/marital.csv",
                                       header=None)),
    "occupation": dict(pd.read_csv("hierarchies/occupation.csv",
                                  header=None)),
    "sex": dict(pd.read_csv("hierarchies/sex.csv",
                            header=None)),
    "native-country": dict(pd.read_csv("hierarchies/country.csv",
                                       header=None)),
}

# Apply k-anonymity:
data_kanon = k_anonymity(
    data, ident, quasi_ident, k, supp_level, hierarchies
    )
data_kanon_ldiv = l_diversity(
    data_kanon, ident, quasi_ident, sens_att, k, l, supp_level,
    hierarchies
    )
data_kanon_ldiv_tclos = t_closeness(
    data_kanon_ldiv, ident, quasi_ident, sens_att, k, t, supp_level,
    hierarchies
    )
```

Example Code 2.3: Example: applying *k-anonymity* for the adult dataset.

Then, executing the code given in Code 2.3, we will get an anonymized dataset verifying *k-anonymity* for $k \geq 10$, $\ell \geq 2$ for ℓ*-diversity* and $t \leq 0.5$ for *t-closeness*.

Subsequently, using the *pyCANON* library, we can check that the desired level of anonymity is achieved. In this example we get the following values: $k = 72$, $\ell = 2$ and $t = 0.47370$.

2.8.4 Further Functionalities

It should be noted that the library provides other complementary functionalities to the application process of the anonymization techniques. In particular, during the anonymization process with the techniques described in Section 2.2, it is important to take into account that hierarchies must be defined to carry out the generalization of the different quasi-identifiers. In addition, once the data have been anonymized, users may need to know the level of generalization applied to each of them, or apply a given transformation. These functions are available in *anjana* to facilitate both the anonymization process and the future work to be done with the data once anonymized.

Working with multiple sensitive attributes Furthermore, although the basic use is intended for a single sensitive attribute, the library can be used sequentially to anonymize a database according to multiple sensitive attributes, as explained below.

When the *pyCANON* library was introduced in Section 2.6, two paradigms were presented that could be applied in the case of multiple sensitive attributes: *harmonization of quasi-identifiers* and *updating the quasi-identifiers*. Although *anjana* is designed primarily to support a single SA, recursive calls can be made including new SA following the desired paradigm (updating or not the set of QIs), so that the database can be anonymized for as many SA as desired.

Creating the hierarchies First of all, in order to anonymize a dataset using generalization, it is necessary to define the hierarchies that we want to apply to each QI. It should be noted that we can have some QIs that act as such for the calculation of the various anonymization techniques described in Section 2.2, but that the users does not want them to be generalized or suppressed. For the QIs that are allowed to be generalized, the user must define the hierarchies that he/she wants to apply. This is a thorough work that must take into account the problem to be addressed, specially in the case of strings, such as the city of birth (that can be generalized to region, country, etc. depending on the problem), the type of employment, or the level of education, where the user must consider which levels of generalization he/she is interested in reaching.

For the case of numerical data, the most common is to apply interval-based hierarchies (of increasing length at higher level of generalization). In this sense, the function `generate_intervals` is implemented in *anjana*, which allows to create as many levels of hierarchies as necessary based on intervals of a user-defined

length at each level. These functionality is showcased in Example 2.8.3.

Example 2.8.3. In this example we show how to generate the hierarchies based on intervals for the QI age assuming that we have a *pandas* dataframe with the database stored as `data`. Suppose we want to establish a hierarchy to anonymize the QI *ZIP code* (zip value in the database `data`) and *age*, so that *ZIP code* has only one level which is suppression and *age* has 5 levels of generalization: starting with intervals of 2 years (level 1), 5 years (level 2), 10 years (level 3), 20 years (level 4), and 50 years (level 5). The dictionary `hierarchy` contains the levels of generalization for the QI age (generated using the function `generate_intervals` from *anjana*) and for ZIP code. Example Code 2.4 illustrates these functionality.

```
from anjana.anonymity import utils

age = data['age'].values
zip_code = data['zip'].values

hierarchy = {
    "age": {
        0: age,
        1: utils.generate_intervals(age, 0, 100, 2),
        2: utils.generate_intervals(age, 0, 100, 5),
        3: utils.generate_intervals(age, 0, 100, 10),
        4: utils.generate_intervals(age, 0, 100, 50),
        5: utils.generate_intervals(age, 0, 100, 20)
    }
    "zip": {0: zip_code,
            1: np.array(["*"] * len(zip_code))
    }
}
```

Example Code 2.4: Usage example: define hierarchies and generate interval-based hierarchies using *anjana*.

Getting the anonymity transformation applied Once the anonymization process is performed, the users may be interested in being able to obtain the transformation or generalization applied for each QI. Suppose that the database to be anonymized has n QI and that each QI i has n_i possible levels of generalization (hierarchies) $\forall i\{1,\ldots,n\}$.

The transformation (T) obtained after carrying out the anonymization process will be given by.

$$T = [t_1, t_2, \ldots, t_n],$$

where is t_i the level of hierarchy applied to the QI i, and with $t_i \in \{0, 1, \ldots, n_i - 1\}$, $\forall i \in \{1, \ldots, n\}$.

In order to obtain the transformation applied to an anonymized database the *anjana* library provides the user with the `get_transformation` function.

Example 2.8.4. The Code Example 2.5 an example on the use of this function is shown considering the data, QIs and hierarchies given in the Code Example 2.3. Note that the data used (`data_kanon_ldiv`) in this example is the anonymized one obtained from the Code Example 2.3 when applying *k-anonymity* with $k = 10$ and *ℓ-diversity* with $\ell = 2$.

```
from anjana.anonymity import utils

transformation = utils.get_transformation(
    data_kanon_ldiv,
    quasi_ident,
    hierarchies)
print(f'The transformation applied is: {transformation}')
for i, qi in enumerate(quasi_ident):
    print(f'QI: {qi}. Level or generalization: {transformation[i]}')
```

Example Code 2.5: Usage example: get the transformation applied to a dataset anonymized using *anjana*.

The result of executing the previous code for the adult dataset with the data and hierarchies as defined above is show in the Code Example 2.6.

```
The transformation applied is: [4, 2, 1, 2, 0, 0]
QI: age. Level or generalization: 4
QI: education. Level or generalization: 2
QI: marital-status. Level or generalization: 1
QI: occupation. Level or generalization: 2
QI: sex. Level or generalization: 0
QI: native-country. Level or generalization: 0
```

Example Code 2.6: Usage example: output obtained in the console when executing the Code Example 2.5 with the inputs as given in the Code Example is 2.3.

Applying a given transformation Obtaining the transformation applied when anonymizing a dataset can be important for multiple reasons. For example, in cases where different data sources have to be centralized to develop an ML/DL model, it is important that the generalizations applied in each case are of the same level (e.g. if a numeric field is generalized to intervals, it must be ensured that all databases have performed the generalization to the same type of interval, i.e., for all common databases to be centralized, the same hierarchy is applied to all the QIs).

This also applies to the case where it is intended to train a model in a distributed way using multiple data sources, in order to obtain robustness in the global model. This type of architectures will be discussed in Chapter 4 in more detail.

Note that if we are working with multiple databases in this way, then we will need to harmonize the transformations applied when anonymizing the data. Suppose that we have N databases to anonymize $\xi_j \ \forall j \in \{1,\ldots,N\}$. Let us assume that on each of these databases ξ_j with the same set of n QI each one of them, the same hierarchies with the same levels have been defined. Thus, when carrying out the local anonymization process individually for each one of them certain transformation T_j will be applied for each $\xi_j \ \forall j \in \{1,\ldots,N\}$. Furthermore, let us assume that in all cases the suppression limit is set to 0%. In order to unify and harmonize the transformation applied prior to data centralization or distributed training, we will have to find the transformation $\overline{T}$ obtained as the transformation that when applied on each of the databases involved $\xi_j \ \forall j \in \{1,\ldots,N\}$ will maintain at least the level of anonymity required locally for each ξ_j initially. Thus, the computation of such a transformation $\overline{T}$ will be obtained as:

$$\overline{T} = [\overline{t}_1, \overline{t}_2, \ldots, \overline{t}_n].$$

Be $T_j = [t_1^{(j)}, t_2^{(j)}, \ldots, t_n^{(j)}]$ the transformation applied locally when anonymizing the database ξ_j to verify the required anonymity level, $\forall j \in \{1,\ldots,N\}$. Then, $\overline{t}_i$ is given by $\overline{t}_i = \max_{j\in\{1,\ldots,N\}}(t_i^{(j)}) \ \forall i \in \{1,\ldots,n\}$. Thus, in order to harmonize the applied transformations, to each original ξ_j database, the transformation $\overline{T}$ will be applied as summarized in Figure 2.6.

In *anjana*, the function `apply_transformation` allows users to apply the given transformation to a dataset, either the anonymized one (in which certain records may have been deleted), or the raw dataset. Example 2.8.5 demonstrates how to use this function.

Example 2.8.5. In this example we show how to apply a given transformation to certain database using the function `apply_transformation` and given the set of QIs and the hierarchies associated. Again, to follow coherently the main thread of this section, we will consider the adult dataset with the QIs and the hierarchies defined and applied in the Code Example 2.3.

Suppose that given the raw dataset stored as a dataframe in the variable *data*, we want to apply a transformation so that the QIs are generalized to the following levels: *age* to level 2, *education* to level 2, *marital-status* to level 1, *occupation* to level 2, *sex* to level 1, and *native-country* without generalizing (0).

This can be done using *anjana* as given in the following Code Example (2.7), with the list of QIs and the hierarchies dictionary as given in the Code Example 2.3, and with the QIs in the same order as detailed above.

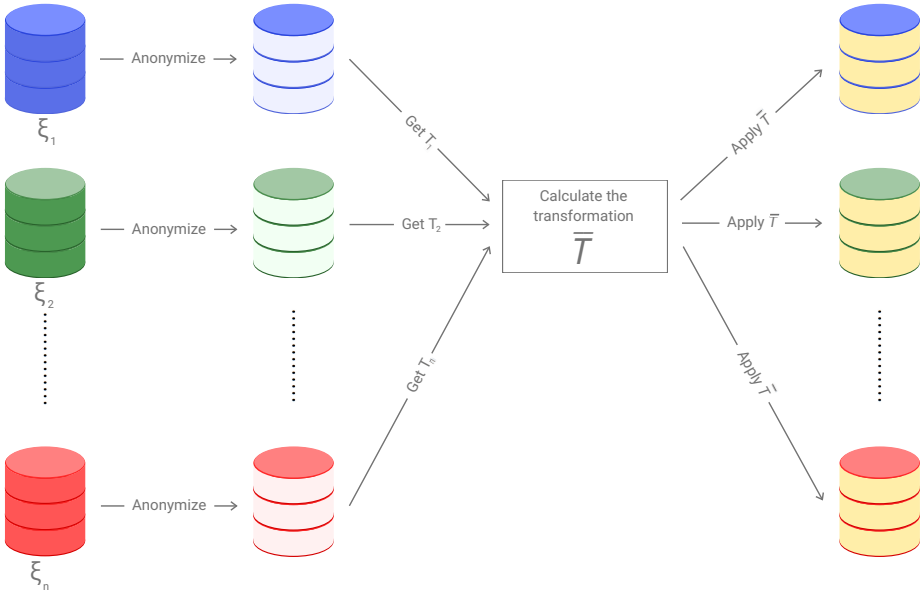


Figure 2.6: Summary: harmonizing the transformation to be applied to multiple databases $\xi_j \ \forall j \in \{1, \ldots, N\}$.

```
from anjana.anonymity import utils

transformation = [2, 2, 1, 2, 1, 0]
data_transformed = utils.apply_transformation(
    data,
    quasi_ident,
    hierarchies,
    transformation
)
```

Example Code 2.7: Use example: apply a given transformation applied to the adult dataset given the identifiers and the hierarchies.

It should be noted that the above applies if and only if the suppression limit established for all the databases involved is 0%. Otherwise, the records corresponding to the locally applied transformation must be deleted in each case. This may cause that when anonymizing with the strictest transformation, a higher level of anonymization than required is being imposed, so auxiliary libraries such as the case of *pyCANON* can be used to analyze the resulting level of anonymization. In particular, the global transformation can be applied on the anonymized data (applying the appropriate generalization level on each QI), or on the contrary it can

be applied on the raw data and then analyze the anonymization level to see if it is necessary to remove any record.

2.8.5 Summary

The implementation of this library arises from the need to provide users working with sensitive data with a set of tools for their proper anonymization available in open source through a library written in Python, as this is one of the most widely used programming languages currently used for the development of data-driven models and applications.

Thus, *anjana* provides users with nine anonymization techniques that can be applied to tabular databases, as well as multiple functionalities that facilitate the integration of anonymized data into data life-cycle flows, even prior to their use as input for ML/DL models (such as the possibility to obtain or apply certain transformations).

2.9 Conclusions

Throughout this chapter we have analyzed in depth the theoretical background associated with the anonymization and pseudonymization of data, with special attention to those of a tabular nature.

First, a review of classical methods used for pseudonymization of data containing identifying attributes has been carried out. In relation to anonymization, two software components developed as open source Python libraries have been presented. Specifically, *pyCANON* is a Python library to check the anonymization level of a dataset, and *anjana* is library than can be used to anonymize a tabular dataset using nine of the most commonly used tabular data anonymization techniques. In addition, we have reviewed common attacks that can be suffered by anonymized databases, as well as metrics to measure the privacy-utility balance obtained and methods to assess the risk of disclosure and re-identification. Furthermore, the impact of the anonymization carried out with some of these techniques on the performance of ML models trained with anonymized data has been analyzed.

Regarding future work in this field, in addition to the maintenance and improvement of the functionality of the two implemented libraries, we will pursue the integration and implementation of anonymity techniques that take into account auxiliary populations for their use and application, as well as other metrics and their evaluation as detailed in Section 2.5. In addition, we have focused on the use of tabular data, but future work includes extrapolating to other types of data, such

as time series, via statistics mixing of synthetic data; images, which may include masking techniques; or focusing on membership inference attacks on training data within ML models and ways to mitigate them.

CHAPTER 3

Differential Privacy

Tú querías que yo te dijera
el secreto de la primavera.

Federico García Lorca,
Idilio
Canciones

Contents

3.1 Theoretical Basis 103
3.1.1 Introducing Differential Privacy 104
3.1.2 Input Privacy and Output Privacy 106
3.2 Global Differential Privacy 108
3.2.1 Laplace Mechanism 108
3.2.2 Gaussian Mechanism 111
3.2.3 Exponential Mechanism 112
3.3 Local Differential Privacy 114
3.3.1 Randomized Response 114
3.3.2 Histogram Representation 116
3.4 Rényi Differential Privacy 118
3.5 Metric Differential Privacy 120
3.6 Differential Privacy for Data Publishing 122
3.7 Differential Privacy for Data Analysis 123
3.7.1 Deep Learning Meets Differential Privacy 123
3.8 Review of Differential Privacy Frameworks 127
3.9 Summary and Conclusions 130

Abstract

This chapter presents the theoretical foundations of the privacy enhancement technique known as differential privacy. We will distinguish between methods for local and global differential privacy, highlighting for the latter the Laplace, Exponential and Gaussian mechanisms. We will explore metric differential privacy or d-privacy for addressing the geo-indistinguishability problem and we will study the theoretical basis of Rényi differential privacy, which provides stronger flexibility of the definition especially in case of multiple mechanism composition. Then, an overview of the differential privacy approach applied to data publishing will be given, followed by its use in data analysis, with special emphasis on deep learning by means of the inclusion of this definition within the stochastic gradient descent process. Finally, different open source frameworks available in the state of the art will be reviewed.

3.1 Theoretical Basis

In the previous chapter we presented several techniques that are applied to raw data before they are shared or published, in order to ensure that they have been properly privatized. These techniques provide privacy safeguards that protect information prior to its analysis through data mining, machine learning, statistical analysis and other applications. While pseudonymization techniques are based in obfuscating the identifiers, with anonymization we aim to reduce the risk of identifying a user in the database under analysis. Specifically, anonymization techniques focus on applying generalizations based on different hierarchies in order to avoid the extraction of valuable information as well as to prevent different attacks, as explored in Sections 2.2 and 2.3. As we have seen, in many cases this can be a complex process due to the amount of auxiliary data that can be used, leading to linkage attacks. Furthermore, an important point to keep in mind is that it is necessary to achieve a compromise between anonymization and data usability.

The need for alternative techniques to guarantee data privacy while maintaining data utility arises. In this sense, differential privacy seeks to protect privacy by adding statistical noise to the data, either before sharing or during the learning or analysis process. Thus, with differential privacy, a balance is sought between the amount of information that is lost and therefore the usefulness of the data, and its privacy. Usually, differential privacy is applied on the output of queries on databases, or during the learning process (for example by applying noise during the

gradient decent when training a neural network). The following is an introduction differential privacy (DP) and to the two main approaches: local and global.

3.1.1 Introducing Differential Privacy

The motivation for including the technique of DP arises to prevent different types of attacks that can be performed on a database or during the analysis. For example, suppose that we have a database *DB* with information about $\{x_i\}_{i=1}^n$ records. Suppose that an attacker has background information about all but one of these records, x_j, for a certain $j \in \{1,\dots,n\}$. Then, the adversary can make a query on the original database to infer information about the record x_j. To do so, it would be enough to perform the same query on the database of which he/she has information, that is *DB'*=*DB*$\backslash\{x_j\}$, and compare the result with the one obtained by performing the query on *DB*.

Now, suppose a database with n records containing clinical information about whether a certain patient suffers certain disease. The attacker can make a query to that database about the number of patients who passed the disease and gets a value $n_1/2$. This same attacker can make the same query after some time, when he/she knows that a new patient might have been included in the database (so that both databases differ only by one record). Suppose he/she gets again a value $n_1/2$, then he/she would not be able to deduce if a new patient has been added and does not have the disease, or if no patient has been added. However, if a value $n_1/2+1$ is obtained, the attacker would know that a new patient who has the disease has been added to the database.

Clearly, this is a very simple example, but it reflects the problem we want to highlight. With differential privacy, we want to ensure that when two databases differ only in one record (i.e., they are neighbors databases), the output of performing the same query on both is as indistinguishable as possible, so that the attacker cannot identify whether the record x_j for which he/she has no information, is in the database or not (see [79]).

The intuitive idea is to add statistical noise to the data. As shown in the following definition, we must have a privacy budget (ϵ), with which we can control the balance between privacy and usefulness of the data.

Definition 3.1.1. ϵ-differential privacy. A randomized algorithm $\mathscr{M}$, with domain $\mathscr{D}$ and range $\mathscr{R}$, satisfies **ϵ-differential privacy** (see, for instance, [80]) if for any two adjacent inputs $Y, Y' \in \mathscr{D}$ and for any subset of outputs $S \subseteq \mathscr{R}$ it is satisfied that:

$$\mathbb{P}[\mathscr{M}(Y) \in S] \leq e^{\epsilon}\mathbb{P}[\mathscr{M}(Y') \in S],$$

with $\epsilon \geq 0$.

In the previous definition the value of ϵ is the privacy budget, which means that this parameter allows to control the level of privacy (the amount of privacy loss allowed). Typically (but not necessarily) a value of $\epsilon < 1$ is taken.

Remark 3.1.1. The lower the ϵ value, the higher the privacy, but the lower the usefulness of the data for the analysis. A value of $\epsilon = 0$ implies total privacy: $\frac{\mathbb{P}[\mathscr{M}(Y)\in S]}{\mathbb{P}[\mathscr{M}(Y')\in S]} \leq 1$.

$$\boxed{Privacy} \longleftarrow \epsilon \longrightarrow \boxed{Utility}$$

From this definition we can directly infer the simple composition theorem (Theorem 3.1.1) which allow as to control the accumulated privacy loss when performing sequential analyzes on the same dataset.

Theorem 3.1.1. *Be $\mathscr{M}_i$ $\forall i \in \{1,\dots,n\}$ n randomized algorithms, with domain $\mathscr{D}$ and range $\mathscr{R}$ and $\mathscr{M}_i$ satisfying ϵ_i-differential privacy for each $i \in \{1,\dots,n\}$. The mechanism defined as the composition of $\mathscr{M}_i$ $\forall i \in \{1,\dots,n\}$, $\mathscr{M}(x) = (\mathscr{M}_1,\dots,\mathscr{M}_n)$, where each randomized algorithm is applied sequentially and independently, will verify ϵ-differential privacy with $\epsilon = \sum_{i=1}^{n} \epsilon_i$.*

Proof. For simplicity, the demonstration for the discrete case is shown:

Let $Y, Y' \in \mathscr{D}$ be two adjacent inputs, verifying that $\mathbb{P}[\mathscr{M}_i(Y) = s_i]$ $\forall i \in \{1,\dots,n\}$ with $s_i \in S$, and $s = (s_1,\dots,s_n)$, considering s the sequential outputs.
Note that assuming that the randomized mechanism are independent, we get that $\mathbb{P}[(\mathscr{M}(Y) = s] = \mathbb{P}[(\mathscr{M}_1(Y),\dots,\mathscr{M}_n(Y)) = (s_1,\dots,s_n)]$, then:

$$\mathbb{P}[(\mathscr{M}_1(Y),\dots,\mathscr{M}_n(Y)) = (s_1,\dots,s_n)] = \mathbb{P}[(\mathscr{M}_1(Y) = s_1] \cdot \ldots \cdot \mathbb{P}[(\mathscr{M}_n(Y) = s_n]$$

Then, as $\mathscr{M}_i$ verifies ϵ_i-differential privacy $\forall i \in \{1,\dots,n\}$:

$$\begin{aligned}
&\mathbb{P}[(\mathscr{M}(Y) = s] \leq e^{\epsilon_1}\mathbb{P}[(\mathscr{M}_1(Y') = s_1] \cdot \ldots \cdot e^{\epsilon_n}\mathbb{P}[(\mathscr{M}_n(Y') = s_n] \Rightarrow \\
\Rightarrow\ &\mathbb{P}[(\mathscr{M}(Y) = s] \leq e^{\sum_{i=1}^{n}\epsilon_i}\mathbb{P}[(\mathscr{M}_1(Y') = s_1] \cdot \ldots \cdot \mathbb{P}[(\mathscr{M}_n(Y') = s_n] \Rightarrow \\
\Rightarrow\ &\mathbb{P}[(\mathscr{M}(Y) = s] \leq e^{\sum_{i=1}^{n}\epsilon_i}\mathbb{P}[(\mathscr{M}_1(Y'),\dots,\mathscr{M}_n(Y')) = (s_1,\dots,s_n)] \Rightarrow \\
\Rightarrow\ &\mathbb{P}[(\mathscr{M}(Y) = s] \leq e^{\sum_{i=1}^{n}\epsilon_i}\mathbb{P}[(\mathscr{M}(Y') = s]
\end{aligned}$$

□

In addition, a parameter δ in order to represent the probability of exceeding the privacy budget ϵ can also be introduced, i.e., with probability $1-\delta$ the privacy loss will not be greater than ϵ. Then the concept of (ϵ, δ)-*differential privacy* arises, and can be defined as follows:

Definition 3.1.2. (ϵ, δ)-differential privacy. Be $\mathscr{M}$ a randomized algorithm with domain $\mathscr{D}$ and range $\mathscr{R}$. It satisfies **(ϵ, δ)-differential privacy** (see [80]) if for any two adjacent inputs $Y, Y' \in \mathscr{D}$ and for any subset of outputs $S \subseteq \mathscr{R}$ it is satisfied that:

$$\mathbb{P}[\mathscr{M}(Y) \in S] \leq e^{\epsilon}\mathbb{P}[\mathscr{M}(Y') \in S] + \delta,$$

with $\epsilon \geq 0$ and $\delta \in [0, 1]$.

3.1.2 Input Privacy and Output Privacy

In the following we define two privacy concepts depending on whether this applies to the inputs or outputs of a data-driven model or computation process especially acting on sensitive or personal data.

Definition 3.1.3. Input privacy. It is a guarantee that one or more persons can participate in a computation process, so that none of the parties knows anything about the inputs of the others.

With input privacy we are then preserving the privacy of the database. In some cases we can achieve this task via data anonymization, as explained in the previous chapter, applying certain transformation to the data. Other examples of input privacy involve more complex computing schemes, such as secure multi-party computation (SMPC), homomorphic encryption (HE) or trusted/secure execution environments (TEEs/SEEs).

Definition 3.1.4. Output privacy. It attempts to ensure that the output of an information stream cannot be reversed to learn specific attributes about the input.

One of the most common ways to guarantee output privacy is the use of differential privacy (DP), so that it is used as a method of perturbation or addition of noise on the computation/statistical method or query performed, allowing statistical disclosure control.

Moreover, regarding differential privacy, we can mainly distinguish two types: local and global DP. On the one hand, let us assume different data owners, who are going to send their data to a central server which they do not trust to perform

some statistical or computing process. In this sense, to ensure input privacy they can add controlled noise to it before sending it to that server (local DP), for example applying this technique to the distributions/histogram representations. On the other hand, if the server is trusted and the aim is to guarantee output privacy when performing queries on the data, such noise can be added to the response to the given query or computing process. This applies not only to queries, but also to statistical analysis, visualizations, and applications of ML or DL models, among others, that can be performed on a centralized dataset.

The intuitive idea behind local and global DP is summarized in the scheme given in Figure 3.1. We can note that with local DP (LDP) we add noise to the data before sending it to an untrusted central server, while with global DP (GDP) we assume that we trust the server and we add noise to the computation process or to the queries performed.

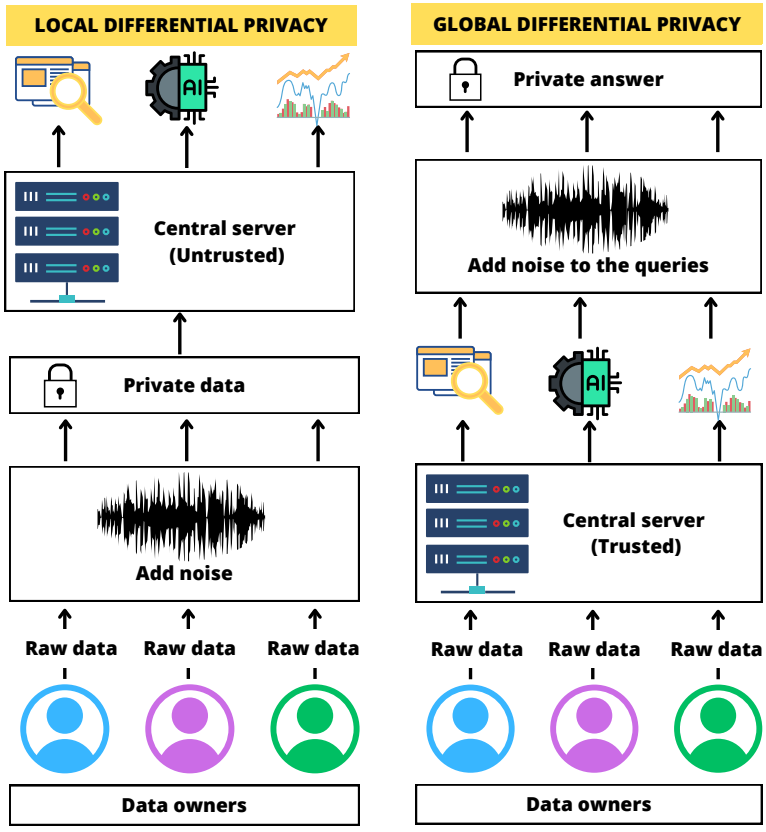


Figure 3.1: Schematic idea of the local and global differential privacy approaches.

3.2 Global Differential Privacy

The idea behind global differential privacy is to ensure that the results of a statistical analysis or queries performed on a database maintain the privacy of the individuals involved. This means that the noise is added to the result of the database queries, i.e., it is introduced by the data curator (assumed to be trustworthy) just before sharing it with third parties. In this section three mechanisms that are widely used to guarantee (global) differential privacy are presented, namely Laplace, Gaussian and Exponential mechanisms.

3.2.1 Laplace Mechanism

In order to ensure DP, one of the most commonly used mechanisms is based on the use of the Laplace distribution.

Remark 3.2.1. Remember that the probability density function (PDF) of a Laplace distribution is given by $L(\mu, b) = \frac{1}{2b} exp\left(-\frac{|X-\mu|}{b}\right)$, with μ and b the location and scale parameters respectively. In Figure 3.2 the graphical representation of the PDF for different values of μ and b is shown.

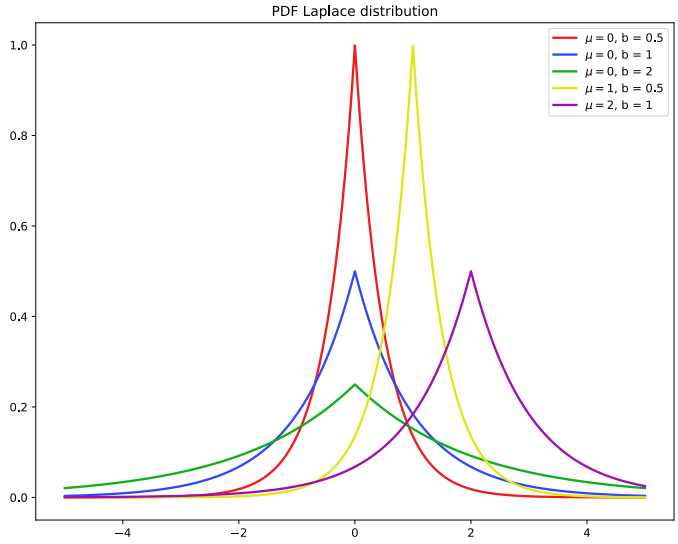


Figure 3.2: Probability density function (PDF) of the Laplace distribution.

Then, let us start with the definition of l_1-*sensitivity* prior to introducing the Laplace mechanism.

Definition 3.2.1. l_1**-sensitivity.** Be $f : \mathscr{D} \longrightarrow \mathbb{R}^k$, we define the l_1**-sensitivity** (see [81]) as follows:

$$\Delta_1(f) := \max_{||x-y||_1} ||f(x)-f(y)||_1$$

Definition 3.2.2. Laplace mechanism. Given any function $f : \mathscr{D} \longrightarrow \mathbb{R}^k$, we define the **Laplace mechanism** (see [81]) as:

$$\mathscr{M}_L(x, f(\cdot), \epsilon) := f(x) + (Y_1, \ldots, Y_k),$$

where $Y_i, \forall i \in \{1, \ldots, k\}$ are independent and identically distributed (i.i.d.) variables from the Laplace distribution with location 0 and scale $\Delta_1(f)/\epsilon$ (equivalently $Laplace(0, \Delta_1(f)/\epsilon)$).

Theorem 3.2.1. *The Laplace Mechanism or Laplace Algorithm outlined above, maintains ϵ-differential privacy (see [82] and [81]).*

Proof. The proof of this theorem is very simple considering the following:

- The PDF of the Laplace distribution with $\mu = 0$ and $b = \Delta_1(f)/\epsilon$ is given by:

$$L(0, \Delta_1(f)/\epsilon) = \frac{\epsilon}{2\Delta_1(f)} exp\left(-\epsilon \frac{|X|}{\Delta_1(f)}\right).$$

- If $X \sim L(\mu, b)$, then $X + k \sim L(\mu + k, b)$.
- If X and Y are two independent identically distributed variables, then the joint PDF is the product of each PDF.

Be $z \in \mathbb{R}^k$, D and D' two neighboring databases. Be $P_D(z)$ and $P_{D'}(z)$ the PDF of D and D' respectively. Let us proof that $P_D(z)/P_{D'}(z) \leq exp(\epsilon)$:

$$\frac{P_D(z)}{P_{D'}(z)} = \prod_{i=1}^{k} exp\left(\frac{-\epsilon(|f(X)_i - z_i| - |f(Y)_i - z_i|)}{\Delta_1(f)}\right) \leq$$
$$\leq \prod_{i=1}^{k} exp\left(\frac{\epsilon(|f(Y)_i - f(X)_i|)}{\Delta_1(f)}\right) = exp\left(\frac{\epsilon}{\Delta_1(f)} \sum_{i=1}^{k} |f(X)_i - f(Y)_i|\right) =$$
$$= exp\left(\epsilon \frac{||f(X)_i - f(Y)_i||_1}{\Delta_1(f)}\right) \leq e^{\epsilon}$$

□

Note that this method is one of the most widely used in the implementation of differential privacy techniques.

Example 3.2.1. In this example we are going to use the stroke dataset [83], which contains clinical information of patients: whether or not they have suffered a stroke, if they have a heart disease, their body mass index (BMI), their average glucose levels, etc, along with certain demographic factors such as type of job, the marital status or the residence type.

Specifically, we will calculate the average BMI of those who have suffered a stroke. To add an additional layer of privacy to these data, as they are sensitive variables, we will apply differential privacy for different ϵ values.

In Figure 3.3 we can see how the value obtained by applying DP, in particular using the Laplace mechanism and the Python *PyDP* library [84], tends to converge to the real value (without applying DP) as ϵ increases (in particular for $\epsilon >> 1$).

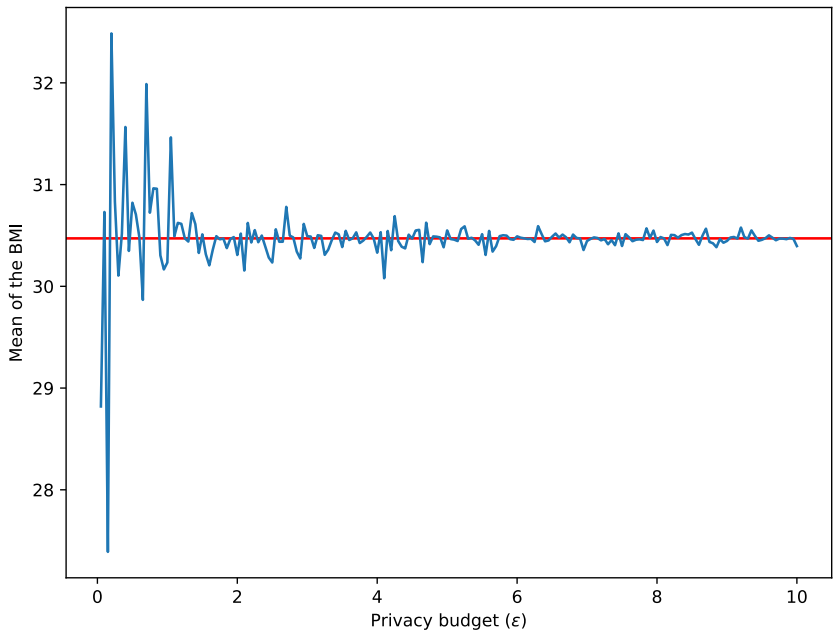


Figure 3.3: Example: mean BMI of all persons in the database who suffered a stroke. Results applying DP with the Laplace mechanism for different values of the privacy budget vs the actual value.

3.2.2 Gaussian Mechanism

As we have exposed previously, Laplace mechanism preserves ϵ-differential privacy. However, with the Gaussian Mechanism we can ensure that (ϵ, δ)-differential privacy is preserved. First, instead of using l_1-sensitivity, in this case we need to define l_2-sensitivity as follows:

Definition 3.2.3. l_2-sensitivity. Be $f : \mathscr{D} \longrightarrow \mathbb{R}^k$, we define the **$l_2$-sensitivity** as follows:

$$\Delta_2(f) := \max_{||x-y||_1} ||f(x) - f(y)||_2$$

Definition 3.2.4. Gaussian mechanism. Given any function $f : \mathscr{D} \longrightarrow \mathbb{R}^k$, we can define the **Gaussian mechanism** (see [85], [86]) as:

$$\mathscr{M}_G(x, f(\cdot), \epsilon, \delta) := f(x) + (Y_1, \ldots, Y_k),$$

where $Y_i, \forall i \in \{1, \ldots, k\}$ are independent and identically distributed (i.i.d.) variables from the Gaussian distribution $N(0, \sigma^2)$, with $\sigma = \frac{\Delta_2(f)\sqrt{2\log(1.25/\delta)}}{\epsilon}$.

Theorem 3.2.2. *The Gaussian Mechanism given in Definition 3.2.4, maintains (ϵ, δ)-differential privacy.*

Proof. A detailed proof can be found in [85]. □

Example 3.2.2. Let us considered the same data as in the Example 3.2.1. Suppose that we want to calculate the mean BMI (of the dataset corresponding to patients who have suffered stroke) in this case using the Gaussian mechanism. In this case, in addition to the privacy budget (ϵ), we will need to introduce a parameter δ in order to adjust the probability of exceeding the value fixed for ϵ.

Note that in the case of the mean, the l_2-sensitivity will be given by $\frac{x_{max} - x_{min}}{n}$, with n the number of data, x_{max} and x_{min} the maximum and minimum possible values in the dataset respectively.

For applying this mechanism in this example we will use the *diffprivlib* library [87] [88]. Then, taking a value $\delta = 1e-3$, Figure 3.4 shows value obtained for the mean of the BMI by applying DP with the Gaussian mechanism when varying the value of ϵ. Again we can notice how the values calculated by applying DP converge to the real value as the ϵ value increases.

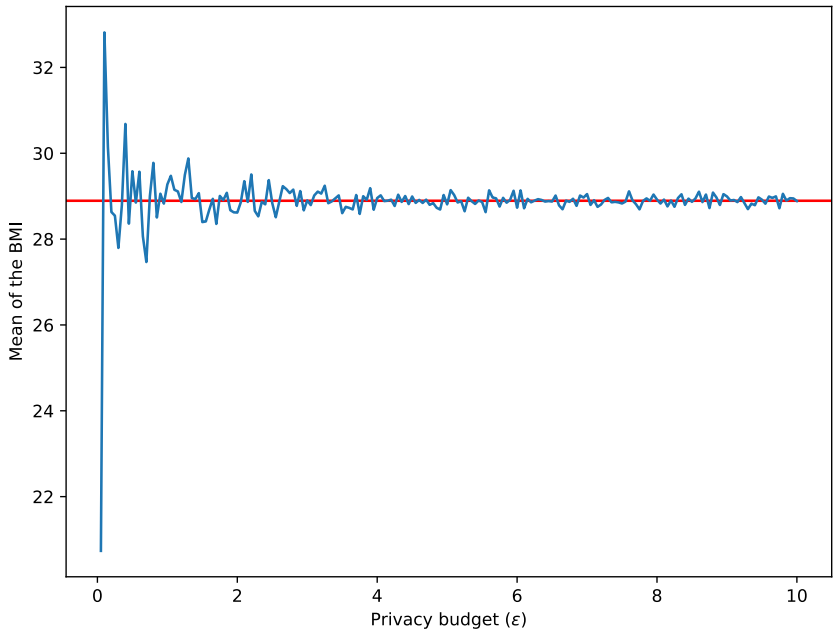


Figure 3.4: Example: mean BMI of all persons in the database who suffered a stroke. Results applying DP with the Gaussian mechanism for different values of the privacy budget vs the actual value.

3.2.3 Exponential Mechanism

Until now we have analyzed the Laplace and Gaussian mechanisms, which are based on output perturbation (i.e. $\mathscr{M}(x) := f(x) + Y$). However, the main limitation of these methods is that they only work for numeric queries. For non-numeric queries we can introduce the exponential mechanism. First, let us start by introducing some key concepts:

Definition 3.2.5. Utility function. Be $\mathscr{R}$ and output space and $f : \mathscr{D} \to \mathscr{R}$ (in the following we can assume $f : \mathbb{N}^{|X|} \to \mathscr{R}$). We define an **utility function** associated to the abstract output space as:

$$g := \mathbb{N}^{|X|} \times \mathscr{R} \to \mathbb{R}$$

Be $\mathscr{D} \subseteq \mathbb{N}^{|X|}$ a dataset, and $r \in \mathscr{R}$ the output of a query, $g(D, r)$ represents the score function regarding returning r as output for a query $f(\mathscr{D})$.

Remark 3.2.2. Regarding the definition of the utility or score function g given in definition 3.2.5:

- The function g must be defined for each use case.
- If $r = f(D)$, we assign the maximum score.

Then, we need to define the sensitivity for the utility or score function g defined above:

Definition 3.2.6. Sensitivity of the utility function. Be $\mathscr{R}$ the abstract space of outputs. Be $g : \mathbb{N}^{|\mathcal{X}|} \times \mathscr{R} \to \mathbb{R}$ a score or utility function, we define the **sensitivity** as follows:

$$\Delta(g) := \max_{r \in \mathscr{R}} \max_{D,D' : ||D-D'|| \leq 1} |g(D,r) - g(D',r)|$$

Then, we define the exponential mechanism as follows:

Definition 3.2.7. Exponential mechanism. Be D the set of inputs, $\mathscr{R}$ the set of outputs, $r \in \mathscr{R}$ and $\Delta(g)$ the sensitivity of the utility or score function g. The **exponential mechanism** [86] outputs r with probability $\mathbb{P}[r]$ defined as follows:

$$\mathbb{P}[r] = \frac{exp\left(\frac{\epsilon \cdot g(D,r)}{2\Delta(g)}\right)}{\sum_{r' \in \mathscr{R}} exp\left(\frac{\epsilon \cdot g(D,r')}{2\Delta(g)}\right)}$$

Theorem 3.2.3. *The exponential mechanism ($\mathscr{M}_E$) verifies ϵ-differential privacy.*

Proof. For this proof and for simplicity, let us suppose that $\mathscr{R}$ is finite [86]:

Be D and D' two neighboring databases, $D, D' \in \mathscr{D}^n$, $\mathscr{M}_E$ the exponential mechanism, and $r \in \mathscr{R}$, being $\mathscr{R}$ the set of outputs. Be g the score function and $\Delta(g)$ its sensitivity. The probability of r being selected is $\mathbb{P}[r] = \frac{exp\left(\frac{\epsilon \cdot g(D,r)}{2\Delta(g)}\right)}{\sum_{\bar{r} \in \mathscr{R}} exp\left(\frac{\epsilon \cdot g(D,\bar{r})}{2\Delta(g)}\right)}$, then:

$$\frac{P(\mathscr{M}_E(D) = r)}{P(\mathscr{M}_E(D') = r)} = \frac{\frac{exp(\frac{\epsilon \cdot g(D,r)}{2\Delta(g)})}{\sum_{\bar{r} \in \mathscr{R}} exp(\frac{\epsilon \cdot g(D,\bar{r})}{2\Delta(g)})}}{\frac{exp(\frac{\epsilon \cdot g(D',r)}{2\Delta(g)})}{\sum_{\bar{r} \in} exp(\frac{\epsilon \cdot g(D',\bar{r})}{2\Delta(g)})}} =$$

$$= exp\left(\frac{\epsilon \cdot (g(D,r) - g(D',r))}{2\Delta(g)}\right) \frac{\sum_{\bar{r} \in \mathscr{R}} exp(\frac{\epsilon \cdot g(D',\bar{r})}{2\Delta(g)})}{\sum_{\bar{r} \in \mathscr{R}} exp(\frac{\epsilon \cdot g(D,\bar{r})}{2\Delta(g)})} \leq$$

$$\leq e^{\epsilon/2} \frac{\sum_{\bar{r} \in \mathscr{R}} exp(\frac{\epsilon \cdot g(D',\bar{r})}{2\Delta(g)})}{\sum_{\bar{r} \in \mathscr{R}} exp(\frac{\epsilon \cdot g(D,\bar{r})}{2\Delta(g)})} \leq e^{\epsilon/2} e^{\epsilon/2} = e^{\epsilon}$$

□

Then, one of the most commonly encountered example of use of the exponential mechanism is the selection of a value from a set of possible outputs regarding a utility function, maximizing its reliability while respecting privacy.

3.3 Local Differential Privacy

As introduced in Section 3.1.1, local differential privacy (LDP) aims to protect the privacy of each individual's record in the dataset. Specifically, it involves applying DP before centralizing the data to an external server. To this end, through the addition of noise, the response to each query is perturbed before it is sent to a central server. Then, we can modify the definitions of DP to the case of local DP as follows:

Definition 3.3.1. Local ϵ-differential privacy. A randomized algorithm $\mathscr{M}$, with domain $\mathscr{D}$ and range $\mathscr{R}$, satisfies **local ϵ-differential privacy** if for any inputs $y, y' \in \mathscr{D}$ and for any possible output $r \in \mathscr{R}$ it is satisfied that:

$$\mathbb{P}[\mathscr{M}(y) = r] \leq e^{\epsilon}\mathbb{P}[\mathscr{M}(y') = r],$$

with $\epsilon \geq 0$.

Definition 3.3.2. Local (ϵ, δ)-differential privacy. A randomized algorithm $\mathscr{M}$, with domain $\mathscr{D}$ and range $\mathscr{R}$, satisfies **local (ϵ, δ)-differential privacy** if for any inputs $y, y' \in \mathscr{D}$ and for any possible output $r \in \mathscr{R}$ it is satisfied that:

$$\mathbb{P}[\mathscr{M}(y) = r] \leq e^{\epsilon}\mathbb{P}[\mathscr{M}(y') = r] + \delta,$$

with $\epsilon \geq 0$ and $\delta \in [0, 1]$.

A first approach to the most classical methods to achieve local differential privacy is shown in this section. We can note that in cases where we are dealing with numerical data, the Laplace or Gaussian mechanisms can be used to achieve local DP. However, when working with categorical data, the concept of randomized response should be introduced.

3.3.1 Randomized Response

The randomized response method was stated by Stanley L. Warner in 1965 as a proposal to add privacy to structured survey interviews, and thus avoid inaccurate responses from individuals answering a survey [89].

We can mainly think about two approaches concerning the mechanism of randomized response for applying DP: for binary and non-binary data (including categorical), but in both cases for finite sets. First, for the case of binary data we can think on the classic approach given by the coin algorithm: Suppose we want the records of people who have passed certain disease. In this case, for each individual, we flip a coin, if it comes up heads we give the real answer, but if it comes up tails we flip a new coin. For this second coin, if it comes up heads we answer no, and if it comes up tails we answer yes. In this way, in no case can we know if the saved answer is the real answer for each user. The probability of occurrence of each event using this *randomized response mechanism* is shown below:

$\mathbb{P}(Yes|No) = \mathbb{P}(tails)\mathbb{P}(heads) = \frac{1}{4}$.
$\mathbb{P}(Yes|Yes) = \mathbb{P}(heads) + \mathbb{P}(tails)\mathbb{P}(heads) = \frac{3}{4}$.
$\mathbb{P}(No|Yes) = \mathbb{P}(tails)\mathbb{P}(tails) = \frac{1}{4}$.
$\mathbb{P}(No|No) = \mathbb{P}(heads) + \mathbb{P}(tails)\mathbb{P}(tails) = \frac{3}{4}$.

In view of the above, we can immediately note that the previously exposed random response process verifies ϵ-differential privacy for $\epsilon = \log(3)$.

Other mechanism for randomized response can be considered taking into account a pre-defined threshold for the probability of answering or not the ground truth, specifically for binary data. For example, be Y_i the ground truth and $\overline{Y_i}$ the randomized one. Be $\lambda \in [0, 1/2]$, one randomized mechanism for binary data can be given as follows parametrized by certain value of λ (if $\lambda = 0$ it is uniformly random and if $\lambda = 1/2$ we always return the real value) [90]:

$$\overline{Y_i} = \begin{cases} Y_i & \text{with probability } 1/2 + \lambda \\ \neg Y_i & \text{with probability } 1/2 - \lambda \end{cases}.$$

Now, we can generalize the previous approach to a no binary setting. Then, we can introduce the *k-ary Randomized Response algorithm* as follows [91]:

Definition 3.3.3. Randomized response algorithm. Be k the number of different values in the data domain ($\mathscr{D} = \{y_1, \ldots, y_k\}$). The **k-ary randomized response algorithm** for the value y given ϵ is the true value y if $b = 0$ and otherwise it is sampled from $\mathscr{D}$ following an uniform distribution. In this approach we get $b \sim Ber(k/(e^\epsilon + k - 1))$, with *Ber* the Bernoulli distribution.

Theorem 3.3.1. *The k-ary Randomized Response algorithm verifies ϵ-local-differential privacy [91].*

Proof. Be $p = k/(e^\epsilon + k - 1)$ the parameter of the Bernoulli distribution. First, note that the output of the method is a uniformly distributed random value with

probability $p = \frac{k}{e^\epsilon+k-1}$ (case $b = 1$) and the true value with probability $1-p = 1-\frac{k}{e^\epsilon+k-1} = \frac{e^\epsilon-1}{e^\epsilon+k-1}$ (this is given by definition of the Bernoulli distribution).

Be y, y' two inputs (the real values) and $r \in \mathscr{R}$ the output (randomized value). If $y = y' = r$ or $y \neq r$ and $y' \neq r$ the proof is trivial, so we will proof these theorem for the case in which $y = r$ and $y' \neq r$. We want to find out that:

$$\frac{\mathbb{P}[\mathscr{M}(y) = r]}{\mathbb{P}[\mathscr{M}(y') = r]} \leq e^\epsilon,$$

with $\mathscr{M}$ the k-ary randomized response algorithm. Then:

$$\mathbb{P}[\mathscr{M}(y) = r] = 1-p+\frac{p}{k} = \frac{e^\epsilon-1}{e^\epsilon+k-1} + \frac{1}{e^\epsilon+k-1} = \frac{e^\epsilon}{e^\epsilon+k-1},$$

Note that as $y = r$ this is the probability of getting the real value if $b = 0$ $(1-p)$ and if $b = 1$ and the random value outputs r, p/k. In the same line, for $\mathbb{P}[\mathscr{M}(y') = r]$ as $y' \neq r$:

$$\mathbb{P}[\mathscr{M}(y') = r] = \frac{p}{k} = \frac{1}{e^\epsilon+k-1},$$

Finally, we get the following ratio that verifies ϵ-local differential privacy:

$$\frac{\mathbb{P}[\mathscr{M}(y) = r]}{\mathbb{P}[\mathscr{M}(y') = r]} = \frac{\frac{e^\epsilon}{e^\epsilon+k-1}}{\frac{1}{e^\epsilon+k-1}} = e^\epsilon.$$

□

Finally, the k-ary Randomized Response is also know in the literature as flat mechanism or k-RR mechanism ($\mathscr{M}_{k-RR}$) [92]. In view of the above, given $r \in \mathscr{R}$ and $y \in \mathscr{D}$, we can give a simplified version of this mechanism as follows:

$$\mathbb{P}[\mathscr{M}_{k-RR}(y) = r] = \begin{cases} \frac{e^\epsilon}{e^\epsilon+k-1} & \text{if } y = r \\ \frac{1}{e^\epsilon+k-1} & \text{if } y \neq r \end{cases}.$$

Applying this kind on LDP techniques allow users to add noise from the data owner side, even before sending such data to a central server or curator.

3.3.2 Histogram Representation

A problem that is frequently addressed in the field of differential privacy is the representation of privatized histograms.

Histograms can provide important insights concerning the data under study, from outliers to the general trend of the data. One way of addressing this problem via adding differential privacy can be simply done by thinking of the mathematical

definition of an histogram. Histograms are graphical representations of a set of data in such a way that the values are grouped into a certain number of classes or bins. The widely known Sturges' Rule is commonly used to choose the number of bins, which states that we should construct as many classes as $1 + log_2(N)$, with N being the number of data. Additionally one can vary the width of each class, or simply take the same for all bins.

If we consider histograms as sets of classes that group the data in a database according to certain characteristics, a common technique for preserving privacy is to add Laplacian noise to each disjoint bin of the histogram, following the Laplace mechanism given in Section 3.2.1. In particular, since each class is disjoint and can reveal information about the data represented, adding noise on each class is a very simple and effective solution to this problem.

In particular, it is immediate to note that the sensitivity of adding a new element to the database (remember that two inputs are disjoint if they vary in a single element), can only vary by at most 1 in a class or bin of the histogram. Thus, the sensitivity l_1 in this case is 1, according to the definition given in Definition 3.2.1. Thus, following the Laplace mechanism given in Definition 3.2.2 we have that we have to add noise from a Laplace distribution $L(0, 1/\epsilon)$ to guarantee that differential privacy is verified.

An example of a randomized histogram taking 3 different values of ϵ is shown below, demonstrating the impact of the privacy budget on the randomized histogram.

Example 3.3.1. As in the Example 3.2.1, we consider again the Stroke Dataset. In this case, we will consider (according to the Sturges' rule) 13 bins. The idea is to create a histogram showing the distribution of the *"average glucose level"* feature. Then, we consider three ϵ values, a particularly small one that adds a large amount of noise $\epsilon_1 = 0.01$, and two more significant values $\epsilon_2 = 0.1$ and $\epsilon_3 = 0.5$.

As explained before, to each count in the various initial bins, we will add Laplacian noise with 0 as location and $1/\epsilon$ as scale. In order to add this noise we will use the function `laplace` from the package `random` from the *numpy* Python library. Once we add the noise form such distribution to each bin we take the result rounded to the nearest integer. It is immediate to see that the trend with $\epsilon_3 = 0.5$ is closer to the real trend of the original histogram, as it adds less noise. The four histograms are shown in Figure 3.5.

It can be seen that the total number of records when adding DP in the different scenarios varies from the initial number of records. it should be noted that the application of DP aims to provide results that are statistically similar to the original

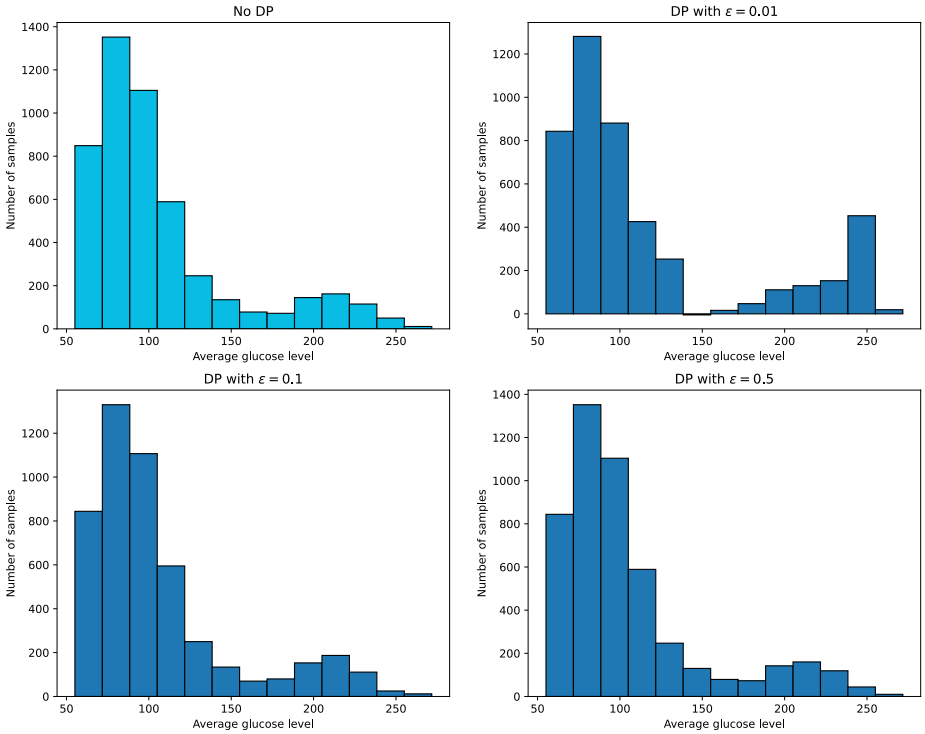


Figure 3.5: Example: original and DP randomized histograms for the average glucose level of the Stroke Dataset.

data, but do not disclose information about specific individuals in the database. In this sense the randomized histogram should be useful to perform data analysis through it, without compromising privacy.

3.4 Rényi Differential Privacy

The idea behind the concept of Rényi Differential Privacy (RDP) is to generalize the basic idea of differential privacy so that it entails a natural relaxation of the classic definition by using the Rényi divergence, as proposed in [93]. Thus, RDP allows to give a strictly stronger privacy definition. In particular, this technique is very useful when multiple randomized algorithms are applied sequentially due to the guarantees it provides with respect to composition of heterogeneous mechanisms [93]. In order to introduce this concept, let us begin by highlighting the formal definition of Rényi divergence.

Definition 3.4.1. Rényi divergence. Let P and Q be two probability distributions. The **Rényi divergence** of order α of P over Q is defined as follows:

$$D_\alpha(P||Q) = \frac{1}{\alpha-1}\log \mathbb{E}_{x\sim Q}\left(\frac{P(x)}{Q(x)}\right)^\alpha,$$

with $\alpha > 1$.

If we transfer this to the notation followed in the previously given differential privacy definitions, we get the following definition for (α,ϵ)-Rényi differential privacy:

Definition 3.4.2. (α,ϵ)-Rényi differential privacy. Be $\mathscr{M}$ a randomized algorithm with domain $\mathscr{D}$ and range $\mathscr{R}$. It satisfies **(α,ϵ)-Rényi differential privacy** (see [93, 94]) if for any two adjacent inputs $Y, Y' \in \mathscr{D}$ it is satisfied that:

$$D_\alpha(\mathscr{M}(Y)||\mathscr{M}(Y')) \leq \epsilon,$$

with $\alpha > 1$ and $\epsilon \geq 0$.

As can be seen from the above, by introducing the α parameter, RDP allows privacy to be quantified in a more flexible way than with classical DP, where we are limited by the ϵ parameter.

In view of the proof given for the Proposition 3 of [93], it can be seen that the following theorem relating (α,ϵ)-Rényi differential privacy and (ϵ,δ)-differential privacy is fulfilled:

Theorem 3.4.1. *If a mechanism $\mathscr{M}$ verifies (α,ϵ)-Rényi differential privacy, will also verify (ϵ',δ)-differential privacy for $\epsilon' = \epsilon + \frac{1}{\alpha-1}\log\left(\frac{1}{\delta}\right)$ and $\delta \in (0,1)$.*

Proof. A detailed proof of this theorem is given in the original work by I. Mironov in [93]. □

Regarding the mechanism to obtain classical RDP, the main method is again the Gaussian mechanism. In this case, we define it as follows:

Definition 3.4.3. Gaussian Mechanism for (α,ϵ)-Rényi differential privacy. Given any function $f : \mathscr{D} \longrightarrow \mathbb{R}^k$, we can define the **Gaussian Mechanism for (α,ϵ)-Rényi differential privacy** as:

$$\mathscr{M}_{G_{RDP}}(x, f(\cdot), \alpha, \epsilon) := f(x) + (Y_1, \ldots, Y_k),$$

where $Y_i, \forall i \in \{1,\ldots,k\}$ are independent and identically distributed (i.i.d.) variables from the distribution $N(0,\sigma^2)$, with $\sigma^2 = \frac{\Delta_2(f)^2\alpha}{2\epsilon}$.

As already mentioned, the usefulness of this technique resides especially in cases where multiple queries or mechanisms are performed on the same database executing multiple algorithms. This makes it particularly useful in machine learning systems. This is because RDP satisfies the sequential composition of mechanisms. This is detailed in the following theorem, again proved in [93]:

Theorem 3.4.2. *Be $\mathscr{M}_1$ and $\mathscr{M}_2$ two mechanisms verifying (α, ϵ_1)-Rényi differential privacy and (α, ϵ_2)-Rényi differential privacy respectively. Then, the composition of $\mathscr{M}_1$ and $\mathscr{M}_2$ (the results of running these two mechanism sequentially), verifies $(\alpha, \epsilon_1 + \epsilon_2)$-Rényi differential privacy*

Proof. A detailed proof of this theorem is given in the original work by I. Mironov in [93]. □

If we rely on the previous theorem, it is evident that applying sequentially n times a given mechanism verifying (α, ϵ)-*Rényi differential privacy* will result in $(\alpha, n\epsilon)$-*Rényi differential privacy*.

3.5 Metric Differential Privacy

In this section we explore the notion of *d-privacy*, *metric differential privacy* or simply *metric privacy* [95]. The motivation behind this concept is that, in multiple situations, we can consider a distance relationship between two databases or even a pair of inputs. Naturally we can think of a geometric distance given by the distance between two databases as a function of data location. However, we can also define a metric space that can be used to measure the distance between two datasets as a function of the different distributions of the inputs, among other cases.

The main objective pursued with metric-DP or *d-privacy* is to take into account the geographical distance in the development of privacy preserving solutions, with special focus to the context of location privacy and geo-indistinguishability in order to protect the users location in location-based services [92].

Definition 3.5.1. ***ϵ*-d-privacy.** A randomized algorithm $\mathscr{M}$, with domain $\mathscr{D}$ and range $\mathscr{R}$, with $\mathscr{D}$ provided with a metric $d : \mathscr{D}^2 \longrightarrow \mathbb{R}_{\geq 0}$, satisfies ***$\epsilon$*-d-privacy** or ***$\epsilon$*-metric-differential-privacy** (see [96]) if for any inputs $y, y' \in \mathscr{D}$ and for any possible output $r \in \mathscr{R}$ it is satisfied that:

$$\mathbb{P}[\mathscr{M}(y) = r] \leq e^{\epsilon \cdot d(y,y')} \mathbb{P}[\mathscr{M}(y') = r],$$

with $\epsilon \geq 0$. Note that we assume that $\mathscr{D}$ is provided with a metric space.

From this definition, we can note that inputs that are closer in relation to the given metric d will be more indistinguishable to an attacker, while inputs that are more distant will be more easily discernible. According to [96], we can note that if the domain and the range verify $\mathscr{D} = \mathscr{R} = \mathbb{R}^2$, and d in the Euclidean metric, then we get the definition of geo-indistinguishability. Specifically, we can define this property as follows:

Definition 3.5.2. Geo-indistinguishability. A mechanism provides guarantees of **geo-indistinguishability** if and only if for any radius $x > 0$ we can ensure ϵ *x-privacy* within the radius x [97].

In fact, the Euclidean metric is one of the most commonly used in *d-privacy*, together with the Hamming distance or discrete metrics for the distance.

Several mechanisms are know for achieving *d-privacy* for some specific metrics. Examples are the α-randomized response mechanism and the geometric and truncated geometric mechanisms explained in [98]. Adding geometric noise is a commonly used approach for ensuring *d-privacy*. For example, in [92] two mechanism are defined for achieving *d-privacy*, the Laplacian and Geometric ones, for continuous and discrete metric spaces respectively.

Definition 3.5.3. Laplacian Mechanism for *d*-privacy. The **Laplacian Mechanism for *d*-privacy** ($\mathscr{M}_{dL}$) [92] is given by the following probability density function given $y \in \mathscr{D}$ the real location that outputs $r \in \mathscr{R}$, with $\mathscr{D}$ provided with a metric space with the metric d and with λ a normalization factor:

$$dP_y(r) = \lambda e^{-\epsilon d(y,r)}.$$

Definition 3.5.4. Geometric Mechanism for *d*-privacy. The **Geometric Mechanism for *d*-privacy** ($\mathscr{M}_{dG}$) [92] is given by the following probability distribution given $y \in \mathscr{D}$ the real location that outputs $r \in \mathscr{R}$, with $\mathscr{D}$ provided with a metric space with the metric d and with λ a normalization factor:

$$\mathbb{P}[\mathscr{M}_{dG}(y) = r] = \lambda e^{-\epsilon d(y,r)}.$$

In the previous section we have presented the basis concerning four classic differential privacy methods, global, local, Rényi and metric DP. In this line, Figure 3.6 seeks to serve as an intuitive overview of the main distinctive aspect associated with each of these techniques

In the following sections, we will distinguish between the use of DP for publishing sanitized databases and for the data analysis process, with special attention to its incorporation in the training of DL models.

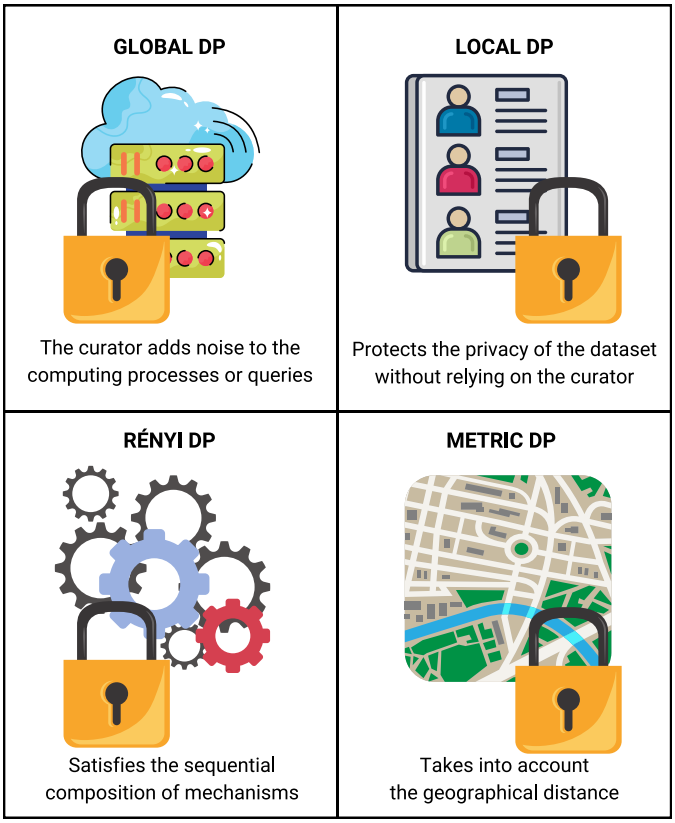


Figure 3.6: Wrap up: global, local, Rényi and metric differential privacy.

3.6 Differential Privacy for Data Publishing

Concerning differential privacy for data publishing, we can first think of publishing statistics on the data, to which GDP applies, or even histograms representing the data distributions previously sanitized, as we have seen previously in Section 3.3.2. Otherwise, if we want to release a complete sanitized dataset using DP, we will need to apply LDP.

Essentially, in terms of adding differential privacy prior to the database publication, we can think of the case where we add noise directly on the dataset. In a tabular database, numeric attributes can have noise added to them using the Laplace or Gaussian mechanisms seen in Section 3.2, while for binary or categorical attributes with multiple classes we can apply different randomized response methods (as the k-ary randomized response mechanism) seen in Section 3.3. In the same way, dif-

ferent partitions of the dataset can be made and sanitization can be performed on each partition individually before re-aggregating it for publication.

3.7 Differential Privacy for Data Analysis

From a data analytics point of view, we can think of the application of global differential productivity for the calculation of different statistics on a database, as well as response to queries performed on them, for which the GDP techniques seen previously can be employed. The same applies to the representation of the data by means of histograms or other statistics, as already mentioned when explaining different LDP methods.

However, beyond all this, the integration of DP during the training of ML or DL models is a growing field with multiple applications. For example, in [99] the authors explore an image classification use case using a pediatric pneumonia dataset integrating DP during the training of the models via the DP-Stochastic Gradient Descent methods (DP-SGD).

In particular, in the following sections we show the integration of DP in DL architectures during the gradient descent process, as well as a possible way to implement this approach.

3.7.1 Deep Learning Meets Differential Privacy

The most common way to incorporate DP in a machine or deep learning pipeline using artificial neural networks focuses on its application during the gradient descent process, as presented below.

Differentially-private stochastic gradient descent Let us start by explaining the theoretical background concerning the step of stochastic gradient descent (SGD) during the training of a ML/DL model based on ANN.

As we know from basic optimization notions, for minimizing a function f we need to move in the negative direction of the gradient of f $(-\nabla f)$. Additionally, we can add a learning rate (μ) for such descent. This process is followed in the gradient descent method for optimizing the loss function of the learning model. The idea is to modify the model weights according to the direction of the error gradient. It is important to note that this process can be very time consuming, which motivates introducing the concepts of stochastic or batch gradient descent.

The main objective of SGD is to optimize the learning model iteratively [100]. In addition, SGD aims to approximate the gradient by a single sample (or even a

batch). This methods is usually much faster than the classic gradient descent, and it also allow to scape from local minima. However, the gradient updates with SGD are more noisy as it does not use the whole dataset, and then they can have a high variance and lower convergence. This can cause the optimization process to be less stable. Algorithm 2 shows an intuitive pseudocode of this process. Note that for simplicity, we show the example assuming that we approximate the gradient using a single sample instead of a batch, but it will be analogous.

Algorithm 2 Stochastic Gradient Descent (SGD)

INPUT: M, the model to be trained
INPUT: D, the training dataset
INPUT: F, the loss function
INPUT: N_e, the number of epochs
INPUT: μ, the learning rate

function SGD(M, D, F, N_e, μ)
    Get N the number of samples of D
    Shuffle the initial dataset $D = \{(x_i, y_i) : \forall i\{1, \ldots, N\}\}$
    Initialize some random weights for the model $w^{(0)}$
    for t $\in [1, \ldots, N_e]$ **do**
        Randomly select a single sample (x_i, y_i)
        Compute $\overline{y}_i$ the prediction obtained for x_i with M initialized with w^{t-1}
        $g_i^{(t)} = \nabla_{w^{(t-1)}} F(\overline{y}_i, y_i)$ ▷ Compute the gradients of the loss
        $w^{(t)} \leftarrow w^{(t-1)} - \mu g_i^{(t)}$
    end for
end function

Then, for adding DP during the gradient descent, the Gaussian mechanism can be used by introducing the noise multiplier and the bound of the norm of the gradient (clipping norm). The pseudocode given in Algorithm 3 details this implementation.

Note that before adding Gaussian noise to the gradients, it is essential to clip the gradients to ensure that the sensitivity is bounded. Similarly, the noise added by the noise multiplier (n_ϵ) must be carefully calibrated according to the value of ϵ to be verified. It is important to remember that, according to definition 3.2.4 we have to choose $\sigma = \frac{\Delta_2(f)\sqrt{2\log(1.25/\delta)}}{\epsilon}$ in order to guarantee (ϵ, δ)-DP. Then, in this approach we will need then to take into account also the size of the batches selected to to preserve this definition.

Algorithm 3 Differentially Privacy Stochastic Gradient Descent (DP-SGD)

INPUT: M, the model to be trained
INPUT: D, the training dataset
INPUT: F, the loss function
INPUT: N_e, the number of epochs
INPUT: μ, the learning rate
INPUT: C, threshold for the clipping norm for the gradients
INPUT: n_ϵ, noise multiplier

function SGD(M, D, F, N_e, μ, C, n_ϵ)
  Get N the number of samples of D
  Shuffle the initial dataset $D = \{(x_i, y_i) : \forall i\{1, \dots, N\}\}$
  Initialize some random weights for the model $w^{(0)}$
  for t $\in [1, \dots, N_e]$ **do**
    Randomly select a single sample (x_i, y_i)
    Compute $\overline{y}_i$ the prediction obtained for x_i with M initialized with w^{t-1}
    $g_i^{(t)} = \nabla_{w^{(t-1)}} F(\overline{y}_i, y_i)$ ▷ Compute the gradients of the loss
    $\overline{g}_i^{(t)} \leftarrow g_i^{(t)} / max(1, \frac{||g_i^{(t)}||_2}{C})$ ▷ Clip the gradients
    $\tilde{g}_i^{(t)} \leftarrow \overline{g}_i^{(t)} + \mathcal{N}(0, n_\epsilon^2 C^2)$ ▷ Add Gaussian Noise
    $w^{(t)} \leftarrow w^{(t-1)} - \mu \tilde{g}_i^{(t)}$
  end for
end function

Example 3.7.1 shows a simple example implemented using the *TensorFlow Privacy* Python library in which we train a CNN on a openly available dataset and we compare the performance when using DP-SGD with different values for the noise scaler or noise multiplier.

Example 3.7.1. For this example we are going to use the Alzheimer dataset [101], in which the objective is, given an MRI image, to predict the level of Alzheimer of an individual classified as follows: non-demented, very mild-demented, mild-demented, moderate-demented Specifically, Table 3.1 shows the distribution of the four predictive classes in both train and test datasets. Figure 3.7 shows an image of each class in the dataset.

In view to the distribution of the classes in the initial train and test datasets given in Table 3.1, for this simple example we propose to perform a binary classification on whether the patient presents any type of Alzheimer or not.

We train a DL model after performing a model and hyperparameter optimization using 20% of the training set as validation. The model has been optimized using

Data	*Non-demented*	*Very mild demented*	*Mild demented*	*Moderate demented*	*Total*
Train	2566	1781	724	49	5120
Test	634	459	172	15	1280

Table 3.1: Example: number of data from each class in the train and test datasets.

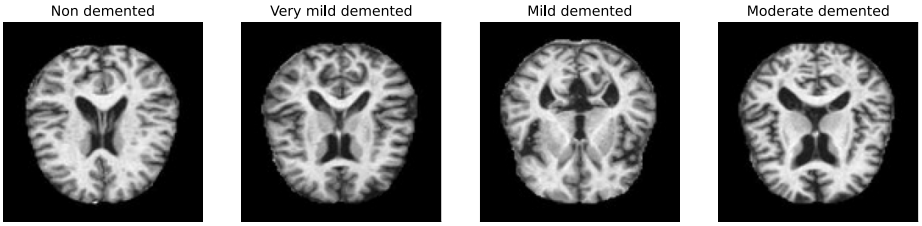


Figure 3.7: Example: samples extracted from the training dataset.

SGD with a batch size of 16. In addition, the loss function is the binary cross-entropy and the evaluation metric is the accuracy. The model is a simple CNN with the following layers:

- **Conv2D layer**. Filters: 32. Kernel size: (3, 3). Activation: *ReLU*. Input shape: (128,128,1).
- **MaxPooling2D layer**. Pool size: (2,2). Strides: 2.
- **Conv2D layer**. Filters: 64. Kernel size: (3, 3). Activation: *ReLU*.
- **MaxPooling2D layer**. Pool size: (2,2). Strides: 2.
- **Conv2D layer**. Filters: 128. Kernel size: (3, 3). Activation: *ReLU*.
- **MaxPooling2D layer**. Pool size: (2,2). Strides: 2.
- **Flatten layer**.
- **Dense layer**. Units: 64. Activation: *ReLU*.
- **Dropout layer**. Rate: 0.1.
- **Dense layer**. Units: 32. Activation: *ReLU*.
- **Dropout layer**. Rate: 0.1.
- **Dense layer**. Units: 1. Activation: *sigmoid*.

For performing the training using DP-SGD we used the TensorFlow Privacy library. We have taken 0.2 as the clipping norm for the gradients (this hyperparameter has been optimized taking into account the validation set) and we have tested different values for the noise multiplier. Table 3.2 shows the results obtained in terms of different metrics such as the accuracy, the precision, the recall, the F1-score and the AUC for these two cases: simple SGD with no DP and the scenario

applying DP-SGD with noise multiplier 0.1. For the case in which we do not apply DP, the models has been training during 10 epochs. However, for the case of DP-SGD, we have run the model during 50 epochs in order to try to reach a better convergence of the training, that is slower due to the noise added. Finally, the number of batches for DP-SGD is 1.

Optimizer	*Accuracy*	*Precision*	*Recall*	*F1-score*	*AUC*
SGD	0.95000	0.94769	0.95356	0.95062	0.98755
DP-SGD	0.68125	0.68536	0.68111	0.68323	0.76060

Table 3.2: Example: results obtained when performing the binary prediction in the test set in terms of different metrics with SGD and DP-SGD as optimizers for the gradient descent process.

In view of Table 3.2 we can note the impact of adding DP in the performance of the training in this very simple example: even when training five times more epochs than in the standard approach, it still shows a lower performance. However, at the same time, with this assumption we can guarantee a level of privacy for the resulting training model that will reduce the risk of extracting information from the training data and the individual represented in the training database.

The example presented above aims to simply show the impact of adding DP in the SGD process, which have been done by means of the *TensorFlow Privacy* Python library [102]. In the following section we present a curated review of different state of the art frameworks for differential privacy.

3.8 Review of Differential Privacy Frameworks

In this section some widely known frameworks for differential privacy and for training ML/DL models under DP assumptions are presented. Specifically, these libraries are listed along with the license under which they are developed as well as a brief summary of their purpose or main features (highlights) in Tables 3.3 and 3.4.

These table summarize some of the state of the art libraries in relation to DP both from the point of view of local and global application on data, queries, statistics or histogram visualization (*Google differential privacy*, *IBM Diffprivlib*, *PyDP*, *OpenDP*, *AutoDP*), as well as from the point of view of their incorporation in the gradient descent process during the training of DL models (*Opacus*, *Tensorflow Privacy*, *FastDP*) or in ML models (*IBM Diffprivlib*, *Sarus DP-XGBoost*).

At the moment, these are the most significant from the point of view of stability, maintainability and popularity, but other libraries are emerging and should be continuously tracked.

Framework	***Highlights***	***License***
Google Differential Privacy [103]	It is a set of libraries that allows to get ϵ-DP and (ϵ, δ)-DP statistics. Laplace and Gaussian mechanisms are implemented together with different privatized versions of statistics such as the mean, variance and quantiles for Go, C++ and Java. It also includes a library for DP accounting, making easy to track the privacy budget fulfilled.	Apache 2.0
Diffprivlib (IBM) [87] [88]	It implements several DP mechanisms for both ϵ-DP and (ϵ, δ)-DP, such as Laplace, Gaussian, Exponential and randomized response. It also allows to get sanitized histograms and to train ML models (classification and clustering) using DP for supervised learning (e.g. decision trees random forest) and for unsupervised learning (e.g. k-Means, PCA).	MIT License
Opacus [104] [105]	It provides support for training PyTorch models with differential privacy. It allows to move from the classic training to the DP-DL one with few code changes using the *PrivacyEngine* function. Provide tools for calculating ϵ given δ (track the privacy budget). Rényi-DP is supported.	Apache 2.0
Tensorflow Privacy [102]	It allows to train DL models including DP during the SGD process by introducing the clipping norm for the gradient and the noise scaler. The most classic TensorFlow optimizers are available including DP. It also provide some *keras* models with DP. In addition, this library provides users with the functionality of estimating the values of ϵ fulfilled according to the parameters introduced from the optimizer side.	Apache 2.0

Table 3.3: Review of differential privacy frameworks (1/2).

Framework	*Highlights*	*License*
PyDP [84]	It aims to be an open-source software for statistical analysis of sensitive data. It provides differentially private versions of different statistics (mean, max, sum, variance, quantiles, etc) using the Laplacian mechanism. Can be used for numeric data for GDP.	Apache 2.0
FastDP [106] [107] [108] [109]	It provides support for including DP in all PyTorch optimizers. Memory-efficient and scalable. Easy to use to a DP-DL setting using the *PrivacyEngine* function. It also provide support for different models, such as the ones available in torchvision and HuggingFace among others. The code is inspired in the Opacus library.	Apache 2.0
OpenDP [110]	It supports Gaussian and Laplace mechanisms. It allows to apply DP to several statistics, to compute sanitized histograms and quantiles and differentially private principal component analysis (PCA). Implemented in Rust, can be used with Python and R.	MIT License
AutoDP [111]	It implements (ϵ, δ)-DP. The Gaussian mechanism is used and it allows to make an accounting also concerning the composition. It put special focus on the use on RDP: implementation, accountant, noise calibrator, etc.	Apache 2.0
Sarus DP-XGBoost [112] [113]	This library arises as a fork of the Python Library XGBoost (which aims to provide efficient gradient boosting methods). Specifically, this framework adds differential privacy to the gradient boosted trees models from XGBoost. It uses different mechanism, such as the Laplace mechanism, applied to the leaf values, or the exponential one for split selection.	Apache 2.0

Table 3.4: Review of differential privacy frameworks (2/2).

3.9 Summary and Conclusions

In this chapter we have carried out a careful review of the state of the art regarding differential privacy techniques. One of the main reasons for introducing this technique is to deal with the problems of attacks that can still be carried out on anonymized databases, such as de-anonymization in *k-anonymity*, linkage attacks, etc. With DP, the idea is to confuse the attacker by means of obfuscation mechanisms based on adding controlled noise with statistical guarantees, while seeking a good trade-off between privacy and utility. In addition, another motivation is its twofold applicability, being either useful during data processing itself or during data analysis or ML/DL model training.

First, we have started from the concepts of input and output privacy in order to introduce the ideas of global and local DP, together with different mechanisms to fulfill these definitions and to have a theoretical guarantee of the privacy level. In addition, some demonstrations on the guarantees of these mechanisms are included.

Next, we have reviewed two state-of-the-art methods in differential privacy: Rényi differential privacy and metric differential privacy or d-privacy. Regarding the first one, we present it due to its theoretical interest for its good performance in cases of composition of multiple mechanisms, making it of special interest in data analytics processes. Concerning the second one, we are interested in analyzing some mechanisms proposed in the literature for the case in which we are dealing with data provided with a geographical location component to be protected, in order to preserve, for example, the location of an specific individual in a radius, but guaranteeing the statistical significance of the data or of the sanitized analysis performed.

Subsequently, all the previous concepts have been summarized in order to analyze the applicability of these approaches in cases where it is desired to apply these techniques directly on a raw dataset for the safe publication of the data (DP for data publishing), as well as in cases where these techniques are to be applied to perform analysis or to train ML/DL models (DP for data analysis). Regarding the latter, the idea of training a DL model with DP by adding DP during model optimization, in particular during the gradient descent process, has been reviewed. In particular, a benchmark example of that is shown to see the scalability and the impact of this technique on the performance of a DL model for an specific medical imaging use case.

Finally, we have reviewed different openly available frameworks for DP, both

applied to the processing of data or queries performed on a set of data, calculation of statistics or graphical representation of data (e.g. via histograms), as well as in relation to the incorporation of DP in the input of DL models by means of the optimizers for *TensorFlow* and *Keras* as well as for *PyTorch*.

This chapter aims to show the potential of differential privacy in relation to security and privacy concerns in data science environments. The direct applicability of DP in the training of different AI models and in data science pipelines can then be demonstrated from two paradigms: applying DP to the data (LDP) and/or to their analysis (GDP) or the training of the models. The last one is particularly interesting for preventing membership attacks. In particular, by adding DP during the training of the models we prevent an attacker from being able to extract information about whether or not a user's data were used in the model's input. Likewise, it can make interference attacks more difficult, as noise has been added during the resulting training. In addition, adding DP on the raw data allows to reduce the risk of sensitive information extraction in case the data are published, shared or simply intercepted in a security breach.

Part III: Privacy Preserving Machine Learning

CHAPTER 4

Federated Learning

Verde que te quiero verde,
verde viento, verdes ramas,
el barco sobre la mar,
el caballo en la montaña.

Federico García Lorca,
Romance sonámbulo,
Romancero gitano

Contents

4.1 Machine Learning turns Distributed 137
4.1.1 Computing Resources-Aware Distributed Learning 138
4.1.2 Privacy-Aware Distributed Learning 141
4.2 Introduction to Federated Learning 142
4.2.1 Motivation 142
4.2.2 Basis of Federated Learning 145
4.2.3 Basic Implementation 148
4.2.4 Types of Federated Learning 151
4.2.5 Aggregation Functions 153
4.3 Open Challenges in Federated Learning 157
4.4 Federated Learning in Production: Drift Detection 160
4.5 Differential Privacy in Federated Learning 164
4.6 Secure Aggregation via Homomorphic Encryption 166
4.6.1 Homomorphic Encryption and Federated Learning 167
4.7 Python Libraries for Federated Learning 171
4.8 Other Distributed Learning Architectures 173
4.8.1 All-Reduce Architecture 173
4.8.2 Ring-All-Reduce Architecture 174
4.8.3 Neighbor Architecture 174
4.8.4 Gossip Learning 174
4.9 Summary and Conclusions 178

Abstract

In this chapter, we shift to focus from the data to the models or analysis. We begin by establishing the basics regarding the distributed machine learning technique classically known as federated learning, in relation to the implementation, advantages, open problems, aggregation strategies and types of federated learning, among others. Different Python libraries for federated learning are presented and analyzed, followed by a review on how take into account drift detection and monitoring when putting this architecture in production. Different privacy issues that arise in relation to the applicability of this architecture, in connection with the integration of already exposed privacy-preserving methods, such as differential privacy (DP), or even homomorphic encryption (HE), are discussed. The implementation of this architecture in an AI as a service platform is presented, detailing different extensions for including privacy enhancing technologies, carbon footprint monitoring and allowing client authentication. Finally, different non server dependent distributed learning architectures are also presented and compared.

4.1 Machine Learning turns Distributed

Data-driven technologies have undergone a rapid growth over the last years, especially thanks to the availability of large volumes of data available to be processed and analyzed (i.e. *big data*). As emphasized in the introduction, artificial intelligence (AI), machine learning (ML) and deep learning (DL) empower a wide range of applications (like artificial vision, natural language processing, speech recognition, anomaly detection, etc.).

Generally speaking, in order to produce such systems, large amounts of labeled data are needed to build a robust application with an acceptable level of accuracy. This means that existing data have gone through a manual or semi-automatic process where it has been labeled by domain experts. These data need to be centralized or aggregated into a single location, in order to be able to consume it to build the model or applications. In some cases, this is not difficult to achieve (for instance on public image datasets that are collaboratively built, like biodiversity datasets published and curated through the Global Biodiversity Information Facility (GBIF) [114], or datasets that are owned by a single entity). However, in some other cases it is difficult or impossible to obtain a good quality and large enough dataset. This is the case of distributed datasets which cannot be aggregated or merged into a single

location, due to a series of different reasons. These difficulties can be caused due to technical limitations, for instance on Internet of Things (IoT) or edge devices with bad connectivity; or due to legal, ethical, privacy or confidentiality concerns.

In this context, it is therefore difficult to extract information and knowledge from data that may be affected by these impediments. In such situations, data are fragmented and distributed among data owners or different locations, without the possibility to further collaborate in order to build a robust ML application, as data cannot be centralized and labeled accordingly. Because of this, the study and further development of different privacy techniques that allow us to operate with distributed data in a secure way, allowing us to extract their full potential, is a key area for scientific research.

In this line, in this section we introduce the concept of *distributed machine learning (DML)*. It is important to note that DML is not only focused on privacy protection, but other interesting goal is to take advantage of the available computing resources by distributing the training of the models or the data in the different available processors. We can thus define two different types of DML according to the desired objective: computing resources-aware distributed learning and privacy-aware distributed learning.

4.1.1 Computing Resources-Aware Distributed Learning

The term distributed computing denotes a model for addressing large-scale computing problems using a large number of distributed computers, workers or nodes organized in clusters integrated in a distributed communication infrastructure. This system is based on distributing the information to different computers or workers, which solve the calculations and once they have the result they send it to a central server which is orchestrating the process.

In many cases, during the development of artificial intelligence models, it is necessary to resort to the use of specialized hardware to speed up the computation. In particular, we can think of two classical approaches to train a model in a distributed way on multiple processors: *data parallelism* or *model parallelism* [115]:

- **Data parallelism**: This is the most classical approach. In this case an horizontal partition of the initial dataset is performed and distributed to the different available workers. The model is then trained on each worker with each data partition. The results of the different workers are then aggregated to create a global model. The idea is similar to that of the federated learning approach. It is interesting to apply this method when the data are very large and do not

fit in a single worker.

- **Model parallelism**: In this case, instead of splitting the data, the model is partitioned. Thus, the dataset is replicated in different workers and in each of them a part of the model is trained. Subsequently, the different parts of the model are aggregated to create the global one. It is interesting to apply it in cases where there we deal with very large models that do not fit in memory in a single node.

Figure 4.1 intuitively shows the ideas behind the distributed training schemes of data parallelism and model parallelism. Specifically, it is shown how in data parallelism a partition of the data is introduced in each worker and the complete model is trained in all the workers, while in model parallelism all the data are shared in all the workers and a partition of the model is trained in each worker.

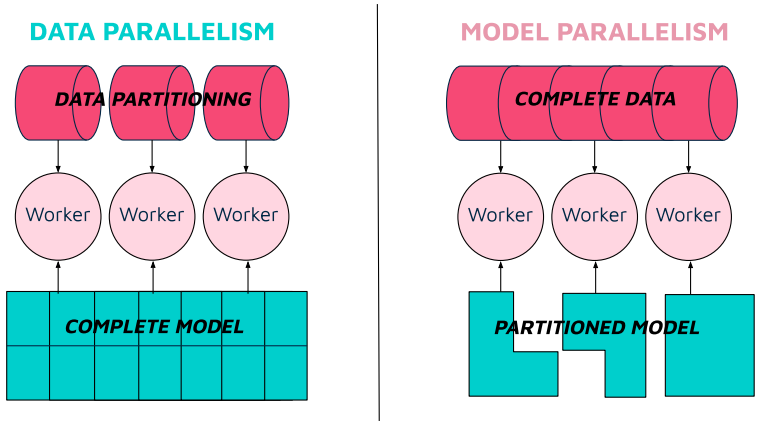


Figure 4.1: Model and data parallelism: intuitive view from the node or worker's perspective.

As already mentioned, the most common approach is to apply data parallelism, since in many cases the data may already be naturally distributed in different nodes. Several frameworks can be considered for distributing training in multiprocessor environments under the scope of ML and DL model development, such as *Horovod* [116], that was originally developed to simplify the parallelization of training scripts for a single GPU and to enable efficient scaling across multiple GPUs on one node or multiple GPUs on multiple nodes. This framework works under the principle presented above of data parallelism, i.e., data are distributed dis-jointly among the different computation units (in this case GPUs), and the model in question is trained on each data split and then aggregated by a master node.

Other notable tools in this sense are the package `distribute` [117] from *TensorFlow* [118] and *PyTorch Distributed* [119]. However, *Horovod* allows scalability to different ML/DL frameworks, as it supports *TensorFlow* and *PyTorch* as well as *keras* and *MXNet*.

Figure 4.2 shows the intuitive idea derived from a training distributed on multiple processors under the data parallelism paradigm.

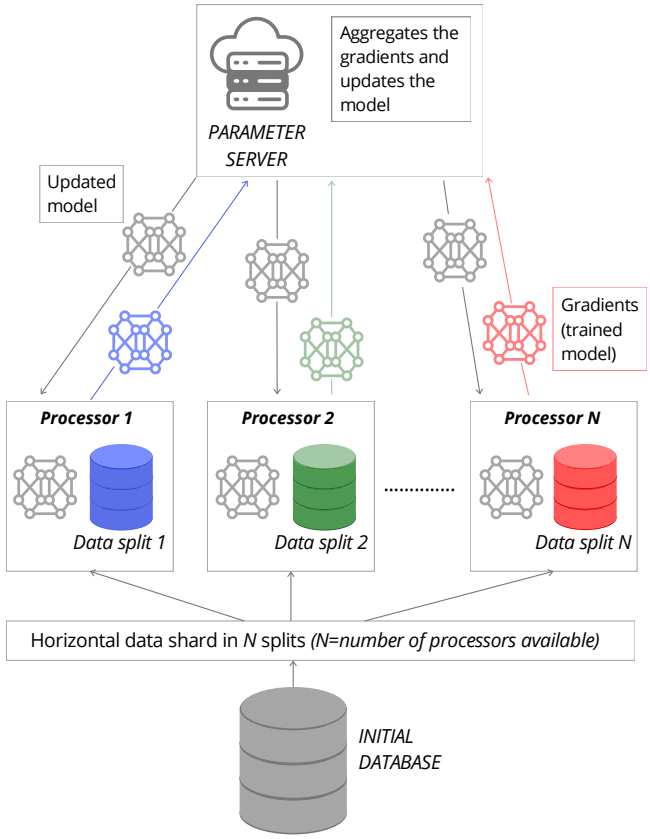


Figure 4.2: Data parallelism scheme: distributing the training across multiple processors.

Note that we can consider two different paradigms, the case where we have a single machine provided by at least 2 GPUs, or the case where we have a cluster with multiple machines, each of them provisioned with at least one GPU.

Regarding the first case, the steps to be followed to parallelize the training using *Horovod* are the following:

1. Initialize Horovod.

2. Pin GPU to be used to process local rank (one GPU per process) .

3. Read and process the data to be trained and create the ML/DL model.

4. When creating the model, modify the optimizer by multiplying the learning rate by the size (hv.size()).

5. Initialize the necessary checkpoints and save them in a directory.

6. Define the training function including Horovod Distributed GradientTape. Then, Broadcast initial variable states from rank 0 to all other processes.

7. Once trained, use Horovod to adjust the number of steps based on number of GPUs.

8. Save checkpoints only on one worker (e.g. worker 0).

While distributed computing motivated by the availability of computing resources is an essential tool for scaling models and processes in environments with large volumes of data and high complexity, these types of methodologies can also present advantages from the privacy point of view. These implications will be explored in detail in the following section and throughout this chapter.

4.1.2 Privacy-Aware Distributed Learning

In view of the above, the idea of using this type of distributed learning architectures for cases in which data cannot be centralized due to privacy issues, as well as to maintain data integrity, naturally arises. Thus, making an analogy with the parameter server presented in Figure 4.2, we can think of an architecture of this style motivated by the availability of distributed data in different nodes or machines, allowing to train the model following a distributed paradigm.

This chapter explores the federated learning (FL) architecture, that emerges as a privacy-aware distributed machine learning solution with the idea of federating training in cases where privacy restrictions apply to distributed data and/or data belonging to multiple clients or data owners.

4.2 Introduction to Federated Learning

4.2.1 Motivation

In this chapter we will explore in depth the *federated learning (FL)* architecture, which can be seen as a specific case of DML where the data are distributed among several machines, workers, or servers. In short, it is a privacy preserving machine learning technique. In order to motivate its adoption, let us start by analyzing the classical centralized model training architecture, to analyze the concerns that arise from the privacy point of view.

Definition 4.2.1. Centralized machine learning. It consists of applying machine learning models in such a way that the data generated by different centers, entities or devices are centralized in a single external central server or data processing center.

The classical centralized machine learning methodology can be summarized as shown in Figure 4.3.

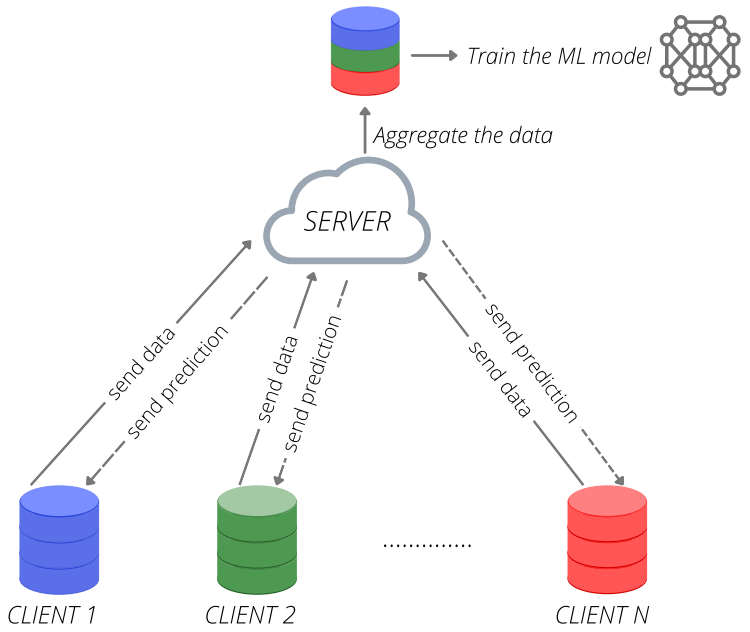


Figure 4.3: Classic scheme of centralized machine learning.

Classically, the ML process in many life areas follows the Cross-Industry Standard Process for Data Mining cycle (CRISP-DM) [120], which consists of six steps:

(1) business understanding, (2) data understanding, (3) data preparation, (4) modeling, (5) evaluation and (6) deployment.

However, there are different drawbacks arising from this methodology, especially (but not only) from the privacy point of view:

- In relation to the privacy and security concerns of the data involved, the information must leave the devices or centers that have generated it and travel to the server or data processing center in charge of carrying out the centralized methodology. This can lead to potential security breaches during data transmission, resulting in security problems, but also in problems from a legal point of view, since in many cases such information cannot be outsourced (for example, this can occur with clinical information).
- Latency problems can occur when working with devices that expect responses from the models in real time, as very fast transmission is required between these devices and the central server.
- Problems related to connectivity, due to the dependence on the Internet for the transmission of information may also arise.
- Resources consumption-related problems derived from the storage, processing and integration of machine/deep learning models on extremely large volumes of data in cases where data from multiple devices and entities are centralized.

Specifically in relation to the last problem presented, the security and privacy of information, we can think about edge computing approach as a potential solution to this issue, as the model can be trained in different data locations without centralizing the data. First, let us introduce the definition of edge computing.

Definition 4.2.2. Edge computing. It is a distributed computing paradigm in which the computing and data storage node is located closer to the actual location that generates the data, with the aim of reducing communication time and therefore latency. An schematic example is given in Figure 4.4.

In this sense, to solve the privacy concerns that may arise from the application of the classic centralized learning architecture presented in Figure 4.3, one may think on the idea of simply performing the individual training with the data from each data owner. Thus, when it comes to systems (such as IoT devices) with sufficient storage and computing capacity at the edge, we can use edge computing, without the need for the data to be sent to a central location. As we will be training

the model individually with the data from each data owner we will be preserving data privacy by not centralizing it. Specifically, once the models have been trained individually with the data captured by the involved data owners, the processing of the data and the training and evaluation of the models can be done on the edge, without compromising data privacy and integrity.

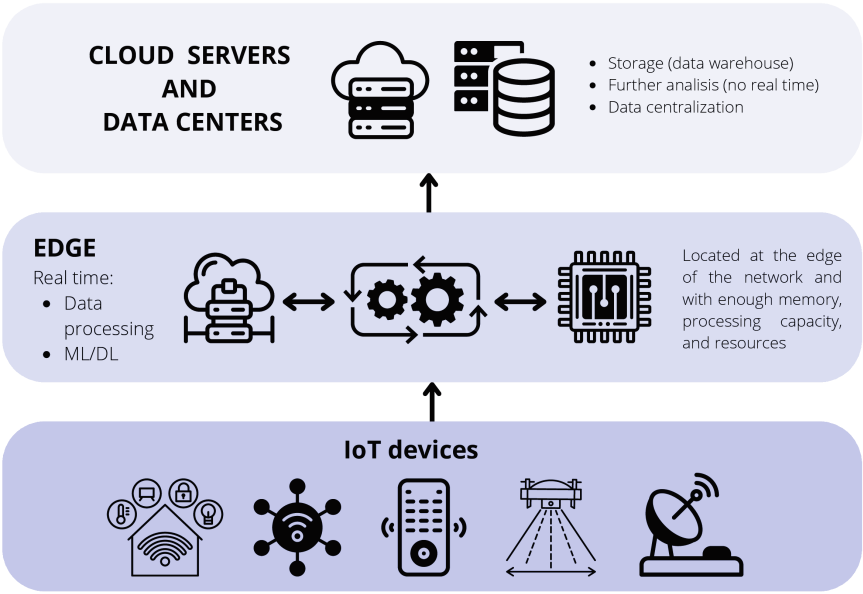


Figure 4.4: Edge computing scheme. (1) The IoT or edge devices capture the data; (2) data are processed and analyzed including the training of ML/DL models at the edge of the network if there is sufficient computing and storage capacity, allowing them to provide real-time response (3) finally, data can be stored in data lakes, data centers and cloud servers for further research, analysis and preservation.

However, by following this approach, in many cases the amount of data needed to build a sufficiently accurate model is not available. In addition, the possibility of collaboration is limited, which prevents the creation of more reliable models that are scalable to new data. Therefore, the need arises to introduce the federated learning paradigm, which allows training a global model on distributed data, seeking a balance between privacy and performance.

In this sense, we want to preserve the privacy offered by training locally on each data owner, without compromising the good performance and generalization capability typical of the centralized approach.

4.2.2 Basis of Federated Learning

The concept of federated learning (FL) first emerged in 2017, introduced by McMahan [121]. Specifically, it follows a similar idea to the one already presented in Section 4.1.1 when introducing the distributed machine learning architecture known as parameter server.

Definition 4.2.3. Federated learning (FL). It is a distributed machine learning technique in which a central server orchestrates the communication with the different clients participating in the architecture. Each of them trains a local model (provided by the server), and the server aggregates the resulting models to create a global one, which is then distributed to the clients. Thus, the training is performed without the need for data sharing or centralization, allowing the collaboration of different entities or devices among others.

FEDERATED LEARNING SCHEME

1. The central server creates a model for addressing the common task of all the involved clients.
2. The server distributes such model with all the clients participating in the training process.
3. Each client trains the model locally with its own data (in parallel for each client).
4. Each client sends to the central server the weights that define the model once it has been trained locally.

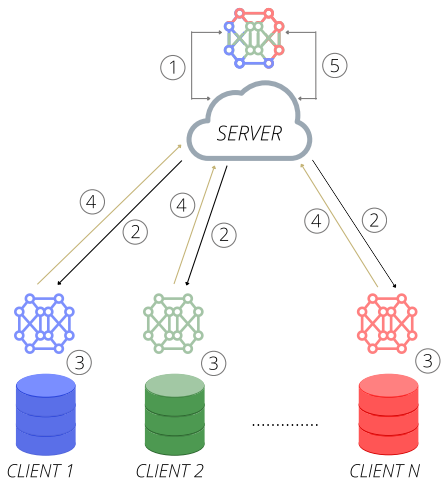


Figure 4.5: Federated learning scheme. Extracted from [6].

5. Once the server has received the weights of the clients involved, it applies an aggregation function to build a global model from the weights of all the clients.
6. Repeat from step 2 as many rounds as needed.

For step 5 of the previous scheme, the aggregation function to be used must be defined previously. As will be explained in Section 4.2.5, several strategies can be considered. Let us start by giving a formal definition for this concept and by introducing one of the most widely used aggregation strategy in federated learning, *FedAvg*.

Definition 4.2.4. Aggregation function (in FL). It is a mathematical function that combines the weights obtained after training each client on the initial model.

In federated learning the aggregation of the models to build a global one (as stated in definition 4.2.4) is classically implemented based on the weighted average of the weights defining the models (*FedAvg*). This weighting can be done, for example, according to the number of data that each client has in order to train the model locally. It is essential to take this into account, since let us suppose a use case in which one client has 1000 data, while another has 100, as a general rule, a higher weighting should be given to the first one with respect to the second one. Therefore, we calculate the weighted average of the weights that define the model after training it locally on each of the clients as follows: Be n the number of clients in the training process, n_i the number of data of client i $\forall i \in \{1,\dots,n\}$. Be $w_i^{(r)}$ the weights of the model obtained after training in round r with the client i $\forall i \in \{1,\dots,n\}$. The, we can define the aggregation function of the weighted average as follows, with $w^{(r)}$ the aggregated weights in round r:

$$w^{(r)} = \sum_{i=1}^{n} w_i^{(r)} x_i, \text{ with } x_i = \frac{n_i}{\sum_{j=1}^{n} n_j} \qquad \forall i \in \{1,\dots,n\} \tag{4.1}$$

The scheme of the classic federated learning architecture is presented below, along with an illustrative scheme (see Figure 4.5).

The main advantage that can be extracted from the application of the FL architecture comes from the privacy perspective. In the FL approach multiple clients can collaborate by building an ML/DL model without the data having to leave the data center, device or institution that generated or owns it. This reduces the risk of these data being intercepted by attackers, as it does not have to be shared with third parties or externalized even for training purposes.

When implementing this technique, it is essential to be aware not only of the potential advantages with respect to the centralized approach, but also of the conflicting aspects and open problems that must be taken into account. For this purpose, a list of these potential issues is given in the reminder (and will be further explored in Section 4.3.

ADVANTAGES

- The data should not leave the data center, terminal or institution that owns or generated it, maintaining privacy by not having to move even for training the models.
- The latency problem is reduced (each client performs the training locally), as well as the connectivity dependency. Although these problems still exist for computing the weights and obtaining the new updated models, in this approach each client has the model locally.
- When dealing with small models, FL can help to reduce the energy consumption as the training is distributed. Although energy savings are not always obtained with FL, this can occur, for example, when distributing data among several nodes requires the use of lighter processors than when they are centralized. However, the costs of communication and the need to have multiple machines must be taken into account.
- The training time of the model is reduced, since it is trained in parallel with smaller volumes of data than if it were centralized.
- Increased data privacy and security, as the information does not have to leave the device/entity that generated and stores it.

OPEN PROBLEMS

- We may encounter problems in relation to the intermittency of certain client(s) (e.g. due the connectivity or availability issues). That is, there may be clients who are available at the right time to send their weights, as well as new clients who join the training.
- From the weights that are computed we can extract information about the data with which they were trained, then it may be necessary to apply further security restrictions, as will be exposed later.
- The server must be trusted, as it could extract information about the different clients through their models/weights.
- There may be cases in which the data from the different clients cannot be used for the same purpose, due to different in the distribution or the data capturing methods among others. This may led to intentional or unintentional poisoning attacks.

4.2.3 Basic Implementation

A basic implementation of the federated learning scheme can be carried out through the construction of the client and server classes. As explained in the previous section, the client class will be responsible for training with the local data, and then transmitting the resulting weights to the server. Thus, the server will be in charge of creating an aggregated global model and sending it back to the clients.

In a simple way to better understand this architecture, let us suppose that each client sends to the server the trained weights via an encrypted communication. The server can access it for each client and then aggregated them. Once they are aggregated, the server will update the model, which will be sent back to each client.

Firstly, let us show a pseudocode concerning the implementation of the Client class in Algorithm 4. For simplicity, in this example we will assume that each client sends to the server a path where its trained weights are stored (and in the same way, the server returns a path to the aggregated weights in each round).

Algorithm 4 Class Client

INPUT: x_train, train data
INPUT: y_train, train label
INPUT: x_test, test data
INPUT: y_test, test label

1: **function** __INIT__($self$, x_train, y_train, x_test, y_test)
2: $self.x\_train \leftarrow x\_train$
3: $self.y\_train \leftarrow y\_train$
4: $self.x\_test \leftarrow x\_test$
5: $self.y\_test \leftarrow y\_test$
6: **end function**

INPUT: $model$, model to be trained (neural network)
INPUT: $epochs$, number of epochs for training the model
INPUT: $batch\_size$, batch size
INPUT: $path\_weight$, path where the weights of the trained model will be saved

1: **function** TRAIN_MODEL($self$, $model$, $epochs$, $batch\_size$, $path\_weights$)
2: $model.fit(self.x\_train, self.y\_train, epochs = epochs, batch\_size = batch\_size)$
3: $model.save\_weights(path\_weights)$
4: **end function**

Note that in Algorithm 4, in the `train_model` function we can evaluate error or precision metrics (such as the MRE or the accuracy) for debugging.

Secondly, in the pseudo-codes of Algorithms 5 and 6 we present an intuitive implementation of the server following the assumptions discussed at the beginning of this section. Specifically, the server will have the data number of each client as well as a copy of the initial model. It will be able to access the weights of each locally trained model (via the path in which each client stores it), and aggregate the models, in order to create the global model by updating the initial one with the aggregated weights (in this particular example *FedAvg* is used for the aggregation). At the end of each round, the aggregated model should be passed as the initial model to the different clients to repeat the process as many times as necessary.

It should be noted that this is simply an intuitive idea that could be applied, for example, to test the federated architecture with all clients and the server deployed on a single machine. However, to implement this architecture in a real example, it will be necessary to use secure communication protocols between the clients and the server to transmit the weights/models. For this purpose, there are multiple Python libraries that can be used, as will be detailed in the Section 4.7. In addition, a detailed implementation together with further functionalities for securing the FL process will be presented in Appendix C.

Algorithm 5 Class Server (1 out of 2)

INPUT: num_data_client, list with the number of data of each client
INPUT: $model$, model to be trained (neural network)

function __INIT__($self$, num_data_client, $model$)
    $self.num\_data\_client \leftarrow num\_data\_client$
    $self.model \leftarrow model$
end function

OUTPUT: copy of the model to be trained

function GET_MODEL($self$)
    return $copy(self.model)$ ▷ Return a copy of the model
end function

OUTPUT: list with the weighting of each client based on the number of data

function GET_XI($self$)
    $x \leftarrow \frac{self.num\_data\_client}{\sum self.num\_data\_client}$
    return x
end function

Algorithm 6 Class Server (2 out of 2)

INPUT: $path\_weights$, list with the path to the weight of each client
OUTPUT: list with weights of each client after training

function GET_WEIGHTS_CLIENTS($self$, $path\_weights$)
    $model\_copy = copy(self.model)$ ▷ Copy of the model
    $client\_weights \leftarrow [\ ]$ ▷ List with the weights of each client
    for $path \in path\_weights$ **do** :
        $model\_copy.load\_weights(path)$
        $client\_weights \leftarrow client\_weights + model\_copy.get\_weights()$ ▷ Weights are added to the list
    end for
    return $client\_weights$
end function

INPUT: $path\_weights$, list with the path to the weight of each client
OUTPUT: aggregated weights after training locally all the clients

function GET_AGG_WEIGHTS($self$, $path\_weights$)
    $x \leftarrow self.get\_xi()$
    $client\_weights \leftarrow self.get\_weights\_clients(path\_weights)$
    $agg\_weights \leftarrow [\ ]$ ▷ Empty list where the aggregated weights will be stored
    for $j \in range(len(client\_weights[0])$ **do**
        $peso \leftarrow 0$
        for $i \in range(len(xi))$ **do**
            $tmp \leftarrow tmp + x[i]client\_weights[i][j]$ ▷ Weighted average
        end for
        $agg\_weights \leftarrow agg\_weights + tmp$ ▷ Add tmp to the list
    end for
    return $agg\_weights$
end function

INPUT: $path\_weights$, list with the path to the weight of each client

function UPDATE_MODEL($self$, $path\_weights$)
    $model\_copy \leftarrow copy(self.model)$
    $agg\_weights \leftarrow self.get\_agg\_weights(path\_weights)$
    $model\_copy.set\_weights(agg\_weights)$
    $self.model \leftarrow model\_copy$
end function

4.2.4 Types of Federated Learning

Regarding the types of federated learning, we can distinguish on the one hand, according to the type of clients involved, and on the other hand, according to the data held by the clients. In the first case, we are interested in taking into account the differential conditions that clients may have if they are institutions, entities, or if they are devices connected online. Therefore, we will distinguish between cross silo and cross device FL. On the other hand, with regard to the type of data and the features held by clients, we will distinguish between horizontal and vertical FL.

4.2.4.1 Cross Silo and Cross Device Federated Learning

Regarding the type of clients or data owners that participate in a federated training, we can distinguish between *cross silo FL* or *cross device FL*.

One the one hand, in the cross silo approach the clients are usually institutions, organizations or companies seeking to collaborate to build a global model. The particularity of this case is that the number of participating clients is usually limited, and all ideally want to collaborate in all rounds of the training.

An example of this approach could be different hospitals collaborating to build a ML/DL model to predict whether or not a patient has a certain disease based on a diagnostic test, so that a global model is built using data from multiple patients from different hospitals without sharing the raw data (of sensitive nature, as it refers to patients). Another example could be the collaboration between different organizations or public entities, such as the different statistical agencies of different countries or regions to perform a global analysis using ML/DL models of their data without centralizing or sharing them, which is not possible in many cases due to legal restrictions.

On the other hand, we have the cross device FL approach, where the collaborating clients in the FL architecture are internet-connected devices. By their very nature, in this case there will be a high number of clients participating in the training, although many of them will not be available in some rounds due to connectivity, availability or latency issues. It should be noted that the devices that can participate in this type of architectures are usually mobile phones, IoT sensors, etc. We have to take into account that such devices in some cases may be offline or present slow or connections [121]. Then, in some real scenarios only 10% of the available clients are taken [122].

An example of this type of cross device federated learning architecture is the classic collaboration of different mobile phones or devices to build a model to pre-

dict the next word the user is going to type using the predictive keyboard. In this sense, the different devices would train the model locally, without the need to share data, applying a FL architecture with a server orchestrating the process. In this case we work with thousands of clients, and we will encounter situations where different devices lose connection and do not send their updates in all rounds.

Another example could be the case of multiple edge devices that collect data and have the computational capacity to analyze them in situ, and due to technical issues of privacy of the collected data they cannot share or centralize them, but they can collaborate in a distributed architecture with other sensors with the same type of data (and with the same characteristics and labels).

A third scenario that is currently emerging is the *cross institution FL* approach, and consists of applying FL on data distributed in different locations (both physical and virtual), belonging to the same institution or organization (an example can be found in [123]).

4.2.4.2 Horizontal and Vertical Federated Learning

Up to now, whenever we have referred to federated learning, from the preliminary Definition 4.2.3, and the scheme presented in Figure 4.5, we have always taken into account the horizontal federated learning architecture.

The *horizontal FL (HFL)* architecture is most intuitive and common approach to FL. It consists of considering the data of all clients with the same features, for example, this is the case of structured data, the data of all the clients will all have the same columns. Another example, if we think about image classification, all clients would have the same type of images and labels.

Regarding *vertical FL (VFL)*, in this case the different clients have data with different characteristics, but with the same identifier. For example, is the case of several institutions which have data from the same users, but each of them has information about different features. Note that the number of clients available will normally be lower than in the previous case. In order to better understand this architecture, the schematic idea on how the different clients should behave is presented in Figure 4.6. Note that in Figure 4.6 we have N clients, all of them having data from the same M individuals or IDs. For each records, each client have a different number of features (e.g. client 1 has n features, client 2 has m features, client N has k features, etc).

From this section on-wards, when we mention the concept of FL, we will always refer to HFL, keeping the notation FL for the sake of simplicity.

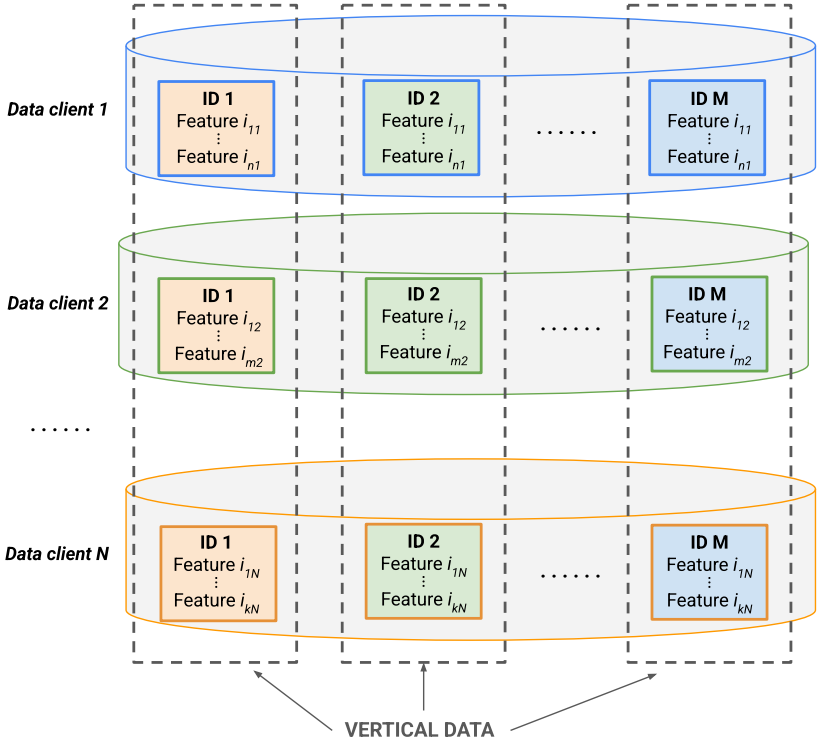


Figure 4.6: Schematic idea of the distribution of the data in the different clients of a VFL architecture. Adapted from [6].

4.2.5 Aggregation Functions

One of the most important steps in an FL architecture is the point at which local models are aggregated to build a global model. This aggregation can in many cases be impaired when dealing with heterogeneous clients with high divergence, both from the point of view of unbalanced data in the different clients or due to heterogeneous class distributions (in cases of classification problems). Therefore, analyzing the most suitable aggregation strategy in each case is a key step.

In addition to the classic aggregation function defined within the simple *FedAvg* or Federated Average approach, there are many strategies that can be considered when performing the aggregation process in an FL architecture. For example, if we are dealing with non-i.i.d. clients, we may find other options that are more convenient including an optimization process or regularization terms.

In the following we present the main ideas behind some of the state of the art

aggregation functions that can be used on the server side in this type of architecture.

4.2.5.1 Federated Average (FedAvg)

As already explained, this is one of the most intuitive ideas that comes to mind when carrying out aggregation: perform a weighted average. Despite its simplicity, it is an effective and widely used method in the literature. Other aggregation functions used in the state of the art are shown below.

4.2.5.2 Federated Average with Momemtum (FedAvgM)

This aggregation function is a modification of the classic *FedAvg* strategy in which the concept of momemtum is introduced [124]. Be $w^{(r)}$ the weights in round r and Δ denotes the difference between two consecutive weights. Be $v^{(r)}$ the momentum vector in round r (initialized as $v^{(0)} = 0$), then, we can calculate the momentum vector in round $r+1$ as $v^{(r+1)} = \beta v^{(r)} + \Delta w^{(r+1)}$ being β the momentum. The aggregated weights in round $r+1$ can be calculated introducing the momemtum as $w^{(r+1)} = w^{(r)} - v^{(r+1)}$.

4.2.5.3 Federated Median (FedMedian)

The Federated Median strategy (*FedMedian*) was proposed in [125] and the proposed algorithm (based on using the median) takes into account byzantine machines so that only in the case of normal worker machines each client computes the local gradient.

Specifically, the aggregated weights for round $r+1$ can be calculated using this strategy as follows: Be $w^{(r)}$ the weights in round r, the central server will be in charge of getting the median of the local gradients in each round r ($g_{median}(w^{(r)})$). Then, once such aggregated gradient is calculated, the weights are updated for round $r+1$ as: $w^{(r+1)} = w^{(r)} - \mu g_{median}(w^{(r)})$ [125], being μ the step-size or learning rate.

4.2.5.4 FedProx

The main objective of the *FedProx* aggregation strategy is to deal with heterogeneous settings in FL schemes [126]. Specifically, it is a generalization and reparametrization of the *FedAvg* strategy. In order to calculate $w_i^{(r+1)}$ (the weights for client i in round $r+1$) we have to minimize the following objective function h_i for each client i, with $F_i(\cdot)$ the global objective function at each client i:

$$h(w, w^{(r)}) = F_i(w) + \frac{\mu}{2}||w - w^{(r)}||^2. \tag{4.2}$$

Thus, $w_i^{(r+1)} = arg\min_w h_i(w)$. Then the server aggregates the values $w_i^{(r+1)}$ $\forall i \in \{1, \ldots, n\}$ by taking the mean.

4.2.5.5 Federated Optimization (FedOpt, FedAdam, FedYogi and FedAdagrad)

Adaptive Federated Optimization (*FedOpt*) [127] is an aggregation function that aims to improve *FedAvg* by using gradient-based optimizers with given learning rates that must be customized both from the client and server side. Be $w^{(r)}$ the global weights at round r, $w_i^{(r)}$ the weights for client i, then we can define $\Delta_i^{(r)} = w_i^{(r)} - w^{(r)}$ and $\Delta^{(r)} = \frac{1}{n}\sum_{i=1}^{n} \Delta_i^{(r)}$. From the server side, we will use an optimizer for calculating $w^{(r+1)}$ given $w^{(r)}$, $\Delta^{(r)}$ and the learning rate.

Note that *FedOpt* allows the use of different adaptive optimizers in the server side, such as *Adam*, *Adagrad* or *Yogi* (*FedAdam*, *FedAdagrad* and *FedYogi*). The adaptive methods are used on the server side while SGD is used from the client side [127].

These strategies require additional parameters, such as the server-side and client-side learning rate, β_1 and β_2 as momentum and second momentum parameters respectively and τ for controlling the degree of adaptability of the algorithm.

4.2.5.6 Scaffold

This strategy stands from Stochastic Controlled Averaging for Federated Learning [128]. The idea is to use control variates to correct for the client drift. This helps to stabilize the local updates and thereby to improve the convergence rates. More details in client drift and how to mitigate will be given in Section 4.4.

4.2.5.7 Ditto

The idea behind the ditto aggregation strategy [129] is to incorporate a regularization term that allows the individual models to approximate the optimal aggregated model. This function seeks that instead of optimizing the loss function, what it optimizes is the loss function plus the regularization. Specifically, this method works by personalizing the learning objective for each client.

4.2.5.8 A Novel Aggregation Strategy: *FedAvgOpt*

The idea of the FedAvgOpt strategy was first proposed in the following preprint: ***Sáinz-Pardo Díaz, J.*** *& López García, Á. (2025). Enhancing the Convergence of Federated Learning Aggregation Strategies with Limited Data. arXiv, 2501.15949 [130].*

The main idea when working with federated learning architectures or different variants of this distributed approach (such as the ones that will be presented in Section 4.8), is to build robust global models that are built from the individual models trained with data from different data owners.

In this line, we propose *FedAvgOpt* as an alternative to the *FedAvg* method to aggregate the weights of different clients in a federated learning architecture and build a global model. This idea was first introduced in [130] together with a use case on medical imaging in order to showcase its performance and robustness.

Be n the number of clients, n_i the number of data of client i $\forall i \in \{1,\dots,n\}$. Be $\{w_i : i \in \{1,\dots,n\}\}$ the weights obtained for each client i after the individual local training. Be w_{FedAvg} the vector with the weights aggregated using *FedAvg*.

$$w_{FedAvg} = \frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i \tag{4.3}$$

In order to improve the convergence of this method, we propose the following approach: let α be a vector such that $\alpha in \mathbb{R}^n$, we define $w_{FedAvgOpt}$ the aggregated weights using the *FedAvgOpt* strategy as follows:

$$w_{FedAvgOpt} = \frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i \alpha_i, \tag{4.4}$$

so the value of α is calculated by solving the following optimization problem:

$$(P) \begin{cases} min \sum_{j=1}^{n} || \frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i \alpha_i - w_j ||_2 \\ s.t.\ \alpha \in \mathbb{R}^n \end{cases}, \tag{4.5}$$

The function to optimize $f(x)$ is defined as follows, being a continuous function. Let us check whether in addition $f(x)$ is coercive and we can therefore guarantee the existence of a global solution for (P):

$$f(x) = \sum_{j=1}^{n} || \frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i x_i - w_j ||_2 = \sum_{j=1}^{n} || w_{FedAvg} x - w_j ||_2 \tag{4.6}$$

A function f is coercive if $f(x^{(k)}) \to +\infty$, $\forall (x^{(k)})_{k\in\mathbb{N}} \subset \mathbb{R}^n$ such as $||x^{(k)}|| \to +\infty$. It is immediate that if $w_{FedAvg} \neq 0$, $f(x^{(k)}) \to +\infty$ if $||x^{(k)}|| \to +\infty$. Then

we can conclude that f es coercive in this case because, under general conditions the aggregated weights obtained using *FedAvg* are non-zero. Therefore, there is a global solution for (P).

In addition, $f(x)$ is convex because it is the sum of convex functions (the norm of a linear function is convex). Therefore, every local solution of (P) is a global solution.

To solve (P) we can use the Nelder-Mead method, which allows one to minimize a function in a multidimensional space.

Additionally, we are interested that the norm defined on the function $f(x)$ is bounded by the norm of the sum as follows:

$$(P_1)\begin{cases} min \sum_{j=1}^{n} \frac{||\frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i \alpha_i - w_j||_2}{||\frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i \alpha_i + w_j||_2} \\ s.t.\ \alpha \in \mathbb{R}^n \end{cases}, \tag{4.7}$$

Then, be $g(x)$ defined as follows:

$$g(x) = \sum_{j=1}^{n} \frac{||\frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i x_i - w_j||_2}{||\frac{1}{\sum_{i=1}^{n} n_i} \sum_{i=1}^{n} w_i n_i x_i + w_j||_2} = \sum_{j=1}^{n} \frac{||w_{FedAvg} x - w_j||_2}{||w_{FedAvg} x + w_j||_2} \tag{4.8}$$

Note that $g(x)$ is continuous in the domain in which it is defined because it is the finite sum of continuous functions.

The proposed aggregation strategy we call *FedAvgOpt* is the one in which we aggregate the weights after optimizing the problem (P_1).

Specifically, we have implemented this aggregation function as a strategy using the Python library *Flower* [131], with the code detailed in Appendix B.

4.3 Open Challenges in Federated Learning

Part of this section has been published in: ***Sáinz-Pardo Díaz, J.*** *& López García, Á. (2023). Study of the performance and scalability of federated learning for medical imaging with intermittent clients. Neurocomputing, 518, 142-154. [6].*

Due to the distributed nature of this technique, there are challenging aspects of FL compared with a traditional centralized ML/DL approach [132]:

- **Expensive communication.**

  Communication (both in terms of privacy and performance) between the server and the client is the most critical point of the federated learning process. One

the one hand, each time the process is repeated, the client sends to the server a new update of the model weights. It must be noted that the difference between the updated weights and the previous ones can cause information to be extracted from the data used for training. It is therefore crucial that this communication is encrypted to ensure its privacy and security. In addition, Differential Privacy (DP) or Homomorphic Encryption (HE) techniques can also be applied to ensure privacy [133, 134]. On the other hand, network performance can be a bottleneck when the model updates are large, therefore a compression or reduction of the information transferred back to the server should be performed in order to save bandwidth or to maximize the chances of a successful transfer when the network quality is low. In this regard, techniques such as data compression, dimensionality reduction or other techniques [121] are used to implement a more efficient communication method.

- **Systems heterogeneity.**

  Another major challenge, caused by the heterogeneity of the underlying systems, is due to the fact that devices can be intermittent throughout the learning cycle. It may happen that one of the clients involved in the training, is idle at a given round and cannot send the corresponding model updates [135]. Initially the number of clients may be limited, and other ones may decide to join the collaboration once started, while others may decide to abandon it, e.g., due to lack of new data or even privacy issues or connectivity problems. This must be taken into account when design the architecture, in order to define the protocol to be followed if this happens (e.g., define a maximum waiting time for clients to send their model updates, decide whether their updates will be removed for future rounds if they leave the system, etc.)

- **Statistical heterogeneity.**

  Another aspect that can affect the training process of a federated learning scheme is the distribution of the data among the different clients. In particular, the fact that these may be non-i.i.d. is a point to be taken into account. In the classic FL approach (horizontal FL, which will be explained later), we usually refer to non-i.i.d. data in terms of the skew of the different labels. For example, in extreme cases where there are clients who do not have all possible categories represented or only have one of them, the overall model may not converge. These problems can also be notably observed when the distributions of the clients are too far away from the distribution of all data globally. The classic *FedAvg* method for aggregating the weights using the weighted mean is quite sensible for non-i.i.d.

data [136].

However, other approaches can be studied in order to analyze whether the data are non-i.i.d. or homogeneous, not only the distribution of the labeling of the data. Although the intuitive idea is to analyze how different the $\mathscr{P}_i$ and $\mathscr{P}_j$ distributions of the labels in clients i and j are, other options that can make the data non-homogeneous are: imbalance in the number of data (clients with much more data than others), differences in the distribution of the features, or the fact that very different features have the same label.

- **Privacy concerns.**

  Apart from the technical challenges that stem from the distributed nature of federated learning described previously, a FL system poses several privacy concerns, that we will describe bellow. It is important to note that FL can be seen as an example of Privacy Preserving Machine Learning PPML method, as we are not sharing data for training. Note that this term refers to machine learning models which incorporate of defensive measures to protect user privacy and data security. However, in some cases, to prevent different types of attacks it is not enough to apply a FL architecture, but additional privacy measures are required. For example, DP, which was explored in Chapter 3, or HE, which will be presented in following sections, can be seen as Privacy Enhancing Technologies (PETs), that can contribute to build PPML systems.

  It should be taken into account that, as will be explored in Section 4.5, we can deal with cases in which the server is not trusted (the clients do not have full confidence in it), and therefore it is interesting to apply DP during the training of the model or on the weights before its transmission. Conversely, it may happen that the server is trusted but not all the clients, and in that case it may be interesting to apply DP on the aggregated model, so that in case of dealing with an attacker acting as a client, a privatized version of the aggregated model will be received.

  Applying this type of privacy enhancing techniques is fundamental during a privacy preserving data science process that involves the usage of sensitive data, since, specifically in machine learning, attacks can occur at any stage of the process, from data publication to model training and inference. In the case of federated learning, one of the main privacy issue concerns the reconstruction attacks. Moreover, during model training, the most common attacks are, in addition to reconstruction attacks, membership inference attacks, model inversion attacks or model poisoning attacks [137, 138]. Let us briefly state what these attacks

consist of:

- **Reconstruction attacks**: in this type of attack, an adversary can reconstruct the original data by intercepting intermediate information [139]. For example, in the case of federated learning, this intermediate information could be the model weight updates or gradient information obtained before each client trains the model locally. Then it is important to apply security measures during training, such as SMPC or HE so that this intermediate information is kept private [137].
- **Membership inference attacks**: in this case, the attacker wants to determine, given a machine learning model, whether or not a certain individual has participated in the training of the model [140]. The most common approach to deal with this type of attack is the application of DP algorithms [137].
- **Model inversion attacks**: in this type of attack, the adversary's goal is to extract training data or feature vectors from the model's training data [137]. Be a model M which takes as input a vector $(x_1, \ldots, x_n)$ from an n-dimensional feature space (e.g. from $\mathbb{R}^n$), and be y the output obtained when predicting: $y = M(x_1, \ldots, x_n)$. In the model inversion attack the adversarial uses black-box access to the model M to extract some feature x_i, $i \in \{1, \ldots, n\}$, given some knowledge about the other features and the value y [141].
- **Model poisoning attacks**: in this type of attack, the objective of the adversary is to affect the learning model during the training process [142]. In the case of federated learning, this could be done by a client that enters deliberate wrong labels for its data in order to damage the overall accuracy of the model, since it would be using incorrect data. The simplest and most straightforward way to prevent this attack is to allow only verified data owners with pre-established trust to participate in the training.

4.4 Federated Learning in Production: Drift Detection

One of the major issues we address when deploying federated learning architectures is the previously mentioned poisoning attacks. In many cases these attacks may be intentional by external parties seeking to damage the overall accuracy of the model

by corrupting it. However, in many other cases this problem derives from the fact that data from different clients are too different from each other to be used in the same federated architecture.

As an example, if we think of a use case concerning medical images that may come from different hospitals, it may happen unintentionally that the way in which the images are taken is so different that they cannot be combined to build an accurate global model. This may be due for the devices used for collecting the data, the data processing followed, etc. In order to preserve the privacy of the data of the different clients, it is not feasible that the server can access even a part of the data to make these preliminary checks before including a new client in the federated scheme. Therefore, in order to try to mitigate this problem, it is necessary to analyze the non i.i.d. nature of the data from the different clients according to the weights and/or gradients they sent to the server.

In this line, the need arises to introduce the concept of data drift. This concept addresses the fact that the data used to train a machine or deep learning model may shift their trend or distribution over time. It is therefore essential to analyze this possible drift or change in the data to determine whether the model needs to be retrained with updated data.

Frequently, when facing a problem involving the use of machine learning, it is assumed that the same concepts learned during the training phase will continue to be valid at the time of inference. However this is not always true and concept drift can occur [143]. Specifically, three variants of this concept are usually studied (see [144] and [145]). Suppose two time points t_i, t_j, with $t_i \neq t_j$, $X \in \mathbb{R}^{nxm}$ the features space (assuming m features) and $y \in \mathbb{R}^n$ the labels:

- **Concept drift:** is produced when there is a change in the joint probability $\mathbb{P}(X, y) = \mathbb{P}(y|X)\mathbb{P}(X)$, so it can be defined as $\mathbb{P}_{t_i}(X, y) \neq \mathbb{P}_{t_j}(X, y)$.
- **Virtual drift:** is defined as $\mathbb{P}_{t_i}(X) \neq \mathbb{P}_{t_j}(X)$ and $\mathbb{P}_{t_i}(y|X) = \mathbb{P}_{t_j}(y|X)$, as it refers to the case in which there is a change in $\mathbb{P}(X)$ that does not influence $\mathbb{P}(y|X)$.
- **Data drift:** in this case we only focus on data, assuming that there are no available labels y, so it can be defined as $\mathbb{P}_{t_i}(X) \neq \mathbb{P}_{t_j}(X)$ (a change in $\mathbb{P}(X)$).

Drift detection methods can be classified according to the type of drift they can detect and also by the form of detection: supervised, semi-supervised or unsupervised. The latter are especially useful for data drift detection, so they are the ones from which we will extrapolate their use for our case of data labeling drift detection.

The collaboration derived from the application of a federated learning architecture requires a clear consensus on the data used to train the models. Suppose a medical imaging use case, if different devices are used to capture the images, there may be a drift between the data of some clients and others, because they present significant differences to enter a global model with the participation of both. In this case data standards and interoperability present one of the biggest barriers that can be found when training deep learning models [146].

In this section, we are interested to take advantage of drift detection techniques to prevent poisoning attacks. This type of attack can be done intentionally or unintentionally. In the first case, it is an attacker acting as a client sending maliciously modified model updates to damage the overall performance of the model during aggregation. In other cases, it may be due to the client using training data that differs from that expected or used by the other clients. Returning to the case of medical imaging, this can happen in the collaboration of different institutions taking imaging data with different equipment, or suffering from wear and tear of the equipment itself over time.

In the first approach, we aim to avoid this type of attack when it occurs intentionally for malicious purposes. In a scenario where the server can access data from all clients, it would be possible to apply classical data drift detection techniques (such as those presented in [143]) to detect if the data from any client should not be used in the training because they would damage the global performance by presenting important divergences. However, the potential of federated learning resides in the fact that there is no need to share raw data while maintaining data privacy. In this sense, we propose the possibility that, once the first weights are received after training the initial model locally on all the clients, one server performs a divergence analysis of the models using the weights that define them [147]. In this first inspection, the inclusion of any client could be rejected in case of significant differences compared to the rest, or even clustering algorithms could be applied to group different clients in subgroups to run federated training in small clusters. This selection can be done by measuring the distance between the weights of the clients between them and with a control dataset available from the server side. In this case, as to initialize the training we will have to have a client acting as a leader, it can provide the server with synthetic or similar data to perform the control. To measure the distance, different metrics can be used, such as cosine similarity, Chebyshev distance, Minkowski distance, or Hamming distance among others [148]. Note that in view of the above, the use of informed gossip learning [149] or other peer-to-peer architectures [150] (see Section 4.8), so that the clients with which each entity involved

must communicate are conditionally selected, can also come into play.

Assuming the privacy-preserving DL approach related with the classic FL architecture, when training the global model in a federated way, from the server side we do not have access to all the raw data. This is an important point in order to preserve privacy, technical or legal considerations that may apply, and to be able of measuring the differences between the data of the different clients, we are going to use the information that the central server that performs the federated training would have available: the weights that define the model once trained. In this sense, we propose to analyze the divergence of the weights between each pair of clients [147]. Be $||\cdot||_F$ the Frobenius norm, and $w_i^{(r)}$ the weights obtained when training with the data from client i in round r, we define the divergence in a symmetric way $d_{i,j}^{(r)}$ as follows:

$$d_{i,j}^{(r)} = \frac{||w_i^{(r)} - w_j^{(r)}||_F}{\frac{1}{2}(||w_i^{(r)}||_F + ||w_j^{(r)}||_F)} \tag{4.9}$$

Note that as it will be explored in Chapter 5, this metric can be used to measure the divergence between the clients, and also we can apply it in order to create cluster of clients according to their similarity.

On the other hand, to prevent unintentional poisoning attacks by clients, we can use data drift detectors in each client deployment. When we talk about data drift detectors we can mainly think of statistical tests or based on distance methods. Some examples are the following [143]:

- **Statistical tests:** Anderson-Darling test, Baumgartner-Weiss-Schindler test, Chi-square test, Cramér-von Mises test, Kolmogorov-Smirnov test, Kuiper's test, Mann-Whitney U test, Welch's t-test, etc. In all cases, we must set a value of α to be compared with the *p-value* obtained with the statistical test (e.g. $\alpha = 0.05$). Be our null hypothesis H_0 that data drift does not occur, and our alternative hypothesis H_a that data drift does occur. In the case that the *p-value* obtained verifies *p-value* $\leq \alpha$ we will reject the null hypothesis, thus concluding that data drift is taking place.

- **Distance based:** Bhattacharyya distance, Earth Mover's distance, Energy distance, Hellinger distance, Jensen-Shannon distance, Kullback-Leibler divergence, Maximum Mean Discrepancy, etc. In this case, a criterion must be established to select the minimum distance beyond which a client is considered to present anomalous values for the model weights and should be eliminated from federated

learning training. Permutation tests are usually used to determine the minimum distance and obtain a *p-value*.

As far as data drift detection is concerned, several Python libraries for this purpose can be highlighted, among others *Alibi-detect* [151], *TorchDrift* [152], *menelaus* [153] and *Frouros* [143]. For this purpose, the idea is that the aforementioned control dataset (with synthetic or simulated non-sensitive data), can be transmitted from the server to the clients so that they can apply an appropriate data drift detector and decide whether their data are suitable for the training carried out. This will allow clients to detect possible deviations in their data from those expected for the federated training and avoid unintentional poisoning attacks.

Then, in the FL architecture we will be interested in having a server that acts prior to the FL one so that it intercepts the weights or the updates sent by the clients, decides if any of the clients has drift, and in that case disconnects it from the FL process and does not transmit the weights to the server. This server (let us call it drift server), will analyze the weights with the previously established criteria in terms of method (distance based or statistical), and the distance or α value from which drift is considered to be occurring. For this purpose, one of the aforementioned Python drift detection libraries will be used. It is interesting that the server has a control data set as a reference in case it is not possible to determine which of the clients is the one producing drift. For example, if there are only two clients and we obtain that there is drift, the server will have to make a decision according to a control group.

Note that this technique is of special interest in cases where there are multiple clients ($N >> 2$ with N the number of clients), because we can make clusters of clients according to their closeness in terms of model weights, but also to detect outliers or potential attackers.

4.5 Differential Privacy in Federated Learning

Although by definition a FL architecture is a privacy preserving machine learning method, during the process we may encounter some security problems, depending on whether or not the server in charge of orchestrating the operation is trusted, as well as on the clients involved. We can distinguish different scenarios:

First, if the central server is not trusted, a possible solution is to apply DP during the training of the models from the client side. As seen in Section 3.7.1, DP can be used during the gradient descent process, so that a model that has been trained under the DP assumption would be sent. Another option would be to apply local

DP to the raw data before training the model. Finally, the last option is to train the model with the original data, without adding DP either during the SGD, and apply DP on the weights resulting from the training, before sending them to the server. All three options are valid and each of them can be more suitable depending on the type of data we handle and the amount of data (e.g. when dealing with limited data it is not convenient to use local DP).

Second, in the case where the central server is trusted but not all clients (e.g. a client that trusts the server does not trust the rest of the clients involved), DP can be applied from the server side. In particular, DP can be applied during the aggregation process. Additionally, as proposed in [154], metric privacy (or metric DP, as defined in Section 3.5) can be applied as a relaxation of server side DP, taking into account the divergence between the weights of the clients. In this case, we need to endow the space of the weights of the different client with a concept of distance. In particular, we can define the distance as between the clients for applying metric privacy as follows: Let $w_i^{(r)}$ be the local weights of the model trained for client i in round r, with $w_i^{(r)}(\ell)$ being the weights of the layer ℓ of the model in round r. Moreover let $\mathscr{L}$ be the set of layers of the trained model and n_c be the number of clients. We define the distance in round r as follows:

$$d^{(r)} = \max_{\substack{i,j\in\{1,\dots,n_c\}\\ i\neq j}} \left(\frac{1}{|\mathscr{L}|} \sum_{\ell\in\mathscr{L}} ||w_i^{(r)}(\ell) - w_j^{(r)}(\ell)||_F \right). \tag{4.10}$$

with $||\cdot||_F$ the Frobenius norm.

As far as the classic implementation of global DP with fixed clipping in FL is concerned, it is necessary to introduce the clipping norm C, the noise multiplier (n_ϵ) and the number of sampled clients n_c. Note that the noise multiplier has the opposite interpretation to the privacy budget, since the lower the value, the higher the utility and the lower the privacy, as less noise is injected.

Thus, we can apply DP using the Gaussian mechanism as well as metric-privacy if we have calculated the metric $d^{(r)}$ dynamically in each round r. Equation 4.11 shows how to add noise to the aggregated parameters from the server side using Gaussian noise (by removing the value of $d^{(r)}$, in blue). In this same equation, if we include $d^{(r)}$ (in blue), we will be adding noise for achieving metric-privacy instead of simple DP. Then, we simple add noise from the Gaussian distribution with mean (μ) 0 and standard deviation (σ) as given in Equation 4.11 to the aggregated parameters calculated in each round.

$$\mathscr{N}(\mu,\sigma) = \mathscr{N}\left(0, \frac{n_e \cdot C}{n_c \cdot d^{(r)}}\right). \tag{4.11}$$

Note that in the case of metric-privacy when adding the distance d, it is included by dividing the noise multiplier. Thus, if $d^{(n)} > 1$, we are adding more noise with metric-privacy than with global DP if we keep the same noise multiplier.

The hypothesis is that by including metric-privacy in FL, we can make a better calibration of the added noise in order to achieve a better performance. In this line, in [154] we show a comparison of server side DP and metric-privacy in a medical imaging use case and analyze their effectiveness against *client inference attacks (CIA)*. In that work we introduced the concept of CIA as a novel attack where a semi-honest client (attacker) receives the global aggregated model from an honest (trusted) server and tries to determine if another client belongs to the list of participants.

Definition 4.5.1. Client inference attack. Given a global aggregated model W and the shadow dataset D_x of client x, a **client inference attack** can be defined as:

$$C(D_x, W) := \mathbb{P}(s_x = 1 | D_x, W) \tag{4.12}$$

Specifically, in the work performed in [154] we analyze how this mechanism can be used to improve the performance obtained with DP, while at the same time providing a similar protection against the proposed attacks.

4.6 Secure Aggregation via Homomorphic Encryption

Data scientists will encounter sensitive data that require treatment to maintain their privacy in many areas, such as business, health, phone and information systems, banking, etc. In terms of information and privacy protection techniques, cryptography is already present in our daily lives, in credit cards, secure connections through the Internet, digital signature, file transfer, etc. When we talk about cryptography, we refer to protect data privacy and also we are focus on authenticity. Essentially, with cryptography we seek to protect data by mathematically encrypting it using a secret key.

Homomorphic encryption (HE) is a form of encryption that allows computation on encoded data [155]. This technique ensures that operating on the encrypted data and decrypting the result is equivalent to performing such operations without enciphering it. Intuitively, an homomorphism is a mathematical function between two objects with the same algebraic structure (i.e. rings) that preserves the operations on them. Important preliminary algebraic notations and definitions in this regard are detailed in Appendix A. It should be noted that depending on the number of operations that can be carried out on the encrypted data, there are different

homomorphic encryption schemes, as will be detailed in the following definitions (see [156] [157]).

Definition 4.6.1. Fully Homomorphic Encryption (FHE). It allows to make any number of operations. In practice it is very expensive computationally. Theoretically, any function can be computed.

Definition 4.6.2. Somewhat Homomorphic Encryption (SWHE). It is an scheme more feasible in practice, but limits the number of operations that can be performed on encrypted data.

Definition 4.6.3. Partially Homomorphic Encryption (PHE). This scheme allows only one type of operation to be performed (any number of times). It is a straightforward and efficient scheme despite its limitations.

4.6.1 Homomorphic Encryption and Federated Learning

In many cases, when considering the incorporation of homomorphic encryption in the aggregation process of a federated learning scheme, it is proposed that all clients have the same public and private key pair [158]. Thus, each of them encrypts its weights/models with their private key, and sends them to the server. The server aggregates these weights/models without decrypting them (and without being able to access them), and returns the encrypted result to each client. Each client uses the private key to decrypt the weights, and continues the process by training again the model. This is possible because the aggregation has being performed on weights that have been encrypted using the same public key, so the decryption process is immediate with the associated private key.

A schematic idea of this initial approach of integrating HE into a FL architecture is shown in Figure 4.7 Note that this figure is analogous to the one presented in Figure 4.5. However, it includes the public and private key pairs needed to encrypt the model on each client once it has been trained. This allows the server to aggregate the encrypted models received, creating a secure aggregated model (encrypted) that is send to the clients for moving to the next round.

Although this is a good approach from the point of view that the weights are shared in encrypted form, and the server cannot decrypt them either, it still raises some issues. First, if all clients have the same public and private key pair, an external server must be in charge of distributing them securely (or otherwise they should communicate with each other in a secure way). In addition, in the event of a potential security breach, if all clients have the same private key, one client could access the weights of another. It would be most appropriate for each client

to have a unique public/private key pair with which to encrypt its weights/models. However, if each client encrypts its information with a different key, the server in charge of aggregating the models will not be able to manage the work with a classic HE approach (as each model will be encrypted with a different key).

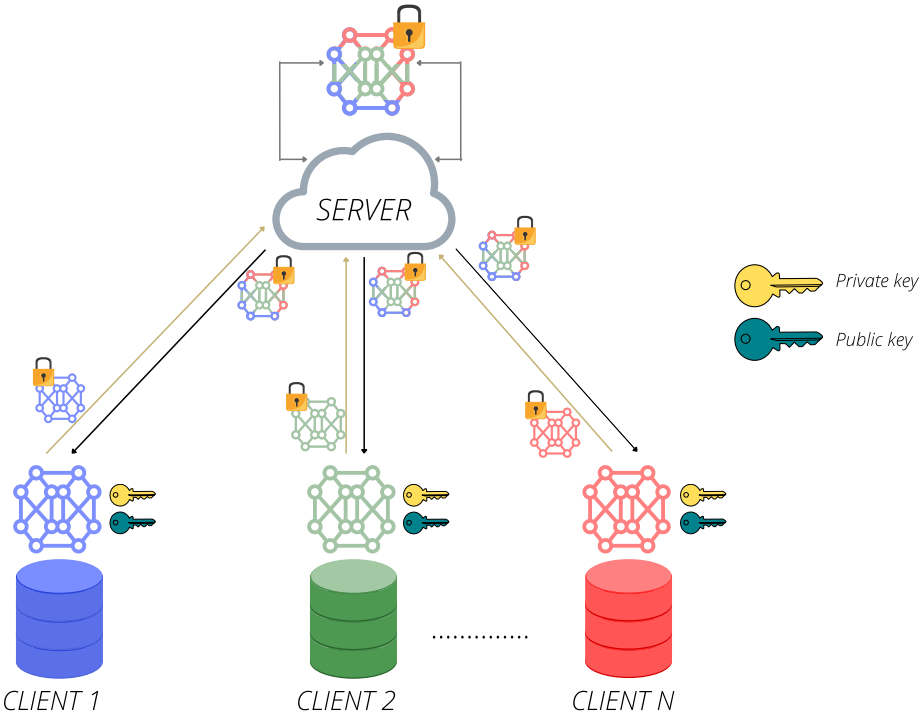


Figure 4.7: Schematic idea of the basic integration of HE in an FL architecture in which all clients use the same public and private key pair.

Therefore, we can intuitively think on the following approach: each client generates its own public and private key pair following the protocol it deems appropriate in each case (e.g. Cheon-Kim-Kim-Song algorithm). Each client sends its public key to the server, which is in charge of making it publicly accessible to the rest of the clients. Thus, we would have that given N clients, each of them has the pair of public and private keys (respectively) defined as (pk_i, sk_i), $\forall i \in \{1,\dots,N\}$. Once the local training has been carried out on each client, the client i encrypts the resulting weights using its public key, but also that of the other clients. Then, be $weights_i$ the weights obtained by client i after training the models locally, and $[\ ]_{pk_j}$ represents encryption using the public key pk_j, client i will send to the server $\{[weights_i]_{pk_j}, j \in \{1,\dots,N\}\}$, $\forall i \in \{1,\dots,N\}$. In this way, each client will send

the server N encrypted models/weights, with the server receiving a total of N^2 models/weights.

In view of the above, this paradigm will only be suitable in case of cross silo FL with a small number of clients, since each client will share N encrypted values, and the server will have to do the aggregation N times, once per client. Following this scheme, the server returns to the client i the aggregation carried out with the N weights encrypted with the key pk_i, and this one can decrypt them using sk_i to continue with the next round of the FL scheme and follow the process.

Specifically, if the aggregation function established is the weighted average, let n_i be the number of i client data, the server will create N aggregations of the weights, each one encrypted with the public key of each of the N clients as follows:

$$agg_i = \frac{1}{\sum_{k=1}^{N} n_k} \sum_{j=1}^{N} n_j [weights_j]_{pk_i}, \forall i \in \{1, \ldots, N\}. \tag{4.13}$$

Thus, each client i can decrypt agg_i using the private key sk_i, while the server has computed this aggregation without decrypting the weights, since they are sum operations between encrypted vectors and multiplication by scalars. In cross silo federated learning scenarios with a low number of clients, we can think about this secure aggregation scheme using HE. However, in the case of cross-device FL, where the number of devices is usually high, the computational cost of the proposed scheme would be unaffordable. In addition, note that this approach involves making the aggregation process N times, requiring an important computing consumption from the server side, which makes it reasonable only for cross-silo setting in which few clients (e.g. $N = 2$, $N = 3$ and indeed $N << 10$) are available.

Therefore, due to the computational and memory cost of this approach, the idea of proposing a multi-key HE paradigm that is scalable to multiple clients (>>2) arises.

4.6.1.1 Multi Key Homomorphic Encryption

The idea of HE is that the server can perform the aggregation of the received models without decrypting them, by being able to perform operations on them. However, the simple HE approach does not allow aggregation of data that has been encrypted with different private keys. Therefore, it is necessary to resort to the Multi Key Homomorphic Encryption (MKHE) technique, thanks to which it is possible to perform operations on the encrypted models even if they have been encrypted with different keys. Thus, aggregation is secure in case the server and some clients are not trusted

(or simply to prevent an external attacker from accessing the models when they are transmitted from the client to the server).

Then, the basic premise behind MKHE for FL is that each client in federated learning training can encrypt the weights it has obtained after local training using its own public and private key pair. Thus, each client would use its own public key to encrypt its model/weights.

The main point of this technology is the key-switching technique, which seeks to convert information encrypted with a certain key e_k to information encrypted under another key e'_k without the need to decrypt the content.

A classic approach to achieve this is based on the Ring Learn With Errors (RLWE) problem (see Section A.5 from Appendix A), so intuitively one seeks to move from one cipher space to another. Note that following this approach, the post-computation result can be decrypted jointly by all participating users.

It should be noted that both HE and MKHE are more computationally expensive than other privacy techniques such as DP, and also in the case of MKHE, all parties are required to participate in collaborative decryption. This makes that in practice, the use of methods such as DP or metric privacy as explained in Section 4.5 is more feasible in real applications, especially when working with cross device FL approaches or cross-silo approaches with a high number of clients.

Finally, since it is beyond the scope of this thesis to go deeper into this broad topic, more details on the application of MKHE in FL with [159] and [160] as introductory references are left for the reader.

In particular, in [159] the authors present the development of an aggregation protocol for FL that can be applied under the assumption of semi-honest parties (parties that are honest-but-curious[1]). In particular, each client uses its own key for encryption in each round, and the subset of clients that contributed in each round must collaborate in order to decrypt the aggregated updates. In the same line, in [160] the authors propose a MKHE scheme for FL in which each client can use its own public-private key pair for encryption the model parameters using its public key. Again in this case they assume that the involved parties are semi-honest. In order to carry out the process, they use the double decryption mechanism proposed by Bresson, Catalano and Pointcheval (BCP) proposed in [161].

[1] This means that such parties will follow the protocol but they will try to learn as much information as possible.

4.7 Python Libraries for Federated Learning

*Part of the review presented is this section has been published in the Journal Artificial Intelligence Review as: Nguyen, G., **Sáinz-Pardo Díaz, J.**, Calatrava, A., Berberi, L., Lytvyn, O., Kozlov, V., Tran, V., Moltó, G., & López García, Á. (2024). Landscape of machine learning evolution: privacy-preserving federated learning frameworks and tools. Artificial Intelligence Review, 58 (2), 51 [4].*

There are several Python libraries which implement the classic federated learning architecture. In the following, seven Python libraries that can be used for perform FL trainings or experiments are discussed:

TensorFlow Federated

TensorFlow Federated [162] is a framework for implementing federated learning powered by *TensorFlow* [118]. It is an open-source framework for ML and other computations on decentralized data. *TensorFlow Federated* enables users to simulate the most commonly used FL algorithms. The framework consists of two layers of interfaces. First, the federated learning API, allows to apply the included implementations of FL and evaluation to the existing models. Secondly, the federated core API is a set of low-level interfaces for concisely expressing novel federated algorithms by combining *TensorFlow* with distributed communication operator within strongly typed functional programming environment. However, this framework is it is very focused on simulation and not on actual implementation with real distributed datasets, as it does not provide communication mechanisms for such cases.

PySyft

PySyft [163] is a privacy-preserving framework for statistical analysis and machine learning that aims to provide users with tools for performing secure and private data science tasks in Python. *PySyft* augments classic DL frameworks for servers and IoT with privacy-preserving capabilities. Specifically, it decouples private data from model training using techniques such as FL, DP, and encrypted computation. This open source framework is easily integrable with *PyTorch* for training a model under a FL architecture. In addition, *SyMPC* is a library that extends *PySyft* with SMPC support. It allows computing over encrypted data and to train and evaluate neural networks.

Flower

Flower [131] is a Python library that provides a unified approach to FL, analytics, and evaluation, offering a framework-agnostic implementation of an FL system.

The framework allows for the rapid transition of existing centralized ML training pipelines into an FL setup to evaluate their convergence properties and training time in a federated setting. Another advantage of *Flower* is its support for extending FL implementations to mobile and wireless clients, with heterogeneous compute, memory, and network resources. *Flower* provides an API wrapper for *TensorFlow*, *TensorFlow Lite*, *PyTorch*, *PyTorch Lightning*, *Hugging Face*, *MXNet*, *JAX*, and *Scikit-Learn*. The way to transition to a federated scheme from a centralized one is very simple, with clients being able to connect to a server deployed on a certain IP using a gRPC protocol.

Nvidia Flare

NVIDIA Flare (NVFlare) [164] is an extensible, open-source domain-independent SDK that enables users to adapt existing ML/DL workflows to a federated paradigm. It is a framework powered by NVIDIA Federated Learning Application Runtime Environment, which allows one to perform research concerning FL applications, but also real-world production deployments. Simulations can be managed through a dashboard and it allows a wide customization of the training as well as integration with monitoring tools.

IBM Federated Learning

IBM Federated Learning [165] provides tools for multiple remote parties to collaboratively train a single ML model without sharing data. Each party trains a local model with a private data set. Only the local model is sent to the aggregator to improve the quality of the global model that benefits all parties. *IBM Federated Learning* library is an open-source Python framework for FL in an enterprise environment.

FedML

FedML (Foundational Ecosystem Design for Machine Learning) [166] has motto: *The unified and scalable ML library for distributed large-scale training, model service, and federated learning*. It is backed by *FEDML Nexus AI*, which is an enterprise pay-as-you-go AI platform for cloud service for LLM and generative AI. It helps developers launch complex model training, deployment, and federated learning anywhere on decentralized GPUs, multiclouds, edge servers, and smartphones, easily, economically, and securely.

FATE

FATE (Federated AI Technology Enabler) [167] is an open-source project initiated

by Webank's AI Department to provide a secure computing framework to support the federated AI ecosystem. It implements multiple secure computation protocols to enable big data collaboration in compliance with data protection regulations. *FATE* is now an open-source project hosted by the Linux Foundation.

4.8 Other Distributed Learning Architectures

In this last theoretical section we will highlight four other privacy-aware distributed learning architectures, which together with FL shape the state of the art. Note that the architectures proposed in this section do not depend on a central server to orchestrate the learning process.

4.8.1 All-Reduce Architecture

When we think of a distributed learning architecture in which we do not want any central server to be in charge of orchestrating the communication, but rather we want the participating clients to coordinate the process, the idea of the *all-reduce architecture* is the first that comes to mind. This architecture, composed of N clients (participating nodes), will consist of each of them sending to the remaining $N-1$ clients their locally trained model. Note that all the clients will train the same initial model. This process can be done only once (one round), and obtain an aggregated global model at each node using certain aggregation function (such as a weighted average of the models), or it can be repeated for a certain number of rounds, starting in each of them from the aggregated global model obtained in the previous round as the initial model to be re-trained at each client. Thus, all clients will have N models at the end of each round (note that this may vary only in the case of a client connecting to the training as a new node, or a client disconnecting from the training).

It should be noted that each client will be in charge of aggregating the received models to build a global one (see Section 4.2.5). The process can be repeated as many rounds as necessary until convergence is obtained, taking each client the new global aggregated model as the initial one for training at the beginning of each round.

It is clear that the main problem of this architecture arises from the communication point of view, as well as the bandwidth needed to share the models among all the clients, especially for large values of N. We can also find memory problems on the nodes/clients side, as they will receive and store $N-1$ models in each round.

4.8.2 Ring-All-Reduce Architecture

In view of the aforementioned architecture, in order to reduce the costs derived from the communication of the models between all the clients, and thus reduce the bandwidth as well as the amount of memory required on the side of each client to store all the models, the *ring-all-reduce architecture* arises. Specifically, in this case, the communication of the models between the clients follows a ring structure. Thus, clients are numbered from 1 to N, whereby client 1 sends its trained model to client 2, client 2 to client 3, client 3 to client 4, client $N-2$ to client $N-1$, client $N-1$ to client N, and client N to client 1.

Following this structure, each client will use different local models to aggregate and create the global model, so each client will get a different aggregated model. However, if this is repeated during several rounds, the information will be propagated along the ring structure.

4.8.3 Neighbor Architecture

The aforementioned architecture allows a considerable reduction in bandwidth and the cost of communicating as many models as in the initial all-reduce approach. However, another architecture that can be proposed is the *neighbor architecture*, in which each node sends its model to its two neighbors in the ring structure. This means that certain client i (c_i for $i \in \{2,\dots,N-1\}$) sends its model/weights of the model (w_i) to clients c_{i+1} and c. If $i = 1$ then this client sends its model to clients 2 (c_2) and N (c_N), and if $i = N$ it is sends it to clients 1 (c_1) and $N-1$ (c_{N-1}).

In this approach each client will have 3 trained models to aggregate each time, instead of only two as in the previous architecture, thus being a more robust model.

4.8.4 Gossip Learning

In this architecture, based on the gossip protocol [168], clients choose conditionally or randomly to which other clients they will send their updates. Therefore, it can be set in advance to which clients each node will send its trained model, it can be selected randomly, and it can be changed in each round (if the process is repeated multiple rounds). Thus, for each client at the beginning of each round, a list of clients to send their models to will be established. At the end of each round, each client will aggregate its model along with the ones received.

It is important to note that each client will have a different overall aggregated model. However, repeating this throughout the rounds, varying or not the communication structure between clients, allows the information obtained (in the form

of a trained model) to be propagated at each node. Even if the architecture does not change in each round, it can happen as in the ring-all-reduce approach, and the information is propagated because different clients have been used to build the base aggregated model for the next round.

In this way, in the four architectures proposed, we have an scheme composed of multiple clients that hold locally the different datasets used to train the desired model.

4.8.4.1 Consensus Algorithms

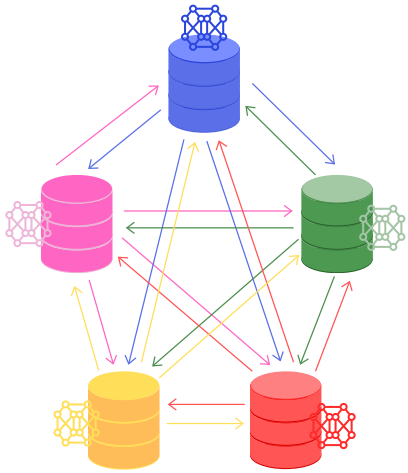
As already explained, the clients in a GL architecture must conditionally or randomly choose how to communicate with each other. In addition, for certain procedures it may be necessary to have a client that acts as a leader. For this purpose, consensus algorithms can be used to select such client. This type of algorithms are widely used in distributed architectures or decentralized networks, as they allows different workers to coordinate themselves in distributed environments.

Consensus algorithms aim to achieve the following properties: validity, agreement, completeness and fault tolerance. Validity implies that the agreed value must be proposed by some node. Agreement means that all nodes must agree on the same value. Completeness implies that all nodes must decide on a final value. Fault tolerance means that the algorithm must work even if some nodes are faulty or malicious. Consensus algorithms work by using various techniques, such as voting, quorum, timeouts and testing.

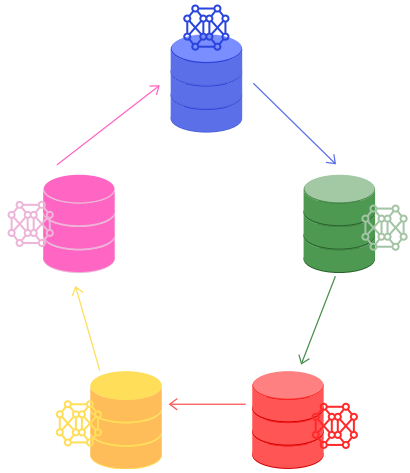
The selection of a consensus algorithm for a distributed system depends on several factors, such as system size, network environment, fault model, performance requirements and security guarantees.

Note that, in contrast to the FL architecture, this four distributed learning architectures does not involve the presence of a central server for orchestrating the network. This may help to mitigate some problems, such as the lack of trust in the server that makes it necessary to apply HE or DP techniques. However, the need to use PETs is still present in the case where not all the clients are trusted and it is necessary to prevent them from disclosing sensitive information. However, these architectures may present some problems from the point of view of communication between the different parties as well as from a memory cost point of view.

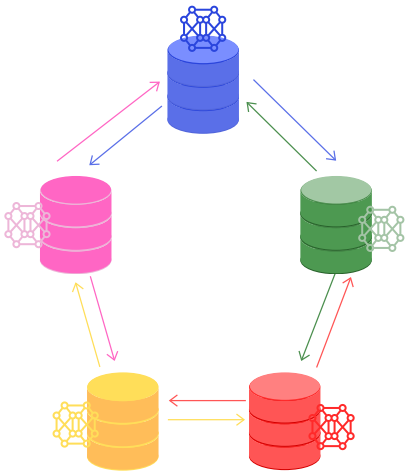
Figure 4.8 intuitively summarizes the behavior of the four server agnostic approaches, taking into account that each arrow represents that the model trained with each dataset is shared. For the gossip learning approach an example of randomized communication is shown.



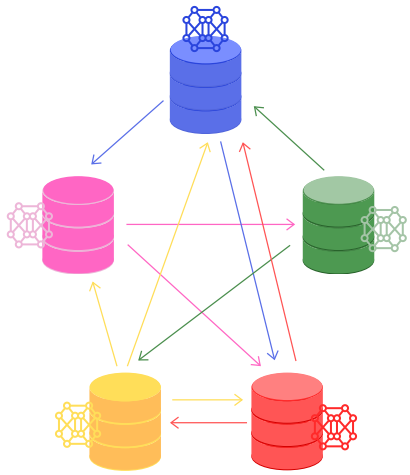
(a) All reduce architecture.



(b) Ring all reduce architecture.



(c) Neighbor architecture.



(d) Gossip learning architecture.

Figure 4.8: Visual overview: server independent distributed learning architectures analyzed.

Finally, the five privacy preserving machine learning architectures analyzed in this chapter are summarized in Table 4.1.

Architecture	**References**	**Description**
Federated Learning (FL)	[6] [121] [132] [135] [137]	This architecture is composed by a central server and multiple clients. All the clients want to train a global model without sharing raw data. In order to do so, each client trains the same model locally and then transmits it to the server which aggregates it to build a global one. This aggregated model is then redistributed to the clients again who repeat the process as many rounds as necessary to achieve convergence.
All-Reduce architecture	[150] [169] [170]	This architecture aim to remove the dependency on the central server of the FL scheme. In this case, all the clients communicate between themselves.
Ring-All-Reduce architecture	[150] [169] [171] [172]	It is an alternative to the all-reduce architecture in order to reduce bandwidth consumption. It consists of setting up the clients on a ring structure.
Neighbor architecture	[150] [173]	This architecture is similar to the GL one, but in this case each client only communicates with its neighbors.
Gossip Learning (GL)	[150] [173] [174] [175]	Decentralized architecture in which no central server is required. Instead, clients directly share their updates among them, and can choose conditionally or randomly with whom they will communicate. In this approach, the aggregation takes place in a distributed manner. Note that this approach is fully decentralized.

Table 4.1: Summary: decentralized learning architectures explored. Adapted from [4].

4.9 Summary and Conclusions

In this chapter we have discussed in detail the theoretical basis behind the distributed learning approach, first from the computing aware perspective and then motivating the privacy preserving machine learning technique known as federated learning (FL).

In particular, we have compared and motivated the introduction of this architecture by focusing on privacy preserving aware development of solutions and models based on data and AI in a distributed manner across multiple data owners. The FL approach has been compared with the classical centralized approach and with edge computing, presenting a number of advantages and open problems of FL with respect to the centralized case. Then, a basic and intuitive implementation idea has been presented following the proposed architecture scheme.

Next, we have stopped to explore different types of federated learning. Firstly, we discussed according to the type and number of clients involved, distinguishing between cross device and cross silo FL. Secondly, according to the type of data of each client involved, in this case distinguishing between horizontal and vertical federated learning.

Following this, some aggregation functions classically used in the state of the art have been presented, together with the proposal of a new form of aggregation, called *FedAvgOpt*, which seeks to optimize the aggregation process by minimizing the similarity between the local weights or parameters of each client and the aggregated ones.

Following this, we have presented different open challenges in federated learning, including expensive communication, systems and statistical heterogeneity, and privacy concerns. We have also discussed aspects that arise when putting into production FL models, including drift detection and monitoring as well as its management in these architectures.

Afterwards, we have focused on analyzing the integration of differential privacy (DP) and metric privacy techniques in federated architectures from different paradigms, depending on whether or not the server that is hosting the process is trusted or not.

Continuing privacy concerns, a brief discussion followed on how homomorphic encryption (HE) could be included in a FL architecture, including the possibility of using multi-key homomorphic encryption (MKHE) so that each client can use its own public and private key pair.

Subsequently, we have reviewed different Python frameworks for development

and experimentation with federated architectures, discussing the contributions of each of them.

Finally, we have reviewed other distributed learning architectures that do not rely on a central server, such as all reduce, ring all reduce or neighbor architectures, or the gossip learning (GL) one.

CHAPTER 5

Federated Learning Applications

Cayó una hoja
y dos
y tres.
Por la luna nadaba un pez.
El agua duerme una hora
y el mar blanco duerme cien.

Federico García Lorca,
Vals en las ramas,
Poeta en Nueva York

Contents

5.1 Medical Imaging: Classification of Chest X-Ray Images 185
5.1.1 Data Availability and Preprocessing 186
5.1.2 Deep Learning Model 186
5.1.3 Federated Learning Approach (3 Clients) 187
5.1.4 Federated Learning Approach (10 Clients) 193
5.1.5 Comparison: Centralized vs Federated 196
5.1.6 Federated Learning with Intermittent Clients 197
5.1.7 Federated Learning Meets Differential Privacy 201
5.1.8 Comparing Different Aggregation Strategies 205
5.1.9 Conclusions 206
5.2 Water Quality: Data Based Estimation of High Frequency Nutrient Concentrations 206
5.2.1 Data Availability and Processing 208
5.2.2 Deep Learning Model Analyzed 214
5.2.3 Methodology 215
5.2.4 Results and Discussion 216
5.2.5 Conclusions 228
5.3 Climate Sciences: Vertical Integrated Liquid Nowcasting using Weather Radar Data 229
5.3.1 Data Availability and Preprocessing 231
5.3.2 Methodology 233
5.3.3 Benchmark Model: COTREC 233
5.3.4 Deep Learning and Federated Learning Models 234
5.3.5 Personalized Federated Learning 236
5.3.6 Convolutional Neural Architecture 239
5.3.7 Results and Analysis 240
5.3.8 Conclusions 247
5.4 Summary and Wrap Up 248

Abstract

In order to show the wide range of applications of the federated learning approach, this chapter presents three use cases from diverse fields: medical imaging, water quality and meteorology through radar images. The complete use cases are presented along with the results obtained in each problem. Firstly, with respect to the medical imaging case, an analysis is carried out by incorporating differential privacy into the training process, in addition to a comparison of different aggregation functionalities. Secondly, regarding the water quality use case, the federated learning approach has been compared with the individual learning and centralized learning settings, further considering different data reduction scenarios, simulating the case where data collection is less frequent. Thirdly, in the case of nowcasting with radar images, we propose the application of a personalized federated learning technique that allows a better adaptability to the data acquisition area to be tested. In this case, the divergence between the different clients has also been analyzed. Finally, the conclusions obtained from the exhaustive analysis of these use cases are drawn.

The fields in which the techniques of federated learning can be applied are manifold. Moreover, the desired objectives with the implementation of this architecture can range from dealing with data security and privacy restrictions that prevent its centralization, to legal aspects or high cost of data acquisition, or even motivated by technical restrictions or a lack of connectivity. It is interesting to compare the effects of the application of this methodology in comparison with classical approaches, such as the centralized one, or the training of the models individually on each client.

In the following, we present a wide range of use cases in which federated learning architectures have been applied to replace the classical versions in which centralized or individual learning is applied. In particular, we seek to reflect the multidisciplinary nature of the field under such a diverse set of use cases that focus on aspects ranging from the application of deep learning methods to medical imaging scenarios (thus deeply related with a privacy preserving use case), to cases involving parameters for water quality monitoring or data captured by weather radars (meteorology or climate sciences).

Concerning the three applications explored in this chapter, the results and analysis presented as well as the methodology followed have been previously published,

during the course of this doctoral thesis, in three scientific publications, as detailed below:

- *Medical imaging. Classification of chest X-ray images:* **Sáinz-Pardo Díaz, J.** & López García, Á. (2023). Study of the performance and scalability of federated learning for medical imaging with intermittent clients. *Neurocomputing*, 518, 142-154 [6].

- *Water quality. Estimation of high frequency nutrient concentrations:* **Sáinz-Pardo Díaz, J.**, Castrillo, M., & López García, Á. (2023). Deep learning based soft-sensor for continuous chlorophyll estimation on decentralized data. *Water Research*, 120726 [176].

- *Climate sciences. Vertical integrated liquid nowcasting from weather radar data:* **Sáinz-Pardo Díaz, J.**, Castrillo, M., Bartok, J., Heredia Cachá, I., Malkin Ondík, I., Martynovskyi, I., Alibabaei, K., Berberi, L., Kozlov, V. & López García, Á. (2024). Personalized Federated Learning for improving radar based precipitation nowcasting on heterogeneous areas. *Earth Science Informatics*, 17, 5561–5584 [177].

The contributions to the federated learning field derived from these studies and this thesis are outlined in the following.

Firstly, with respect to the medical imaging use case, we have compared the performance of the FL architecture simulating different clients, and compared with the performance of the centralized training. In addition, we have further studied two cases of client intermittency, in which different clients enter or leave the training once started. Two ways of dealing with this challenge have been proposed for each case. In addition, we have further conducted an study evaluating the impact of the incorporation of differential privacy (DP) during the training of the models based on the federated learning architecture. In this same case, we have also explored the application of the *FedAvgOpt* aggregation strategy proposed in Section 4.2.5.8 from Chapter 4, and comparing its performance in relation to other classical aggregation functions.

Secondly, concerning the study of water quality, we have applied the FL architecture including features from different sources and also different weather conditions. The performance of the FL architecture has been compared with the individual and centralized approaches, additionally considering a study on three data reduction scenarios, thus simulating the case where data are sampled less frequently. The objective is to analyze if in this case if the FL architecture can provide advantages

(with regard to the centralized architecture or to the individual training) from the point of view of the generalization capacity of the model when we deal with two databases (clients) with different distributions.

Finally, regarding the work done in climate sciences to predict the vertical integrated liquid using radar images, we have simulated four clients, in order to try to extrapolate findings in the case of having multiple radar distributed in different areas. Additionally, we have applied a novel technique of personalized federated learning (PFL) that allows us to obtain a better adaptability in each test zone. We have compared the results of the individual, FL and PFL training architectures with a weather nowcasting state of the art method (COTREC) as a benchmark, testing additionally on a central zone not used in the training. Finally in this case we have also analyzed the divergence between the different clients (which in this case are the different capture zones).

5.1 Medical Imaging: Classification of Chest X-Ray Images

Part of the results presented in this section has been published in the journal Neurocomputing (Elsevier): ***Sáinz-Pardo Díaz, J.*** *& López García, Á. (2023). Study of the performance and scalability of federated learning for medical imaging with intermittent clients. Neurocomputing, 518, 142-154. [6].*

The main objective of this first use case is to present the implementation of a FL scheme from scratch and to compare the results obtained with the centralized case, also when varying the number of clients. Although several Python libraries that implement the federated learning architecture can be used (as exposed in the previous section), in this case, for the sake of completeness of the study and greater customization, we have made our own implementation following by using one class called *Client* and another one called *Server* as presented in Section 4.2.3.

In addition, in this first section we also present different approaches to deal with the problem of intermittent clients. Afterwards, we will detail an example of applying DP to a federated architecture, and finally we will compare different aggregation strategies, with special attention to the *FedAvgOpt* function proposed in Section 4.2.5.8.

Before proceeding to analyze the data and methods used in this section, it is interesting to comment briefly on the state of the art in the field of FL applied to medical imaging. Federated learning is being studied in a wide range of fields [178]

such as medical data [179, 180, 181, 182, 183] cybersecurity [184, 185] or Internet of Things [186, 187] among others. Regarding medical data we can highlight [182], where three benchmark datasets are studied (MNIST, Medical Information Mart for Intensive Care-III dataset and an ECG dataset), [188], which shows experiments using MRI scans and include the use of Differential Privacy (DP) techniques, and [189], which is a survey on the use of FL in smart healthcare.

The work in the field is extensive, but in this section we address contributions from different paradigms, as detailed above, including comparison with the centralized case, depending on the number of clients, approaches for intermittent clients, use of DP and comparison of different aggregation functions.

5.1.1 Data Availability and Preprocessing

In this medical imaging use case we are going to use chest X-Ray images, and the objective will be to classify them according to whether or not the patient has pneumonia. The data under study are openly available in a widely known data repository, and it was obtained from [190]. The different images are initially distributed into three groups: train, test and validation. In Table 5.1 the distribution of the images of each class (pneumonia and normal) in each of these three splits is shown.

	Pneumonia	**Normal**
Train	3875	1341
Test	390	234
Validation	8	8

Table 5.1: Distribution of the X-Ray images in train, test and validation sets.

5.1.2 Deep Learning Model

In order to classify the different Chest X-Ray images into normal or pneumonia, a model consisting of a multi-layer convolutional network is proposed [6].

Note that different tests have been carried out to adjust the final model and the hyper-parameters. Finally, the optimizer *RMSprop* and the binary cross-entropy as loss function were used to compile the model, implemented using the Python library *keras*. the architecture developed is given by the following layers:

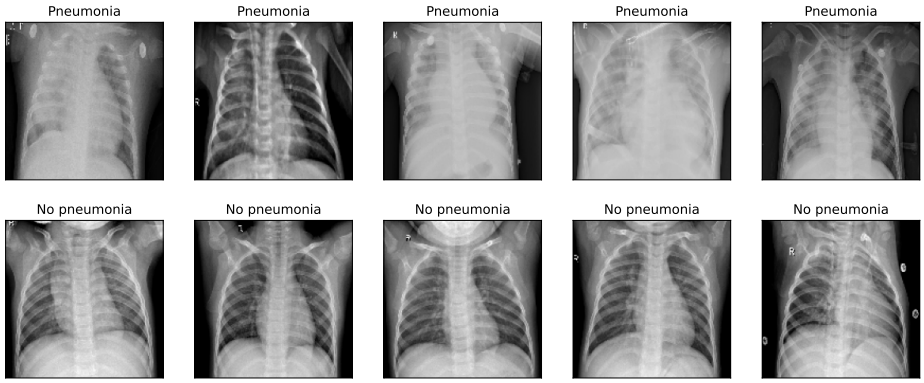
- **Conv2D layer**. Filters: 32. Kernel size: (3, 3). Activation: *ReLU*. Input shape: (150,150,1).

- **BatchNormalization layer**.
- **MaxPooling2D layer**. Pool size: (2,2). Strides: 2.
- **Conv2D layer**. Filters: 64. Kernel size: (3, 3). Activation: *ReLU*.
- **Dropout layer**. Rate: 0.1.
- **BatchNormalization layer**.
- **MaxPooling2D layer**. Pool size: (2,2). Strides: 2.
- **Conv2D layer**. Filters: 64. Kernel size: (3, 3). Activation: *ReLU*.
- **BatchNormalization layer**.
- **MaxPooling2D layer**. Pool size: (2,2). Strides: 2.
- **Conv2D layer**. Filters: 128. Kernel size: (3, 3). Activation: *ReLU*.
- **Dropout layer**. Rate: 0.2.
- **BatchNormalization layer**.
- **MaxPooling2D layer**. Pool size: (2,2). Strides: 2.
- **Conv2D layer**. Filters: 256. Kernel size: (3, 3). Activation: *ReLU*.
- **Dropout layer**. Rate: 0.2.
- **BatchNormalization layer**.published as
- **MaxPooling2D layer**. Pool size: (2,2).
- **Flatten layer**.
- **Dense layer**. Units: 128. Activation: *ReLU*.
- **Dropout layer**. Rate: 0.2.
- **Dense layer**. Units: 1. Activation: *sigmoid*.

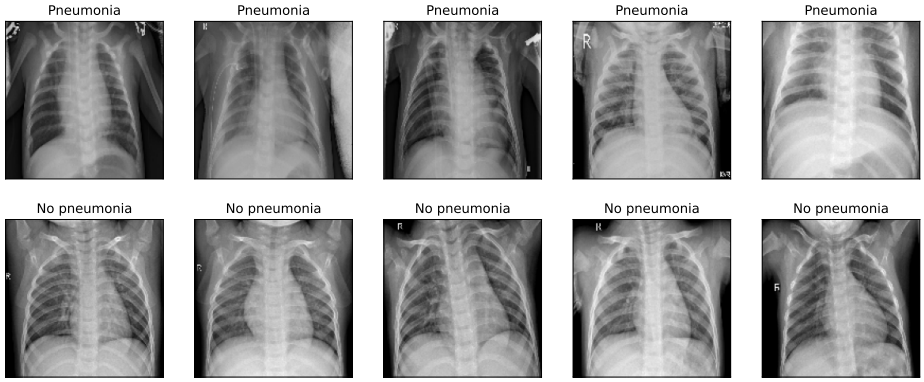
In order to perform a federated learning schema, the training data have been randomly distributed along the different clients, so there may be cases of imbalance in such data. In order to compare the centralized and decentralized cases, and in the latter case the evolution in the initial test set as a function of the number of FL round, the AUC and the accuracy will be computed using the Python library *scikit-learn*.

5.1.3 Federated Learning Approach (3 Clients)

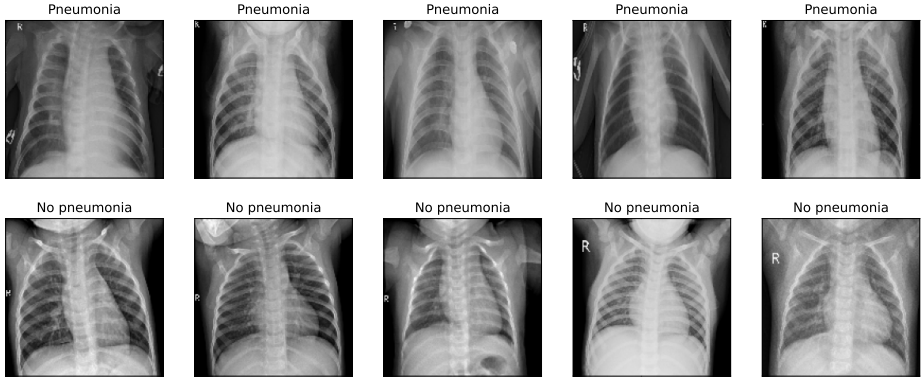
Usually, when we apply FL architectures, it is because we are dealing with distributed data due to privacy restrictions, such as data belonging to different data owners or stored in different locations. However, in this case, we are interested in simulating the clients from the centralized dataset, in order to compare with the centralized scenario and to analyze the performance of the FL architecture depending on the number of clients.



(a) Client 1.



(b) Client 2.



(c) Client 3.

Figure 5.1: Example of the two categories of chest X-ray images under study (pneumonia and normal), obtained from each client train set.

First, note that Figures 5.1(a), 5.1(b) and 5.1(c) show 10 examples extracted from each client train set regarding data of patients with and without pneumonia (5 samples for each class).

In this section we will divide the initial training data into three different clients, with different number of data in each of them, as exposed in Table 5.2, simulating an heterogeneous distribution. In addition, we split each of these into train (75%) and test (25%) sets. Table 5.2 also shows the average time per epoch that it takes to train the model exposed in the previous section with the data from each client.

	Number of data		
	Train	**Test**	**Average training time per epoch (s)**
Client 1	1050	350	23.1
Client 2	1800	600	40.1
Client 3	1062	354	24

Table 5.2: Number of train and test data of each client and average training time per epoch. Case: 3 clients.

Be N_c the number of clients, N_e the number of epochs the model is trained on each client, N_r the number of times the FL schema is repeated and t_i the average time it takes to train each epoch for the data of client i. Since this training is done in parallel, the execution time will be:

$$N_r N_e \max_{i \in \{1,\dots,N_c\}} t_i \tag{5.1}$$

Note that when training the same model in a centralized way (with a training set containing 5216 images) the average time per epoch (average over 10 epochs executed) is 138.6s. In the FL scenario, if the process is repeated 10 times ($N_r = 10$), the average time considering that one epoch is trained in each case ($N_e = 1$) will be 401s (see Table 5.2). However if we train the model in the centralized case during 10 epochs, the execution time will be approximately 1386s. Then, we want to analyze if this substantial saving in execution time will be reflected in an accuracy penalty.

Therefore, let us see in Table 5.3 the results obtained when training these two cases for the test set, which consists of 624 images.

Note that in the centralized approach we are training with more data than with the 3 clients together in the decentralized one. This is because for the decentralized case we have left a part of the data as a test for further analysis. Even so, the

Centralized approach			*Decentralized approach*		
Loss (test)	**Accuracy (test)**	**AUC (test)**	**Loss (test)**	**Accuracy (test)**	**AUC (test)**
3.0694	0.6619	0.9429	2.6034	0.8029	0.9185

Table 5.3: Centralized approach vs decentralized approach. Case: 3 clients.

results for the test accuracy obtained are better in the case where we apply federated learning (i.e. the data are used in a decentralized way). This does not occur with the AUC, which is slightly worse in the case of FL, but at the cost of a runtime reduction of more than 70%. In fact, in addition to improved results in terms of accuracy, a reduction in execution time of approximately 71.07% is obtained.

The evolution of the test loss, accuracy and AUC values obtained for this case, training one epoch each time on each client ($N_e = 1$), and repeating this process N_r times, is shown in Table 5.4.

N_r	**Loss (test)**	**Accuracy (test)**	**AUC (test)**
1	8.5823	0.6250	0.5105
2	8.2261	0.6250	0.5204
3	11.8544	0.6250	0.5987
4	13.6620	0.6250	0.8799
5	8.9979	0.6266	0.8936
6	12.2626	0.6298	0.9210
7	3.0013	0.7660	0.9271
8	4.3232	0.7308	**0.9313**
9	6.2835	0.6987	0.9246
10	**2.6034**	**0.8029**	0.9185

Table 5.4: Decentralized approach. Metrics obtained for the test data varying N_r and with $N_e = 1$. Case: 3 clients.

It can be observed that, contrary to what might be intuitively expected, the best results in terms o loss and accuracy for the set of tests are obtained after 10 repetitions, while the best value for the AUC is reached with $N_r = 8$, being better than the one obtained after 10 rounds. This can be due to the fact that in each repetition the weights of the models are adjusted according to the data of each

client, and the data on which we are predicting have not been seen by any client. In the same way, the convergence of the model can be seen when increasing repetitions of the FL cycle, especially noticeable in the AUC, which goes from 0.5105 to 0.9185 in the last round, and 0.9313 in the best case.

In the same way, as we will observe in the following table, the same thing will happen with the test set of each client, since we are not interested in overfitting the models for a specific client, but rather that based on the data of each client, they generalize in the best possible way. In Table 5.5 the results in terms of loss and accuracy for each client's test set evaluating using the model obtained after N_r repetitions, are presented.

	Client 1 (test)		*Client 2 (test)*		*Client 3 (test)*	
N_r	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Accuracy**
1	6.4700	0.7143	3.7583	0.8333	8.6489	0.6186
2	6.0401	0.7143	3.4773	0.8333	8.0885	0.6186
3	8.4223	0.7143	4.8034	0.8333	11.2106	0.6186
4	7.2142	0.7200	4.1675	0.8333	10.2194	0.6186
5	2.9167	0.7513	1.4952	0.8467	4.1782	0.6441
6	3.1232	0.8000	1.5306	0.8750	4.5520	0.7118
7	**0.1318**	**0.9714**	**0.0357**	0.9883	**0.2064**	**0.9548**
8	0.2194	0.9657	0.0505	**0.9917**	0.3503	0.9463
9	0.3861	0.9457	0.1813	0.9817	0.9088	0.8927
10	0.3553	0.9543	0.0921	0.9783	0.2908	**0.9548**

Table 5.5: Metrics obtained for the test set of each client varying N_r with $N_e = 1$ fixed. Case: 3 clients.

Note that for client 1, the best results in terms of accuracy are obtained for $N_r = 7$, for client 2 with $N_r = 8$ and with $N_r = 7$ and $N_r = 10$ for client 3. In terms of the loss function, the best results are obtained with $N_r = 7$. That is, in most cases the best results for the tests set of each client are reached for $N_r = 7$. It is again observed that more repetitions do not necessarily lead to better results, since these are made by aggregating the different weights obtained for each client, which will vary with each repetition. In addition, the convergence of the method can be clearly seen from the sixth repetition.

On the other hand, if we take $N_e = 10$ and $N_r = 1$, although according to the Equation 5.1 the execution time is the same as in the case where $N_e = 1$ and

$N_r = 10$, the results are actually worse. Specifically, in this case we obtain for the test set a loss of 11.8120, and an accuracy of 0.6250, as opposed to the results obtained for $N_e = 1$ and $N_r = 10$ (see the last row of Table 5.4). Concretely, taking $N_e = 10$ and $N_r = 1$, we obtain the same accuracy as for the case $N_e = N_r = 1$, and a worse value for the loss (see the first row of Table 5.4). This tells us that in this specific use case is more appropriate to perform more repetitions of the federated learning scheme instead of training more epochs on each client. In addition, this may be because training more epochs on each client leads to overfitting on this client, worsening the accuracy for the test set. Intuitively this reinforces the idea of using federated learning to improve results in a decentralized data case.

5.1.3.1 Data Distribution

Let us study the distribution of the two categories present in the data for each of the three clients (see Table 5.6). The objective is to analyze if the two classes are distributed in the same way in the different clients, or if, on the contrary, a different distribution is obtained in each one of them.

	Number of data	**Normal (%)**	**Pneumonia (%)**
Client 1	1400	28.57	71.43
Client 2	2400	16.67	83.33
Client 3	1416	38.21	61.79
ALL	5216	25.71	74.29

Table 5.6: Data distribution for the three clients (without distinguishing train and test sets).

These data have been selected manually, so that the distributions of the data in the different clients are different. For example, in the case of the second client, less than 17% of the samples correspond to images of patients without pneumonia, while this percentage exceeds 38% (more than double) in the case of client 3. In addition, if we compare with the global distribution of the data of the three clients, we can see that between that of the third one and the global distribution, there is more than a 12% difference.

The distribution of the data is one of the most important factors that can affect a federated learning algorithm, in particular, the fact that they are unbalanced (as in this case), is a key aspect to be evaluated. However, as we have seen in the above, despite the unbalanced data, we managed to obtain a substantial improvement by

distributing the data in 3 clients, compared to the case in which they are centralized (passing the accuracy from 0.6619 in the centralized case to 0.7853 in the best performance of the federated learning approach). In the following cases, the process of allocating the data among the different clients will be a random process, so that these differences in the distributions will not be so clear.

5.1.4 Federated Learning Approach (10 Clients)

Now, let us repeat the previous analysis using a larger number of data owners. In this case, instead of decomposing the train set into 3 clients, we will decompose it into 10 of them. Again we will use the model presented in Section 5.1.2. Table 5.7 shows the number of data for each client as well as the average computation time per epoch is shown.

	Number of data		
	Train	**Test**	**Average training time per epoch (s)**
Client 1	442	148	10.1
Client 2	412	138	9.7
Client 3	258	87	6
Client 4	326	109	7.9
Client 5	378	127	9
Client 6	393	132	9
Client 7	356	119	8
Client 8	468	157	11
Client 9	412	138	9.8
Client 10	462	154	11

Table 5.7: Number of train and test data of each client and average training time per epoch. Case: 10 clients.

As we have already explained in the previous case, when training the model in a centralized way, the average time per epoch (average over 10 epochs executed) is 138.6s. In this case, with 10 clients, if we repeat the process 10 times the average time considering that one epoch is trained in each case ($N_e = 1$), will be 110s (see Equation 5.1), while if we train the model for the centralized case 10 epochs, the execution time will be approximately 1386s. Let us see in Table 5.8 the results of training these two cases for the test set.

Centralized approach			*Decentralized approach*		
Loss (test)	**Accuracy (test)**	**AUC (test)**	**Loss (test)**	**Accuracy (test)**	**AUC (test)**
3.0694	0.6619	0.9429	4.7813	0.7212	0.9095

Table 5.8: Centralized approach vs decentralized approach. Case: 10 clients.

We can thus verify (although we will study this in more detail in a later summary of results), that applying federated learning techniques provides substantial advantages over the centralized case. The advantages in terms of execution time are evident (since the different clients will perform their computations in parallel, and the number of data for each one will always be strictly smaller than in the centralized case), and as shown in Table 5.8, there is a very significant improvement also with respect to the accuracy (although the values obtained for the other two metrics are slightly worse). The most significant strength is that it can be observed that in this case, the execution time of the decentralized case reduce by 92.06% the time of the centralized approach, while the accuracy has also increased.

N_r	**Loss (test)**	**Accuracy (test)**	**AUC (test)**
1	14.3417	0.6250	0.4237
2	8.6290	0.6250	0.4328
3	6.7486	0.6250	0.4609
4	16.5803	0.6250	0.5406
5	12.1097	0.6250	0.6825
6	9.6882	0.6250	0.7758
7	5.7260	0.6635	0.8763
8	5.5559	0.6907	0.9055
9	**3.8730**	**0.7340**	**0.9130**
10	4.7813	0.7212	0.9095

Table 5.9: Decentralized approach. Metrics obtained for the test data varying N_r with $N_e = 1$ fixed. Case: 10 clients.

In Table 5.9 we show in detail the test loss, accuracy and AUC values obtained by varying the number of repetitions of the federated learning process (i.e. varying N_r, with $N_e = 1$). Specifically, in this case, the best results for the three metrics are

obtained with the highest number of repetitions performed, $N_r = 10$. Looking at the precision, we can see that it is not until the sixth round that the value of this metric starts to increase, going from 0.6250 during the first 5 repetitions, to 0.7340 in the ninth, and decreasing a little in the tenth to 0.7212. While in the case of 3 clients (see Table 5.4) the accuracy did not exceed 0.6250 until the 5th repetition (when reached 0.6266), in this case it occurs from the 7th repetition onwards (with an accuracy of 0.6635 en this round). This shows that in this second case the convergence is slower, probably because each client has a smaller number of data than in the first case. In this case it is very interesting to note that in terms of the three metrics under study, loss, accuracy and AUC, the best results are obtained for round 9 instead of round 10, which would imply even less execution time, and therefore a temporal reduction of approximately 92.85% compared to the centralized case.

Let us now see in Table 5.10 for which values of N_r (remember that $N_e = 1$) the optimal results (in terms of loss and accuracy) are obtained for each client's test set, and what these values are. Note that the optimum of each metric is not always obtained for the same value of N_r, as is the case of clients 3, 7 and 10.

	Loss (client test set)	**Accuracy (client test set)**	N_r
Client 1	0.2779	0.9662	9
Client 2	0.0001	1.0000	9
Client 3	0.1060	0.9655	10
Client 4	0.0330	0.9908	9
Client 5	0.0192	0.9843	9
Client 6	0.0940	0.9772	10
Client 7	0.3918	0.9412	9
	0.4982	0.9664	10
Client 8	0.2372	0.9618	9
Client 9	0.0029	1.0000	10
Client 10	0.2432	0.9740	9

Table 5.10: Decentralized approach. Optimal values for each client's test set. Case: 10 clients.

In Table 5.10 it can be seen that both in terms of loss and accuracy really successful results are obtained for the test set of each client, with the worst result being Client 7, and the best being Clients 2 and 9 (both with an accuracy of 100%, and

Client 2 with a loss value slightly lower than Client 9).

Again, as in the case of 3 clients, when taking $N_e = 10$ and $N_r = 1$ the estimated execution time is the same as in the case where $N_e = 1$ and $N_r = 10$ (see Equation 5.1), but the results are actually worse. Specifically, in this case we obtain for the test set a loss of 6.5796, and an accuracy of 0.6250, while this values are 4.7813 and 0.7212 respectively for the case of $N_e = 1$ and $N_r = 10$ (see the last row of Table 5.9).

5.1.5 Comparison: Centralized vs Federated

Let us compare the results obtained with the centralized approach and the FL one with 3 and 10 clients. Specifically, Table 5.11 shows the performance obtained in each case under study in terms of loss, accuracy and AUC and the time reduction obtained with the FL approach regarding the centralized one.

	Test Loss	**Test Accuracy**	**Test AUC**	**Execution time (s)**	**Time reduction (%)**
Centralized approach	3.0694	0.6619	0.9429	1386	—
Decentralized approach					
3 clients	**2.6034**	**0.8029**	0.9185	401	71.07
10 clients	4.7813	0.7212	0.9095	110	92.06

Table 5.11: Comparison of the centralized approach and the two FL cases: 3 and 10 clients. Best performance in terms of loss, accuracy and AUC. The time reduction column corresponds with the reduction obtained with the FL architectures regarding the centralized approach.

In the case of 10 clients it is clearly observed that the results for the three metrics are better in the case where 9 repetitions of the FL schema are performed instead of 10, which also leads to a shorter computation time. With respect to the case of 3 clients, it is observed that in terms of loss and accuracy, the best results are obtained in the tenth repetition, but this is not true for the AUC, which reaches its best value in the eighth repetition, again implying less computation time.

In addition, let us study it in more detail the AUC and the ROC curves obtained for the centralized approach and for the round which provides the best results in

terms of AUC for the cases of three and ten clients, analyzing the one obtained for each round $N_r \in \{1,\dots,10\}$, and for $N_r = 10$ (see Figure 5.2). As already mentioned, it can be seen that the best results in terms of this metric are not obtained for the maximum number of rounds performed ($N_r = 10$), but for 8 and 9 rounds respectively for the cases of 3 and 10 clients.

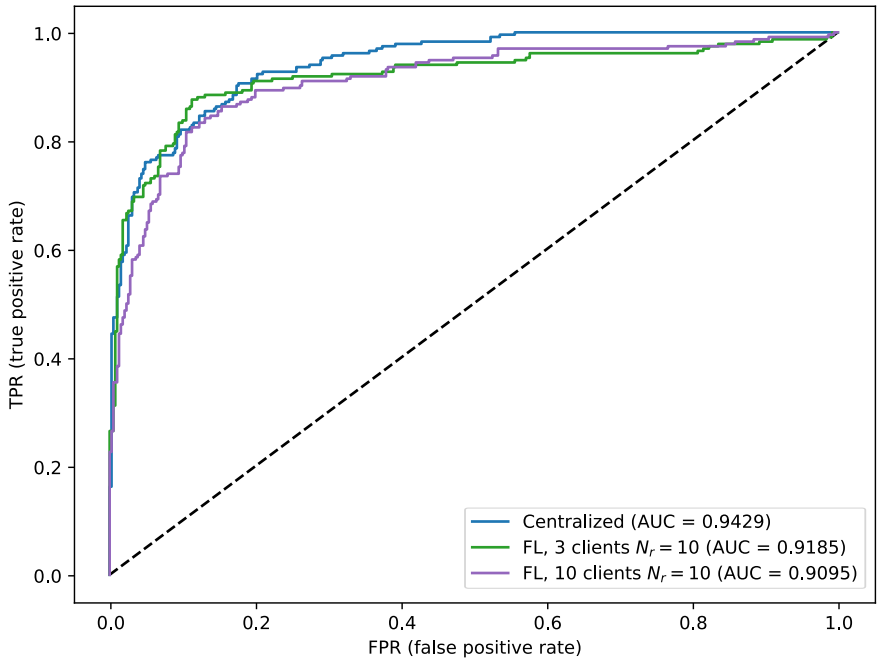


Figure 5.2: ROC curves for each of the cases exposed in Table 5.11

In view of Table 5.2 it can be seen that they are very similar, despite the enormous reduction in execution time provided by the FL approach as the number of clients increases. It is clear that increasing the number of clients will reduce the computation time, but it should be noted that in some cases this may be at the cost of reducing the accuracy.

5.1.6 Federated Learning with Intermittent Clients

In the following, two problems that may arise related to client intermittency will be discussed. As mentioned previously, the problem of intermittent clients can be due to a wide range of reasons. The most common ones can be communication

limitations, connectivity problems or issues related to computing infrastructures. However, in this case concerning the health field, if we assume that this study could involve the collaboration of different hospitals or health or research centers, all of them with X-ray images from different patients, the intermittency could be due to a new center deciding to participate in the training, or others deciding to drop out for other reasons, such as privacy concerns. Therefore, it is important to analyze the problems that may arise and how they could be addressed.

Suppose now that a new client enters the architecture, and one of the clients which was participating in the training, leaves it. This fits with a real case where a data owner decides to leave the training, either voluntarily or involuntarily (e.g. problems with internet connectivity, technical issues, or even privacy concerns). In the case of medical imaging, this can be very common if it is a study in which different hospitals participate, and some leave the training (for any of the reasons stated previously, among others), and others join it once it is started. To exemplify this, first let us come back to the case of a federated learning schema composed by 3 clients. Let us suppose that a client leaves (we will study what happens when client 1, 2 and 3 leave), and a new client enters, client 4, which will build using the validation data.

Note that the new client that we are going to add (client 4), only consists of 16 images, compared to the 1400, 1416 and 2400 of those used previously. Thus, we seek to test the influence of the balancing of the datasets used. We consider two different options:

- **Approach 1:** When a client leaves, the weights obtained for that client in previous repetitions are not taken into account for subsequent repetitions of the training. The weights obtained for the new client are included in the aggregation.

- **Approach 2:** When a client leaves, its last weights calculated are kept and are used in subsequent aggregations to update the model. Again, the weights obtained for the new client are included in the aggregation.

The results obtained for the prediction on the test set with each approach by adding client 4 and removing each of the previous clients are presented in Table 5.12.

In this case we cannot select one approach as better than the other, since by eliminating the first client, the first approach is better in terms of accuracy and loss than the second approach. The same applies to the third client, the second approach is better than the first one. However, the opposite happens when eliminating the

	Approach 1 (test)		Approach 2 (test)	
Client removed	**Loss**	**Accuracy**	**Loss**	**Accuracy**
Client 1	**2.4668**	**0.8092**	2.8923	0.7949
Client 2	4.4945	0.7452	**3.4701**	**0.7821**
Client 3	**2.1746**	**0.8237**	3.5976	0.7340

Table 5.12: Results obtained by eliminating one client and adding a new one with the two approaches described above. Case: 3 clients.

second client (the one that contained a larger number of data), in this case approach 2 produces better results than approach 1, both in terms of loss and accuracy.

Following this same line, the results of removing a client and adding a new one in the case of the federated learning schema with 10 clients are shown. Again in this case the client that we add, client 11, will have as data the images of the validation set (16 images). We consider the same two approaches as in the case of three clients In Table 5.13 the results obtained for the test set when removing each one of the initial ten clients and adding client 11 are shown.

	Approach 1 (test)		Approach 2 (test)	
Client removed	**Loss**	**Accuracy**	**Loss**	**Accuracy**
Client 1	**4.0000**	**0.7500**	4.5387	0.7131
Client 2	**3.6074**	**0.7564**	4.8299	0.7083
Client 3	**3.7131**	**0.7548**	4.3448	0.7228
Client 4	**3.7189**	**0.7548**	4.3951	0.7196
Client 5	**3.7096**	**0.7548**	4.2605	0.7228
Client 6	3.7111	**0.7548**	**3.6957**	0.7340
Client 7	**3.9376**	**0.7436**	4.8988	0.7051
Client 8	**3.4686**	**0.7580**	4.3015	0.7147
Client 9	**3.3949**	**0.7596**	3.7850	0.7260
Client 10	**3.6860**	**0.7548**	8.1755	0.6266

Table 5.13: Results obtained by eliminating one client and adding a new one with the two approaches described above. Case: 10 clients.

In this case, it is clear that in the majority of cases (9 out of 10 for the loss, and 10 out of 10 for the accuracy), the best results are achieved with the first approach

(something that did not seem to be clear when we study the case of 3 clients). That is, without using in further rounds of the FL schema the weights obtained in previous repetitions if one client leaves the training process.

It should be noted that the selection of one criterion or another will depend on the type of data of each client, but in principle, it seems more convenient not to keep the last weights calculated for the eliminated client, rather than not taking them into account (for example, the frequency with which data are updated for each client is key in considering whether or not to discard such weights).

Finally, let us suppose now that one of the clients involved in the process does not send the updates after a pre-established time. As the rest cannot wait indefinitely, a decision must be taken in advance: consider the previous update available or not count on this client. In the same way, once this client sends its update after a certain number of repetitions of the FL schema, it will have to be decided whether it will be included or whether it should not be included until it sends the weights corresponding to the current round. Then, the following two approaches can be studied:

- **Approach A:** Until new weights are received from the client, the last update received by the client is used. Once a new one is received, it is included regardless of the repetition of the process.

- **Approach B:** If a client does not send its updates on time in a certain round, the parameters are not included again, nor are those calculated in other repetitions used. The weights of that client will only be included again when the client re-trains the initialized model with the weights of the corresponding repetition.

Let us illustrate this problem with the following example: suppose that once the model is applied for the fifth time to each client, client i (for certain $i \in \{1, \ldots, N_c\}$) does not send its weights after the pre-established maximum waiting time prior to aggregation. Also, suppose that this client sends the updates at the end of repetition 10. The results obtained with the two approaches exposed for this example are shown for each client in Table 5.14 for the case of three clients and in Table 5.15 for the case of 10 clients.

In this case, we find that approach B achieves better results than approach A in the three clients test sets in terms of accuracy, and in 2 out of 3 regarding the loss. In Table 5.15 the results for the 10-clients case are presented.

In addition, when comparing approaches A and B for 10 clients, we can see that the models fail to converge in terms of accuracy by the end of the FL scheme

	Approach A (test)		Approach B (test)	
Client *i*	**Loss**	**Accuracy**	**Loss**	**Accuracy**
Client 1	12.2229	0.6250	**8.6411**	**0.6522**
Client 2	6.9331	0.6298	**3.5930**	**0.7516**
Client 3	**2.0380**	0.7436	3.0532	**0.7997**

Table 5.14: Results obtained for the test set considering the last repetition of the FL schema and approaches A and B. Client i is the intermittent client considered in this example. Case: 3 clients.

rounds. This is not entirely surprising, since we have already seen that in this case it took 7 repetitions for the model to start converging by increasing the accuracy to 0.6250, and in this case, the model undergoes the change in the configuration of the clients in round 5. However, it is possible to compare these two approaches in terms of loss. Thus, in this example we can see that in 7 out of 10 cases, better results are obtained with approach A than with approach B, i.e. if a client does not send its updates in the corresponding repetition, the previous ones will be used until new weights are received. This is the opposite of what happened in the case of 3 clients. However, what can be clearly observed here is that once the FL scheme undergoes a change, namely some weights are not received in time and/or sent with delay, more rounds will be needed to converge. Again, as already mentioned, it is important to study which approach is suitable depending on the data used, the characteristics of the clients (and the number of data each one owns) and the objective of the problem.

5.1.7 Federated Learning Meets Differential Privacy

In Chapter 3 we exposed the benefits of applying differential privacy to deep learning models when optimizing the gradient descent process. This can also be applied in the context of a federated learning architecture. First, we can apply global DP when aggregating the weights in the server. This can prevent the clients from extracting information from the other clients. In particular this is quite important when there are only two clients participating in the learning process, as both clients will know the model form the other. However, by doing this we are assuming that we trust the central server, which may be not trivial in most cases. In this sense, it is particularly relevant to apply DP when training the model, for example on the optimizer, before transmitting the weights that define the model to the server. In

	Approach A (test)		**Approach B (test)**	
Client *i*	**Loss**	**Accuracy**	**Loss**	**Accuracy**
Client 1	**5.6262**	0.6250	6.7738	0.6250
Client 2	**4.1949**	0.6250	4.3756	0.6250
Client 3	**4.9695**	0.6250	8.2096	0.6250
Client 4	5.1496	0.6250	**4.8699**	0.6250
Client 5	**4.5158**	0.6250	6.4545	0.6250
Client 6	**4.3505**	0.6250	4.3655	0.6250
Client 7	**4.4714**	0.6250	6.3264	0.6250
Client 8	**3.4311**	0.6250	6.0426	0.6250
Client 9	5.1880	0.6250	**4.9086**	0.6250
Client 10	5.4487	0.6250	**4.8596**	0.6250

Table 5.15: Results obtained for the test set considering the last repetition of the FL schema and approaches A and B. Client *i* is the intermittent client considered in this example. Case: 10 clients.

cases where the server is not trusted, sending the model weights to it may make possible the extraction of information by an undesired third party. Thus, by applying DP during the training process, the risks derived from the possible extraction of information by means of the weights will be reduced. However, this may have a negative impact on the results in terms of the error metrics analyzed once the resulting aggregated model is obtained.

In order to test this approach and to evaluate the impact of incorporating DP in the optimizer of a deep learning model in the context of a federated learning architecture, we propose the following: given the present use case, we will take the 3 clients presented in Section 5.1.3, together with the model presented in this case. The idea will be to apply different noise multipliers to an optimizer, namely the *DPKerasAdam* optimizer and analyze the results obtained in each case in terms of the accuracy and the AUC. In particular, it should be noted in this case the experiment will be repeated 5 times with each noise multiplier in order to obtain the mean values of these metrics in test as well as the standard deviation.

Specifically, 5 values of the noise multiplier have been tested (one of them 0, i.e. no noise is applied). The other values are: 0.25, 0.5, 0.75 and 1. In addition, the normalization value l_2, after testing different approaches, has been taken equal to 5.

First, Table 5.16 shows the results as a function of the AUC for the test set of each individual client. Specifically, once the model has been trained for 10 rounds in the federated architecture, it has been tested on the test set of each client (corresponding to 25% percent of the data as explained in Section 5.1.3). Table 5.17 shows the results in the same scenario but in terms of accuracy. In both cases the mean and the standard deviation obtained after performing the training 5 times are shown.

	Mean AUC ± std		
Noise multiplier	**Client 1**	**Client 2**	**Client 3**
0	0.997 ± 0.0002	0.999 ± 0.0002	0.996 ± 0.0003
0.25	0.871 ± 0.0530	0.905 ± 0.0458	0.874 ± 0.0459
0.50	0.785 ± 0.0618	0.801 ± 0.0534	0.757 ± 0.0462
0.75	0.677 ± 0.1191	0.697 ± 0.1307	0.690 ± 0.1307
1.00	0.701 ± 0.0726	0.684 ± 0.1029	0.706 ± 0.0949

Table 5.16: Mean AUC and standard deviation for each client test set when varying the noise multiplier (γ).

	Mean accuracy ± std		
Noise multiplier	**Client 1**	**Client 2**	**Client 3**
0	0.973 ± 0.0046	0.992 ± 0.0033	0.955 ± 0.0075
0.25	0.795 ± 0.0207	0.884 ± 0.0154	0.734 ± 0.0270
0.50	0.767 ± 0.0210	0.829 ± 0.0288	0.654 ± 0.0197
0.75	0.730 ± 0.0293	0.829 ± 0.0190	0.649 ± 0.0287
1.00	0.734 ± 0.0144	0.824 ± 0.0157	0.676 ± 0.0350

Table 5.17: Mean accuracy and standard deviation for each client test set when varying the noise multiplier (γ).

In view of Tables 5.16 and 5.17 we can see, as expected, that the best results with respect to AUC and the accuracy are obtained without applying differential privacy, since we are not including noise in the data. In the same line, it can be seen how the performance worsens as the noise multiplier increases, except for the case of 0.75 and 1 in both cases, which is slightly better in the case where the noise multiplier is 1. This is verified for the mean AUC and accuracy, but it can be noted

that the variance for the case of 0.75 is higher (especially for the AUC, except for the accuracy of client 3). In view of these results, it seems reasonable to apply DP with a noise multiplier of 0.25 in order not to damage the performance so much while adding an additional privacy layer, as the values analyzed which are higher than 0.25 have a very strong impact on the performance of the federated architecture.

On the other hand, given the initial test set (see Table 5.1), Table 5.18 shows the results in terms of accuracy and AUC (both the mean and standard deviation), after evaluating the resulting models on that dataset.

	Accuracy	**AUC**
Noise Multiplier	*Mean ± std*	*Mean ± std*
0	0.72 ± 0.01	0.92 ± 0.03
0.25	0.68 ± 0.02	0.83 ± 0.03
0.50	0.66 ± 0.01	0.74 ± 0.04
0.75	0.64 ± 0.01	0.70 ± 0.07
1.00	0.68 ± 0.03	0.71 ± 0.10

Table 5.18: Mean and standard deviation of the accuracy and AUC obtained in round 10 of federated training with each noise multiplier value analyzed.

In this case, when analyzing the prediction in the initial test set, it can be found that the difference between the cases in which DP is not applied and the cases of a noise multiplier $\in \{0.25, 0.5, 0.75, 1\}$ are not so much disparate (compared to the previous case) for the accuracy, where in the worst case the difference is about 8%. However, this number increases to at least a 9% difference in the case of the AUC, where the most viable option for adding DP of those analyzed is again the one where the noise multiplier is 0.25.

As might be expected, the impact of incorporating DP is clearly reflected in the performance of the model, so assessing the impact and making a trade off between privacy (measured using the noise multiplier), and the minimum expected performance of the model, is a key point to take into account when designing the strategy.

These tests have been carried out by simulating a federated architecture according to the client and server classes presented in Section 4.2.3 from Chapter 4. Specifically, a cloud instance was deployed in the IFCA's cloud via OpenStack so that 4 CPUs, 10.7GB of RAM, and 20GB of disk were available. In addition, in terms of the libraries used, highlight the use of TensorFlow in its version 2.13.0 for compatibility with the TensorFlow Privacy library in its version 0.8.10.

5.1.8 Comparing Different Aggregation Strategies

We will conclude this use case by showing a performance comparison of the FL architecture by training with the 3 clients proposed in Section 5.1.3 and using different aggregation strategies. Specifically, we have slightly modified the neural network proposed in Section 5.1.2 by eliminating the second convolutional layer, and we have trained the model during 10 epochs and 5 rounds. In this line, Table 5.19 below shows the results obtained in terms of accuracy, loss and AUC by applying the FL architecture on these data aggregating with *FedAvg*, *FedMedian*, *FedOpt* and the *FedAvgOpt* strategy proposed in Section 4.2.5.8 from Chapter 4. Additionally, *FedAvgM* was tested with momentum 0.5, showing that convergence was not achieved despite the introduction of some initial parameters for the model from the server side. Likewise with *FedYogi*, which despite converging, did not improve substantially over the rounds, so it was decided not to include them in the final comparison as they did not provide any information of interest in this case. With respect to *FedOpt*, the parameters relating to the learning rate on the server and client side (0.1), as well as the first and second momentum (0) and $\tau = 1e^{-9}$ were left by default according to their default configuration in Flower. Also, in this case, the initial parameters for the model to be trained were introduced from the server side with a random initialization, in order to obtain greater stability when training from the client side.

The results shown correspond to those obtained in the fifth round and for the test set of each client.

		FedAvg	*FedMedian*	*FedOpt*	*FedAvgOpt*
Accuracy	*Client 1*	0.8486	0.8543	0.8886	**0.9600**
	Client 2	0.9233	0.9200	0.9450	**0.9767**
	Client 3	0.7938	0.7599	0.8249	**0.9435**
Loss	*Client 1*	0.9768	1.3829	0.2346	**0.1080**
	Client 2	0.4903	0.6891	0.1195	**0.0492**
	Client 3	1.8409	2.5038	0.4224	**0.1561**
AUC	*Client 1*	0.9828	0.9869	0.9884	**0.9981**
	Client 2	0.9879	0.9844	0.9961	**0.9981**
	Client 3	0.9634	0.9674	0.9833	**0.9941**

Table 5.19: Chest X-Ray use case with 3 clients. Accuracy, loss and AUC in the test set of each client with each aggregation strategy.

In view of the Table 5.19, we can clearly notice the superiority in this case of the aggregation strategy proposed in this thesis (see Section 4.2.5.8 from Chapter 4), named *FedAvgOpt*. In particular, it allows us to obtain an accuracy and AUC much higher in all the test sets of the three clients than the other three strategies explored. In terms of loss, it performs better in all the three clients with the *FedAvgOpt* strategies than with the other functions analyzed. It can therefore be concluded that in this case *FedAvgOpt* allows to reach a final model with better performance than the other explored aggregation methods.

5.1.9 Conclusions

In this first use case we have analyzed the application of a federated learning architecture to a medical imaging case, specifically to a binary classification problem. First, we have explored and compared the performance of the same DL model depending on whether it was applied in a centralized way to the whole training dataset, or under a federated learning architecture simulating 3 and 10 clients respectively from the training dataset. In both cases with FL the accuracy is better than in the centralized case, as well as the loss. The AUC is slightly lower under the FL approach but the reduction in training time derived from the parallel training by dividing the original dataset into different clients stands out. Subsequently, we have studied two client intermittency paradigms and two approaches respectively to deal with them. Then, we have analyzed the impact of incorporating differential privacy (DP) during the training of the DL model to the 3-clients case, using different values for the noise multiplier. Finally, we have tested the applicability of the *FedAvgOpt* aggregation function proposed in Chapter 4, comparing the results obtained in benchmarking with five classic aggregation functions of the state of the art, obtaining in the test set of each client significantly higher results for both the accuracy and the AUC with the proposed aggregation strategy.

5.2 Water Quality: Data Based Estimation of High Frequency Nutrient Concentrations

The results presented in this section has been published in the journal Water Research (Elsevier): ***Sáinz-Pardo Díaz, J.****, Castrillo, M., & López García, Á. (2023). Deep learning based soft-sensor for continuous chlorophyll estimation on decentralized data. Water Research, 120726. [176].*

The following use case presented in this chapter belongs to the field of environmental sciences, specifically, the objective is to use federated learning to estimate continuous chlorophyll concentrations on distributed data.

While freshwater is an essential resource for life, water bodies are increasingly coming under pressure, leading to water scarcity and a deterioration in water quality. In particular, climate change and unpredictable weather patterns and droughts are contributing significantly to the strain on the availability of freshwater arising from urban development and agriculture [191, 192]. Nutrient pollution causing excessive growth of algae, a phenomenon called eutrophication, is a significant challenge to the management of water bodies, threatening the availability of safe drinking water. Eutrophication and Harmful Algal Blooms (HABs) suppose not only environmental, but also societal and economical problems, that are taking place in many natural and artificial water-bodies around the world, like rivers, lakes and reservoirs [193, 194].

Monitoring plays a key role in the contribution to water sustainability. However, the 2022 United Nations' (UN) report [195] about the global progress on the 2030 Agenda for Sustainable Development has recognized that the lack of monitoring, specially in the poorest countries, is limiting the identification of water quality issues at an early stage, and consequently the application of preventive measures before serious deterioration occurs. In addition, the existing sensor technology cannot monitor certain parameters such as the nutrients, pigments or micro-pollutants in real-time and with enough frequency to ensure proper risk management. In the case of HABs, to date there exists barriers to detect them even at the first level of an early warning system [196].

In view of the growing need to carry out this monitoring more accurately and at a higher frequency, different studies have already proposed the use of ML or DL models for this purpose [197]. These techniques allow exploiting the relationship between a target variable and other parameters or surrogates that can be easily and continuously measured with a high time resolution by means of robust and affordable sensors. The software-based sensor that processes the available surrogate data by means of models and allows inferring the target variable(s) is called a *soft-sensor*.

In the field of AI, the models that conform the soft-sensor are data-driven models. Suitable surrogates are mainly hydro-physical indicators like temperature, pH, electric conductivity (EC[1]), turbidity, etc. Regarding chlorophyll (Chl) estimation, ML has been mostly applied to exploit spectral data and other water quality indicators as proxies [198]. Among them, [199] developed ML models based on

[1]Note that when using the term EC in this section we will always refer to electric conductivity.

physical–chemical parameters and different algal species counts. In [200] authors developed an early-warning protocol based on ML models, but they rely on variables (nutrient concentrations) that are not easy to obtain with high-resolution in the long term, as well as in [201]. Recently, [202] applied ML models (Classification and Regression Trees and Random Forests (RF) regressor) for algal bloom predictions using only basic and commonplace sensors as surrogates in two sampling points of a reservoir, demonstrating its feasibility as low-cost and energy-efficient soft-sensor. However, the error of the ML models was high if it is compared with the average values of the Chl concentrations.

One inconvenient that we find when applying ML models to this field is the absence of enough data to train the models. This can happen due to the way in which data are acquired, in most cases involving qualified material and personnel taking the measures in-situ. As this can be an expensive and time-consuming task, the water masses monitoring datasets are not as large as some AI algorithms training requires for achieving a proper convergence. However, gathering the data from different points all together for training a centralizing model can lead to a lack of generalization duo the site-specific nature of such models. A possible solution to this issue can be applying a federated learning architecture for combining individual models built on top of data from different sites.

In order to estimate the Chl concentration we are going to use basic hydrophysical variables as well as in open-access meteorological ones as the input for the data based models, as will be explained in the following section. With the aim of improving the performance and the generalization ability of the models, three ML training paradigms were considered: the individual approach, which only makes use of data from a specific site, the centralized one and the federated learning architecture. In addition, given the large amount of data per site that was available in this use case, different data reduction scenarios were designed, in order to test the different paradigms on more realistic scenarios that may occur specially when the data are obtained by means of manual sampling.

5.2.1 Data Availability and Processing

For carrying out this study we have used an open-access dataset containing information about two tributaries of the River Thames (England): the *River Enborne and The Cut*. Two data files with information about the two tributaries were obtained from the Environmental Information Data Centre platform [203] of the Centre for Ecology and Hydrology (CEH) belonging to the United Kingdom's Natural Environment Research Council. This use case presents two zones with quite different

conditions: while the River Enborne is relatively rural, The Cut is in an urbanized catchment. In both cases we had hourly data from around two years (from November 2009 to February 2012 in the case of the River Enborne and from May 2010 to February 2012 in the case of The Cut), containing information concerning hydro-physical and chemical variables: Chl concentration ($\mu g L^{-1}$), turbidity (NTU), EC ($\mu S cm^{-1}$), water temperature (Temp) (C°), pH, dissolved oxygen concentration ($mg L^{-1}$), percentage saturation of dissolved oxygen (%) and flow rate ($m^3 s^{-1}$). For our study we have selected three hydro-physical features (Temp, pH and EC) as inputs for the DL models. This selection has been made taking into account that these are the easiest to collect due to the robustness and ubiquity of the sensors that record them. The selected variables keep a close relationship with Chl concentration: first, the temperature affects the degradation and solubility of various nutrient compounds, as well as the kinetic parameters of algal metabolism. Second, pH determines the solubility of carbon dioxide and minerals in water and directly or indirectly influences metabolism. Finally, the EC is a proxy for the nutrient concentrations [197] available for algal growth.

In addition, we are interested in using additional variables that can be obtained from open source repositories in order to try to obtain models as informed as possible. In this sense, meteorological variables are of special importance because of their impact on the up-welling of this type of nutrients. Then, three meteorological variables have been used: global irradiance (*Global*) (W/m^2), air temperature (C°) at 2 meters above the ground ($T2m$) and wind speed (m/s) at 10 meters above the ground ($WS10m$). These data were obtained from the Photovoltaic Geographical Information System (PVGIS) [204] which provides hourly data based on data from satellites and reanalysis for given location and dates.

The hydro-physical data of the River Enborne and The Cut had been already curated by the CEH and the University of Reading [205]. Initially, 20412 observations were available for the River Enborne and 15636 for The Cut. However, after removing the missing values (NaN), these were reduced to 18850 and 14162 records in each case respectively.

It is important to note here that although a previous curation of the data has already been carried out, with a first visual inspection of the data it seemed necessary to perform an outlier detection process, as such data could damage the global convergence of the developed models. In this sense, we want to detect values that may come from failures in the sensor measurement (unnatural outliers), without eliminating values that, even if they seem anomalous, may represent the occurrence of an algal bloom. In particular, being able to predict those values that are associ-

ated with the occurrence of an algal bloom is very important in this type of task, then it is critical that these values are not eliminated. Therefore, the InterQuartile Range (IQR) technique or Tukey's test was implemented to detect outliers in the physico-chemical predictor variables as follows: be Q_i for $i \in \{1,2,3\}$ the three data quartiles, then the lower and upper limits established are presented below:

$$\textit{lower limit} = Q_1 - 3(Q_3 - Q_1) \tag{5.2}$$

$$\textit{upper limit} = Q_3 + 3(Q_3 - Q_1). \tag{5.3}$$

Those values below and above the lower and upper boundaries respectively were considered outliers. Just a few outliers were found in the River Enborne (4 in the pH time series and 2 in the WS10m's one) and some more in The Cut (171 in the EC time series and 24 in the pH's one). Considering the large amount of data that is handled in this work, the outliers were directly removed, as the outliers of each variable entailed less than 1% of the observations. Regarding the target variable, Chl concentration, deleting exceptionally high values that could be attributed to an algal bloom would be detrimental to the performance of the soft-sensors. As an absolute value of Chl may be perfectly feasible, the focus was put on the change rate with respect to the previous value, which can give more information about the feasibility of a value. In addition, transforming the Chl concentration into Chl growth rates, considering an exponential behavior, results in a transformation of the distribution of the variable, becoming normal. This allowed implementing parametric methods, as the Tukey's one. Considering x_t the value at instant of time t, t_i and t_j two consecutive time instants ($t_i > t_j$), the growth rate of x is calculated as follows:

$$\textit{growth rate} = \frac{\ln(x_{t_i}) - \ln(x_{t_j})}{t_i - t_j}. \tag{5.4}$$

Once the growth rate had been calculated according to Eq. 5.4, the Tukey's test was applied and eventually the detected outliers were removed. This process was repeated as many times as necessary so that there were no more outliers to eliminate, recalculating the new growth rates, and considering the lower and upper limits calculated with the initial data distribution.

Figures 5.3 and 5.4 shows the time series with the Chl concentration as well as the outliers detected with the criteria indicated above in the first round. In the case of the River Enborne, we have performed the outliers detection and suppression until round number 3 (in which no more anomalous values are detected with the established criteria), while this number rises to 5 rounds for the case of The Cut.

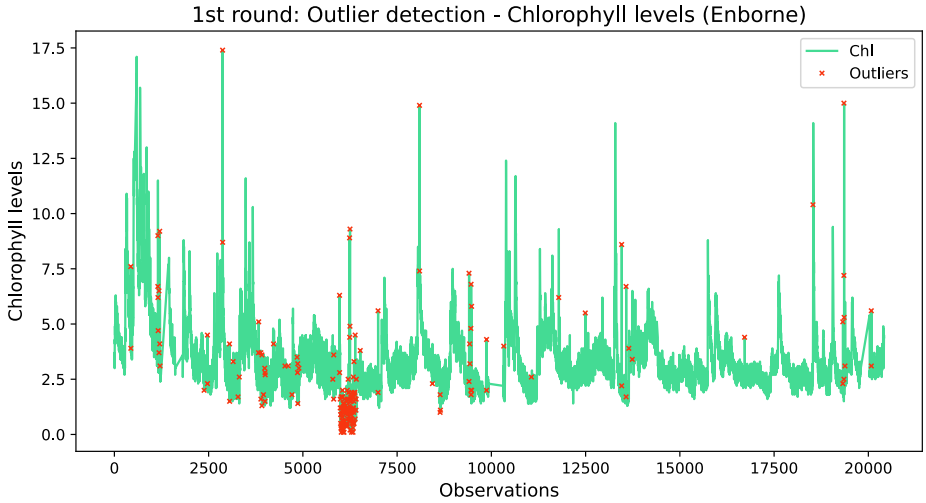


Figure 5.3: Outliers detection of the Chl concentrations for the *river Enborne*.

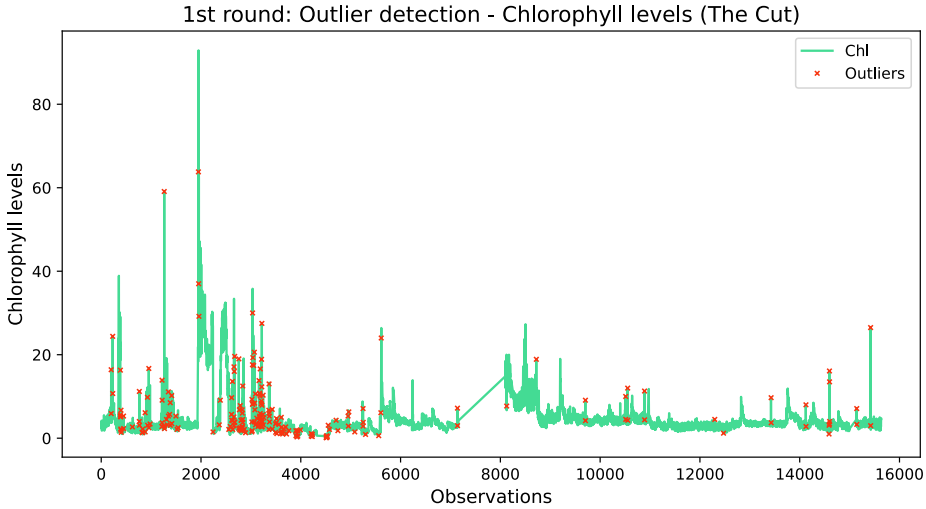


Figure 5.4: Outliers detection of the Chl concentrations for the *river the Cut*.

Note that, concerning the three meteorological features, they were aggregated taking the maximum in the last 24 hours for each observation, representing the extremest conditions each day. It was made like this because the meteorological situation is likely to exert a delayed effect on the algal growth. For example, the wind blowing across the water surface pushes the water from the upper layer away while deeper water rises, but this takes time to occur.

In Figure 5.5 the distribution of the six features that were used to train and test the models is shown respectively once the data have been processed following the steps described above. Specifically, the distribution in the train and test sets for the two rivers is shown by means of violinplots.

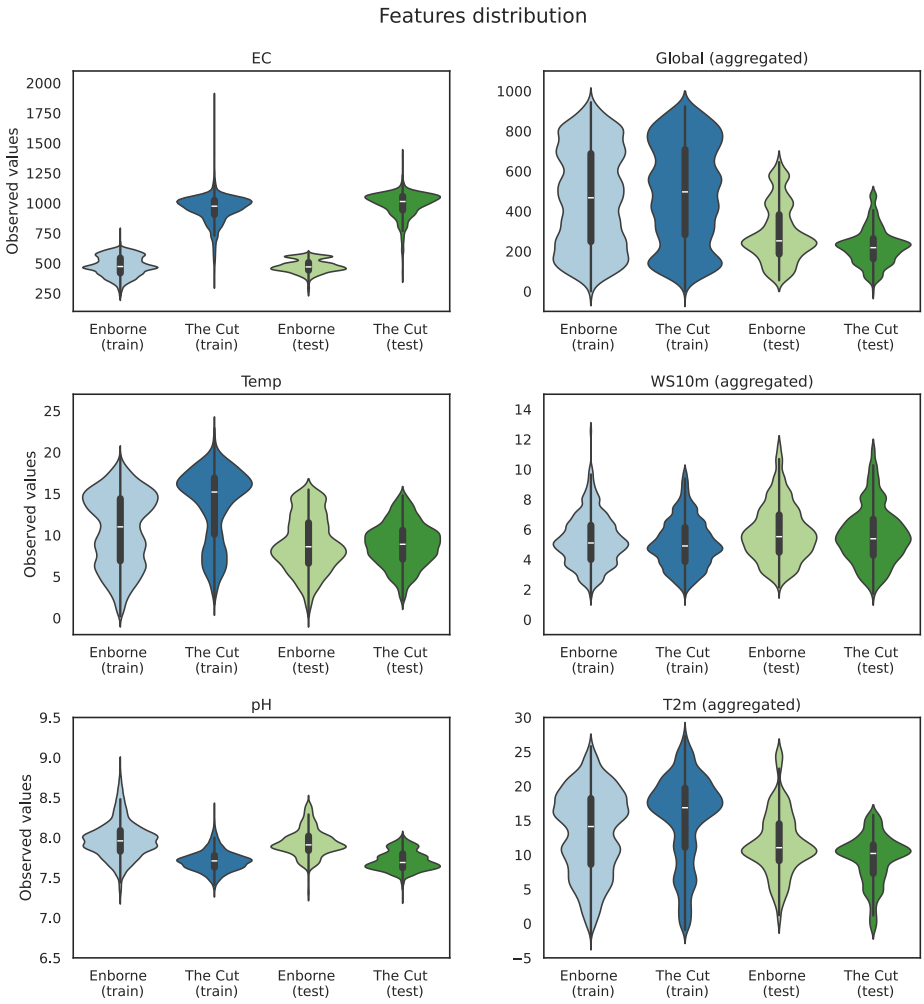


Figure 5.5: Violin plots with the features distribution for the river *Enborne* and *The Cut* in the train and test splits.

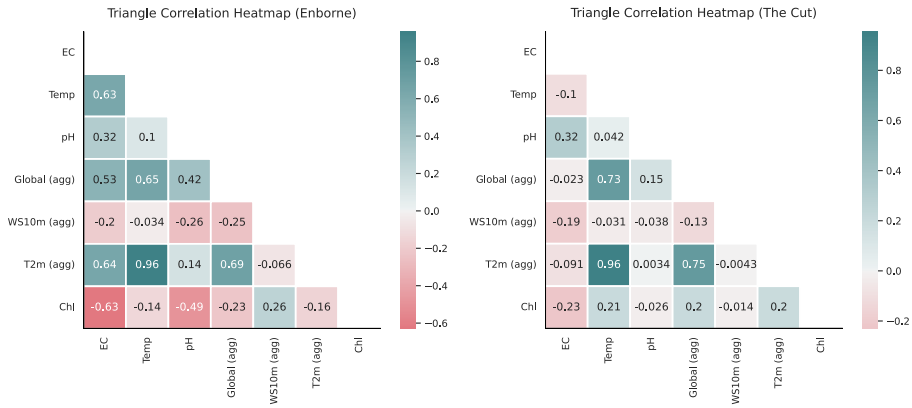
In addition, Table 5.20 shows different statistics associated with the features analyzed, as well as the target variable, calculated after processing the data.

Finally, Figures 5.6(a) and 5.6(b) shown the correlations between the six fea-

Ecosystem	Statistics	EC (μScm^{-1})	Temp ($C°$)	pH	Global (W/m^2)	WS10m (m/s)	T2m ($C°$)	Chl ($\mu g L^{-1}$)
River Enborne	*Min*	215.00	0.20	7.24	0.00	1.45	-2.07	0.50
	Max	769.00	19.50	8.94	944.01	12.62	25.81	17.10
	Mean	476.58	10.27	7.97	431.17	5.36	12.95	3.45
	Std	77.64	4.25	0.22	238.41	1.72	5.83	1.56
River The Cut	*Min*	329.00	1.70	7.23	0.00	1.45	-0.93	0.20
	Max	1877.00	22.90	8.39	924.01	11.24	27.28	92.90
	Mean	959.06	12.68	7.71	435.53	5.17	14.10	4.64
	Std	117.87	4.54	0.13	247.68	1.67	6.38	4.75

Table 5.20: Statistics associated to each parameter for the rivers Enborne and The Cut.

tures by means of triangle heatmaps, obtained using the Pearson correlation. In view of Figures 5.6(a) and 5.6(b) we can note that the linear correlation between the predicting variables and the predicted one is stronger in the case of Enborne than in The Cut, especially highlighting the negative correlation between EC and Chl and between pH and Chl.



(a) River Enborne. (b) River The Cut.

Figure 5.6: Pearson correlations between the six features (Enborne and The Cut). Triangle heat-map.

Additionally, for the data of both rivers, as expected, a positive correlation very close to one is observed between the temperature and T2m, as well as for T2m

and the irradiance (global) and temperature and irradiance. One correlation that stands out for its unusual performance in each of the rivers, is the 0.63 correlation that occurs between EC and temperature in Enborne, which is drastically different in The Cut, where it is -0.1. This gives us a clear evidence of the different nature of the data obtained in each river.

5.2.2 Deep Learning Model Analyzed

The ANN architecture implemented in this work has been selected after analyzing different configurations, varying the number of layers, as well as the number of neurons in each one, the rate in the caps dropout, the inclusion of regularization, the optimizer used and its learning rate, as well as the batch size (finally set to 128), among other factors. The ANN applied to this use case is presented bellow:

- **Dense layer**. Filters: 128. Activation: *ReLU*. Input shape: number of features.
- **Dropout layer**. Rate 0.4.
- **Dense layer**. Filters: 64. Activation: *ReLU*.
- **Dropout layer**. Rate 0.2.
- **Dense layer**. Filters: 32. Activation: *ReLU*.
- **Dropout layer**. Rate 0.1.
- **Dense layer**. Filters: 16. Activation: *ReLU*.
- **Dense layer**. Filters: 1 neuron (the output). Activation: *ReLU*.

The ANN were compiled using the *RMSprop* optimizer with 1×10^{-3} as learning rate and the Mean Squared Error (MSE) as loss function. The Mean Relative Error (MRE) was used as metric to evaluate the performance of the models. Although the use of other error metrics such as MAE was considered, MSE seemed to us the most appropriate due to its higher penalty on the peaks (as it takes the square of the error). Be $y \in \mathbb{R}^n$ the real value and $\overline{y} \in \mathbb{R}^n$ the output obtained with the trained model. Then, the MSE, MAE and MRE are defined as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \overline{y}_i\right)^2, \tag{5.5}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left(\left|y_i - \overline{y}_i\right|\right), \tag{5.6}$$

$$MRE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \overline{y}_i|}{y_i}. \tag{5.7}$$

5.2.3 Methodology

In this use case we will compare the performance of three learning paradigms: individual, centralized and federated learning.

Individual Learning (IL). The first decision that naturally arises when facing environmental ML models is to train using the data from each sampling area independently, due to the site dependence that characterizes ML models. That is, in the case of the rivers Enborne and The Cut, the models are trained individually with data from each river. Thus, two models are obtained and evaluated on data from each river independently.

In this approach, after data curation and pre-processing, the last 20% was kept apart as test set (unseen data to test the models once trained). From the other 80%, the last 15% was used as validation set while the rest was used as training set. The proposed model were initially trained on the training set for 500 epochs and considering a batch size of 128, taking the validation set for quality control through the observation of the loss function. The number of epochs at which the optimum of the loss function is reached (the lowest MSE in the validation set, but also looking at the performance in the train set) is kept to train the models with the whole available data (training + validation). Finally, the models are tested using the test set.

Regarding data scaling, the same process was followed in both cases using *sklearn MinMaxScaler*: the scaler was *fit* and applied individually to each training set. Subsequently, the test dataset was *transformed* using the corresponding adjusted scaler.

Centralized Learning (CL). In the case where no additional privacy or technical restrictions apply to the data, another approach that emerges intuitively is to train the model on data from both zones gathered together. Especially in this case where the data are taken from two rivers with quite different environmental conditions, it is of special interest to test this approach to build a more general model. In this approach, each data partition for both rivers were joined together, but the models were tested on the test data from each river separately. As in the previous approach, the models were initially trained during 500 epochs and considering a batch size of 128. Again, the validation set was used as reference to obtain the optimum number of epochs to train the model with the whole available training data.

It is important to note that in order not to train with biased data so that there is an imbalance whereby too much data from one river falls in the train relative to the other, the same train-validation-test split described in the previous approach (IL) was performed on both rivers before centralizing the data. In this case, the scaler

was fitted with the centralized training data, and afterwards, it was applied to the case of each test set individually.

Federated Learning (FL). Finally, we are going to explore the applicability of FL to this use case. Specifically, there are two potential clients: each one of the rivers (Enborne and The Cut). The same model introduced previously was applied but in this case under a FL architecture. The model was trained in each client for a certain number of epochs N_e, and the scheme was repeated for a number of rounds N_r. Three casuistry were considered regarding the number of epochs and rounds: $N_e \in \{10, 50, 100\}$, and associated to each of them a maximum of $N_r = 500/N_e$ training rounds. The number of rounds of the scheme was optimized according to the MSE obtained for the validation and train sets of each client. In the same way, the number of epochs for training the DL model was selected among the three analyzed values using the validation MSE, seeking a balance between the two rivers. As in the CL approach, the model was tested with the test set from each river independently.

5.2.4 Results and Discussion

In this section the main results for each scenario (training with three or six features) are presented in terms of the MRE for both train and test sets and for each learning approach: IL, CL and FL. Finally, we will show some examples simulating scenarios of data reduction in order to analyze the impact on models and learning architectures of having data sampled at lower or higher frequencies

Training with 3 features Table 5.21 shows the results for the case where the input are the three hydro-physical variables mentioned in Section 5.2.1: EC, Temp and pH.

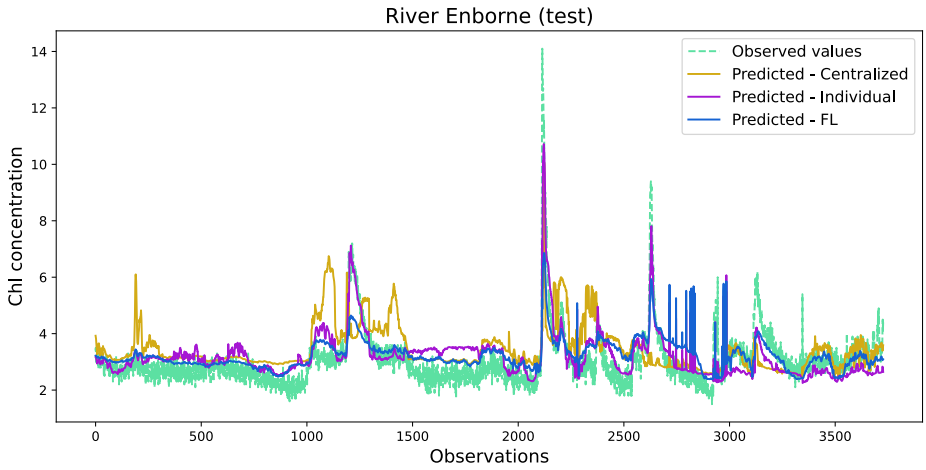
	Enborne		***The Cut***	
Approach	**MRE (train)**	**MRE (test)**	**MRE (train)**	**MRE (test)**
IL	**0.16**	0.19	0.56	**0.17**
CL	0.21	0.18	0.65	**0.17**
FL	0.19	**0.17**	**0.40**	0.23

Table 5.21: Summary of results in terms of MRE for train and test with the three approaches and using three features.

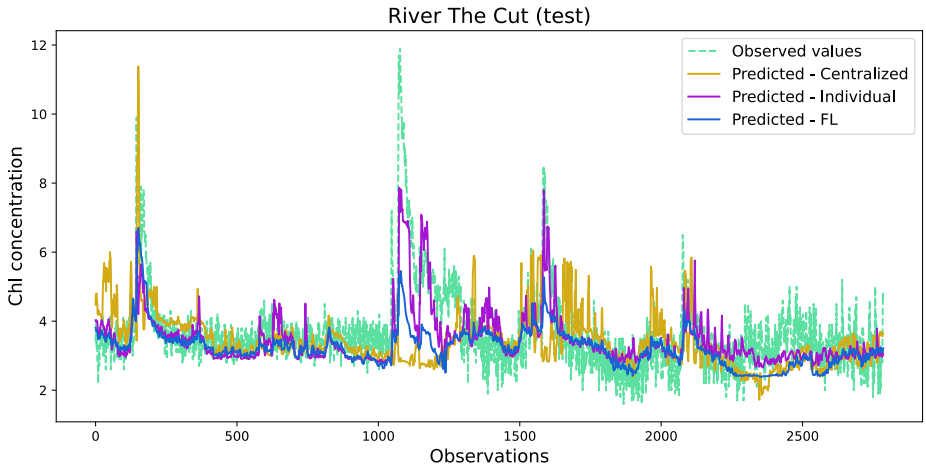
For the FL case the results are shown for the number of epochs in which the

optimum is reached for the validation set for each river (verifying $N_r \leq 500/N_e$). Specifically, in this case it was reached with the same number of epochs for both rivers.

In order to analyze the performance of the predictions on the test set in a more visual way, Figure 5.7 shows the predictions obtained with each learning approach and the observed values.



(a) River Enborne.



(b) River The Cut.

Figure 5.7: Prediction obtained for the test set of each river with the three learning approaches using the three physico-chemical features.

In both cases (Figures 5.7(a) and 5.7(b)) it can be observed that the centralized model creates peaks (specially in the first observations in both rivers), while it is not able to reproduce accurately the significant peaks that are present in both rivers in the first 500 observations. Motivated by these inaccuracies, the meteorological variables were added in order to enhance the models.

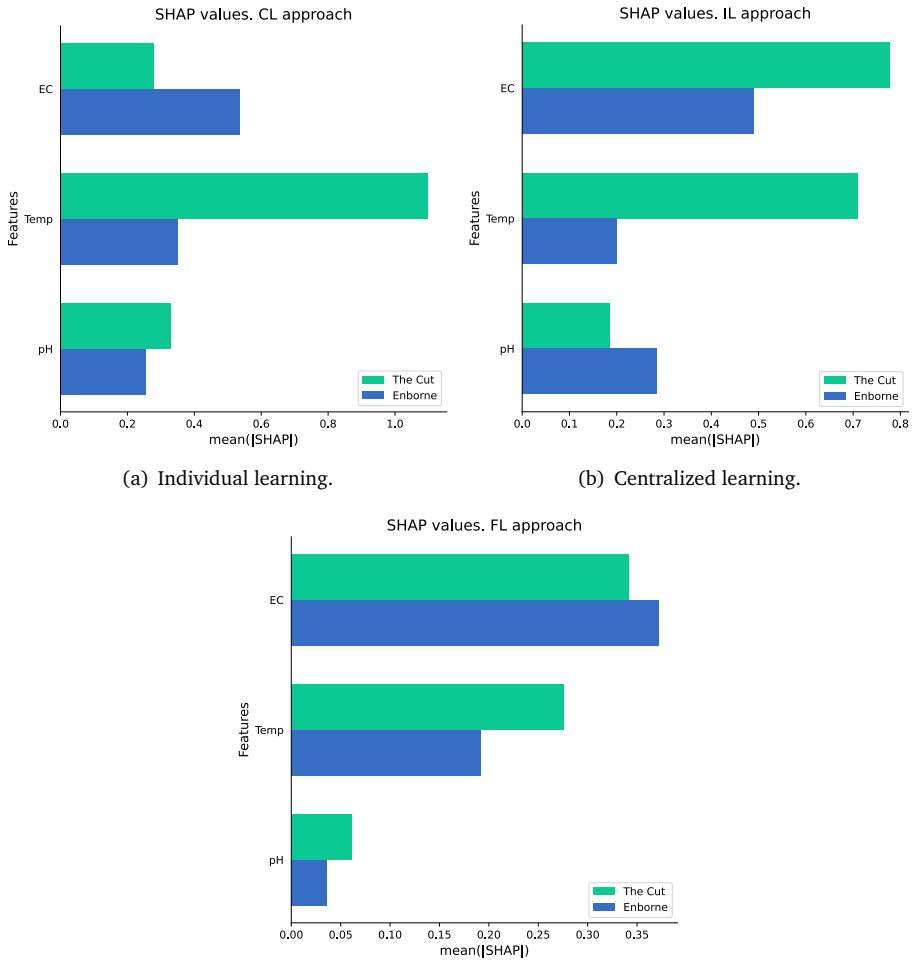
Coming back to the results presented in Table 5.21, we can note that in the case of River Enborne the best approach in the training phase was the IL, however, when it was trained in a FL scheme with The Cut, although the training error was slightly worse, the test error improved. In the case of The Cut, the training error with the IL and the CL approaches were notably high, which to some extent was expected as already occurred with the nutrients estimations in a previous study using the same data [197]. Additionally, the statistics presented in Table 5.20 show that the maximum value for the Chl concentration (92.90 $\mu g L^{-1}$) is far from the mean value (4.64 $\mu g L^{-1}$), revealing that there must exist a peak that may have caused the high error. This is graphically shown in Figure 5.4, where the peak is seen around observation 2000. However, the test error in these cases was lower, which is attributable to the absence of such high peaks in the test set. The FL approach had different effects in River Enborne and The Cut. In the first case the train error was not as low as with the IL, but the test error was lower, avoiding the overfitting to the train fraction obtained with the IL. On the contrary, in The Cut the FL approach did not provide any advantage in terms of the performance of the model for the test, but a large reduction of the train error and a better balance between these two is reached.

In order to check whether each feature contribute to the performance of the models, we have studied the Shapley Additive Explanations (SHAP) values [206].

The SHAP values are a technique that can be used to compute how much each feature has contributed to the current prediction, considering that the features with the largest mean absolute Shapley values are the most important ones. Specifically, the SHAP values assign to each feature the change in the expected model forecast when conditioned on that feature [206]. For calculating this values the Python library *shap* has been used (version 0.42.1). The SHAP values obtained with the four approaches analyzed (IL, CL and FL) for the case of three features are shown in Figure 5.8.

In view of Figure 5.8, EC is the more relevant feature independently of the learning paradigm in the case of River Enborne. It is also the more relevant in The Cut for the IL and FL paradigms, but closely followed by Temp, which is the more relevant for the CL approach in this river. According to Figure 5.5 Temp is the

variable that shows more difference between the two rivers, and the CL approach seems to rely on this variable to assign the intrinsic characteristics of each river.



(a) Individual learning.

(b) Centralized learning.

(c) Federated learning.

Figure 5.8: Mean of the absolute value of the SHAP values obtained for each feature with the three different approaches exposed in Table 5.21 for each river (3 features).

Training with 6 features Following the same scheme as in the previous case, Table 5.22 shows the results obtained when the meteorological variables are added. For the FL approach, the optimum configuration obtained in view of the validation set for River Enborne is shown in the third row while the optimum for The Cut

is shown in the fourth row. In both cases the model has been trained during 10 epochs but different number of rounds (50 for Enborne and 40 for The Cut). The optimum of the validation MSE was obtained for Enborne with $N_e = 10$, while an improvement was obtained for The Cut with $N_e = 100$. However, since the evolution of the validation seemed clearer with $N_e = 10$ than in the cases where the model is trained for 50 and 100 epochs, we have selected this first scenario (in addition to the fact that in the production use case it makes more sense to train the same number of epochs in both cases, in order to perform the training once). It should be noted that the results obtained in terms of training and test in each of these three cases (varying the number of epochs) were very similar.

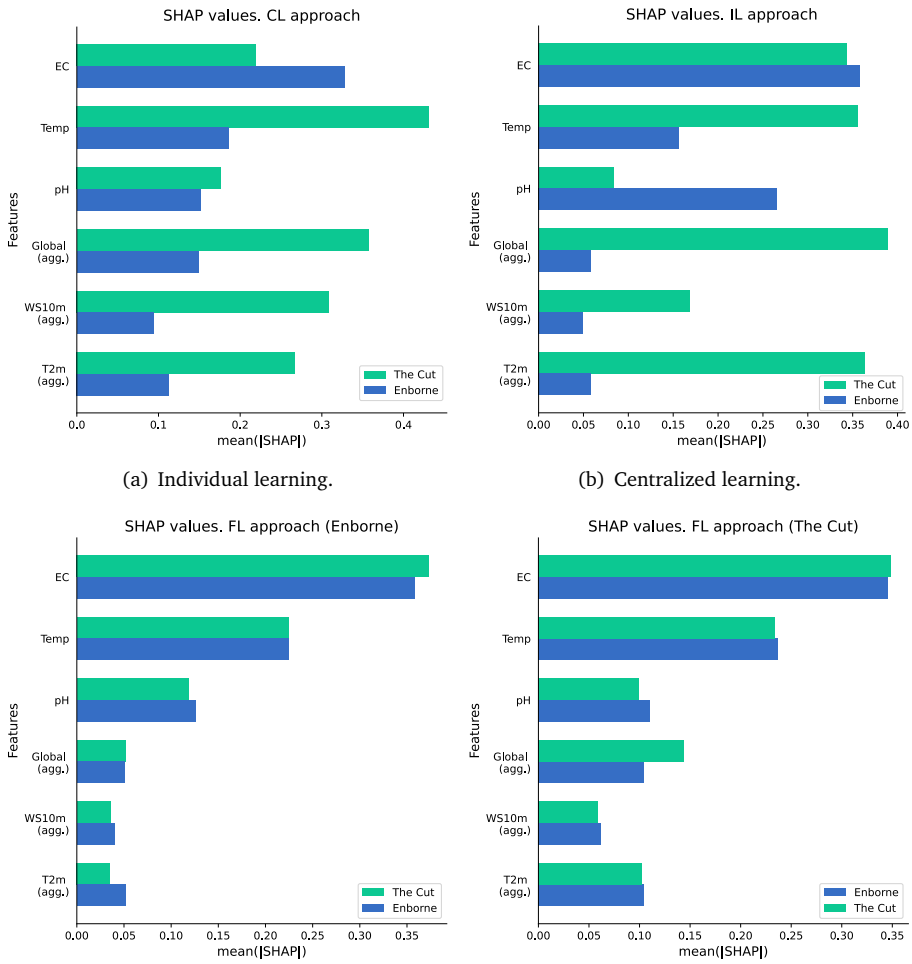
	Enborne		***The Cut***	
Approach	**MRE (train)**	**MRE (test)**	**MRE (train)**	**MRE (test)**
IL	**0.15**	**0.18**	**0.36**	0.21
CL	0.20	**0.18**	0.45	**0.19**
FL (Enborne)	0.18	0.19	0.39	0.26
FL (The Cut)	0.19	**0.18**	0.39	0.26

Table 5.22: Summary of results in terms of MRE for train and test with the three approaches and using six features.

In accordance with this reduction in the train error of all the approaches in The Cut, the best results for this river are those obtained with CL for train and with IL for the test set. Additionally, the best results for River Enborne were achieved through the IL for the train test but with the IL, CL and FL schemes for the test set, the latter providing a better balance between the errors of the two splits.

In view of Table 5.22 it can be seen that the inclusion of the meteorological variables had a positive impact in the train set, especially in the IL and CL approaches in The Cut (see the train MRE). This is also supported by the SHAP values.

Note that Figure 5.9 shows the mean of the absolute values of the SHAP values for each feature and each learning paradigm. Specifically, it can be seen that with the IL and the CL approaches in The Cut, the meteorological variables have a high impact, on the contrary than in River Enborne. However, in the FL approaches, EC and Temp are the most significant features in both rivers. A possible explanation is that, as the rivers are close to each other, the meteorological variables distribution is very similar (as shown in Figure 5.5) and the FL approach does not obtain relevant information from them to differentiate each site.



(a) Individual learning.

(b) Centralized learning.

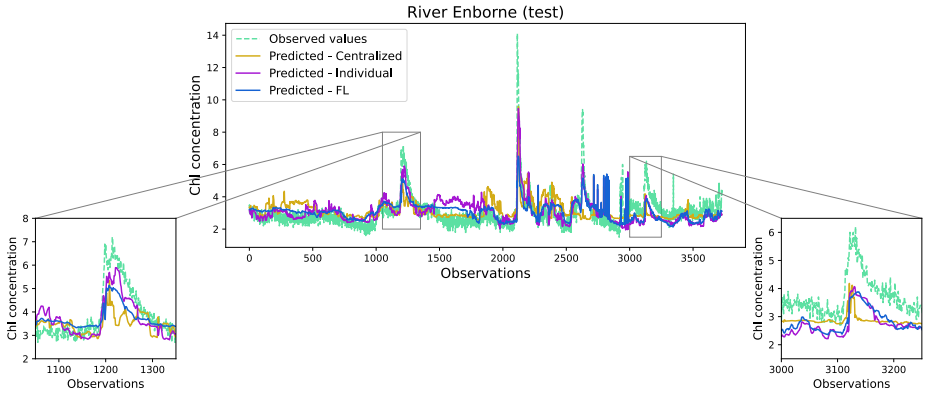
(c) Federated learning (Enborne).

(d) Federated learning (The Cut).

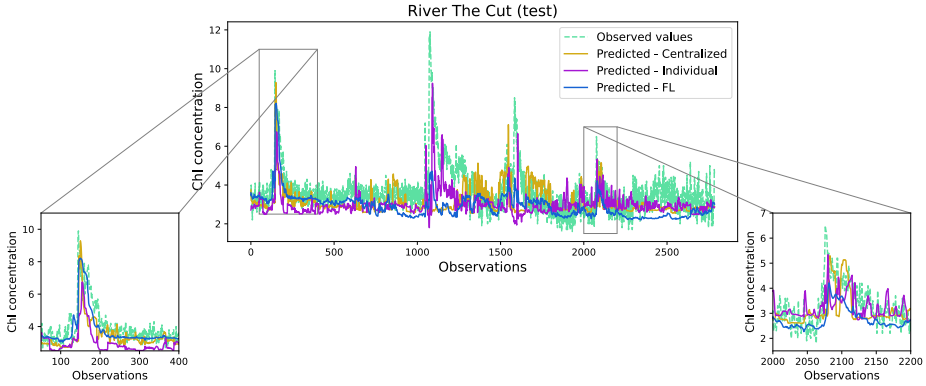
Figure 5.9: Mean of the absolute value of the SHAP values obtained for each feature with the four learning approaches presented in Table 5.22 for both Enborne and The Cut.

In order to visualize the performance of the models along the time line in the test sets we have plotted the observed Chl values together with those predicted with each approach in Figure 5.10, making zoom in some specific areas to show the forecast both in stable regions and in the peaks. Note that for the FL case only the model obtained for the optimum in the validation set of each river is shown. In view of Figure 5.10, it is remarkable that the prediction with the FL approach is smoother

than that obtained with the other learning approaches. This fact resulted in the underestimation of several prominent peaks but a much more accurate prediction in stable periods, although still being able to represent the occurrence of peaks. On the contrary, both the individual and the centralized approaches present more fluctuating predictions, and in the case of River Enborne they tend to overestimate the predictions and produce some peaks that are not seen in the observed values, as in the cases shown in the enlarged views.



(a) River Enborne.



(b) River The Cut.

Figure 5.10: Prediction obtained for the test set of both rivers with each learning approach using six features.

Again, regarding Figure 5.10, for the test set of Enborne in the enlarged section on the left it can be seen that the IL captures the magnitude of the peak better than the FL one, although this approach is also able to reproduce it (although at a lower

magnitude than the real one). In the enlarged section of the right (around the observation 3100) it can be observed that IL and FL are able to detect the presence of the peak better that CL. In the case of The Cut, in the flat periods all the models tend to underestimate the predictions.

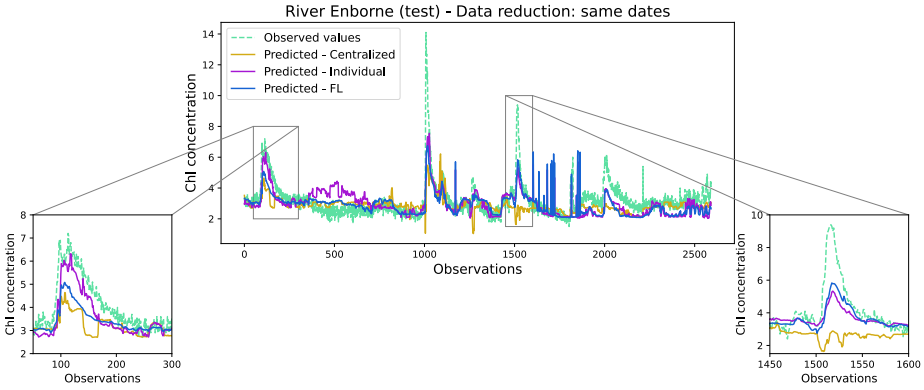
Additionally, between the observations 1000 and 1500 there is a sharp peak followed by a stepped descent, where only the IL approach is able to get close to the peak. The data have been checked in order to elucidate the reason of this mismatching and it was found that this peak was characterized by an unusual low EC and high WS10m. In particular, when the peak reached its maximum (11.9 μgL^{-1}) the EC was 579 μScm^{-1} and the WS10m had reached 10.28 m/s in the last 24 hours. If we compare with the values shown in Figure 5.5 it can be confirmed that these values are scarcely represented in the data, being in the very low range of EC values and in the very high range of WS10m values. The second period of mismatching is around the observation 2500. This period is characterized by a continuous high EC, with values above 1000 μScm^{-1} that are not well represented in the training fraction of the data. In the case of The Cut and first zoom made in the figure, it can be seen how the the models are able to reproduce such peak, as well as its subsequent downward zone and stability.

Data reduction It is important to note that in several use cases we will probably have less data available for training the models. Then, in order to study the performance of each learning approach when we have few data available, the results of three data reduction scenarios are shown in the following. Note that in the case of the River Enborne initially there were 18636 observations available, while in the case of The Cut there were fewer observations (13928).

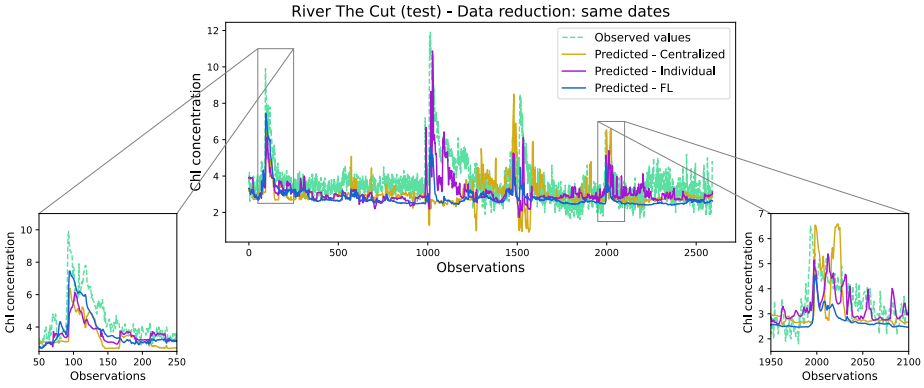
The first scenario analyzed consisted in equalize both datasets by taking only the observations that were taken in common dates. This scenario was intended to avoid the possible influence of the number of data in the proposed models. It should be remarked that the data from November 2009 to May 2010 in River Enborne are eliminated in this scenario, as these dates were not available in The Cut. This reduces the variability of data in the Enborne's dataset, as practically the first half of a water year is removed. With this reduction both datasets have 12960 observations, which implies a reduction of 30.5% in *River Enborne* and 6.9% in *The Cut*. The results obtained as MRE of train and test are shown in Table 5.23, when the six features are used, and the predictions obtained in each case (IL, CL and FL) and the related observed values are displayed in Figure 5.11.

	Enborne ($n_t = 10368$)		***The Cut*** ($n_t = 10368$)	
Approach	**MRE (train)**	**MRE (test)**	**MRE (train)**	**MRE (test)**
IL	**0.13**	0.20	0.44	0.20
CL	0.19	0.22	0.40	**0.19**
FL (Enborne)	0.19	0.18	0.40	0.24
FL (The Cut)	0.19	**0.17**	**0.39**	0.22

Table 5.23: Summary of results in terms of MRE for train and test with the three approaches and using six features in a scenario with the data reduced to coincident dates in both sites.



(a) River Enborne.



(b) River The Cut.

Figure 5.11: Prediction obtained for the test set of *rivers Enborne and The Cut* with each model analyzed. Data reduction: same dates.

In the first scenario analyzed the results remain very similar to those obtained with the original dataset, which allows dismissing the effect of the amount of data to explain the better results obtained in River Enborne. The results for River Enborne continue being better than those for The Cut, which is attributable to the distribution of the data represented by the statistics shown in Table 5.20. Again in this case the best performance for the Enborne test set is obtained with the FL architecture.

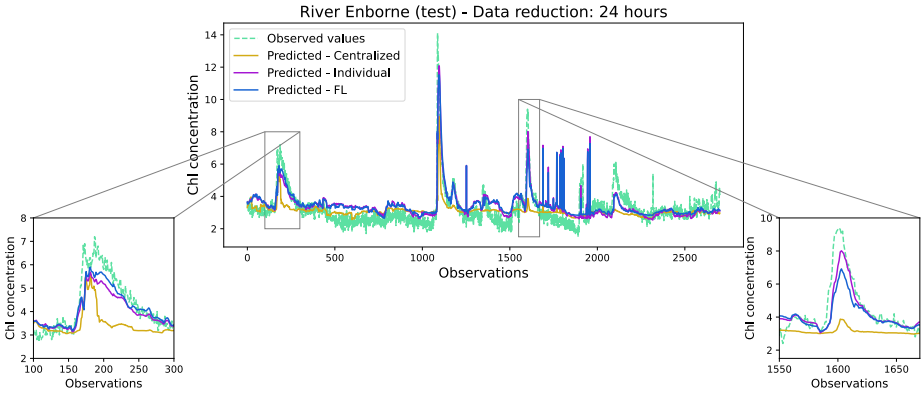
It is important to highlight that in all the scenarios presented in this section, the train, validation and test sets have been recalculated, again taking the last 20% of the complete dataset for the test set and the last 15% of the train set for validating. To be able of comparing this new scenario with the previous one (in which we have more data for training the model), during all this three data reduction experiments we have taken in the FL approach $N_e = 10$, and optimized in each case the optimal number of rounds.

The second scenario was based on the idea that the data are collected with low frequency, having in mind that in many situations the monitoring is carried out manually. Therefore the data were resampled with a frequency of 24 hours (starting at 8 a.m.), simulating a sampling frequency of once a day. The observations were reduced to 617 for training in the case of *River Enborne* and 466 in *The Cut*. As one of the main advantages of soft-sensors is their ability to predict with high frequency, the predictions in the test set are made hourly. That is to say, in this scenario the possibility to use low frequency data to predict high frequency data is tested with the different approaches. The results obtained in terms of the MRE for this data reduction scenario are detailed in Table 5.24.

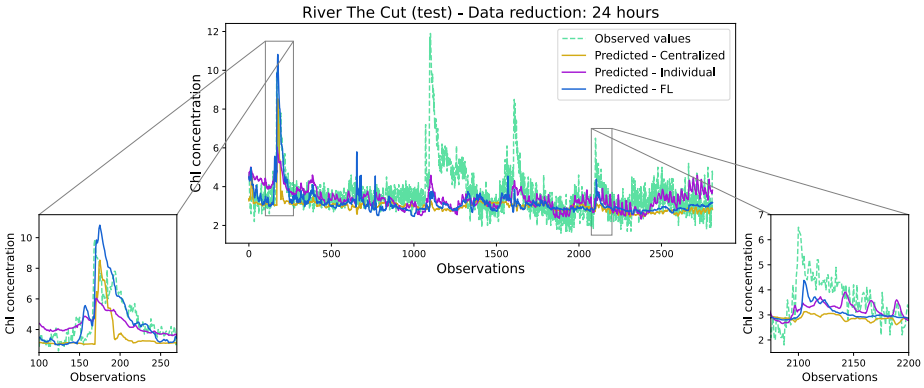
	Enborne ($n_t = 617$)		***The Cut*** ($n_t = 466$)	
Approach	**MRE (train)**	**MRE (test)**	**MRE (train)**	**MRE (test)**
IL	0.28	**0.19**	0.61	**0.19**
CL	0.21	0.25	**0.51**	**0.19**
FL (Enborne)	**0.18**	**0.19**	0.53	0.21
FL (The Cut)	**0.18**	**0.19**	0.53	0.21

Table 5.24: Summary of results in terms of MRE for train and test with the three approaches and using six features in a scenario with data reduced to a periodicity of 24 hour for training, and testing every one hour.

Again, in this case we can make a visual inspection of the predictions obtained with each approach in the test set with each architecture, zooming in on different areas of interest, as shown in Figure 5.12.



(a) River Enborne.



(b) River The Cut.

Figure 5.12: Prediction obtained for the test set of *rivers Enborne and The Cut* with each model analyzed. Data reduction: 24 hours.

The pronounced reduction of data has a negative effect on the train error in all the scenarios except for the FL approach in River Enborne. The penalty is more remarkable in The Cut, which is attributable to the lower capacity to learn about the pronounced peak. However, in this same scenario the test error is not practically penalized, which suggests that one observation a day can be enough, at least with the amount of data available in this study. The optimum models considering the test error are obtained with the FL approaches for River Enborne, and with the CL one

for The Cut. In addition, with respect to the train set the best results are obtained with the IL approach for both rivers, although in the case of River Enborne the FL approach is less overfitted, in other words it presents a higher generalization ability. Again, the predictions obtained in each case and the observed values can be seen in Figure 5.12.

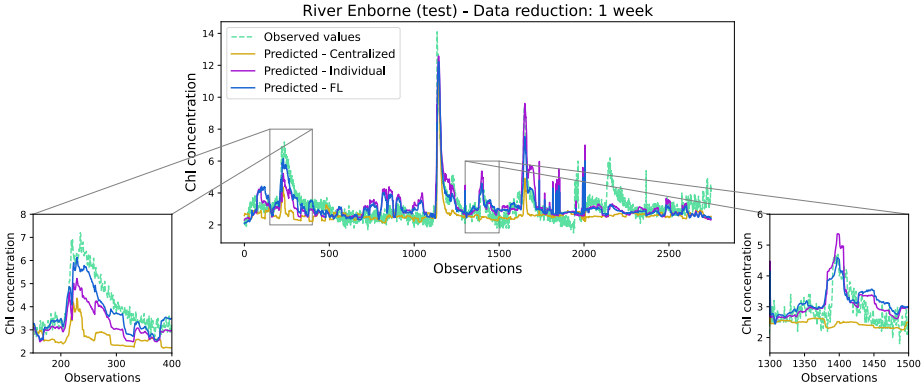
Following with the focus on the ability to predict at high resolution when few and widely spaced data are available, the next scenario uses data collected weekly for the training phase. Although this seems a drastic data reduction, this sampling scenario can be considered realistic enough, as it is not uncommon to perform manual sampling only once a week, where access to the sampling area is difficult or when there are limited resources. As in the previous example, the prediction with the test set is made hourly. The MRE for both train and test datasets is shown in Table 5.25.

	Enborne ($n_t = 91$)		***The Cut*** ($n_t = 66$)	
Approach	**MRE (train)**	**MRE (test)**	**MRE (train)**	**MRE (test)**
IL	**0.21**	0.19	0.60	**0.23**
CL	0.22	0.24	**0.48**	0.26
FL (Enborne)	0.22	**0.17**	0.55	0.24
FL (The Cut)	0.22	**0.17**	0.57	**0.23**

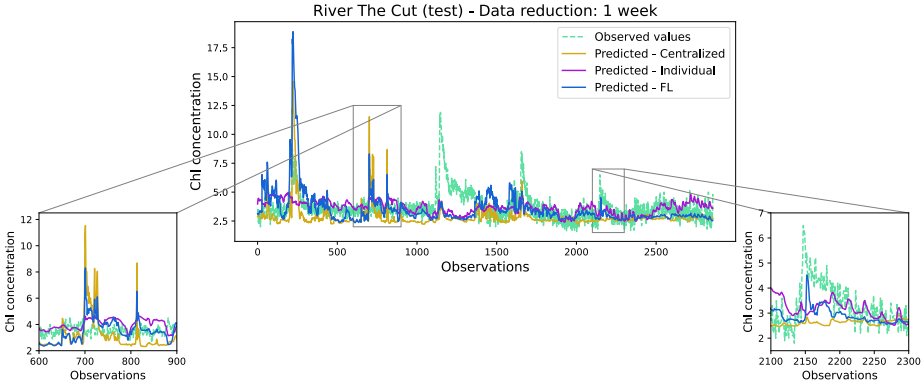
Table 5.25: Summary of results in terms of MRE for train and test with the three approaches and using six features in a scenario with data reduced to a periodicity of one week. Prediction: every one hour.

Again in this case, the optimum for Enborne test set is reached with the FL approach and for The Cut test set the FL approach performs better than IL and the same as CL. As for both training set, the optimum is obtained with the IL approach as expected. We can observe that the error for Enborne test set has decreased (using the FL approach for Enborne) with respect to the previous case despite the reduction of the number of training data, and we can conclude that in this case a reliable generalization to higher resolution data is obtained. This example clearly shows the generalization capacity of the FL approach in both cases with respect to the IL architecture (with the number of rounds in which the optimum was obtained for the validation of each river respectively) in cases of low resolution, closely spaced and few data.

Finally, the predictions for the test set of this data reduction scenario can be found in Figure 5.13.



(a) River Enborne.



(b) River The Cut.

Figure 5.13: Prediction obtained for the test set of *rivers Enborne and The Cut* with each model analyzed. Data reduction: 24 hours.

5.2.5 Conclusions

ML and DL techniques show a strong capability to predict Chl concentrations based on basic hydro-physical variables that can be measured with robust and economic sensors. Additionally one can introduce openly available meteorological features, which can have a notably positive impact.

Regarding the learning paradigm, IL normally provides better results on the training set, but combining the information of two sites in a FL paradigm can re-

sult in a higher generalization ability, as it has been demonstrated for the case of River Enborne. In the case of The Cut, in spite of not having achieved such advantage, the FL models are not notably penalized in comparison with the IL. Therefore, the FL paradigm can be recommended when data privacy issues occur, as well as any other difficulty to share the data from a site, without imposing a penalty on the performance of the models. Additionally, in all cases under this approach, the number of epochs multiplied by the number of rounds of the scheme is fewer than the number of epochs used by the IL or the CL, resulting in a lower computational cost and time. In general, in view of the results obtained in this use case, it is more convenient to apply the federated architecture instead of the centralized one in the case where it is intended to use the data from both zones.

These conclusions are maintained in the tested data reduction scenarios, where it has been also concluded that low frequency data can be used to predict high frequency data.

As already mentioned, a case study which such different ecosystems, *a priori* is not the most suitable one for the application of the FL architecture, due to the different distribution of the data in both sites or clients. In a natural way, in order to have a larger amount of data for training, the use of the CL could arise. However, this paradigm has provided the poorest prediction ability in this study, concluding that the FL architecture is generally more convenient. Thus, in this use case we have seen that the data from the urban environment (The Cut) can help the generalization by including new and more varied casuistry to the more rural environment (River Enborne). The results show the interest of FL (especially compared to the centralized one) with special focus on its application when privacy or technical conditions require it, as well as in the case where there is little data available at a particular point, contributing to a better generalization of the models.

5.3 Climate Sciences: Vertical Integrated Liquid Nowcasting using Weather Radar Data

The results exposed in this section had been published in the journal Earth Science Informatics (Springer): ***Sáinz-Pardo Díaz, J.****, Castrillo, M., Bartok, J., Heredia Cachá, I., Malkin Ondík, I., Martynovskyi, I., Alibabaei, K., Berberi, L., Kozlov, V. & López García, Á. (2024). Personalized Federated Learning for improving radar based precipitation nowcasting on heterogeneous areas. Earth Science Informatics, 17, 5561–5584 [177].*

In this third use case we are going to consider an scenario related to the meteorology field. Specifically, we are going to use weather radar images that, among other variables, capture the vertical integrated liquid (VIL), which is the reflectivity recalculated to water content by Marshall-Palmer formula and summed vertically. Then the VIL (kg/m^2) is defined as the vertical integrated quantity of liquid water content (LWC) (kg/m^3). The objective of this analysis is to make short-term predictions (specifically, for the next 5 minutes) of this variable, in order to anticipate meteorological catastrophes.

Weather radars are a useful tool for precipitation nowcasting, whose frequency and intensity is expected to be affected by climate change, as well as various other types of extreme weather events [207, 208, 209, 210]. Reviewing the literature we can find clear evidence about this, such as a fatal flash flood (see for instance [211]) and the deadliest European tornado since 2001 [212, 213].

On the other hand, regarding meteorology-oriented DL, blurry predictions are an interesting challenge [214], requiring a special treatment of loss functions, for example using Generative Adversarial Networks (GAN) [215, 216]. Regarding radar-based storm nowcasting, the approach called DGMR [217] is currently the state-of-the-art, and concerning precipitation nowcasting using DL models, several works analyze the use of different models to better accomplish this task. In [218] the authors propose the use of the *Trajectory GRU* model instead of the classic Convolutional LSTM which was used for example in [219]. However, to our knowledge, our work published as [177] is the first study in which FL and PFL architectures have been applied to try to improve the predictions from classical models using meteorological radar images.

The motivation for applying a FL architecture to this use case comes from the increasing demand for collaboration in the climate sciences field without sharing data, as in many cases such data can be private or have restricted access. In particular, there is a growing demand for weather forecasts with high spatial accuracy, which is driving private weather forecasting products, for example through the installation of private infrastructures such as radars. Their distributed nature, as well as the often proprietary nature of the data, makes them an ideal fit for testing and applying a FL architecture. In addition, high-resolution radars produce large volumes of data, which are not convenient to share in order to train a centralized model.

In addition, due to the inherent nature of the radar data, in this case we have analyzed the use of a novel Personalized Federated Learning (PFL) approach. The aim of PFL is to address some of the major challenges that faces FL, like its poor con-

vergence on highly heterogeneous data and its lack of generalization ability beyond the global distribution of the data [220]. In the case where we have different radars located in multiple locations, it is clear that the adaptability of a global model will vary depending on the test area. Therefore, we are interested in comparing this novel approach with the classical federated scheme (as in the previous sections), as well as in the case of training the individual model in each data capture area, and using as benchmark a classical model of the state of the art of radar image nowcasting, the COTREC method.

5.3.1 Data Availability and Preprocessing

Throughout this study, we will use compact X-band meteorological radar. The radar used in this study is located in the borderland between Czech and Slovak republic. The radar features a parabolic antenna with a diameter 1160 mm which results in half power beam width 1.8°. The device sensitivity is 10 dBZ at 200km range.

The radar detects precipitation as 3D volumes of high reflectivity (radar beam reflections on water droplets, ice crystals and hail). Data are available as hdf5 files storing each day's information. In each file there are measurement products (ground truth) approximately every five minutes throughout the day and 5 minutes nowcasts obtained using the COTREC method which will be exposed within the methodology (Section 5.2.3).

As we stated above, in this use case we are going to focus on the vertically integrated liquid (VIL), given in kg/m^2, which is defined as the vertical integrated quantity of liquid water content (kg/m^3).

Radar image data related to VIL are highly dependent on the season and the time of the year. Therefore, in this case, having images captured every 5 minutes, we have chosen to use data from four months of the first available year. Specifically, we have selected the first four months available, April, May, June and July 2016, which can also represent significant periods in terms of precipitation.

The original radar images are 400km wide square centered at the radar position. However, since most of the information is concentrated in the center of the images, they were cropped from 400×400 to 100×100 (centered in the middle of the original image). Afterwards, they were used on the different models described in Section 5.3.2 to make 5-minute predictions (nowcast) of those images. Some examples of the images obtained concerning the VIL after re-scaling the resolution to the new dimensions (100×100) are shown in Figure 5.14.

From Figure 5.14 we can note that a large part of each image does not contain any precipitation (no rain present), with several images being totally blank and ex-

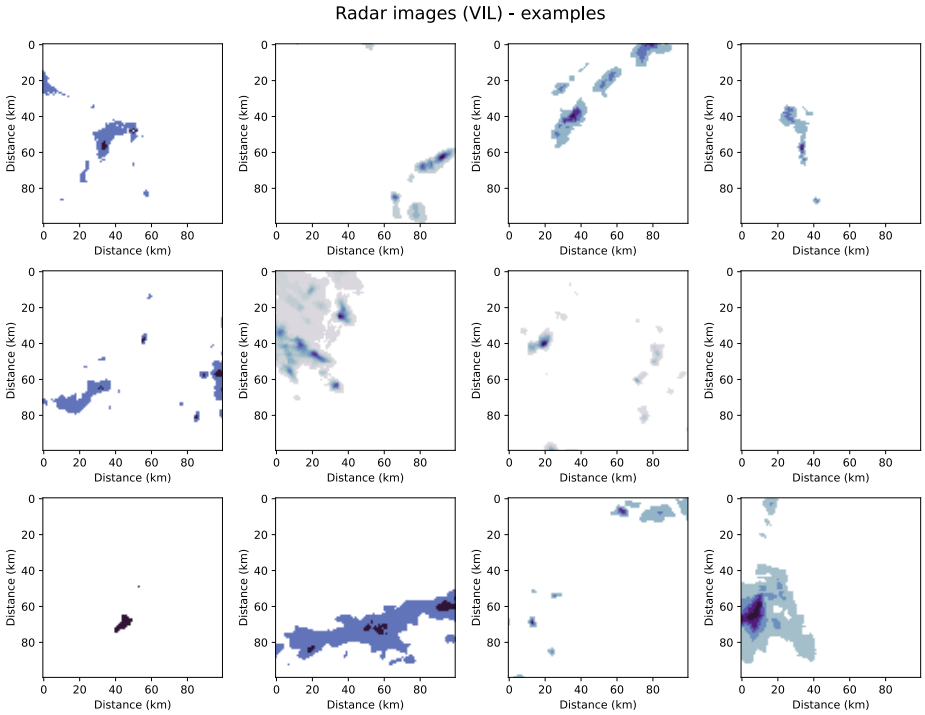


Figure 5.14: Radar images examples (regarding VIL). Images obtained after re-scaling to 100×100 resolution.

posing a high seasonal component. This goes hand in hand with the meteorological events that occurred at the time the radar image was acquired.

For training and testing the models, a split has been made following the temporal order of the images, so that the first 80% of images (28025) is used as the training set, and the last 20% (7007 images) is used as test set. In addition, since many of the available images are blank (no rainfall), additional processing has been performed on them, which is explained in Section 5.3.4. In addition, the validation process was performed once the data had been distributed, in order to apply the approach that should be followed in a real-world scenario. This is explained in the methodology (Section 5.3.2).

Table 5.26 shows some relevant statistics according to the images to be predicted in the test set, after data processing. Specifically, these statistics are those related to the mean of the VIL in each of the images.

An important particularity of the data used in this study is that the information contained in the images usually lies in the center of the initial full image. Therefore,

	Min	*Max*	*Mean*	*Median*	*Variance*	*Skewness*	*Kurtosis*
Zone 1	0	4.2192	0.1201	0.0224	0.0500	3.1100	12.2513
Zone 2	0	3.1011	0.1193	0.0236	0.0592	4.0793	22.2058
Zone 3	0	3.7380	0.1133	0.0249	0.0452	3.0060	10.0654
Zone 4	0	3.2316	0.0956	0.0158	0.0316	2.8355	9.2811

Table 5.26: Statistics regarding the mean VIL in each of the images to be predicted in the test set of each of the four zones under study. In the case of the minimum and the maximum, the average of the min and max values for each of the images are shown.

dividing them in four quadrants to have four artificial clients results in a high divergence among them. In consequence, while the client with the data from the upper left quadrant of the original image will have more information in the lower right corner, this is reversed for the client with the images in the lower right quadrant. However, this is convenient in order to compare this case and use it as a benchmark to extrapolate to different scenarios with multiple radars, since in both cases such divergences will also be appreciated, being data captured in different regions or countries.

5.3.2 Methodology

In order to perform the 5-minutes nowcast of the VIL based on the radar images from the immediately last 5, 10 and 15 minutes, we propose to compare four different approaches. We will use the classic COTREC method as benchmark and compare the training of a DL model under three different architectures, namely individual learning, federated learning and a novel personalized federated learning approach.

5.3.3 Benchmark Model: COTREC

First, highlight that the *Tracking of Radar Echoes by Correlation* (TREC) method [221] is a nowcasting technique that is based on a comparison of two consecutive images of radar reflectivity. For each block of radar pixels, this method identifies a motion vector by maximizing the cross-correlation coefficient between the two consecutive reflectivity images. It can be considered as an image processing algorithm.

As [222] emphasizes, "while TREC is successful in tracking the movements of individual radar echoes, in practice, it usually captures the direction of individual rain cells instead of moving the entire meteorological system on a larger scale". Due to this drawback of the TREC method, various alternatives such as Continuity Tracking Radar Echoes by Correlation (COTREC) [223] have been developed. A recent paper [224] presents a detailed overview of the features of the novel refined methods of the TREC concept that emerged during the last decades.

In this use case we will analyze the COTREC method as a benchmark for comparing the three deep learning paradigms presented bellow.

5.3.4 Deep Learning and Federated Learning Models

First, note that in this use case we have data from one single radar. However, as we are interested in analyzing the scalability of a FL and PFL architecture in a real use case with multiple clients with distributed data depending on the capture area, we have divided the data artificially into 4 different zones. Figure 5.15 shows how the examples displayed in Figure 5.14 are divided by splitting each image into four quadrants.

In this sense, in order to carry out the nowcasting of the VIL using DL we will apply the same ANN following three learning approaches considering 4 zones (as we have split the initial images into 4 new areas): training individually in each of the areas where the data are distributed (IL), under a FL architecture and with the novel PFL method proposed.

In addition, Figure 5.16 shows the average of all the radar images available in April, May, June and July 2016 by zone once processed. In view of this figure we can note that the distributions of the images in each quadrant are very different from each other, being quite heterogeneous zones. Then, when training the model in a federated way, this may difficult the model convergence, since each model would have seen data with different statistics. However, we are interested in keeping these differences because they better reflect real use cases, where radar images will have very distinct distributions if they come from clearly differentiated areas, such as radars located in various countries.

It is important to note that, for training the deep learning models, we have processed the input data in the following way: for predicting the image captured in the next 5 minutes, we introduced the 3 images available immediately before. Thus, by taking the three images we are not only be capturing the speed of the movement and the change in the amount of vertical integrated liquid, but also the acceleration.

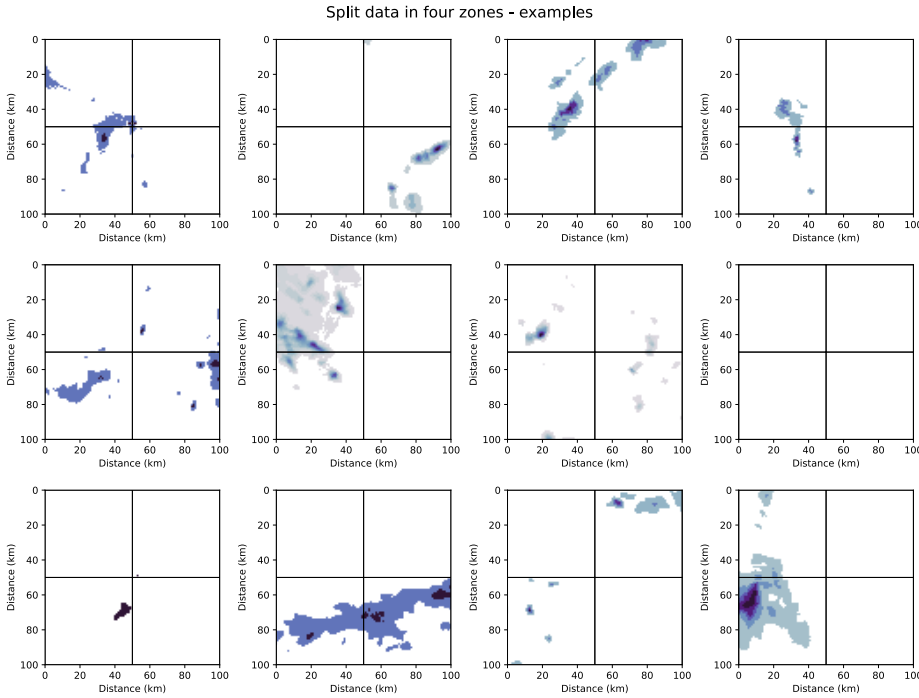


Figure 5.15: Example of an image where the four quadrants into which it has been divided to create the four zones are shown in blue.

In addition, another important part of the processing carried out is that we have eliminated from both the train and test sets those records in which the three images prior to the predictor were empty (blank, no rain). This has also been extrapolated to analyze the results with the COTREC method, taking into account only the error obtained when predicting with the remaining images. The motivation lies in the fact that in these cases, the natural answer would be to predict that there will be no rain (as the three previous images were without rain). Including in the neural network so many empty images can lead to a lack of appropriate adjustment in cases where there are heavy rain events, which as can be seen in the distribution are infrequent even after carrying out this processing, but at the same time are an important factor that nowcasting models must predict accurately.

Table 5.27 shows the number of data for train and test in each of the four zones after carrying out this data pre-processing.

Concerning the training of the DL model, in order to select the model to be used for testing purposes, we need the carry out a validation process. In this case, we

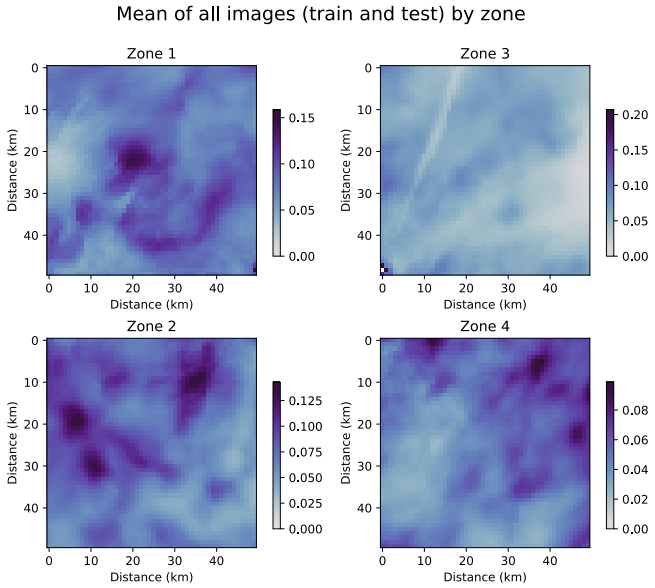


Figure 5.16: Average of all the radar images available in April, May, June and July 2016 by zone once processed.

have performed k-fold cross validation with five folds. Time Series cross-validator from *scikit-learn* has been used for this purpose in order to keep the temporal order of the images. A key point here is to decide on which dataset to perform the validation (as we would not centralize the data). Following a real use case, the four data owners (corresponding to the four zones involved) must agree on the model to be trained. In this case, in view of Table 5.27, we have decided to carry out the validation process using the data from zone 4, as it has the largest number of data for training after processing. Several simple convolutional network architectures have been tested as well as different values for the batch size. The selected one is presented in Section 5.3.6.

5.3.5 Personalized Federated Learning

The implemented FL scheme follows the idea presented in [6], such that the aggregation of the models trained in each zone (in this use case each zone represents a client), is performed using a weighted average according to the number of training data of each client. In addition, a novel PFL paradigm is considered and applied. In these two cases but also for the individual training we will consider the same DL model based on a convolutional neural network, as will be explained in the next

	Train	*Test*
Zone 1	4194	1585
Zone 2	4055	1286
Zone 3	4022	1380
Zone 4	4457	1494

Table 5.27: Number of data after processing in each zone for train and test.

section.

PFL includes a wide group of different strategies to further tailor an FL model to better fit each client's data, while maintaining privacy and security [225]. According to the taxonomy proposed in [220], PFL approaches can be classified in two main types: Global Model Personalization and Learning Personalized Models.

The approach presented in this use case and already discussed in [177] falls into the first group because, at least initially, there is a global model. In fact, it could be considered a kind of Transfer Learning in which the knowledge learned from a source domain (the weights from the whole ensemble of clients) is transferred to a target domain (each particular client). In the following, we will refer to the approach followed in this study regarding PFL as *adapFL* (standing for adaptive federated learning).

In the *adapFL* architecture (proposed in [177]), once the federated training has been carried out for each client during N_e epochs and for N_r rounds. In this use case, the variability in each zone with respect to the others is high, as shown in Figure 5.14. Therefore, it is appropriate that after conducting the federated training, in order to achieve a more robust model by having been trained with more data, these are trained a certain number of epochs on the data of each zone locally, which we have called *adaptive federated training (adapFL)*. The main goal is to be able to capture in each case the peculiarities of each zone before evaluating the results in the test set of that area, but starting with the knowledge and generalization ability provided by the FL model. An intuitive idea of the *adapFL* architecture is summarized in the pseudocode given in Algorithm 7.

In particular, we are interested in analyzing the feasibility of the *adapFL* approach for the case in which images from different areas are available. The goal is to study whether we can allow improving the results that would be obtained by training locally in each zone while maintaining the privacy of the images by not sharing them, or if on the contrary, it is more convenient to carry out an individual

training in each zone, respecting the differential factors of each one of them.

Algorithm 7 Adaptive federated learning (*adapFL*)

INPUT: *clients* (list with all the clients participating on the training, for each client x contains the features and y the labels); n (total number of training data from all clients); *model* (ML/DL model to be trained); N_e (number of epochs); *bs* (batch size); N_r (number of rounds of the FL training).

function AdaptiveFederatedTraining($clients$, n, $model$, n_e, $batch\_size$, n_r)
  for $i \in [1, \ldots, N_r]$ **do**
    $w \leftarrow [\ ]$ ▷ empty list for saving the weights
    for $client \in clients$ **do**
      model.train(client['x'], client['y'], epochs=n_e, batch_size=batch_size)
      w_client $\leftarrow$ *model.get_weights()*
      w.append $\left(\frac{|client['y']|}{\sum n} \cdot w\_client\right)$
    end for
    $w \leftarrow \sum w$
    model.set_weights(w) ▷ Update the model with the aggregated weights
  end for
  $j \leftarrow 1$
  for $client \in clients$ **do**
    model.train(client['x'], client['y'], epochs=N_e, batch_size=bs)
    model.save(f'model_ind_{j}.h5')
    $j \leftarrow j + 1$
  end for
end function

In short, the three learning schemes (including the individual one) are presented in Figure 5.17 in order to compare the different training paradigms of the deep learning model.

In this sense, as the federated training can be carried out in each zone in parallel, in order to make the comparison as fair as possible, it is desirable that the total number of epochs that are trained individually ($N_e^{(I)}$) is equal to the number of epochs that the model is trained locally in the federated scheme ($N_e^{(FL)}$) times the number of rounds (N_r) of the FL training plus the number of epochs the model is trained later in each zone ($N_e^{(L)}$) in the *adapFL* approach. That means:

$$N_e^{(I)} = N_r \cdot N_e^{(FL)} + N_e^{(L)}$$

In the current use case analyzed the following values have been fixed: $N_e^{(I)} = 100$, $N_e^{(FL)} = N_e^{(L)} = 10$ and $N_r = 10$ for the classic FL architecture and $N_r = 9$ for the *adapFL* method.

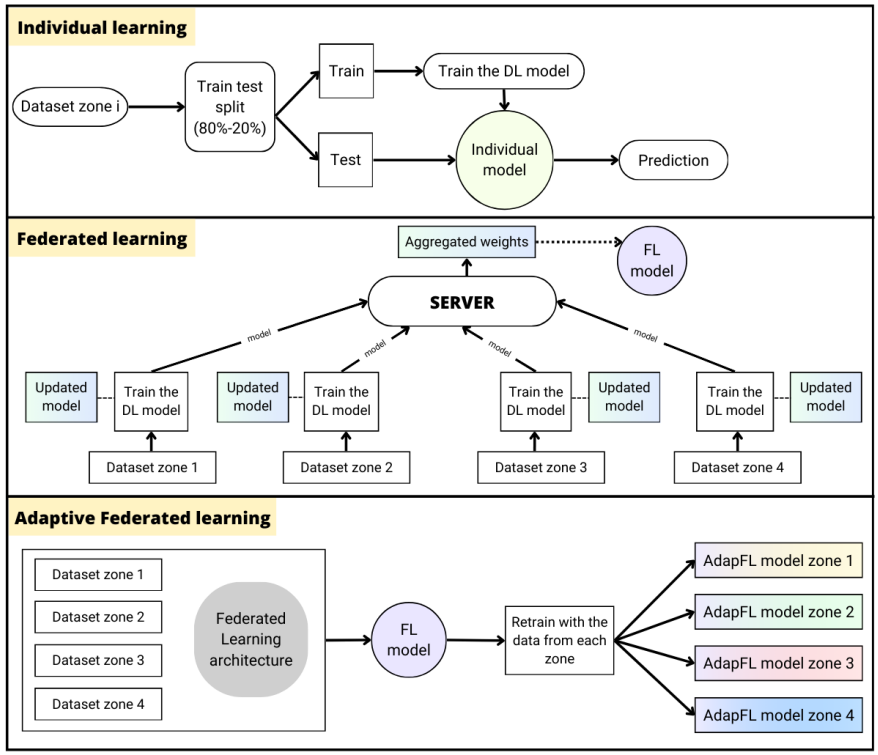


Figure 5.17: Schema of the three learning paradigms implemented: individual, federated and adaptive federated learning.

5.3.6 Convolutional Neural Architecture

Both in the case of individual learning in each zone and in the case of the *adapFL* schema, we have developed a neural network architecture composed of the following convolutional layers:

- **Conv2D layer**. Filters: 128. Kernel size: (3, 3). Activation: *ReLU*. Input shape: (50,50,3).
- **Conv2D layer**. Filters: 64. Kernel size: (3, 3). Activation: *ReLU*.
- **Conv2D layer**. Filters: 32. Kernel size: (3, 3). Activation: *ReLU*.

- **Conv2D layer**. Filters: 16. Kernel size: (3, 3). Activation: *ReLU*.
- **Conv2D layer**. Filters: 16. Kernel size: (3, 3). Activation: *ReLU*.
- **Conv2D layer**. Filters: 32. Kernel size: (3, 3). Activation: *ReLU*.
- **Conv2D layer**. Filters: 64. Kernel size: (3, 3). Activation: *ReLU*.
- **Conv2D layer**. Filters: 128. Kernel size: (3, 3). Activation: *ReLU*
- **Conv2D layer**. Filters: 1. Kernel size: (3, 3). Activation: *ReLU*

The architecture is kept purposely simple, as the intent of this paper is to evaluate the potential of the FL and PFL approaches for this use case, not to establish a new state of the art model in precipitation or vertical integrated liquid nowcasting based on radar images.

The model has been compiled with *Adam* [226] as optimizer and the *mean squared error (MSE)* as loss function and this metric and the *mean absolute error (MAE)* as quality metrics for monitoring. In addition, the *skill score* (calculated as given in [227]) will be calculated. The skill score can be obtained from the MSE and allow us to compare the DL models with a baseline one. Then, the skill score for the model $\mathscr{M}$ ($SkillScore_{\mathscr{M}}$ can be defined as given in Equation 5.8, being $MSE_{\mathscr{M}}$ the MSE for the model $\mathscr{M}$ and $MSE_{\mathscr{B}}$ the MSE for a baseline model. In our study we have considered COTREC as the baseline model for calculating this metric.

$$SkillScore_{\mathscr{M}} = 1 - \frac{MSE_{\mathscr{M}}}{MSE_{\mathscr{B}}} \tag{5.8}$$

Again, as previously explained, this model has been validated using the data from zone 4. Specifically, the mean MSE obtained in the five folds is 0.0326.

5.3.7 Results and Analysis

In this section, the results obtained in each of the zones test sets when evaluating each of the following approaches are exposed and analyzed:

- The COTREC method as benchmark,
- An individual learning model trained for 100 epochs in each area (IL),
- A FL model trained 10 epochs during 10 rounds (FL),
- A PFL model in which the FL model is trained 10 epochs for 9 rounds, ending with a local training of 10 epochs on the data of the testing zone (*adapFL*). We will have 4 models for this approach (one for each area), while we will only obtain one FL model for the previous approach.

Tables 5.28 and 5.29 and 5.30 summarize the results obtained in terms of the MSE and MAE respectively in the test set of each of the four zones and with the four approaches. Note that we have calculated these two metrics taking the mean error in each image, and then the mean of all of them as stated in [228] and [229].

	COTREC	*Individual*	*FL*	*AdapFL*
Zone 1	0.2306	0.1399	0.2000	**0.1352**
Zone 2	0.1756	0.1057	0.1612	**0.1005**
Zone 3	0.2235	0.1220	0.1286	**0.1082**
Zone 4	0.1095	0.0852	0.0958	**0.0815**

Table 5.28: MSE (kg/m^2) obtained in the test set with the different approaches in each zone.

	COTREC	*Individual*	*FL*	*AdapFL*
Zone 1	0.0708	0.0618	0.0649	**0.0606**
Zone 2	0.0650	0.0530	0.0590	**0.0517**
Zone 3	0.0729	0.0607	0.0576	**0.0565**
Zone 4	0.0514	0.0482	0.0484	**0.0464**

Table 5.29: MAE (kg/m^2) obtained in the test set with the different approaches in each zone.

In view of the results shown in Tables 5.28 and 5.29, the best performance is reached using the *adapFL* architecture. Moreover, all the three DL approaches achieve better results than the benchmark model (COTREC). It should be noted that with the conventional FL architecture, the results of the COTREC model were improved substantially, but not those of the IL, since the latter has a better capacity to adapt to the data captured in its corresponding area. Thus, with the adaptive step of the *adapFL* architecture, we managed to improve the results of the FL architecture, but also those of the IL approach. Note that the adaptive step in *adapFL* can be seen as a kind of transfer learning from the FL model.

From Table 5.28 we can see that the MSE is reduced in the *adapFL* approach with respect to the FL approach by more than 30% for zones 1 and 2, by more than 20% for zone 3, and by more than 10% for zone 4. The results closest to those of *adapFL* are those obtained with individual training, with *adapFL* providing a slight

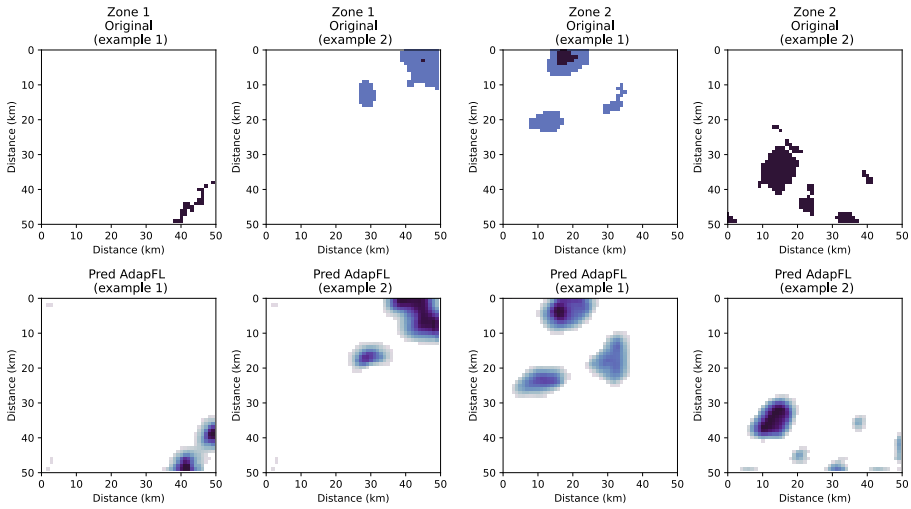
improvement in the generalization ability derived from the fact that it relies on more data during the training of the initial FL architecture.

Finally, we have analyzed the skill score (defined as given in Equation 5.8) in order to check which model given a greater improvement regarding the baseline model (COTREC). The values obtained for the skill score are shown in Table 5.30 for the individual, federated and *adapFL* approaches. In view of Table 5.30 we can note that the greatest improvement in relation to the COTREC model is obtained in the four cases with the *adapFL* architecture.

	COTREC	*Individual*	*FL*	*AdapFL*
Zone 1	-	0.3933	0.1327	**0.4137**
Zone 2	-	0.3981	0.0820	**0.4277**
Zone 3	-	0.4541	0.4246	**0.5159**
Zone 4	-	0.2219	0.1251	**0.2557**

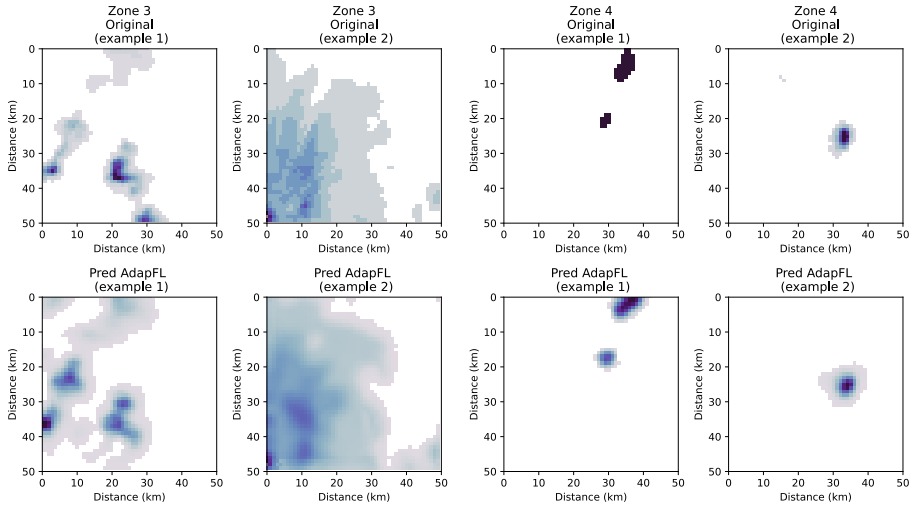
Table 5.30: Skill score obtained with each approach in the test set of each zone.

In order to check the predictions obtained for each area, Figures 5.18(a), 5.18(b), 5.19(a) and 5.19(b) show two examples for each area in which the image to be predicted is displayed together with the predictions obtained with the *adapFL* method.



(a) Zone 1. (b) Zone 2.

Figure 5.18: Example of predictions for the zones 1 and 2 with *adapFL*.



(a) Zone 1. (b) Zone 2.

Figure 5.19: Example of predictions for the zones 3 and 4 with *adapFL*.

Predicting a central area Once we have the four individual models and the models trained in a federated way, we are interested in seeing how the last ones perform in a different region. To do this, we take the 50×50 central crop of the original image (so that it is of the same resolution as the images of each of the 4 zones). An example of an image where this division is taken is shown in Figure 5.20, where the central area selected is the one framed within the red square.

Let us consider all the images within this quadrant and evaluate the predictions that would be obtained with the FL model and the four adaptive FL models for each of the 4 initial areas. Also, we compare it with the IL model on that area and the *adapFL* one with the adaptive phase performed in such zone (the central one), but the FL model only on the first 4 zones. Since part of the images that we evaluate are part of the training set of each individual zone, we will take the same test set as in the previous cases, corresponding to the last 20%. Again, the same processing will be carried out in each zone regarding the blank (empty, no-rain) images and the input for the DL models. The results obtained in each case for the train and test set and both for the MSE, MAE and skills core are shown in Table 5.31. Note from this table that the first three rows correspond to models that have explicitly seen data from the central zone training set (COTREC, IL over the central zone, and *adapFL* with FL over the four initial zones and personalized with the adaptive phase over the central area with a configuration (9,10)). The rest of the models shown

in Table 5.31 have not directly seen training data from the central zone, although each of the four initial zones contain part of the data from the central area. We have highlight the best results obtained for each metric in these two cases.

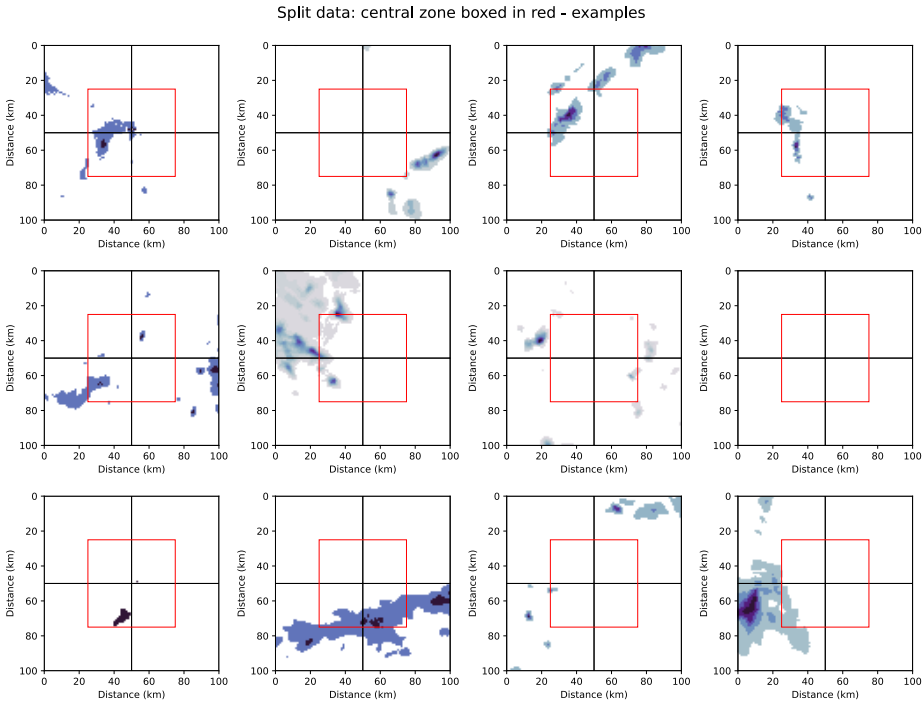


Figure 5.20: Example of images showing the four quadrants into which it the initial images have been divided to create the four zones (black) and the central area (marked in red).

From Table 5.31, we can highlight several points. First, note that it makes sense to analyze the performance of the FL model since each initial area has part of the central zone analyzed in this case. Intuitively, the best results should be obtained with the models trained in the same zone, in this case IL in the central zone and *adapFL* on the central zone. In addition, we can note that the COTREC model is largely worse than the other approaches analyzed, so it can be extrapolated that in this scenario it is more convenient to apply DL models.

Concerning the training dataset, the *adapFL* approach is the one that performs best on the central zone in terms of MSE, and the individual one with respect to the MAE (followed by *adapFL* over the central zone). In the same line, regarding the test set the best results for MSE are obtained with the individual model, followed

by the *adapFL* ones over zones 3 and 4. In the test set the best result with respect to the MAE is obtained with the *adapFL* model over zone 4 followed by the basic FL model, being in this case the FL approach better than the individual model trained over the current area of study.

The skill score (calculated using with COTREC as baseline) shows a greater difference between the MSE for the baseline model and the other approaches than in the previous case where the four initial zones were analyzed (greater than 0.83 in all cases for the train set and greater than 0.94 in all cases for the test set).

Table 5.31: Comparison of the MSE (kg/m^2), MAE (kg/m^2) and skill score obtained in the train and test sets of the central zone with the different approaches analyzed.

	Central zone					
Model	*MSE Train*	*MSE Test*	*MAE Train*	*MAE Test*	*Skill Score Train*	*Skill Score Test*
COTREC	0.0798	0.2547	0.0308	0.0735	-	-
IL zone central	0.0118	**0.0121**	**0.0130**	0.0207	0.8521	0.9525
adapFL (central)	**0.0067**	0.0129	0.0149	**0.0199**	**0.9160**	0.9494
FL (four areas)	0.0128	0.0140	**0.0161**	0.0195	0.8396	0.9450
adapFL (zone 1)	0.0111	0.0130	0.0174	0.0210	0.8609	0.9490
adapFL (zone 2)	**0.0103**	0.0134	0.0165	0.0201	**0.8709**	0.9474
adapFL (zone 3)	0.0112	**0.0123**	0.0169	0.0200	0.8596	**0.9517**
adapFL (zone 4)	0.0114	**0.0123**	0.0159	**0.0187**	0.8571	**0.9517**

Coming back to the predictions obtained in the central area, in order to analyze in a visual way the predictions obtained with FL and with the proposed PFL approach, we present Figure 5.21. This figure shows three images obtained from the central quadrant of resolution 50×50 and their corresponding prediction obtained using the FL approach applied to the four initial areas and the *adapFL* one (adapted to the central area).

In view of Figure 5.21 we can observe how the predictions are very similar to the actual observed image to be predicted. Moreover, if we focus on the first example, we can see how *adapFL* is able to predict more accurately, for example in the zone near (40,40), or in the example 3 in the zone near (20, 30), while in those two regions the FL model does not predict any occurrence of VIL.

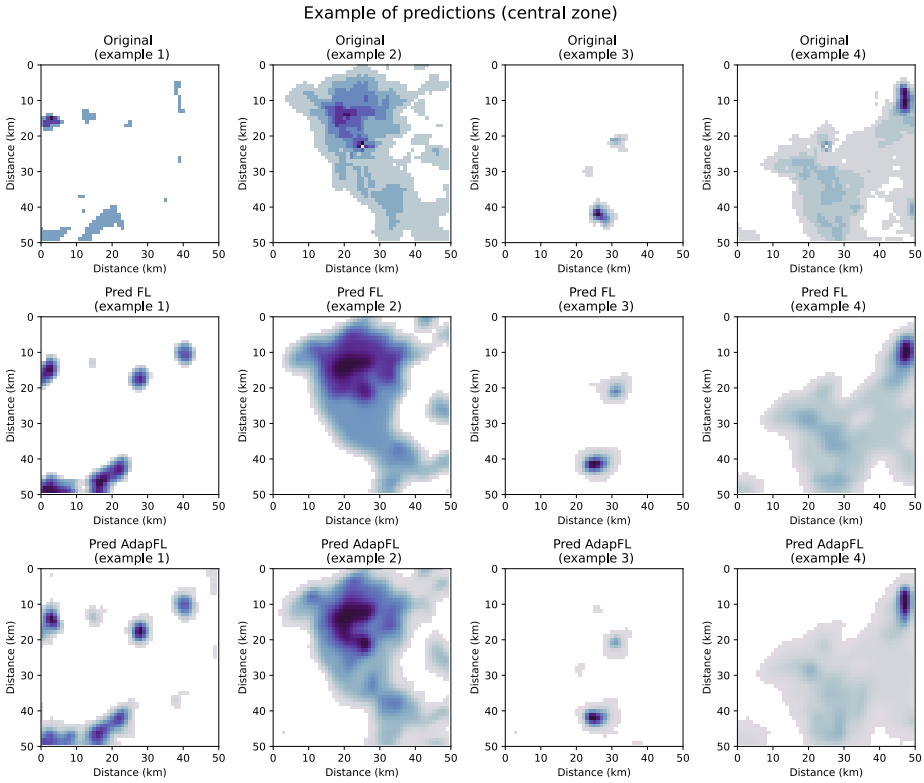


Figure 5.21: Example of predictions of the test set of the central zone. Federated learning approach with $N_r = 10$ and $N_e = 10$ and *adapFL* method with the configuration (9,10).

It is also interesting to analyze the non-independent identically distributed (i.i.d.) nature of each of the five zones proposed. Assuming the privacy-preserving deep learning approach (FL and *adapFL*), we are going to calculated the divergences ($d_{i,j}$) following the Equation 4.9 proposed in Section 4.4 from Chapter4, which was inspired by the equation proposed in [147].

The results obtained in each case for $d_{i,j}$ in the first round are summarized in Table 5.32. In view of the values obtained for $d_{i,central}$ $\forall i \in \{1,2,3,4\}$, the lowest value is obtained with respect to the third zone, which is consistent with this being the *adapFL* model that gives the best results in the central zone in the test set regarding the MSE. However, the best test result is obtained with zone 4 (for MAE and 3 and 4 for the MSE), which is the second with the greatest divergence, and is also the model with the greatest error in the train (of the indexed ones).

This high divergence in training is reflected in the MSE of the train, while allowing a better generalization in the test. Presumably, this is due to the rainfall levels in zone 4 during the period incorporated in the training, which will be similar to that of the central zone test (covering summer periods). It is evident that the divergence between the central zone and the other four areas represents the lowest values (last row of Table 5.32), since each of the 4 initial zones contains a quarter of the information of the central one. The table above provides information on the differences between the different zones, highlighting for example that the smallest divergence regarding the initial four areas is reached between zones 3 and 4, so it may be reasonable to create models agreed between these two areas, separately from the other two, making clients' clusters. In addition, note that zone 1 presents the greater divergence with respect to the other three initial areas, so including this one may be damaging the performance of the overall FL model.

i,j	*1*	*2*	*3*	*4*	*Central*
1	-	0.6563	0.6310	0.6413	0.5729
2	0.6563	-	0.6275	0.6296	0.5850
3	0.6310	0.6275	-	0.6055	0.5677
4	0.6413	0.6296	0.6055	-	0.5743
Central	0.5729	0.5850	0.5677	0.5743	-

Table 5.32: $d_{i,j}$ calculated for each combination of zones.

5.3.8 Conclusions

In this section our main objective was to analyze the feasibility of a novel PFL architecture, in particular the so-called *adapFL* approach, to a use case of meteorological radar images. Specifically, the objective was to use the available images to predict the precipitation expressed as VIL in the next 5 minutes. The complexity of this task, involving the use of images, makes it a good fit for using DL models based on convolutional neural networks.

In particular, to analyze the applicability of the PFL technique to this case, the results obtained in the four artificially distributed zones were compared with the COTREC method, as well as with the training of the same convolutional network on each of the zones individually and the training with a conventional FL architecture. Specifically, for the test set, we obtained improvements in all four cases when applying the *adapFL* architecture both for the MSE and MAE. Moreover, when we

extrapolated these results to the central zone, which has a part in each of the four initial ones, the optimal results for the training set regarding the MSE are also achieved with the application of the *adapFL* architecture trained on all the four initial areas and adapted to the central one. Regarding only the individual models, in this case, the best results are achieved with the individual training on the fourth zone, which is in contrast with the results obtained for the divergence between the different artificial regions. This may be attributed to these regions having more in common in the test set than in the train, e.g. due to seasonality.

The results obtained throughout this study give us a promising idea about the applicability of this PFL architecture to this type of meteorological radar images, since as mentioned, the results improve those of individual training in all the analyzed cases and that of the classic COTREC method. In this sense, future work has been drawn from the very approach of this benchmark study, and it is to extrapolate this type of analysis to radar data distributed in different countries or regions. In that case, the differential behavior of each zone may lead to the introduction of measures considering these divergences, like the creation of different clusters of clients. It should be noted that in this study one of the zones was very different regarding the data used for training from the others on average (see Figure 5.16, zone 3), but the model was not degraded. Instead, this variability in the data distribution in the different clients in some cases may even lead to better generalization ability in the event of unseen data.

5.4 Summary and Wrap Up

In this chapter, the federated learning (FL) architecture presented in detail in Chapter 4 has been put into practice. Its potential has been shown in a wide variety of fields, from medical imaging, where critical privacy restrictions usually apply, water quality prediction and nutrient estimation on distributed data, where technical issues that impose data centralization may occur, and finally climate sciences in a case of radar images, which in many cases come from private companies that would not want to share them with other parties.

Additionally, in the first case (medical imaging), the issue concerning the presence of intermittent clients has been analyzed, but also the incorporation of differential privacy (DP) to the training of the models under a FL architecture. Finally, in this case we have also studied different aggregation strategies and their impact in the prediction of the global model, showing that the approach proposed in Subsection 4.2.5.8 from Chapter 4 and named *FedAvgOpt*, can improve the convergence re-

garding other classical aggregation functions in certain scenarios (as demonstrated in [130] in another case using medical images, specifically brain magnetic resonance images).

In the second use case studied, the performance of the individual learning (IL), centralized learning (CL) and FL paradigms has been compared in a case where the data belong to two fundamentally different ecosystems. Then, in this case we show the advantages that the FL architecture can bring even in the case of non i.i.d. data. Furthermore, different data reduction scenarios have been explored, simulating the case where the data are sampled at different frequencies.

Finally, in the third use case presented (regarding vertical integrated liquid nowcasting), we have tested a novel PFL architecture, named *adapFL*. This has been compared with the classical federated training case and with the case of individual training in each of the artificially distributed zones of data capture (simulating different areas or regions). In this case it is observed that *adapFL* provides a better convergence than the other of the architectures analyzed and than the one of the vertical integrated liquid (VIL) nowcasting state of the art models, the COTREC method. Additionally, the DL models have been tested in a central area of the initial data and the weight divergence between the different distributed areas has also been analyzed following the equation proposed in Section 4.4 from Chapter 4.

All in all, the advantages of the application of the FL architecture in different use cases have been shown to be particularly useful to improve the predictions made with classical machine learning approaches (usually individual or centralized). Furthermore, this technique allows to take into account the privacy of the data involved by not making them leave the centers, institutions or devices that have generated them even for training purposes.

Part IV: Conclusions

CHAPTER 6

Conclusions and Future Work

Eran tres.
...
Eran dos.
...
Era uno.
...
Era ninguno.

Federico García Lorca,
Cortaron tres árboles,
Canciones

Contents

6.1 Main Contributions 255
6.2 Publications and Other Achievements 257
6.2.1 Scientific Journals 257
6.2.2 Peer Reviewed Conference Papers 259
6.2.3 International Stay as Visiting Researcher at INRIA 259
6.2.4 Invited Talks 260
6.2.5 Scientific Posters 261
6.2.6 Software 262
6.3 Future Work and Perspective 262

Abstract

This last chapter presents the contributions made in this doctoral thesis divided into two parts: those concerning privacy aspects related to data processing and curation, and those related to privacy preserving machine learning and the development of machine and deep learning models with privacy restrictions and using privacy enhancing technologies. In order to detail the contributions derived from the pre-doctoral stage in the form of scientific papers, the publications carried out within the framework of this thesis are presented, both in terms of publications in high impact scientific journals and peer reviewed international conferences. Also, the contributions carried out during the period of international stay are discussed, as well as the invited talks at national, international and online conferences and the posters presented. Next, the main open source software developed is presented. Finally, some lines for future work in the field and perspectives are presented.

6.1 Main Contributions

This thesis has as a conductive thread the study, analysis and implementation of privacy techniques in a context of data science, artificial intelligence, machine and deep learning (AI/ML/DL), followed by practical implementations in real applications.

In this sense, several lines of action are distinguished in this document in the different chapters, depending on whether the focus is on the processing of the data itself or on the algorithms and models developed and served. The aim is to cover the whole machine learning lifecycle with special focus on privacy and security issues, from data processing to the analysis, model development and deployment (even exploring the advantages that some privacy preserving architectures can also provide in terms of model performance). Then, the contributions made in this thesis, distinguished by area, are summarized in key items as follows:

- Regarding privacy aspects in a data processing context:
  - I have developed and implemented an open source Python library for checking the level of anonymity of a dataset: *pyCANON*.
  - I have developed and implemented an open source Python library for anonymizing sensitive tabular data: *anjana*.

  - I have studied and reviewed different pseudonymization techniques and best practices of use.
  - I have analyzed the state of the art concerning differential privacy, including local and global DP, but also Rényi DP and *d-privacy* or *metric-DP*. I have introduced the development of differentially private DL models and the use of DP for data publishing and analysis.

  Part of the work presented above is framed within the context of the Horizon Europe project EOSC SIESTA, which seeks to provide users with a cloud platform composed of a set of tools, services and methodologies for the effective exchange of sensitive data in the EOSC. Thus, a prototype integration of these tools (anonymization, pseudonymization and differential privacy) has been developed in the context of such project together with their application to different use cases.

- Regarding privacy aspects in the development of data-based AI/ML/DL models:

  - I have implemented a complete FL architecture, following the concepts introduced in Chapter 4, as part of the Horizon Europe project AI4EOSC, offering this functionality within the cloud platform developed by the project. Additionally, a secret management system that allows the participation of only authenticated clients with a token has been integrated. More information about this integration is given in Appendix C.
  - I have integrated different functionalities for the training of FL architectures in AI4EOSC, including support for metric differential privacy and carbon footprint monitoring. Concerning the former, a proposal for integrating metric privacy into FL architectures is introduced with the additional goal of preventing client inference attacks (in addition to the privacy preservation derived from the inclusion of classical differential privacy). See Appendix C for more details.
  - I have carried out a detailed analysis of different FL application use cases: medical imaging, water quality and weather nowcasting.
  - I have studied the impact of including DP in a FL scheme applied to a medical imaging use case.
  - I have designed a new aggregation method (*FedAvgOpt*) that can improve the convergence of the classic aggregation strategies used in FL in the use cases analyzed (see Section 4.2.5.8). In this line I have compared

the performance of different aggregation strategies in two a medical imaging use cases (one presented in Chapter 5 and one in Section 6.2).

- I have evaluated the impact of the application of a FL architecture instead of simple individual training in a use case concerning water quality monitoring in the case of two heterogeneous clients. I have analyzed the impact in the model performance in different data reduction scenarios and the significance of each feature according to the applied training architecture (including FL).
- I have proposed a novel PFL architecture (*adapFL*) based on the idea of transfer learning applied to a weather nowcasting based on a radar images use case. The idea here was to present how the PFL architecture can improve predictions in different areas of the radar with respect to the centralized training.

6.2 Publications and Other Achievements

This section highlights different achievements accomplished during the course of the doctoral thesis related to publications in scientific journals, international conferences, invited lectures and talks, scientific posters presented and software development.

6.2.1 Scientific Journals

The following are articles which I have published as main author in high impact journals that are directly related to the scope of this thesis:

2025 Nguyen, G.*, **Sáinz-Pardo Díaz, J.***, Calatrava, A., Berberi, L., Lytvyn, O., Kozlov, V., Tran, V., Moltó, G., & López García, Á. (2025). Landscape of machine learning evolution: privacy-preserving federated learning frameworks and tools. *Artificial Intelligence Review*, 58 (2), 51. https://doi.org/10.1007/s10462-024-11036-2.
*These authors contributed equally to this work.

2024 **Sáinz-Pardo Díaz, J.** & López García, Á. (2024). An Open Source Python Library for Anonymizing Sensitive Data. *Scientific Data*, 1289. https://doi.org/10.1038/s41597-024-04019-z.

2024 **Sáinz-Pardo Díaz, J.**, Castrillo, M., Bartok, J., Heredia Cachá, I., Malkin Ondík, I., Martynovskyi, I., Alibabaei, K., Berberi, L., Kozlov, V. & López Gar-

cía, Á. (2024). Personalized Federated Learning for improving radar based precipitation nowcasting on heterogeneous areas. *Earth Science Informatics*, 17, 5561–5584.
https://doi.org/10.1007/s12145-024-01438-9.

2023 **Sáinz-Pardo Díaz, J.**, Castrillo, M., & López García, Á. (2023). Deep learning based soft-sensor for continuous chlorophyll estimation on decentralized data. *Water Research*, 120726.
https://doi.org/10.1016/j.watres.2023.120726.

2023 **Sáinz-Pardo Díaz, J.** & López García, Á. (2023). Study of the performance and scalability of federated learning for medical imaging with intermittent clients. *Neurocomputing*, 518, 142-154.
https://doi.org/10.1016/j.neucom.2022.11.011.

2022 **Sáinz-Pardo Díaz, J.** & López García, Á. (2022). A Python library to check the level of anonymity of a dataset. *Scientific Data*, 9(1), 785.
https://doi.org/10.1038/s41597-022-01894-2.

Other articles to which I have contributed substantially and that are related to machine learning lifecycle and data analysis and processing are as follows:

2025 Berberi, L., Kozlov, V., Nguyen, G., **Sáinz-Pardo Díaz, J.**, Calatrava, A., Moltó, G., Tran, V. & López García, Á. (2025). Machine learning operations landscape: platforms and tools. *Artificial Intelligence Review* 58, 167.
https://doi.org/10.1007/s10462-025-11164-3.

2023 Heredia Cacha, I., **Sáinz-Pardo Díaz, J.**, Castrillo, M., & López García, Á. (2023). Forecasting COVID-19 spreading through an ensemble of classical and machine learning models: Spain's case study. *Scientific Reports* 13, 6750.
https://doi.org/10.1038/s41598-023-33795-8.

Preprints (under review):

2025 **Sáinz-Pardo Díaz, J.** & López García, Á. (2025). Enhancing the Convergence of Federated Learning Aggregation Strategies with Limited Data. arXiv preprint arXiv:2501.15949.
https://arxiv.org/abs/2502.01352.
Accepted for presentation and publication at the proceedings of the 3rd IEEE International Conference on Federated Learning Technologies and Applications (FLTA25).

2025 **Sáinz-Pardo Díaz, J.**, Athanasiou, A., Jung, K., Palamidessi, C., & López García, Á. (2025). Metric Privacy in Federated Learning for Medical Imaging: Improving Convergence and Preventing Client Inference Attacks. arXiv preprint arXiv:2502.01352. https://arxiv.org/abs/2501.15949.

6.2.2 Peer Reviewed Conference Papers

With respect to the presentation and publication of papers at international high impact peer reviewed conferences, the following works have been presented:

2024 **Sáinz-Pardo Díaz, J.**, Heredia Canales, A. Heredia Cachá, I., Tran, V., Nguyen, G., Alibabaei, K., Obregón Ruiz, M., Rebolledo Ruiz, S., López García, Á. (2024). Making Federated Learning Accessible to Scientists: The AI4EOSC Approach. *In Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '24)*. Association for Computing Machinery, New York, NY, USA, 253–264. https://doi.org/10.1145/3658664.3659642.

2023 **Sáinz-Pardo Díaz, J.** & López García, Á. (2023). Comparison of machine learning models applied on anonymized data with different techniques. *IEEE International Conference on Cyber Security and Resilience (CSR), Venice, Italy, 2023,* pp. 618-623. https://doi.org/10.1109/CSR57506.2023.10224917.

6.2.3 International Stay as Visiting Researcher at INRIA

Visiting researcher at INRIA Paris-Saclay center. This 3-month stay has been supervised by the Director of Research Catuscia Palamidessi, PhD. During this period I joined the group Cométe (Privacy, Fairness and Robustness in Information Management) and we collaborated in privacy aspects related with the incorporation of differential privacy in federated learning architectures. We also analyzed the concepts of metric-privacy and Rényi-DP, and the implications of including global-DP and metric-privacy from the server side with fixed clipping by varying the aggregation function. Furthermore, we studied the protection of this methods against client inference attacks in federated learning. This collaboration led to a preprint that is currently under review [154].
Dates: September 6, 2024 to December 5, 2024 (3 months).
Location: INRIA Saclay. Bâtiment Alan Turing, Campus de l'École Polytechnique. Palaiseau, Île-de-France, France.

6.2.4 Invited Talks

The following are invited talks that I have given at national and international workshops, seminars or meetings:

2025 **Sáinz-Pardo Díaz, J.** (2025). Event: NeuroAI, from neurons and connectomes to engrams (in Spanish: NeuroAI, desde las neuronas y conectomas a los engramas). Summer courses International University Menéndez Pelayo (UIMP). Talk: *Federated Learning*. Santander, Spain. 25-29 August, 2025.

2025 **Sáinz-Pardo Díaz, J.** (2025). Event: Accelerating Research with AI4EOSC: Real Use Cases Exploiting the Platform. Talk: *Introducing the AI4EOSC platform*. Online, May 9, 2025.

Agenda of the event: https://indico.ifca.es/event/3441/.

2025 **Sáinz-Pardo Díaz, J.** (2025). Event: Exploring AI4EOSC: AI and LLMs from Theory to Practice. Talk: *Why AI4EOSC? Take advantage of the platform*. Online, 7 March, 2025.

Agenda of the event: https://indico.ifca.es/event/3390/.

2025 **Sáinz-Pardo Díaz, J.** (2025). Event: EOSC SIESTA All Hands Meeting. Talk: *Data privacy tools in EOSC SIESTA, anjana, pyCANON and differential privacy*. Instituto de Física de Cantabria (IFCA), Santander, Spain, 4-5 February, 2025.

Agenda of the event: https://indico.ifca.es/event/3384/.

2025 **Sáinz-Pardo Díaz, J.**, Aguilar Gómez , F. & Lloret Iglesias, L. (2025). Event: Artificial Intelligence and Software Engineering Winter School. Workshop: *the eyes of AI. Topic: Federated learning applications*. University of Doha for Science and Technology (UDST), Doha, Qatar, 19-22 January, 2025.

Webpage of the event: https://www.udst.edu.qa/AiSeschool.

2024 **Sáinz-Pardo Díaz, J.** (2024). Event: Joint Workshop PTI Digital Science - PTI Green Horizon (in Spanish: Workshop conjunto de la PTI Ciencia Digital y la PTI Horizonte verde). Talk: *AI and development of AI-based solutions*. Online, 17 September, 2024.

Webpage of the event: https://pti-cienciadigital.csic.es/evento/ jornada-pti-horizonteverde-cienciadigital/.

2024 **Sáinz-Pardo Díaz, J.** (2024). Event: Data in research: challenges and opportunities (in Spanish: Los datos en investigación: retos y oportunidades).

Summer courses International University Menéndez Pelayo (UIMP). Talk: *Distributed Learning: advantages and risks*. Practical session: *Federated Learning demonstration*. Santander, Spain. 26-28 August, 2024.

Programme of the event: https://wapps001.uimp.es/uxxiconsultas/ficheros/5/6842965SI.pdf.

2024 **Sáinz-Pardo Díaz, J.** & Heredia Cachá, I. (2024). Event: Zero-code tools & BMZ Community partners workshop. Talk: *Advanced AI for scientists: the AI4EOSC platform approach*. Madrid, Spain. 10-11 June, 2024.

Webpage of the event: https://ai4life.eurobioimaging.eu/event/workshop-hackathon-uploathon-madrid/.

2024 **Sáinz-Pardo Díaz, J.** (2024). Event: AI4EOSC webinars (3): Introduction to Federated Learning. Talk: *Demo: Federated Learning in AI4EOSC*. Online, 22 April, 2024.

Agenda of the event: https://indico.ifca.es/event/3134/

2024 **Sáinz-Pardo Díaz, J.** (2024). Event: Flower AI Summit 2024. Talk: *Federating AI in the European Open Science Cloud*. London, United Kingdom. 14-15 March 2024.

Webpage of the event: https://flower.ai/conf/flower-ai-summit-2024/

2024 López García, Á. & **Sáinz-Pardo Díaz, J.** (2024). Event: Flower monthly (January 2024). Talk: *Federated Learning in the European Open Science Cloud*. Online, 3rd January, 2024.

Online talk: https://www.youtube.com/watch?v=4OC2y0kCvPY.

2023 **Sáinz-Pardo Díaz, J.** (2023). Event: AI4EOSC user's workshop. Talk: *Federated Learning with Flower*. Institute of Informatics of the Slovak Academy of Sciences (IISAS), Bratislava, Slovakia. 15-16 November 2023.

Agenda of the event: https://indico.scc.kit.edu/event/3845.

6.2.5 Scientific Posters

Regarding the presentation of scientific posters in congresses and conferences, the list of contributions is detailed below:

2023 **Sáinz-Pardo Díaz, J.** & López García, Á. (2023). Application of federated learning to medical imaging scenarios. AIHUB CSIC Summer School. CaixaForum Macaya, Barcelona, Spain. 4-8 July, 2023. [7].

2023 **Sáinz-Pardo Díaz, J.** & López García, Á. (2023). Privacy preserving techniques for data science. X Doctoral Symposium and V Dissemination Symposium of the G9 Group of Universities (in Spanish: X Jornadas Doctorales y V Jornadas de Divulgación Grupo de Universidades del G9). Universidad de Oviedo, Oviedo, Spain. 31 May - 2nd June, 2023. [8].

6.2.6 Software

As for the software developed within the framework of the present doctoral thesis, the following contributions should be highlighted (curated list including only relevant Python libraries and software products and extensions developed):

- **Sáinz-Pardo Díaz, J.**, & López García, Á. *anjana*.
  DOI: https://zenodo.org/records/11186382.
- **Sáinz-Pardo Díaz, J.**, & López García, Á. *pyCANON*.
  DOI: https://doi.org/10.20350/digitalCSIC/15280.
- **Sáinz-Pardo Díaz, J.**, Heredia Cachá, I., & Kozlov, V. *AI4EOSC federated server*.
  GitHub repository: https://github.com/ai4os/ai4os-federated-server.
- **Sáinz-Pardo Díaz, J.**, Heredia Cachá, I., Tran, V. & López García, Á. *AI4EOSC extensions to the Flower library*.
  GitHub repository: https://github.com/ai4os/ai4-flwr.
- **Sáinz-Pardo Díaz, J.** & López García, Á. *AI4EOSC Flower (maintained fork of the original Flower library)*.
  GitHub repository: https://github.com/AI4EOSC/flower.

6.3 Future Work and Perspective

The field of data science, especially enhanced by the development of solutions based on artificial intelligence models, is a constantly growing field with applications in multiple sectors. This, together with the increasing generation of data and information, often of a sensitive nature because it is associated with identified or identifiable persons, or because it contains confidential data of entities or institutions, highlights the importance of preserving privacy in these environments. Ensuring the protection of information throughout the data lifecycle, from its acquisition to the implementation, deployment and production of models, is becoming increasingly essential. The future of artificial intelligence and data science is also closely

tied to alignment with the regulatory framework. It is essential that technological progress and the development of advanced solutions go hand in hand with legislation and emerging ethical issues.

In addition, the availability of more powerful and affordable processors fosters on-device processing, which drives the development of AI models in a distributed way. The application of federated learning (FL) architectures and other decentralized methods without dependence on a central server becomes crucial, expanding the scope of possible applications concerning IoT and edge devices. In addition, complementary measures such as differential privacy (DP), metric differential privacy, homomorphic encryption (HE) and its multi-key variant (MKHE), whose applicability will depend on suitable computational conditions, should be explored in practical scenarios. From a more general perspective, privacy restrictions in data processing and analysis in such devices are becoming increasingly relevant, ensuring security in distributed systems that process data locally rather than in centralized servers.

Regarding data processing and analysis, future work involves further developing and updating the two libraries developed and presented within the scope of this thesis, *pyCANON* and *anjana*, incorporating additional functionalities to evaluate both the usefulness of the data and its resistance to different types of attacks. It is also important to continue exploring the scalability and impact of differential privacy in different scenarios, its integration with FL and its combined use with other privacy preserving technologies (PETs) such as HE. Moreover, privacy in data exchange between organizations requires the inclusion of PETs that allow information sharing without compromising confidentiality, also including advanced techniques based on secure multi party computation (SMPC).

In terms of data acquisition, the generation of synthetic data is positioned as a key solution for training models without the risks associated with data protection regulations, while ensuring adequate statistical representativeness. It is also key to further investigate different types of privacy attacks, such as inference, membership and model inversion attacks, and to develop mitigation strategies within distributed architectures such as FL or other decentralized serverless solutions.

Along the same lines, future work (already in progress) on FL is extensive. In addition to PETs and attack prevention, future research focuses on developing aggregation functions for dealing with heterogeneous data, designing measures for drift detection, monitoring and prevention, and optimizing cryptographic tools such as HE to reduce computational costs while maintaining integrity. All this development is also relevant for serverless decentralized architectures, where other practical im-

plementation issues take center stage.

Privacy in generative AI environments and large-scale language models (LLMs) is another emerging challenge. Training these models requires enormous computational resources, and federated learning arises as a suitable alternative to optimize their efficiency. The development of this type of models is a key area of scientific research concerning artificial intelligence and enhancing its development with distributed training (and also taking into account privacy issues) will be key for taking advantage of federated resources and for improving access to research information, for example through the training and integration of specialized agents.

Another key aspect is the development of efficient algorithms for secure data exchange, as well as research in trusted and secure execution environments (TEEs and SEEs). In the same line, developing cryptographic methods resistant to the challenges arising from quantum computing is a current challenge that cannot be lost sight of when working in data science environments where sensitive data is handled.

The tools and methodologies presented throughout this thesis must be available (and indeed they are) both to the general public and, in particular, to the research community, as every day, researchers work with increasing volumes of data. The scientific method ranges from systematic observation and data collection to experimentation, analysis, hypothesis formulation and subsequent revision or modification. Since data are an essential component of the scientific process, it is essential to consider the challenges that arise when dealing with sensitive information, either because of its nature or because it is protected in the context of research. In these cases, privacy issues take on a central role, and the solutions proposed in this thesis and concerning the future work are quite useful. This becomes even more relevant when seeking to exploit the potential of data through its analysis, as well as in the development and deployment of artificial intelligence models. Therefore, it is important that all the tools presented in this thesis are accessible to the research community, so that they can take advantage of their potential and carry out their research with a focus on the privacy and security of the information handled.

Part V: Appendix

APPENDIX A

BASIC ALGEBRA FOR CRYPTOGRAPHY

This aim of this appendix is to highlight basic notions of algebra that are essential to understand and deepen some of the cryptographic aspects presented when introducing the pseudonymization methods (Chapter 2), and the concept of Homomorphic Encryption (HE) and Multi-Key Homomorphic Encryption (MKHE) (Chapter 4).

A.1 Algebraic Structures

Let us start introducing the basic algebraic structures: *semigroup*, *monoid*, *group*, *ring* and *ideal*.

Definition A.1.1. Monoid, semigroup. A **monoid** is a semigroup for which the existence of a neutral element is verified. A **semigroup** is a pair $(S, \star)$ with $S \neq \emptyset$ and $\star$ an application from $S \times S$ to S, with $S \times S = \{(x, y) : x, y \in S\}$ the cartesian product of S with itself, such that given the pair $(a, b) \in S \times S$, the result will be $a \star b$. In addition, the associative property is verified for $\star$. The existence of a neutral element implies that the semigroup $(S, \star)$ is a monoid, and is given by the assumption that $\exists e \in S$ such as $e \star x = x \star e = x$, $\forall x \in S$.

Definition A.1.2. Group. A **group** is a monoid given by the pair $(G, \star)$ which verifies the existence of a inverse element. This means that $\forall x \in G$, $\exists \tilde{x} \in G$ such

as $x \star \tilde{x} = \tilde{x} \star x = e$, with e the neutral element of the monoid.

If in addition it is verified that the monoid is commutative, then the group is abelian.

Definition A.1.3. Ring. A **ring** is a tern $(R, +, \cdot)$ where:

- $(R, +)$ is an abelian group.
- $(R, \cdot)$ is a commutative monoid.
- The distributive property is verified:

$$x \cdot (y + z) = x \cdot y + x \cdot z, \quad \forall x, y, z \in R.$$

Definition A.1.4. Ideal. An **ideal** I of a ring $(R, +, \cdot)$ is a subset of R such as $(I, +)$ is a subgroup of $(R, +)$ and $x \cdot y, y \cdot x \in I$, $\forall x \in I$ and $\forall y \in R$. An ideal I of a ring R is **principal** if $\exists x \in I$ such as $I = (x)$.

Definition A.1.5. Integral domain. Be $(R, +, \cdot)$ a ring and A all the elements $a \in R$ such as $\exists a' \in R$ with $aa' = a'a = 1_R$. Then $x \in R$ is a divisor of zero if $\exists y \in A$, $y \neq 0$ such as $x \cdot y = 0$. A ring without non-zero divisors is known as **integral domain** (or domain).

Now, let us introduce the notions of quotient ring and polynomial ring:

Definition A.1.6. Quotient ring. Be I an ideal I of the ring $(R, +, \cdot)$. The **quotient ring** noted as R/I is the set of cosets given by:

$$R/I = \{r + I | r \in R\},$$

verifying:

- $(r + I) + (s + I) = (r + s) + I \ \forall r, s \in R$.
- $(r + I) \cdot (s + I) = (r \cdot s) + I \ \forall r, s \in R$.

Definition A.1.7. Polynomial ring. Be $(R, +, \cdot)$ a ring. The **polynomial ring** over R, denoted as $R[x]$ is the ring of all the polynomials with coefficients in R. Formally:

$$R[x] = \{a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} + a_n x^n | n \geq 0 \wedge a_i \in R, \forall i \{1, \ldots, n\}\}.$$

Finally, let us introduce the definition of cyclotomic polynomial, which will be used during the brief introduction to ring learning with errors (RLWE).

Definition A.1.8. Cyclotomic polynomial. For $n \in \mathbb{N}$ the **n-th-cyclotomic polynomial** is defined as:

$$\phi_n(x) = \prod_{o(w)=n} (x - \omega),$$

with $o(\omega) = min\{n \geq 1 | \omega^n = 1\}$ (i.e. the order of $\omega \in \mathbb{C}^*$).

Equivalently, it is the unit polynomial whose roots are all the n-th primitive roots of the unit (i.e. $e^{\frac{2\pi ik}{n}}$ with $gcd(k,n) = 1$).

A.2 Homomorphisms and Isomorphisms

Definition A.2.1. Morphism. A **morphism (or homomorphism) of groups** between the two groups $(G_1, \star)$ and $(G_2, \diamond)$ is a function $\psi : G_1 \to G_2$ verifying $\psi(e_{G_1}) = e_{G_2}$ with e_{G_i} the neutral element of the group with the set G_i $\forall i \in \{1,2\}$, and $\forall x, y \in G_1, \psi(x \star y) = \psi(x) \diamond \psi(y)$.

Definition A.2.2. Ring homomorphism. Be $(R_1, +, \cdot)$ and $(R_2, +, \cdot)$ two rings, an application $\psi : R_1 \to R_2$ is a **ring homomorphism** if it is a group homomorphism for the abelian groups $(R_1, +)$ and $(R_2, +)$ and it is verified that:

- $\psi(r \cdot s) = \psi(r) \cdot \psi(s)$ $\forall r, s \in R_1$.
- $\psi(r + s) = \psi(r) + \psi(s)$ $\forall r, s \in R_1$.
- Be 1_{R_1} and 1_{R_2} the unity elements of R_1 and R_2 respectively, then: $\psi(1_{R_1}) = 1_{R_2}$.

If the function ψ is inyective, it is a ring *monomorphism*, if ψ is suprayective it is a ring *epimorphism*, and if ψ is biyective it is a ring *isomorphism*.

A.3 Introduction to Cryptosystems

Let us start this section by introducing the concepts of cryptosystem and cryptoanalysis.

Definition A.3.1. Cryptosystem. A **cryptosystem** [230] is defined by the tuple $(\mathscr{P}, \mathscr{C}, \mathscr{K}, \mathscr{E}, \mathscr{D})$ verifying:

- $\mathscr{P}$ contains all the possible *plaintexts*.
- $\mathscr{C}$ contains all the possible *ciphertexts*.

- $\mathscr{K}$ contains all the possible *keys*.
- $\forall k \in \mathscr{K}$, $\exists e_k \in \mathscr{E}$, $e_k : \mathscr{P} \longrightarrow \mathscr{C}$, an encryption rule and a corresponding decryption rule $d_k \in \mathscr{D}$, $d_k : \mathscr{C} \longrightarrow \mathscr{P}$, such as $d_k(e_k(x)) = x \ \forall x \in \mathscr{P}$.

Definition A.3.2. Cryptoanalysis. We can define **cryptoanalysis** as the study of different methods and techniques which can be used for breaking cryptosystems. In particular, modern cryptography employs many techniques from computational mathematics, as well as from algebra and number theory.

As already introduced in Chapter 2, cryptographic systems are typically based on 3 types of algorithms: symmetric key, asymmetric key and hash functions. While hash functions where already introduced in Chapter 2, in the following the present the schemes of symmetric and asymmetric key.

- **Private or symmetric key**, where a single key is used. In this approach, a private key is used to encrypt the information and this same key is used to decrypt it. The main disadvantage concerns the difficulty of sharing keys (since the key has to be known by both parties in the communication). Thus, the major problem with symmetric key schemes is that before transmitting a ciphertext between two agents, it is necessary to securely communicate the key k between them, which in practice is difficult to achieve. However, it is a very efficient scheme that requires small key size.
- **Public or asymmetric key**, where a pair of keys is used, one public key and one private key. The idea behind asymmetric key schemes, consisting of a private key and a public key, is to find a cryptosystem in which it is computationally infeasible to determine d_k given e_k, with e_k being the public key (accessible to anyone in order to determine that a person is who he/she claims to be), and d_k the private key, known only to the owner of that key pair (as defined in Definition A.3.1). Thus, if Bob is the owner of the key pair e_k and d_k, he can encrypt information using d_k, and decrypt it using e_k. Other party, e.g. Alice, will only know about Bob's e_k, but not his d_k. Regarding signatures schemes, Bob can use his private key d_k to sign a message and Alice can use his public key e_k in order to verify that it was Bob who signed it.

  The main disadvantage asymmetric key scheme is that it is not efficient, being the encryption and decryption much slower than in the case of symmetric key, also requiring a larger key size. The major advantage is that the public keys can be distributed publicly and only the private key must be remain protected.

A.4 Essential (and Beautiful) Results from Algebra for Cryptography

The following definitions and theorems are essential in cryptographic algebra. For example, they are fundamental for defining and applying the RSA cryptographic system enunciated in Chapter 2.

Definition A.4.1. Euler's function. The **Euler's function** ϕ is a function defined as $\phi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ verifying:

$$\phi(n) = |\{m \in \mathbb{N} | m \leq n \wedge \gcd(m,n) = 1\}|.$$

Proposition A.4.1. *The Euler's function is the cardinal of the multiplicative group $\mathbb{Z}/n\mathbb{Z}$ (also noted as $\mathbb{Z}_n$).*

Theorem A.4.1. ***Fermat's Little Theorem.*** *Be p a prime number. Then $\forall a \in \mathbb{Z}$ it is verified $a^p \equiv a \pmod p$.*

Theorem A.4.2. ***Euclidean Algorithm.*** *Be (R, ϕ) an euclidean domain and be $a, b \in R$ such as $\phi(a) \geq \phi(b)$. Let us define the following succession of elements from R.*

- *$r_0 = a$ and $r_1 = b$.*
- *$\forall i > 1$, $r_i = r_{i-2} - q_{i-1} r_{i-1}$ if $r_{i-1} \neq 0$, with $\phi(r_i) > \phi(r_{i-1}) - 1$.*

Then the following is fulfilled:

- *The succession given by $\{r_i | i \in \mathbb{N}\}$ is finite (i.e. $\exists k \in \mathbb{N}$) such as $r_k = 0$.*
- *Be $j \in \mathbb{N}$ the maximum value with $r_j \neq 0$. Then $r_j = gcd(a, b)$.*

Finally, as many HE schemes rely on polynomial rings over modular integers, let us recall the Chinese Remainder Theorem:

Theorem A.4.3. ***Chinese remainder theorem.*** *If $I_1, \ldots, I_n$ are ideals of a commutative ring R such as $I_i + I_j = R \; \forall i \neq j$, the the following application ψ is an isomorphism:*

$$\psi : R \longrightarrow \prod_{i=1}^{n} R/I_i,$$

with $\psi(r) = (r + I_1, r + I_2, \ldots, r + I_n) \; \forall r \in R$.

A.5 Ring Learning With Errors

The idea behind the ring learning with errors approach (RLWE) is to introduce structured noise into computations over rings. Specifically, the following quotient ring is usually considered when working with ring-with-errors schemes: $R_q = \mathbb{Z}_q[x]/(p(x))$, with $p(x)$ a cyclotomic polynomial. For example, $R_q = \mathbb{Z}_q[x]/(1-x^n)$ can be considered (with q prime).

Then, given a pair $(a(x), b(x))$ with $b(x) = a(x)\cdot s(x)+e(x)$ with $a(x)$ a random polynomial uniformly extracted from R_q and $e(x)$ the error distribution (e.g. from a Gaussian distribution), it is difficult to distinguish from other pair $(a(x), c(x))$, with $c(x)$ a uniformly random extraction from R_q. Given this assumption, the security of RLWE is stated. Indeed, the search RLWE problem refers to recovering the secret polynomial $s(x)$ given multiple RLWE samples $(a_i(x), b_i(x)$. Assume that we have m samples, $s(x)$ can be calculated solving the following equations system:

$$
\begin{aligned}
b_1(x) &= a_1(x)\cdot s(x) + e_1(x) \\
b_2(x) &= a_2(x)\cdot s(x) + e_2(x) \\
&\cdots\cdots \\
b_m(x) &= a_m(x)\cdot s(x) + e_m(x)
\end{aligned}
$$

Note that the ring structure of RLWE enables homomorphic cryptography with a ring homomorphism supporting both addition and multiplication of ciphertexts [231].

APPENDIX B

FEDAVGOPT IMPLEMENTATION

In this Appendix we show how to integrate the aggregation strategy proposed in Section 4.2.5.8 and in [130], known as *FedAvgOpt*, within the Python Library for FL *flower*.

First, we define the class `FedAvgOpt`, which inherits from the strategy `FedAvg`. In order to properly apply the *FedAvgOpt* strategy, we also define the function that will be minimized ($g(x)$ in Section 4.2.5.8) for calculating the value of α as given in problem (P_1) from Section 4.2.5.8, using the Nelder-Mead optimization method. This class is presented in Code B.1.

```
1  import flwr as fl
2  from typing import Dict, List, Optional, Tuple, Union
3  from flwr.common import (
4      FitRes,
5      NDArrays,
6      Parameters,
7      Scalar,
8      ndarrays_to_parameters,
9      parameters_to_ndarrays,
10 )
11 from flwr.server.client_proxy import ClientProxy
12 from flwr.common.logger import log
13 from logging import WARNING
14 from functools import reduce
15 from scipy.optimize import minimize
16 from numpy.linalg import norm
17 import numpy as np
18
19
20 class FedAvgOpt(fl.server.strategy.FedAvg):
21
22     def __repr__(self) -> str:
```

```
23          """Compute a string representation of the strategy."""
24          rep = f"FedAvgOpt(accept_failures={self.accept_failures})"
25          return rep
26
27      def aggregate_fit(
28          self,
29          server_round: int,
30          results: List[Tuple[ClientProxy, FitRes]],
31          failures: List[Union[
32              Tuple[ClientProxy, FitRes], BaseException
33          ]],
34      ) -> Tuple[Optional[Parameters], Dict[str, Scalar]]:
35
36          fedavg_parameters_aggregated, metrics_aggregated = (
37              super().aggregate_fit(
38                  server_round=server_round,
39                  results=results,
40                  failures=failures
41              ))
42          if fedavg_parameters_aggregated is None:
43              return None, {}
44
45          weights_results = [
46              (
47                  parameters_to_ndarrays(fit_res.parameters),
48                  fit_res.num_examples
49              )
50              for _, fit_res in results
51          ]
52
53          num_clients = len([num_examples
54                             for (_, num_examples) in results])
55          x0 = [1] * num_clients
56          value = minimize(
57              min_aggregation, x0,
58              method='Nelder-Mead', args=weights_results
59          )
60          alpha = np.array(value['x'])
61
62          # FedAvgOpt strategy
63          num_examples_total = sum(
64              num_examples
65              for (_, num_examples) in weights_results
66          )
67
68          weighted_weights = [
69              [layer * num_examples * alpha_i for layer in weights]
70              for (weights, num_examples), alpha_i
71              in zip(weights_results, alpha)
72          ]
73
74          # Compute average weights of each layer
75          weights_aggregated: NDArrays = [
76              reduce(np.add, layer_updates) / num_examples_total
77              for layer_updates in zip(*weighted_weights)
78          ]
```

```
79
80          parameters_aggregated = ndarrays_to_parameters(
81              weights_aggregated
82          )
83
84          metrics_aggregated = {}
85          if self.fit_metrics_aggregation_fn:
86              fit_metrics = [
87                  (res.num_examples, res.metrics) for _, res in results
88              ]
89              metrics_aggregated = self.fit_metrics_aggregation_fn(
90                  fit_metrics
91              )
92          elif server_round == 1:  # Only log this warning once
93              log(WARNING, "No fit_metrics_aggregation_fn provided")
94
95          return parameters_aggregated, metrics_aggregated
96
97
98  def min_aggregation(alpha, weights_results):
99      weighted_weights = [
100         [layer * num_examples * alpha_i for layer in weights]
101         for (weights, num_examples), alpha_i
102         in zip(weights_results, alpha)
103     ]
104
105     num_examples_total = sum(
106         num_examples for (_, num_examples) in weights_results
107     )
108
109     wg: NDArrays = [
110         reduce(np.add, layer_updates) / num_examples_total
111         for layer_updates in zip(*weighted_weights)
112     ]
113
114     agg_weights = 0
115     for wj, _ in weights_results:
116         agg_weights += norm(
117             [norm(a - b) / norm(a + b)
118              for a, b in zip(wg, wj)]
119         )
120
121     return agg_weights
```

Example Code B.1: Defining the `FedAvgOpt` class in flower. This strategy inherits from the class `FedAvg`.

Thus, to use this strategy with the *Flower* library, we simply import it and pass it to the server as follows, introducing the function to find the aggregate metric (if any) and the minimum number of clients, as detailed in the following code:

```
1  # Create the strategy. In this example we consider 3 clients as the
2  # minimum, and we introduce and function which should define the
3  # aggregation metric to be used
```

```
strategy = FedAvgOpt(
        min_available_clients=3,
        min_fit_clients=3,
        evaluate_metrics_aggregation_fn=aggregation_metric,
    )

# Define the Flower server
# (runs in port 5000 during 10 rounds):
fl.server.start_server(
    server_address="0.0.0.0:5000",
    config=fl.server.ServerConfig(num_rounds=10),
    strategy=strategy,
)
```

Example Code B.2: Starting the Flower FL server using the aggregation strategy *FedAvgOpt*.

APPENDIX C

FEDERATED LEARNING IN AI4EOSC

Part of this appendix was presented in the 12th ACM Workshop on Information Hiding and Multimedia Security (2024) and published in the conference proceedings as follows: ***Sáinz-Pardo Díaz, J.****, Heredia Canales, A., Heredia Cachá, I., Tran, V., Nguyen, G., Alibabaei, K., Obregón Ruiz, M., Rebolledo Ruiz, S. and López García, Á. Making Federated Learning Accessible to Scientists: The AI4EOSC Approach. In Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '24). Association for Computing Machinery, New York, NY, USA, 253–264.[232].*

Due to the rise of this privacy preserving machine learning technique, within the framework of the European project AI4EOSC, this functionality was implemented as a tool for allowing to deploy a federated learning to which the participating clients can connect for training.

C.1 AI4EOSC Platform

The AI4EOSC platform arises within the European project of the same name, and aims to be a user friendly and intuitive platform for AI model development within the framework of the EOSC. With this objective, it provides users with a simple and intuitive IDE for developing AI/ML/DL models (*VSCode* or *JupyterLab*) allowing seamless access to computational resources to accelerate model development,

including access to GPUs for model training.

The platform has been developed following the Platform-as-a-Service (PaaS) cloud computing model, whereby hardware and software resources are managed to provide an environment in which users can run, deploy and instantiate in an easy way.

For the development of this platform, different open-source software solutions have been used, such as Nomad, employed as scheduler and orchestrator for managing containers; Consul, the service for automating network configurations and *Traefik*, the load balancer and reverse proxy that exposes the jobs. It should be noted that the platform has a dashboard for intuitive user access, being a single web application developed using *Angular*, *Jest*, and the *Material Angular UI* component library. In addition, it has an API written in Python using *FastAPI* (logic API) and the Nomad Python client to deploy jobs to the cluster.

The main objective of the platform is to provide users with advanced features for distributed, federated, composite learning, metadata provenance, MLOps, event-driven data processing, and provision of AI/ML/DL services. Thus, as far as we are concerned in this chapter, we are going to focus on the integration of the functionality of FL to allow users to carry out federated training using the platform in an intuitive and simple way. Specifically, the idea is that users can deploy in a few simple interactive steps from the dashboard a FL server with the desired characteristics in terms of aggregation function, number of clients, etc. Subsequently the clients can be deployed on external cloud machines, locally, or within the platform in different deployments.

In the following sections the successful integration of the *Flower* framework for FL into the AI4EOSC platform is presented, provide an overview of the challenges faced during the implementation and the development of various extensions and modifications to the library in order to add more functionalities.

C.2 Federated Learning in AI4EOSC

For the allowing users to test and deploy a federated learning architecture on the platform, the Python framework *Flower* has been used. Specifically, a tool called "federated server" has been created, so that once the different clients have agreed to start the federated training, one of them will be in charge of initializing the server on the platform. When initializing the server, different functionalities can be customized, such as the aggregation function used, the number of training rounds, the way to measure the evaluation metrics, the minimum number of clients needed

to start the training, etc. Thus, once the server is initialized with the features agreed upon by the different participating clients, the server can be started and the clients can connect to it for starting the process.

The following is a step-by-step guide on how to deploy the federated learning server in the AI4EOSC platform, as well as the process by which clients are connected to the training:

C.2.1 Deploying and Starting the Federated Learning Server

As already stated, the server is deployed from the dashboard, following a few simple steps to configure it:

- **Deployment options:** deployment title, deployment description, custom domain for the endpoint (if needed) and service to run (*fedserver*, *JupyterLab* or *Visual Studio Code*).

  Then, the user can select the docker image and docker tag to use for deployment. As will be explained later, we have implemented a secret management systems so that the federated server can be configured including tokens for authenticating the clients when connecting with the server. In order to make use of this option, the docker tag *tokens* must be selected.

  In addition, in this step users can select if they want to monitor the carbon footprint (CO_2 emissions), as will be detailed later.

- **Hardware configuration:** for running the server no special hardware requirements are necessary, as the model training is done on the clients' side, and they will have the highest computational requirements, for example in terms of RAM, CPUs and GPUs. However, the users can select the resources needed from the server side (number of CPUs, RAM memory and disk memory).

- **Federated configuration:** is the most important part, as it includes the specifications that the federated training must satisfy. Specifically, the parameters to be filled in by the user and additional options are the following.

  - *Federated rounds:* number of rounds that the training process will be repeated.
  - *Minimum number of clients:* number of clients that must be connected to the server and ready to train the models to start the process.

- *Federated metric:* is the metric used for monitoring and validating. More than one metric can be included.
- *Federated aggregation strategy:* is the function or strategy that the server will apply to aggregate the models received from the clients. It must be selected among different predefined options, including FedAvg, FedProx or Adaptive Federated Optimization using Adam among others.
- The user can select if they want to apply differential privacy or metric privacy with fixed clipping from the server side. Then, they can customize the noise multiplier, the number of sampled clients and the clipping norm.

If the server has been created with tokens, the user will have the possibility to create labeled tokens (e.g. one token for each client), which can be distributed to the participating clients to allow only those who have a token to connect.

Once the server is created, the user can access the IDE (if selected), or access the monitoring terminal (if *fedserver* was selected). In the first case, it will be necessary to run the server to start waiting for the connection of the clients as well as to receive the information from them:

```
/srv# cd ai4os-federated-server/
/srv/ai4os-federated-server# python3 fedserver/server.py
```

Example Code C.1: Running the FL server from an AI4EOSC deployment using an IDE.

Once the server is running, the clients can start connecting, obtaining from the server an evolution similar to the one shown in the Code Example C.2, in which 2 clients connect to the server in the first round.

```
INFO :      Starting Flower server, config: num_rounds=5, no round_timeout
INFO :      Flower ECE: gRPC server running (5 rounds), SSL is disabled
INFO :      [INIT]
INFO :      Requesting initial parameters from one random client
INFO :      Received initial parameters from one random client
INFO :      Starting evaluation of initial global parameters
INFO :      Evaluation returned no results (`None`)
INFO :
INFO :      [ROUND 1]
INFO :      configure_fit: strategy sampled 2 clients (out of 2)
```

Example Code C.2: First round of a FL training started from the AI4EOSC platform with two clients connected.

C.2.2 Creating and Running the Clients

To create the clients it is only necessary to create a class that inherits from the class `fl.client.NumPyClient` from *Flower*. Specifically, once each client have read the data and created the model, he/she can create `Client` class, and then connect to the server using the endpoint where it is deployed, as shown below:

```
(...)
import flwr as fl
from pathlib import Path
import tensorflow as tf
import certifi
(...)
class Client(fl.client.NumPyClient):
    def get_parameters(self, config):
        return model.get_weights()

    def fit(self, parameters, config):
        model.set_weights(parameters)
        model.fit(x_train, y_train, epochs=3, batch_size=16)
        return model.get_weights(), len(x_train), {}

    def evaluate(self, parameters, config):
        model.set_weights(parameters)
        loss, accuracy = model.evaluate(x_test, y_test)
        return loss, len(x_test), {"accuracy": accuracy}

# Start -> connecting with the server
# Fill with the UUID of the deployment of the FL server (AI4EOSC):
uuid = "..."
# Fill with the name of the site in with the server is deployed
# (ifca or iisas). This can be found in the URL:
center = "..."
# Degine the endpoint and start the connection with the server:
end_point = f"fedserver-{uuid}.{center}-deployments.cloud.ai4eosc.eu"
fl.client.start_numpy_client(
    server_address=f"{end_point}:443",
    client=Client().to_client(),
    root_certificates=Path(certifi.where()).read_bytes(),
)
```

Example Code C.3: Creating the `Client` class and connecting with the server deployed in the given endpoint.

C.2.3 Extensions and Modifications to the Flower Library

In order to include different functionalities to the federated training performed in the platform, various modifications to the original library and extensions have been implemented. The most relevant ones are presented below.

C.2.3.1 Running the Server Behind a Proxy

In order to effectively manage the distribution of resources according to the requests of the platform users, on the AI4EOSC platform *Traefik* is used as reverse proxy or load balance that exposes jobs under the specific domain. Thus, for the correct deployment and exposure of the `fedserver` in the platform, as well as the connection of the clients to it, it was necessary to make some modifications to the *Flower* library.

First, note that the connection of the different clients to the federated learning server using *Flower* is performed using the gRPC (gRPC Remote Procedure Call) protocol, which facilitates efficient communication between distributed applications. Specifically, a gRPC server is deployed to carry out this connection, which is responsible for listening and answering to the client requests. If the gRPC server is running behind a load balancer (as in the case of AI4EOSC, using *Traefik* as reverse proxy), clients may not be able to connect to the server. In version 1.6.0, *Flower* used the `peer()` method from `grpc.ServicerContext` in order to identify unique *Flower* clients and avoid duplicate clients. However, in some situations (like when running the gRPC server behind a load balancer or a proxy), different clients can have the same peer identifier (in this case corresponding to the same IP port), as HTTP/2 connections are multiplexed (gRPC uses HTTP/2 as transport protocol). The solution that we proposed in order to solve this issue was already included in version 1.7.0 of the *Flower* library.

C.2.3.2 Client Authentication

In the simplest scenario the clients only need to introduce the IP of the server (or endpoint) to connect to the training. However, when working with FL architectures where multiple parties are involved (one server and multiple clients), it is important to maintain the integrity and security of the process.

In this regard, poissoning attacks are a major challenge in FL. These attacks can be performed by a client that enters deliberately wrong labels for its data in order to damage the overall accuracy of the model, by introducing other kind of data for training, or just by sending random weights or parameters for the model. The solution proposed within AI4EOSC consists of developing an authentication system for clients prior to their incorporation into the federated training through the use of tokens.

The tokens themselves are considered as secrets and must be managed in a secure and systematic manner. In the case of a security breach where the secret or token is exposed, it has to be revoked, and there would be no direct way to

modify the server to accept another secret. Therefore, the implementation of a secret management service, specifically *HashiCorp's Vault* [233], has been adopted for storing the tokens. The federated learning server retrieves a list of authorized tokens for client authentication from the secret management service and verifies if the client's provided token is included. Users can manage tokens (add or revoke) through the service's native command-line client, using the graphical user interface or simple from the AI4EOSC dashboard.

In order to integrate this service with the *Flower* library, an extension has been implemented to work alongside the *Vault* service that stores the secrets (it can be found in [234]). In addition, some modifications to the *Flower* framework has been developed in order to manage the credentials that allow to enable or disable the connection to the clients based on the token introduced (see [235]).

In the Example Code C.4 we show how the token interceptor is created using the secrets stored in *Vault* and how the server is started including the interceptor. Note that the strategy and the other federated settings (e.g. number of rounds) are generated using the information provided when creating the deployment and can be retrieved by means of the environment variables in which this information is stored (as well as parameters specific to each aggregation function, as the case of μ for *FedProx* given in `MU_FEDPROX`).

In addition, note that in order to get the secret path, we need to get the deployment universally unique identifier (`UUID`) and the *ID* of the user (`USER`), together with the virtual organization to which he/she belongs (given in the `NOMAD_NAMESPACE` parameter and obtained automatically from this environment variable).

```
(...)
FEDERATED_ROUNDS: int = int(os.environ['FEDERATED_ROUNDS'])
FEDERATED_METRIC = os.environ['FEDERATED_METRIC']
FEDERATED_MIN_FIT_CLIENTS: int = int(
    os.environ["FEDERATED_MIN_FIT_CLIENTS"]
)
FEDERATED_MIN_AVAILABLE_CLIENTS: int = int(
    os.environ["FEDERATED_MIN_AVAILABLE_CLIENTS"]
)
FEDERATED_STRATEGY: str = os.environ['FEDERATED_STRATEGY']
MU_FEDPROX = os.environ["MU_FEDPROX"]
FEDAVGM_SERVER_FL = os.environ["FEDAVGM_SERVER_FL"]
FEDAVGM_SERVER_MOMENTUM = os.environ["FEDAVGM_SERVER_MOMENTUM"]
UUID: str = os.environ['NOMAD_JOB_NAME'][8:]
USER: str= os.environ['NOMAD_META_owner']
VAULT_TOKEN: str = os.environ['VAULT_TOKEN']
(...)
from ai4flwr.auth.vault import VaultBearerTokenInterceptor
# Include token interceptor
nomad_path = f"users/{USER}/vo.{NOMAD_NAMESPACE}.eu"
```

```
token_interceptor = ai4flwr.auth.vault.VaultBearerTokenInterceptor(
    vault_addr="https://vault.services.fedcloud.eu:8200/",
    vault_token=VAULT_TOKEN,
    vault_mountpoint="/secrets/",
    secret_path=f"{nomad_path}/deployments/{UUID}/federated",
)

log(INFO, "Token interceptor created")

)
# Start the server
fl.server.start_server(
    server_address="0.0.0.0:5000",
    config=fl.server.ServerConfig(
        num_rounds=FEDERATED_ROUNDS
    ),
    strategy=strategy,
    interceptors=[token_interceptor],
)
```

Example Code C.4: *Flower* server with token interceptor.

However, different modification have been performed in the current code depending on whether metric privacy or DP is used or not, and also depending on whether the user wants to monitor the carbon emissions derived from the training, as will be explained in the following sections.

It is important to note that this kind of authentication will be useful to prevent unauthorized clients from participating in the training for malicious purposes. However, it will not prevent poisoning attacks in case an authenticated and trusted client misbehaves and sends misleading weights. However, if this happens and the server has a module capable of detecting anomalous behavior in a client (e.g. drift), it will be able to revoke the token that such client used to authenticate and thus prevent the connection in the upcoming rounds.

C.2.3.3 Integrating Metric Privacy

As mentioned in Section 4.5, there are different ways to include DP in a FL architecture. In particular, in *Flower* we have the possibility to use it from the client side as well as from the server side, with fixed or with adaptive clipping of the norm of the gradients.

In order to integrate this functionality into the AI4EOSC platform, when the clients initialize an FL server to perform the federated training, we have included the function for adding server side DP with fixed clipping. In order to use it, we have integrated the class `DifferentialPrivacyServerSideFixedClipping` from *Flower* (specifically from `flwr.server.strategy`). Thus, when deploying the

server from the dashboard the user can select whether to use DP or not, and in the first case, they must introduce the clipping norm, the noise multiplier and the number of sampled clients.

Additionally, we were interested in introducing the notion of metric privacy, in order to, as discussed in Section 4.5 and tested in [154], improve the performance of the model with respect to the case with server side DP, while preventing client inference attacks.

For this purpose, a new class has been implemented for this purpose, namely `MetricDifferentialPrivacyServerSideFixedClipping`, also considering the distance metric proposed in Section 4.5). Then, we can ensure that metric privacy is fulfilled regarding the metric given in Equation 4.10 in each round of the FL scheme. Thus, to integrate this functionality with the use of the *Flower* library to perform a federated training, this new class can be found in the folder in which we can find the different aggregation functions for the server (see Appendix F from [154]). This class inherits from `Strategy`, so it can be used with the different strategies defined. The main changes in this function regarding the function for including server side DP can be found in the function that computes the updates, clip, and pass them for aggregation (`aggregate_fit`). Specifically, we modify it in such a way that when adding the Gaussian noise, instead of introducing the noise multiplier, we enter it divided by the distance calculated in each round. The function implemented for calculating the distance is shown in Example Code C.5:

```
def distance_metric(
    self,
    results: list[tuple[ClientProxy, FitRes]],
    ) -> float:

    parameters = []
    for _, res in results:
        parameters.append(parameters_to_ndarrays(res.parameters))

    norm_params = []
    for i, param in enumerate(parameters):
        param_i = parameters[i]
        for j in range(i+1, len(parameters)):
            param_j = parameters[j]
            norm_wij = np.mean([
                np.linalg.norm(param_i[k]-param_j[k])
                for k in range(len(param_j))
            ])
            norm_params.append(norm_wij)

    print(f'Distance metric: {max(norm_params)}')
    return max(norm_params)
```

Example Code C.5: Function for calculating the distance for integrating metric privacy from the server side with fixed clipping.

This process is integrated when deploying the server, once the user has selected to use metric privacy from the server side, in a transparent way for the user, and does not require any modification on his/her part, nor from the client side.

Not that in order to use this functionality we maintain our own fork of the *Flower* library including this new feature.

C.2.3.4 Carbon Footprint Monitoring

Training ML/DL models, especially large-scale ones, require significant computational resources, often relying on energy from fossil fuels. This contributes to greenhouse gas emissions, aggravating climate change, which makes it crucial to monitor the training emissions and energy consumption (and thus put the focus os the sustainability challenges arising from the computing requirements).

In a FL architecture the processing is distributed among multiple machines, which avoids overloading centralized servers and reduces cooling demand in data centers. Unlike centralized learning, where all data must be transmitted and stored in a central server (which involves high energy consumption in transmission and storage), in FL only model updates are exchanged, which reduces the communication bandwidth and transmission requirements. Furthermore, FL takes advantage of the processing power of local devices, potentially reducing the requirements of a centralized infrastructure. However, while this may suggest energy gains, some studies such as [236] shows that depending on the configuration (also taking into account the aggregation strategy used), FL can emit more carbon emissions that centralized learning. Specifically in FL we have to take into account the communication cost and that we will need to have multiple machines or nodes available and connected in order to perform the training. Then, if the model requires many iterations/rounds between the server and the clients, the energy cost per transmission may exceed the local processing savings.

Thus, with the purpose of making users aware of the carbon emissions derived from their training, when the user deploys a the FL server, the possibility of monitoring the carbon emissions is integrated. For this, we have used the *codecarbon* Python library [237]. We encountered two problems in this regard: (1) clients may not want to share their emissions with each other or with the server, for privacy reasons, as this may reveal information about their computer systems, or for other technical reasons. This makes it difficult in practice to monitor the consumption derived from the complete FL scheme; (2) following the previous point, from the platform we cannot force the clients to use this functionality from the client side if they do not want to, since the idea would be to monitor the local consumption of the

training in each client, and that this information would be sent by all clients to the server that would aggregate it together with the emissions of the server (specially during the aggregation process) to get the global consumption of the training.

However, as this information is key in order to align with the ML/DL best practices concerning sustainability, we have integrated this functionality into the FL server deployed in AI4EOSC. As emission monitoring often goes hand in hand with the optimization of model architectures, training algorithms and hardware usage, this can motivate the development of more efficient models that achieve similar performance with fewer resources.

Thus, users can choose to deploy the server using the CO_2 emissions monitoring functionality. Specifically, on the server side we monitor the aggregation process of the models, and aggregate the results received in terms of emissions derived from the local training in each client (clients can decide whether or not to send this information). For this purpose, we have created a class that inherits from the different aggregation strategies and patches the `aggregate_fit` function, where the information is received from all participating clients in terms of model parameters and metrics. If the clients have tracked their emissions using *codecarbon* they can send them together with the metrics and model parameters, and the server would sum up the emissions of all the clients together with the emissions derived by performing the aggregation. Then, the total emissions per round can be obtained. To be able to send this information, the client class is defined as follows:

```
(...)
from codecarbon import EmissionsTracker
(...)

class Client(fl.client.NumPyClient):
    def get_parameters(self, config):
        return model.get_weights()

    def fit(self, parameters, config):
        model.set_weights(parameters)
        tracker = EmissionsTracker()
        tracker.start()
        model.fit(x_train, y_train, epochs=3, batch_size=16)
        em_co2 = tracker.stop()
        print(f"Client Carbon Emissions: {em_co2} kg CO2")
        return model.get_weights(), len(x_train), {"emissions": em_co2}

    def evaluate(self, parameters, config):
        model.set_weights(parameters)
        loss, accuracy = model.evaluate(x_test, y_test)
        return loss, len(x_test), {"accuracy": accuracy}
```

Example Code C.6: Class of the *Flower* client including carbon emissions monitoring with *codecarbon*.

On the other hand, from the server side a class has been created to be used instead of the strategy defined as the corresponding *Flower* class, in case the user selects the option of monitoring the CO_2 emissions. Specifically, in order to contemplate the cases of integration with DP from the server side and metric privacy from the server side (in both cases with fixed clipping), three classes have been implemented for each of these scenarios (aggregation without DP, with DP or with metric privacy). The Example Code C.7 shows the class implemented for a base *strategy*, defined on the basis of the parameters introduced by the user when deploying the server.

```
(...)
from codecarbon import OfflineEmissionsTracker
DATA_CENTER = os.environ['NOMAD_DC']
if DATA_CENTER == 'iisas-ai4eosc':
    COUNTRY = 'SVK'
else:
    COUNTRY = 'ESP'
(...)
class AggregateEmissions(type(strategy)):
    def __init__(self, strategy):
        params = strategy.__dict__
        super().__init__(**params)

    def aggregate_fit(self, server_round, results, failures):
        server_tracker = OfflineEmissionsTracker(
            country_iso_code=COUNTRY, save_to_file=False
        )
        server_tracker.start()
        aggregated_parameters, aggregated_metrics = (
            super().aggregate_fit(server_round, results, failures)
        )

        # Emissions from clients
        client_emissions = [
            res.metrics["emissions"] for _, res in results
            if "emissions" in res.metrics
        ]
        total_client_emissions = (
            sum(client_emissions) if client_emissions
            else 0.0
        )

        # Server-side emissions
        server_emissions = server_tracker.stop()

        emissions = total_client_emissions + server_emissions
        print(f"ROUND {server_round}.")
        print(f"Total Carbon Emissions: {emissions} kg CO2.")

        return aggregated_parameters, {"total_emissions": emissions}
```

Example Code C.7: Tracking CO_2 emissions from the server side in a FL training.

In the case where aggregation including server side DP or metric privacy is used, the function `aggregate_fit` will be analogous to the previous case, but the class constructor will be re-defined as given in the Code Example C.8. Note that in the Code Example C.8, the version for for server side-DP is given, and using the class `MetricDifferentialPrivacyServerSideFixedClipping` (also from `flwr.server.strategy`, from the AI4EOSC version of the *Flower* library [235]) instead of `DifferentialPrivacyServerSideFixedClipping` we will have the implementation for the case of metric privacy.

```
(...)
from flwr.server.strategy import (
    DifferentialPrivacyServerSideFixedClipping
)
from codecarbon import OfflineEmissionsTracker
(...)
NOISE_MULTIPLIER = os.environ["NOISE_MULT"]
CLIPPING_NORM = os.environ["CLIP_NORM"]
SAMPLED_CLIENTS = os.environ['SAMPLED_CLIENTS']
DATA_CENTER = os.environ['NOMAD_DC']
if DATA_CENTER == 'iisas-ai4eosc':
    COUNTRY = 'SVK'
else:
    COUNTRY = 'ESP'
(...)
class AggregateEmissionsDP(
    DifferentialPrivacyServerSideFixedClipping
):
    def __init__(self, strategy):
        super().__init__(
            strategy,
            noise_multiplier=float(NOISE_MULTIPLIER),
            clipping_norm=float(CLIPPING_NORM),
            num_sampled_clients=SAMPLED_CLIENTS
        )
```

Example Code C.8: Tracking CO_2 emissions from the server side in a FL training. Case with server side DP with fixed clipping (class constructor).

C.3 Summary and Conclusions

This section presents the integration of a federated learning server within the framework of the AI4EOSC platform, specially implemented using the *Flower* library presented in Section 4.7.

Within the platform, users can request the computational resources needed for their applications, including GPUs to train the models from the client side. Regarding federated training, the server can be deployed in an intuitive and user-friendly way on the platform, and clients can be deployed on external machines

in the cloud, locally, or within the same platform in different deployments with different resources.

In addition, it has been described how several modifications and extensions have been integrated to the original framework in order to provide more functionalities to the users, as well as to solve some issues that arose during the integration:

- First, modifications were made to allow the server to run behind a proxy, which were already integrated into the *Flower* library.
- Subsequently, a secret management service was integrated to manage tokens to be used by clients to connect to the server (if this functionality is enabled). Tokens for authenticating are created from the dashboard in an intuitive way, so that tokens may be revoked if needed as well as new ones can be created.
- While server side DP was available in the *Flower* library, we have first integrated this option so that the server is deployed with such functionality if required by the user, who has to enter the associated parameters when deploying the server from the dashboard. Additionally we have also incorporated the functionality to apply metric privacy with fixed clipping from the server side, following what was presented in Section 4.5.
- Finally, the possibility to monitor the CO_2 emissions derived from the training has been integrated. This requires changes on the clients side, but with a few lines of code the emissions of each client can be monitored and then the server can receive them and add them in each round together with its own.

All these functionalities allow users to have at their disposal a platform that allows them to carry out a complete FL workflow, including client authentication, PETs and emissions monitoring, all framed within the scope of the EOSC, thus enhancing collaboration as a key pillar for the development of open science.

APPENDIX D

COMPUTING FACILITIES

In this last appendix I would like to acknowledge the computing facilities that I disposed of during the development of this PhD thesis.

First, I had access to the IFCA cloud computing system, that operates under OpenStack and which is composed of 158 computing nodes, 4734 CPU cores with 8256 GB of RAM on machines of different brands. In addition, concerning storage capacity, I used a NextCloud instance at IFCA provided with 200GB of memory.

On the other hand, I had access to the computing facilities available within the AI4EOSC platform, which has the following resources: 716 CPU cores with 2080GB of RAM, 7315GB of disk memory and 27 GPUs (3 NVIDIA A100, 16 NVIDIA Tesla T4 and 8 NVIDIA Tesla V100). These resources are distributed in two data centers, one located at IFCA and the second one in the Institute of Informatics of the Slovak Academy of Sciences (IISAS), in Bratislava, Slovakia.

BIBLIOGRAPHY

[1] R. Kitchin, *The data revolution: Big data, open data, data infrastructures and their consequences*. Sage, 2014. DOI: https://doi.org/10.4135/9781473909472.

[2] P. Tripathi, "Drug classification dataset." https://www.kaggle.com/datasets/prathamtripathi/drug-classification/data. [Accessed: 15-05-2025].

[3] **J. Sáinz-Pardo Díaz** and Á. López García, "A python library to check the level of anonymity of a dataset," *Scientific Data*, vol. 9, no. 1, pp. 1–12, 2022. DOI: https://doi.org/10.1038/s41597-022-01894-2.

[4] G. Nguyen, **J. Sáinz-Pardo Díaz**, A. Calatrava, L. Berberi, O. Lytvyn, V. Kozlov, V. Tran, G. Moltó, and Á. López García, "Landscape of machine learning evolution: privacy-preserving federated learning frameworks and tools," *Artificial Intelligence Review*, vol. 58, no. 2, p. 51, 2024. DOI: https://doi.org/10.1007/s10462-024-11036-2.

[5] **J. Sáinz-Pardo Díaz** and Á. López García, "An open source python library for anonymizing sensitive data," *Scientific Data*, vol. 11, no. 1, p. 1289, 2024. DOI: https://doi.org/10.1038/s41597-024-04019-z.

[6] **J. Sáinz-Pardo Díaz** and Á. López García, "Study of the performance and scalability of federated learning for medical imaging with intermittent clients," *Neurocomputing*, vol. 518, pp. 142–154, 2023. DOI: https://doi.org/10.1016/j.neucom.2022.11.011.

[7] **J. Sáinz-Pardo Díaz** and Á. López García, "Application of federated learning to medical imaging scenarios." https://confluence.ifca.es/spaces/IC/pages/

128385199/2023+05+-+Privacy+preserving+techniques+for+data+science, July 2023. [Accessed: 18-05-2024].

[8] **J. Sáinz-Pardo Díaz** and Á. López García, "Privacy preserving techniques for data science." https://confluence.ifca.es/spaces/IC/pages/128385437/2023+07+-+Application+of+federated+learning+to+medical+imaging+scenarios, May 2023. [Accessed: 18-05-2024].

[9] D. Gupta and R. Rani, "A study of big data evolution and research challenges," *Journal of Information Science*, vol. 45, no. 3, pp. 322–340, 2019. DOI: https://doi.org/10.1177/0165551518789880.

[10] A. De Mauro, M. Greco, and M. Grimaldi, "What is big data? a consensual definition and a review of key research topics," *AIP Conference Proceedings*, vol. 1644, no. 1, pp. 97–104, 2015. DOI: https://doi.org/10.1063/1.4907823.

[11] J. W. Tukey *et al.*, *Exploratory data analysis*, vol. 2. Springer, 1977. DOI: https://doi.org/10.14778/2350229.2350255.

[12] L. Berberi, V. Kozlov, G. Nguyen, **J. Sáinz-Pardo Díaz**, A. Calatrava, G. Moltó, V. Tran, and Á. López García, "Machine learning operations landscape: platforms and tools," *Artificial Intelligence Review*, vol. 58, no. 6, p. 167, 2025. DOI: https://doi.org/10.1007/s10462-025-11164-3.

[13] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, December 2002. ISBN: 9780130803023.

[14] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[15] C. D. Martínez, P. D. García, and P. N. Sustaeta, "Hidden gender bias in big data as revealed through neural networks: Man is to woman as work is to mother?," *Revista Española de Investigaciones Sociológicas (REIS)*, vol. 172, no. 172, pp. 41–76, 2020. DOI: https://doi.org/10.5477/cis/reis.172.41.

[16] UN General Assembly, "Universal Declaration of Human Rights." https://www.refworld.org/legal/resolution/unga/1948/en/11563, Dec. 1948. [Accessed: 14-04-2025].

[17] European Union, "Charter of Fundamental Rights of the European Union." https://eur-lex.europa.eu/eli/treaty/char_2012/oj/eng. [Accessed: 14-04-2025].

[18] European Union, "Consolidated version of the Treaty on the Functioning of the European Union." https://eur-lex.europa.eu/legal-content/ENG/TXT/PDF/?uri=CELEX%3A12012E%2FTXT. [Accessed: 14-04-2025].

[19] European Parliament and Council of the European Union, "General Data Protection Regulation (GDPR)." https://gdpr.eu/. [Accessed: 20-04-2022].

[20] European Parliament and Council of the European Union, "Art. 5 gdpr principles relating to processing of personal data.." https://gdpr-info.eu/art-5-gdpr/. [Accessed: 16-05-2023].

[21] Information Commissioner's Office (UK), "Uk gdpr guidance and resources.." https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/. [Accessed: 16-05-2023].

[22] European Parliament and Council of the European Union, "Data Governance Act regulation." https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32022R0868. [Accessed: 17-10-2024].

[23] European Parliament and Council of the European Union, "Data Act regulation." https://eur-lex.europa.eu/eli/reg/2023/2854. [Accessed: 18-10-2024].

[24] European Parliament and Council of the European Union, "Eu artificial intelligence act." https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32024R1689. [Accessed: 18-10-2024].

[25] Ministerio de Ciencia, Innovación y Universidades, "Estrategia española de ciencia, tecnología e innovación." https://www.ciencia.gob.es/dam/jcr:e8183a4d-3164-4f30-ac5f-d75f1ad55059/EECTI-2021-2027.pdf, 2021. [Accessed: 11-04-2025].

[26] Ministerio de Ciencia, Innovación y Universidades, "Estrategia española de ciencia, tecnología e innovación." https://www.ciencia.gob.es/dam/jcr:c30b29d7-abac-4b31-9156-809927b5ee49/ENCA_acc_ES.pdf, 2023. [Accessed: 11-04-2025].

[27] Ministère de l'Enseignement supérieur, de la Recherche et de l'Innovation, "Deuxième plan national pour la science ouverte." https://www.enseignementsup-recherche.gouv.fr/sites/default/files/2021-09/2e-plan-national-pour-la-science-ouverte-12968.pdf, 2021. [Accessed: 11-04-2025].

[28] European Commission, "European Union's Horizon 2020 Programme." https://research-and-innovation.ec.europa.eu/funding/funding-opportunities/funding-programmes-and-open-calls/horizon-2020_en. [Accessed: 14-04-2025].

[29] UK Government Digital Service, "Higher Education Funding Council for England." https://www.gov.uk/government/organisations/higher-education-funding-council-for-england. [Accessed: 14-04-2025].

[30] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, *et al.*, "The fair guiding principles for scientific data management and stewardship," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016. DOI: https://doi.org/10.1038/sdata.2016.18.

[31] European Commission, "Horizon 2020 online manual. open access and data management.." https://ec.europa.eu/research/participants/docs/h2020-funding-guide/cross-cutting-issues/open-access-data-management/data-management_en.htm. [Accessed: 07-11-2023].

[32] C. Barthonnat, J. Berti, N. Couëdel, R. Coutanson, M. Dacos, A. Danciu, G. Gallezot, M. Géroudet, S. Granger, J. Janik, C. Josserand, A. L. y Barella, Émilie Lerigoleur, J.-F. Lutz, V. Mansard, C. Okret-Manville, F. Pellegrini, S. Perrin, and N. Thiboud, "Passeport pour la Science Ouverte: Guide pratique à l'usage des doctorants," February 2024. URL: https://www.ouvrirlascience.fr/passeport-pour-la-science-ouverte-guide-pratique-a-lusage-des-doctorants/.

[33] F. Pellegrini, R. D. Cosmo, L. Romary, J. Janik, S. Hodencq, R. Coutanson, M. Géroudet, and S. Granger, "Passeport pour la Science Ouverte: Science ouverte – Codes et logiciels," August 2022. URL: https://www.ouvrirlascience.fr/science-ouverte-codes-et-logiciels/.

[34] EOSC Association, "European Open Science Cloud." https://eosc.eu/. [Accessed: 21-10-2024].

[35] EOSC Association, "EOSC Federation." https://eosc.eu/building-the-eosc-federation/. [Accessed: 11-04-2025].

[36] L. Sweeney, "Simple demographics often identify people uniquely," *Health (San Francisco)*, vol. 671, no. 2000, pp. 1–34, 2000. DOI: https://doi.org/10.1184/R1/6625769.v1.

[37] A. Narayanan and V. Shmatikov, "How to break anonymity of the netflix prize dataset," *arXiv preprint cs/0610105*, 2006. DOI: https://doi.org/10.48550/arXiv.cs/0610105.

[38] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pp. 111–125, IEEE, 2008. DOI: https://doi.org/10.1109/SP.2008.33.

[39] M. Douriez, H. Doraiswamy, J. Freire, and C. T. Silva, "Anonymizing nyc taxi data: Does it matter?," in *2016 IEEE international conference on data science and advanced analytics (DSAA)*, pp. 140–148, IEEE, 2016. DOI: https://doi.org/10.1109/DSAA.2016.21.

[40] Agencia Española de Protección de Datos, "Misunderstandings related to anonymization." https://edps.europa.eu/system/files/2021-04/21-04-27_aepd-edps_anonymisation_en_5.pdf. [Accessed: 20-04-2022].

[41] M. E. Nergiz, M. Atzori, and C. Clifton, "Hiding the presence of individuals from shared databases," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, (New York, NY, USA), p. 665–676, Association for Computing Machinery, 2007. DOI: https://doi.org/10.1145/1247480.1247554.

[42] European Union Agency for Cybersecurity (ENISA), "Pseudonymisation techniques and best practices." https://www.enisa.europa.eu/publications/pseudonymisation-techniques-and-best-practices, December 2019.

[43] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication." https://www.rfc-editor.org/info/rfc2104, Feb. 1997.

[44] Google Support, "HMAC in Google Ad Manager." https://support.google.com/admanager/answer/7637490?hl=en, 2024.

[45] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. W.-C. Fu, "Utility-based anonymization for privacy preservation with less information loss," *SIGKDD Explor. Newsl.*, vol. 8, p. 21–30, dec 2006. DOI: https://doi.org/10.1145/1233321.1233324.

[46] L. Sweeney, "k-anonymity: A model for protecting privacy," *International journal of uncertainty, fuzziness and knowledge-based systems*, vol. 10, no. 05, pp. 557–570, 2002. DOI: https://doi.org/10.1142/S0218488502001648.

[47] R. C.-W. Wong, J. Li, A. W.-C. Fu, and K. Wang, "(α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, (New York, NY, USA), p. 754–759, Association for Computing Machinery, 2006. DOI: https://doi.org/10.1145/1150402.1150499.

[48] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 3–es, 2007. DOI: https://doi.org/10.1145/1217299.1217302.

[49] A. Øhrn and L. Ohno-Machado, "Using boolean reasoning to anonymize databases," *Artificial Intelligence in Medicine*, vol. 15, no. 3, pp. 235–254, 1999. DOI: https://doi.org/10.1016/S0933-3657(98)00056-6.

[50] O. Johnson and Y. Yu, "Monotonicity, thinning, and discrete versions of the entropy power inequality," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5387–5395, 2010. DOI: https://doi.org/10.1109/TIT.2010.2070570.

[51] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd International Conference on Data Engineering*, pp. 106–115, 2007. DOI: https://doi.org/10.1109/ICDE.2007.367856.

[52] J. Cao and P. Karras, "Publishing microdata with a robust privacy guarantee," *Proc. VLDB Endow.*, vol. 5, p. 1388–1399, July 2012. DOI: https://doi.org/10.14778/2350229.2350255.

[53] J. Brickell and V. Shmatikov, "The cost of privacy: destruction of data-mining utility in anonymized data publishing," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 70–78, 2008. DOI: https://doi.org/10.1145/1401890.1401904.

[54] V. Ayala-Rivera, P. McDonagh, T. Cerqueus, L. Murphy, *et al.*, "A systematic comparison and evaluation of k-anonymization algorithms for practitioners," *Transactions on data privacy*, vol. 7, no. 3, pp. 337–370, 2014. DOI: https://dl.acm.org/doi/10.5555/2870614.2870620.

[55] V. S. Iyengar, "Transforming data to satisfy privacy constraints," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Dis-*

covery and Data Mining, KDD '02, (New York, NY, USA), p. 279–288, Association for Computing Machinery, 2002. DOI: https://doi.org/10.1145/775047.775089.

[56] J. Domingo-Ferrer, *Disclosure Risk*, pp. 848–849. Boston, MA: Springer US, 2009. DOI: https://doi.org/10.1007/978-0-387-39940-9_1506.

[57] L. Sugavaneswaran, "Mathematical modeling of gene networks," in *Encyclopedia of Biomedical Engineering* (R. Narayan, ed.), pp. 33–55, Oxford: Elsevier, 2019. DOI: https://doi.org/10.1016/B978-0-12-801238-3.64118-1.

[58] J. Domingo-Ferrer and V. Torra, "Disclosure risk assessment in statistical data protection," *Journal of Computational and Applied Mathematics*, vol. 164, pp. 285–293, 2004. DOI: https://doi.org/10.1016/S0377-0427(03)00643-5.

[59] K. El Emam and F. K. Dankar, "Protecting privacy using k-anonymity," *Journal of the American Medical Informatics Association*, vol. 15, no. 5, pp. 627–637, 2008. DOI: https://doi.org/10.1197/jamia.M2716.

[60] M. E. Nergiz, M. Atzori, and C. Clifton, "Hiding the presence of individuals from shared databases," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pp. 665–676, 2007. DOI: https://doi.org/10.1145/1247480.1247554.

[61] **J. Sáinz-Pardo Díaz** and Á. López García, "*pyCANON*: A Python library to check the level of anonymity of a dataset (GitHub repository).." https://github.com/IFCA-Advanced-Computing/pycanon, 2025. [Accessed: 24-03-2025].

[62] **J. Sáinz-Pardo Díaz** and Á. López García, "*pyCANON*: A Python library to check the level of anonymity of a dataset (repository archived in Zenodo).." https://zenodo.org/records/14769827, 2025. [Accessed: 24-03-2025].

[63] **J. Sáinz-Pardo Díaz** and Á. López García, "Comparison of machine learning models applied on anonymized data with different techniques," in *2023 IEEE International Conference on Cyber Security and Resilience (CSR)*, pp. 618–623, 2023. DOI: https://doi.org/10.1109/CSR57506.2023.10224917.

[64] D. Slijepčević, M. Henzl, L. D. Klausner, T. Dam, P. Kieseberg, and M. Zeppelzauer, "k-anonymity in practice: How generalisation and suppression affect machine learning classifiers," *Computers & Security*, vol. 111, p. 102488, 2021. DOI: https://doi.org/10.1016/j.cose.2021.102488.

[65] H. Wimmer and L. Powell, "A comparison of the effects of k-anonymity on machine learning algorithms," in *Proceedings of the Conference for Information Systems Applied Research ISSN*, vol. 2167, p. 1508, 2014. DOI: https://dx.doi.org/10.14569/IJACSA.2014.051126.

[66] A. Goldsteen, G. Ezov, R. Shmelkin, M. Moffie, and A. Farkash, "Anonymizing machine learning models," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology: ESORICS 2021 International Workshops, DPM 2021 and CBT 2021, Darmstadt, Germany, October 8, 2021, Revised Selected Papers*, pp. 121–136, Springer, 2022. DOI: https://doi.org/10.1007/978-3-030-93944-1_8.

[67] D. Dua and C. Graff, "UCI machine learning repository." http://archive.ics.uci.edu/ml, 2017.

[68] ARX-deidentifier, "ARX - Open Source Data Anonymization Software (GitHub repository).." https://github.com/arx-deidentifier/arx, 2022. [Accessed: 19-03-2025].

[69] R. E. Schapire, "Explaining AdaBoost," in *Empirical Inference*, pp. 37–52, Springer Berlin Heidelberg, Jan. 2013. DOI: https://doi.org/10.1007/978-3-642-41136-6_5.

[70] J. H. Friedman, "Stochastic gradient boosting," *Computational statistics & data analysis*, vol. 38, no. 4, pp. 367–378, 2002. DOI: https://doi.org/10.1016/S0167-9473(01)00065-2.

[71] **J. Sáinz-Pardo Díaz** and Á. López García, "*Anjana*: A Python library for anonymizing sensitive data (GitHub repository).." https://github.com/IFCA-Advanced-Computing/anjana, 2025. [Accessed: 24-03-2025].

[72] **J. Sáinz-Pardo Díaz** and Á. López García, "*Anjana*: A Python library for anonymizing sensitive data (repository archived in Zenodo).." https://zenodo.org/records/14793355, 2025. [Accessed: 24-03-2025].

[73] F. Prasser and F. Kohlmayer, "Putting statistical disclosure control into practice: The arx data anonymization tool," in *Medical data privacy handbook*, pp. 111–148, Springer, 2015. DOI: https://doi.org/10.1007/978-3-319-23633-9_6.

[74] NAV IT - The Norwegian Labour and Welfare Directorate, "GitHub repository: ARXaaS." https://github.com/navikt/arxaas, 2024. [Accessed 08-05-2024].

[75] OpenAIRE, "Amnesia Anonymization Tool." https://amnesia.openaire.eu/, 2024. [Accessed 14-05-2024].

[76] F. Singhofer, A. Garifullina, M. Kern, and A. Scherp, "A novel approach on the joint de-identification of textual and relational data with a modified mondrian algorithm," in *Proceedings of the 21st ACM Symposium on Document Engineering*, DocEng '21, (New York, NY, USA), Association for Computing Machinery, 2021. DOI: https://doi.org/10.1145/3469096.3469871.

[77] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *22nd International conference on data engineering (ICDE'06)*, pp. 25–25, IEEE, 2006. DOI: https://doi.org/10.1109/ICDE.2006.101.

[78] L. Sweeney, "Datafly: A system for providing anonymity in medical data," *Database Security XI: Status and Prospects*, pp. 356–381, 1998. DOI: https://doi.org/10.1007/978-0-387-35285-5_22.

[79] T. Zhu, G. Li, W. Zhou, and S. Y. Philip, *Differential privacy and applications*. Springer, 2017. DOI: https://doi.org/10.1007/978-3-319-62004-6.

[80] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, (New York, NY, USA), p. 308–318, Association for Computing Machinery, 2016. DOI: https://doi.org/10.1145/2976749.2978318.

[81] N. Rodríguez, G. Stipcich, D. Jiménez, J. Ruiz-Millán, E. Martínez-Cámara, G. González-Seco, M. Luzon, M. Veganzones, and F. Herrera, "Federated learning and differential privacy: Software tools analysis, the sherpa.ai fl framework and methodological guidelines for preserving data privacy," *Information Fusion*, vol. 64, 08 2020. DOI: https://doi.org/10.1016/j.inffus.2020.07.009.

[82] A. Friedman and A. Schuster, "Data mining with differential privacy," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, (New York, NY, USA), p. 493–502, Association for Computing Machinery, 2010. DOI: https://doi.org/10.1145/1835804.1835868.

[83] F. Soriano, "Stroke dataset." https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset. [Accessed: 12-05-2025].

[84] PyDP, "OpenMined PyDP Python Differential Privacy Library (GitHub repository).." https://github.com/OpenMined/PyDP, 2023. [Accessed: 13-03-2025].

[85] Juba Ziani, "Approx DP: Gaussian Mech and Advanced Comp." http://www.juba-ziani.com/isye8813/lectures/06_GaussianMech.pdf, 2021. [Accessed 18-10-2024].

[86] C. Dwork, A. Roth, *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014. DOI: http://dx.doi.org/10.1561/0400000042.

[87] N. Holohan, S. Braghin, P. Mac Aonghusa, and K. Levacher, "Diffprivlib: the IBM differential privacy library," *ArXiv e-prints*, vol. 1907.02444 [cs.CR], July 2019. DOI: https://doi.org/10.48550/arXiv.1907.02444.

[88] IBM, "Diffprivlib: The IBM Differential Privacy Library (GitHub repository).." https://github.com/IBM/differential-privacy-library, 2024. [Accessed: 13-03-2025].

[89] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American statistical association*, vol. 60, no. 309, pp. 63–69, 1965. DOI: https://doi.org/10.2307/2283137.

[90] G. Kamath, "Algorithms for private data analysis. lecture 3: Some algorithms and complexity.." http://www.gautamkamath.com/CS860notes/lec3.pdf, 2020. Accessed: 2024-11-04.

[91] P. Abellet, "Privacy Preserving Machine Learning. Lecture 6: Approximation and Complexity. Master 2 Data Science, University of Lille. INRIA.." http://researchers.lille.inria.fr/abellet/teaching/ppml_lectures/lec6.pdf, 2023. [Accessed: 04-11-2024].

[92] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and A. Pazii, "Metric-based local differential privacy for statistical applications," *arXiv preprint arXiv:1805.01456*, 2018. DOI: https://doi.org/10.48550/arXiv.1805.01456.

[93] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th computer security foundations symposium (CSF)*, pp. 263–275, IEEE, 2017. DOI: https://doi.org/10.1109/CSF.2017.11.

[94] D. Jiang, S. Sun, and Y. Yu, "Functional rényi differential privacy for generative modeling," *Advances in Neural Information Processing Systems*,

vol. 36, pp. 14797–14817, 2023. DOI: https://dl.acm.org/doi/10.5555/3666122.3666774.

[95] K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi, "Broadening the scope of differential privacy using metrics," in *Privacy Enhancing Technologies* (E. De Cristofaro and M. Wright, eds.), (Berlin, Heidelberg), pp. 82–102, Springer Berlin Heidelberg, 2013. DOI: https://doi.org/10.1007/978-3-642-39077-7_5.

[96] S. Biswas and C. Palamidessi, "Privic: A privacy-preserving method for incremental collection of location data," *Proceedings on Privacy Enhancing Technologies*, vol. 2024, no. 1, pp. 582–596, 2023. DOI: https://doi.org/10.56553/popets-2024-0033.

[97] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 901–914, 2013. DOI: https://doi.org/10.1145/2508859.2516735.

[98] N. Fernandes, *Differential privacy for metric spaces: information-theoretic models for privacy and utility with new applications to metric domains*. PhD thesis, École Polytechnique Paris; Macquarie University, 2021. URL: https://theses.hal.science/tel-03344453.

[99] A. Ziller, D. Usynin, R. Braren, M. Makowski, D. Rueckert, and G. Kaissis, "Medical imaging deep learning with differential privacy," *Scientific Reports*, vol. 11, no. 1, p. 13524, 2021. DOI: https://doi.org/10.1038/s41598-021-93030-0.

[100] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pp. 177–186, Springer, 2010. DOI: https://doi.org/10.1007/978-3-7908-2604-3_16.

[101] F. G. Salieh, "Alzheimer MRI Dataset, v1.0," 2023. URL: https://huggingface.co/datasets/Falah/Alzheimer_MRI.

[102] TensorFlow, "TensorFlow Privacy Library (GitHub repository).." https://github.com/tensorflow/privacy, 2023. [Accessed: 13-03-2025].

[103] Google, "Google Differentual Privacy libraries (GitHub repository).." https://github.com/google/differential-privacy, 2024. [Accessed: 13-03-2025].

[104] PyTorch, "Opacus Library (GitHub repository).." https://github.com/pytorch/opacus, 2025. [Accessed: 13-03-2025].

[105] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine, K. Prasad, M. Malek, J. Nguyen, S. Ghosh, A. Bharadwaj, J. Zhao, G. Cormode, and I. Mironov, "Opacus: User-friendly differential privacy library in PyTorch," *arXiv preprint arXiv:2109.12298*, 2021. DOI: https://doi.org/10.48550/arXiv.2109.12298.

[106] AWS Labs, "FastDP library (GitHub repository).." https://github.com/awslabs/fast-differential-privacy, 2024. [Accessed: 13-03-2025].

[107] Z. Bu, Y.-X. Wang, S. Zha, and G. Karypis, "Differentially private optimization on large model at small cost," in *International Conference on Machine Learning*, pp. 3192–3218, PMLR, 2023. DOI: https://dl.acm.org/doi/10.5555/3618408.3618538.

[108] Z. Bu, J. Chiu, R. Liu, S. Zha, and G. Karypis, "Zero redundancy distributed learning with differential privacy," *arXiv preprint arXiv:2311.11822*, 2023. DOI: https://doi.org/10.48550/arXiv.2311.11822.

[109] Z. Bu, Y.-x. Wang, S. Zha, and G. Karypis, "Differentially private bias-term fine-tuning of foundation models," in *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, JMLR.org, 2024. DOI: https://dl.acm.org/doi/10.5555/3692070.3692259.

[110] OpenDP, "OpenDP Library (GitHub repository).." https://github.com/opendp/opendp, 2025. [Accessed: 13-03-2025].

[111] AutoDP developers, "AutoDP Library (GitHub repository).." https://github.com/yuxiangw/autodp, 2025. [Accessed: 13-03-2025].

[112] Sarus Technologies, "Sarus DP-XGBoost Library (GitHub repository).." https://github.com/sarus-tech/dp-xgboost, 2024. [Accessed: 13-03-2025].

[113] N. Grislain and J. Gonzalvez, "Dp-xgboost: Private machine learning at scale," *arXiv preprint arXiv:2110.12770*, 2021. DOI: https://doi.org/10.48550/arXiv.2110.12770.

[114] Global Biodiversity Information Facility, "GBIF webpage." https://www.gbif.org/, 2025. [Accessed: 15-05-2025].

[115] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *ACM Comput. Surv.*, vol. 53, Mar. 2020. DOI: https://doi.org/10.1145/3377454.

[116] A. Sergeev and M. Del Balso, "Horovod: fast and easy distributed deep learning in tensorflow," *arXiv preprint arXiv:1802.05799*, 2018. DOI: https://doi.org/10.48550/arXiv.1802.05799.

[117] TensorFlow, "TensorFlow Distributed documentation." https://www.tensorflow.org/guide/distributed_training, 2025. [Accessed: 15-05-2025].

[118] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from https://tensorflow.org.

[119] PyTorch, "PyTorch Distributed documentation." https://docs.pytorch.org/tutorials/beginner/dist_overview.html, 2025. [Accessed: 15-05-2025].

[120] C. Shearer, "The crisp-dm model: the new blueprint for data mining," *Journal of data warehousing*, vol. 5, no. 4, pp. 13–22, 2000.

[121] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017. URL: https://proceedings.mlr.press/v54/mcmahan17a.html.

[122] P. Jobic, A. Mayoue, S. Tucci-Piergiovanni, and F. Terrier, "Extending the scope of gradient reconstruction attacks in federated averaging," in *Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '24, (New York, NY, USA), p. 235–246, Association for Computing Machinery, 2024. DOI: https://doi.org/10.1145/3658664.3659636.

[123] W.-K. Lee, J.-S. Hong, Y.-H. Lin, Y.-F. Lu, Y.-Y. Hsu, C.-C. Lee, H.-C. Yang, C.-C. Wu, C.-F. Lu, M.-H. Sun, *et al.*, "Federated learning: A cross-institutional feasibility study of deep learning based intracranial tumor delineation framework for stereotactic radiosurgery," *Journal of Magnetic Resonance Imaging*, vol. 59, no. 6, pp. 1967–1975, 2024. DOI: https://doi.org/10.1002/jmri.28950.

[124] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019. DOI: https://doi.org/10.48550/arXiv.1909.06335.

[125] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International conference on machine learning*, pp. 5650–5659, Pmlr, 2018. URL: https://proceedings.mlr.press/v80/yin18a.html.

[126] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020. URL: https://proceedings.mlsys.org/paper_files/paper/2020/hash/1f5fe83998a09396ebe6477d9475ba0c-Abstract.html.

[127] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," *arXiv preprint arXiv:2003.00295*, 2020. DOI: https://doi.org/10.48550/arXiv.2003.00295.

[128] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proceedings of the 37th International Conference on Machine Learning* (H. D. III and A. Singh, eds.), vol. 119 of *Proceedings of Machine Learning Research*, pp. 5132–5143, PMLR, 13–18 Jul 2020. URL: https://proceedings.mlr.press/v119/karimireddy20a.html.

[129] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 6357–6368, PMLR, 18–24 Jul 2021. URL: https://proceedings.mlr.press/v139/li21h.html.

[130] **J. Sáinz-Pardo Díaz** and Á. López García, "Enhancing the convergence of federated learning aggregation strategies with limited data," *arXiv preprint arXiv:2501.15949*, 2025. DOI: https://doi.org/10.48550/arXiv.2501.15949.

[131] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, H. L. Kwing, T. Parcollet, P. P. d. Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020. DOI: https://doi.org/10.48550/arXiv.2007.14390.

[132] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020. DOI: https://doi.org/10.1109/MSP.2020.2975749.

[133] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020. DOI: https://doi.org/10.1109/TIFS.2020.2988575.

[134] A. Madi, O. Stan, A. Mayoue, A. Grivet-Sébert, C. Gouy-Pailler, and R. Sirdey, "A secure federated learning framework using homomorphic encryption and verifiable computing," in *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)*, pp. 1–8, 2021. DOI: https://doi.org/10.1109/RDAAPS48126.2021.9452005.

[135] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. A. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *CoRR*, vol. abs/1912.04977, 2019. DOI: https://doi.org/10.48550/arXiv.1912.04977.

[136] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021. DOI: https://doi.org/10.1016/j.neucom.2021.07.098.

[137] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, pp. 1–207, 2019. DOI: https://doi.org/10.1007/978-3-031-01585-4.

[138] M. Rigaki and S. Garcia, "A survey of privacy attacks in machine learning," *ACM Comput. Surv.*, vol. 56, Nov. 2023. DOI: https://doi.org/10.1145/3624010.

[139] H. Oh and Y. Lee, "Exploring image reconstruction attack in deep learning computation offloading," in *The 3rd International Workshop on Deep Learning for Mobile Systems and Applications*, EMDL '19, (New York, NY, USA), p. 19–24, Association for Computing Machinery, 2019. DOI: https://doi.org/10.1145/3325413.3329791.

[140] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, 2017. DOI: https://doi.org/10.1109/SP.2017.41.

[141] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, (New York, NY, USA), p. 1322–1333, Association for Computing Machinery, 2015. DOI: https://doi.org/10.1145/2810103.2813677.

[142] W. Jiang, H. Li, S. Liu, Y. Ren, and M. He, "A flexible poisoning attack against machine learning," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2019. DOI: https://doi.org/10.1109/ICC.2019.8761422.

[143] J. Céspedes Sisniega and Á. López García, "Frouros: An open-source python library for drift detection in machine learning systems," *SoftwareX*, vol. 26, p. 101733, 2024. DOI: https://doi.org/10.1016/j.softx.2024.101733.

[144] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern recognition*, vol. 45, no. 1, pp. 521–530, 2012. DOI: https://doi.org/10.1016/j.patcog.2011.06.019.

[145] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014. DOI: https://doi.org/10.1145/2523813.

[146] M. I. Razzak, S. Naz, and A. Zaib, "Deep learning for medical image processing: Overview, challenges and the future," *Classification in BioApps: Automation of Decision Making*, pp. 323–350, 2018. DOI: https://doi.org/10.1007/978-3-319-65981-7_12.

[147] W. Zhang, X. Wang, P. Zhou, W. Wu, and X. Zhang, "Client selection for federated learning with non-iid data in mobile edge computing," *IEEE Access*,

vol. 9, pp. 24462–24474, 2021. DOI: https://doi.org/10.1109/ACCESS.2021.3056919.

[148] K. Chomboon, P. Chujai, P. Teerarassamee, K. Kerdprasop, and N. Kerdprasop, "An empirical study of distance metrics for k-nearest neighbor algorithm," in *Proceedings of the 3rd international conference on industrial application engineering*, vol. 2, 2015. DOI: http://dx.doi.org/10.12792/iciae2015.051.

[149] I. Hegedűs, G. Danner, and M. Jelasity, "Decentralized learning works: An empirical comparison of gossip learning and federated learning," *Journal of Parallel and Distributed Computing*, vol. 148, pp. 109–124, 2021. DOI: https://doi.org/10.1016/j.jpdc.2020.10.006.

[150] W. Su, L. Li, F. Liu, M. He, and X. Liang, "AI on the edge: a comprehensive review," *Artificial Intelligence Review*, vol. 55, no. 8, pp. 6125–6183, 2022. DOI: https://doi.org/10.1007/s10462-022-10141-4.

[151] A. Van Looveren, J. Klaise, G. Vacanti, O. Cobb, A. Scillitoe, R. Samoilescu, and A. Athorne, "Alibi detect: Algorithms for outlier, adversarial and drift detection." https://github.com/SeldonIO/alibi-detect, 2019.

[152] TorchDrift Team, "Torchdrift: drift detection for pytorch.." https://github.com/torchdrift/torchdrift, 2024. [Accessed 17-01-2024].

[153] L. Nicholl, T. Schill, I. Lindsay, A. Srivastava, and S. J. Kodie P McNamara, "Menelaus.." https://github.com/mitre/menelaus, 2024. [Accessed 17-01-2024].

[154] **J. Sáinz-Pardo Díaz**, A. Athanasiou, K. Jung, C. Palamidessi, and Á. López García, "Metric privacy in federated learning for medical imaging: Improving convergence and preventing client inference attacks," *arXiv e-prints*, pp. arXiv–2502, 2025. DOI: https://doi.org/10.48550/arXiv.2502.01352.

[155] X. Yi, R. Paulet, and E. Bertino, *Homomorphic Encryption and Applications*. Springer, 2014. DOI: https://doi.org/10.1007/978-3-319-12229-8.

[156] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Computing Surveys (Csur)*, vol. 51, no. 4, pp. 1–35, 2018. DOI: https://doi.org/10.1145/3214303.

[157] K. Munjal and R. Bhatia, "A systematic review of homomorphic encryption and its contributions in healthcare industry," *Complex & Intelligent Systems*, vol. 9, no. 4, pp. 3759–3786, 2023. DOI: https://doi.org/10.1007/s40747-022-00756-z.

[158] IBM Research, "Federated learning meets homomorphic encryption." https://research.ibm.com/blog/federated-learning-homomorphic-encryption, 2022. [Accessed: 11-03-2025].

[159] A. Pedrouzo-Ulloa, A. Boudguiga, O. Chakraborty, R. Sirdey, O. Stan, and M. Zuber, "Practical multi-key homomorphic encryption for more flexible and efficient secure federated average aggregation," in *2023 IEEE International Conference on Cyber Security and Resilience (CSR)*, pp. 612–617, 2023. DOI: https://doi.org/10.1109/CSR57506.2023.10224979.

[160] Q. Zhang, S. Jing, C. Zhao, B. Zhang, and Z. Chen, "Efficient federated learning framework based on multi-key homomorphic encryption," in *Advances on P2P, Parallel, Grid, Cloud and Internet Computing* (L. Barolli, ed.), (Cham), pp. 88–105, Springer International Publishing, 2022. DOI: https://doi.org/10.1007/978-3-030-89899-1_10.

[161] E. Bresson, D. Catalano, and D. Pointcheval, "A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications," in *Advances in Cryptology - ASIACRYPT 2003* (C.-S. Laih, ed.), (Berlin, Heidelberg), pp. 37–54, Springer Berlin Heidelberg, 2003. DOI: https://doi.org/10.1007/978-3-540-40061-5_3.

[162] The TensorFlow Federated Authors, "Tensorflow federated." https://github.com/google-parfait/tensorflow-federated, 2018. [Accessed: 13-03-2025].

[163] A. Ziller, A. Trask, A. Lopardo, B. Szymkow, B. Wagner, E. Bluemke, J.-M. Nounahon, J. Passerat-Palmbach, K. Prakash, N. Rose, T. Ryffel, Z. N. Reza, and G. Kaissis, *PySyft: A Library for Easy Federated Learning*, pp. 111–139. Cham: Springer International Publishing, 2021. DOI: https://doi.org/10.1007/978-3-030-70604-3_5.

[164] H. R. Roth, Y. Cheng, Y. Wen, I. Yang, Z. Xu, Y.-T. Hsieh, K. Kersten, A. Harouni, C. Zhao, K. Lu, *et al.*, "Nvidia flare: Federated learning from simulation to real-world," *arXiv preprint arXiv:2210.13291*, 2022. DOI: https://doi.org/10.48550/arXiv.2210.13291.

[165] H. Ludwig, N. Baracaldo, G. Thomas, Y. Zhou, A. Anwar, S. Rajamoni, Y. Ong, J. Radhakrishnan, A. Verma, M. Sinn, *et al.*, "Ibm federated learning: an enterprise framework white paper v0. 1," *arXiv preprint arXiv:2007.10987*, 2020. DOI: https://doi.org/10.48550/arXiv.2007.10987.

[166] C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, *et al.*, "Fedml: A research library and benchmark for federated machine learning," *arXiv preprint arXiv:2007.13518*, 2020. DOI: https://doi.org/10.48550/arXiv.2007.13518.

[167] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, "Fate: An industrial grade platform for collaborative learning with data protection," *Journal of Machine Learning Research*, vol. 22, no. 226, pp. 1–6, 2021. URL: http://jmlr.org/papers/v22/20-815.html.

[168] K. Birman, "The promise, and limitations, of gossip protocols," *SIGOPS Oper. Syst. Rev.*, vol. 41, p. 8–13, Oct. 2007. DOI: https://doi.org/10.1145/1317379.1317382.

[169] T. T. Nguyen and M. Wahib, "An allreduce algorithm and network co-design for large-scale training of distributed deep learning," in *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pp. 396–405, 2021. DOI: https://doi.org/10.1109/CCGrid51090.2021.00049.

[170] P. Patarasuk and X. Yuan, "Bandwidth optimal all-reduce algorithms for clusters of workstations," *Journal of Parallel and Distributed Computing*, vol. 69, no. 2, pp. 117–124, 2009. DOI: https://doi.org/10.1016/j.jpdc.2008.09.002.

[171] Y. Jiang, H. Gu, Y. Lu, and X. Yu, "2d-hra: Two-dimensional hierarchical ring-based all-reduce algorithm in large-scale distributed machine learning," *IEEE Access*, vol. 8, pp. 183488–183494, 2020. DOI: https://doi.org/10.1109/ACCESS.2020.3028367.

[172] M. Yu, Y. Tian, B. Ji, C. Wu, H. Rajan, and J. Liu, "Gadget: Online resource optimization for scheduling ring-all-reduce learning jobs," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 1569–1578, IEEE, 2022. DOI: https://doi.org/10.1109/INFOCOM48880.2022.9796785.

[173] I. Hegedűs, G. Danner, and M. Jelasity, "Gossip learning as a decentralized alternative to federated learning," in *Distributed Applications and Interoperable Systems: 19th IFIP WG 6.1 International Conference, DAIS 2019*, pp. 74–90, Springer, 2019. DOI: https://doi.org/10.1007/978-3-030-22496-7_5.

[174] M. Blot, D. Picard, M. Cord, and N. Thome, "Gossip training for deep learning," *CoRR*, vol. abs/1611.09726, 2016. DOI: https://doi.org/10.48550/arXiv.1611.09726.

[175] L. Giaretta and Š. Girdzijauskas, "Gossip learning: Off the beaten path," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 1117–1124, IEEE, 2019. DOI: https://doi.org/10.1109/BigData47090.2019.9006216.

[176] **J. Sáinz-Pardo Díaz**, M. Castrillo, and Álvaro López García, "Deep learning based soft-sensor for continuous chlorophyll estimation on decentralized data," *Water Research*, vol. 246, p. 120726, 2023. DOI: https://doi.org/10.1016/j.watres.2023.120726.

[177] **J. Sáinz-Pardo Díaz**, M. Castrillo, J. Bartok, I. H. Cachá, I. M. Ondík, I. Martynovskyi, K. Alibabaei, L. Berberi, V. Kozlov, and Á. López García, "Personalized federated learning for improving radar based precipitation nowcasting on heterogeneous areas," *Earth Science Informatics*, pp. 1–24, 2024. DOI: https://doi.org/10.1007/s12145-024-01438-9.

[178] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020. DOI: https://doi.org/10.1016/j.cie.2020.106854.

[179] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, *et al.*, "The future of digital health with federated learning," *NPJ digital medicine*, vol. 3, no. 1, pp. 1–7, 2020. DOI: https://doi.org/10.1038/s41746-020-00323-1.

[180] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *International journal of medical informatics*, vol. 112, pp. 59–67, 2018. DOI: https://doi.org/10.1016/j.ijmedinf.2018.01.007.

[181] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *Journal of Healthcare Informatics Research*, vol. 5, no. 1, pp. 1–19, 2021. DOI: https://doi.org/10.1007/s41666-020-00082-4.

[182] G. H. Lee and S.-Y. Shin, "Federated learning on clinical benchmark data: Performance assessment," *J Med Internet Res*, vol. 22, p. e20891, Oct 2020. DOI: https://doi.org/10.2196/20891.

[183] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, "Federated learning for healthcare: Systematic review and architecture proposal," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 4, pp. 1–23, 2022. DOI: https://doi.org/10.1145/3501813.

[184] M. Alazab, S. P. RM, M. Parimala, P. Reddy, T. R. Gadekallu, and Q.-V. Pham, "Federated learning for cybersecurity: concepts, challenges and future directions," *IEEE Transactions on Industrial Informatics*, 2021. DOI: https://doi.org/10.1109/TII.2021.3119038.

[185] B. Ghimire and D. B. Rawat, "Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things," *IEEE Internet of Things Journal*, 2022. DOI: https://doi.org/10.1109/JIOT.2022.3150363.

[186] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive iot networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4641–4654, 2020. DOI: https://doi.org/10.1109/JIOT.2020.2964162.

[187] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained iot devices," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1–24, 2021. DOI: https://doi.org/10.1109/JIOT.2021.3095077.

[188] W. Li, F. Milletarì, D. Xu, N. Rieke, J. Hancox, W. Zhu, M. Baust, Y. Cheng, S. Ourselin, M. J. Cardoso, *et al.*, "Privacy-preserving federated brain tumour segmentation," in *International workshop on machine learning in medical imaging*, pp. 133–141, Springer, 2019. DOI: https://doi.org/10.1007/978-3-030-32692-0_16.

[189] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, "Federated learning for smart healthcare: A survey," *ACM Comput. Surv.*, vol. 55, feb 2022. DOI: https://doi.org/10.1145/3501296.

[190] D. S. Kermany, M. Goldbaum, W. Cai, C. C. Valentim, H. Liang, S. L. Baxter, A. McKeown, G. Yang, X. Wu, F. Yan, J. Dong, M. K. Prasadha, J. Pei, M. Y. Ting, J. Zhu, C. Li, S. Hewett, J. Dong, I. Ziyar, A. Shi, R. Zhang, L. Zheng, R. Hou, W. Shi, X. Fu, Y. Duan, V. A. Huu, C. Wen, E. D. Zhang, C. L. Zhang, O. Li, X. Wang, M. A. Singer, X. Sun, J. Xu, A. Tafreshi, M. A.

Lewis, H. Xia, and K. Zhang, "Identifying medical diagnoses and treatable diseases by image-based deep learning," *Cell*, vol. 172, no. 5, pp. 1122–1131.e9, 2018. DOI: https://doi.org/10.1016/j.cell.2018.02.010.

[191] P. Visser, J. Verspagen, G. Sandrini, L. Stal, H. Matthijs, T. Davis, H. Paerl, and J. Huisman, "How rising CO2 and global warming may stimulate harmful cyanobacterial blooms.," *Harmful Algae*, vol. 54, pp. 145–159, 2016. DOI: https://doi.org/10.1016/j.hal.2015.12.006.

[192] P. M. Glibert, "Harmful algae at the complex nexus of eutrophication and climate change," *Harmful Algae*, vol. 91, p. 101583, 2020. DOI: https://doi.org/10.1016/j.hal.2019.03.001.

[193] J. Jin, S. Wells, D. Liu, G. Yang, S. Zhu, J. Ma, and Z. Yang, "Effects of water level fluctuation on thermal stratification in a typical tributary bay of three gorges reservoir, china," *PeerJ*, vol. 7, 04 2019. DOI: https://doi.org/10.7717/peerj.6925.

[194] M. Barros, A. Wilson, J. Leitão, S. Pereira, R. Buley, E. Fernandez-Figueroa, and J. Capelo-Neto, "Environmental factors associated with toxic cyanobacterial blooms across 20 drinking water reservoirs in a semi-arid region of brazil.," *Harmful Algae*, vol. 86, pp. 128–137, 2019. DOI: https://doi.org/10.1016/j.hal.2019.05.006.

[195] United Nations, Department of Economic and Social Affairs (DESA), "The sustainable development goals report 2022." https://unstats.un.org/sdgs/report/2022/The-Sustainable-Development-Goals-Report-2022.pdf, 2022.

[196] H. Almuhtaram, F. A. Kibuye, S. Ajjampur, C. M. Glover, R. Hofmann, V. Gaget, C. Owen, E. C. Wert, and A. Zamyadi, "State of knowledge on early warning tools for cyanobacteria detection," *Ecological Indicators*, vol. 133, p. 108442, 2021. DOI: https://doi.org/10.1016/j.ecolind.2021.108442.

[197] M. Castrillo and Á. López García, "Estimation of high frequency nutrient concentrations from water quality surrogates using machine learning methods," *Water Research*, vol. 172, p. 115490, 2020. DOI: https://doi.org/10.1016/j.watres.2020.115490.

[198] S. S. Khruschev, T. Y. Plyusnina, T. K. Antal, S. I. Pogosyan, G. Y. Riznichenko, and A. B. Rubin, "Machine learning methods for assessing photosynthetic activity: environmental monitoring applications," *Biophysical Reviews*, vol. 14, p. pages 821–842, 2022. DOI: https://doi.org/10.1007/s12551-022-00982-2.

[199] P. García Nieto, E. García-Gonzalo, J. Alonso Fernández, and C. Díaz Muñiz, "Water eutrophication assessment relied on various machine learning techniques: A case study in the englishmen lake (northern spain)," *Ecological Modelling*, vol. 404, pp. 91–102, 2019. DOI: https://doi.org/10.1016/j.ecolmodel.2019.03.009.

[200] Y. Park, K. H. Cho, J. Park, S. M. Cha, and J. H. Kim, "Development of early-warning protocol for predicting chlorophyll-a concentration using machine learning models in freshwater and estuarine reservoirs, korea," *Science of The Total Environment*, vol. 502, pp. 31–41, 2015. DOI: https://doi.org/10.1016/j.scitotenv.2014.09.005.

[201] K.-M. Kim and J.-H. Ahn, "Machine learning predictions of chlorophyll-a in the han river basin, korea," *Journal of Environmental Management*, vol. 318, 2022. DOI: https://doi.org/10.1016/j.jenvman.2022.115636.

[202] A. Mozo, J. Morón-López, S. Vakaruk, Á. G. Pompa-Pernía, Á. González-Prieto, J. A. P. Aguilar, S. Gómez-Canaval, and J. M. Ortiz, "Chlorophyll soft-sensor based on machine learning models for algal bloom predictions," *Scientific Reports*, vol. 12, p. 13529, Aug 2022. DOI: https://doi.org/10.1038/s41598-022-17299-5.

[203] UK CEH, "Environmental Information Data Centre." https://eidc.ac.uk/, 2024. [Accessed 24-07-2024].

[204] European Comission, "Photovoltaic Geographical Information System." https://re.jrc.ec.europa.eu/pvg_tools/en/, 2024. [Accessed 24-07-2024].

[205] A. J. Wade, E. J. Palmer-Felgate, S. J. Halliday, R. A. Skeffington, M. Loewenthal, H. P. Jarvie, M. J. Bowes, G. M. Greenway, S. J. Haswell, I. M. Bell, E. Joly, A. Fallatah, C. Neal, R. J. Williams, E. Gozzard, and J. R. Newman, "Hydrochemical processes in lowland rivers: insights from in situ, high-resolution monitoring," *Hydrology and Earth System Sciences*, vol. 16, no. 11, pp. 4323–4342, 2012. DOI: https://doi.org/10.5194/hess-16-4323-2012.

[206] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 4768–4777, Curran Associates Inc., 2017. DOI: https://dl.acm.org/doi/10.5555/3295222.3295230.

[207] V. Svoboda, M. Hanel, P. Máca, and J. Kyselý, "Projected changes of rainfall event characteristics for the czech republic," *Journal of Hydrology and Hydromechanics*, vol. 64, no. 4, pp. 415–425, 3916. DOI: https://doi.org/10.1515/johh-2016-0036.

[208] J. Rajczak and C. Schär, "Projections of future precipitation extremes over europe: A multimodel assessment of climate simulations," *Journal of Geophysical Research: Atmospheres*, vol. 122, no. 20, pp. 10,773–10,800, 2017. DOI: https://doi.org/10.1002/2017JD027176.

[209] M. Hanel and T. A. Buishand, "On the value of hourly precipitation extremes in regional climate model simulations," *Journal of Hydrology*, vol. 393, no. 3, pp. 265–273, 2010. DOI: https://doi.org/10.1016/j.jhydrol.2010.08.024.

[210] P. Hosseinzadehtalaei, H. Tabari, and P. Willems, "Climate change impact on short-duration extreme precipitation and intensity–duration–frequency curves over europe," *Journal of Hydrology*, vol. 590, p. 125249, 2020. DOI: https://doi.org/10.1016/j.jhydrol.2020.125249.

[211] V. Bačová Mitková, P. Pekárová, D. Halmová, and P. Miklánek, "Reconstruction and post-event analysis of a flash flood in a small ungauged basin: a case study in slovak territory," *Natural hazards*, vol. 92, no. 2, pp. 741–760, 2018. DOI: https://doi.org/10.1007/s11069-018-3222-2.

[212] M. Korosec, "The most powerful tornado on record hit the czech republic, leaving several fatalities and 200+ injured across the hodonin district," *Severe Weather Europe, Weather Report*, 2021. URL: https://www.severe-weather.eu/weather-report/europe-severe-weather-tornado-hodonin-czech-republic-mk/.

[213] K. Komjáti, Á. J. Varga, L. Méri, H. Breuer, and S. Kun, "Investigation of a supercell merger leading to the ef4 tornado in the czech republic on june 24, 2021 using radar data and numerical model outputs.," *Időjárás*, vol. 126, no. 4, 2022. DOI: http://dx.doi.org/10.28974/idojaras.2022.4.2.

[214] P. Pavlík, V. Rozinajová, and A. B. Ezzeddine, "Radar-based volumetric precipitation nowcasting: A 3d convolutional neural network with u-net architecture," in *2nd Workshop on Complex Data Challenges in Earth Observation (CDCEO 2022)*, 2022. URL: https://ceur-ws.org/Vol-3207/paper10.pdf.

[215] S. Agrawal, L. Barrington, C. Bromberg, J. Burge, C. Gazen, and J. Hickey, "Machine learning for precipitation nowcasting from radar images," 2019. DOI: https://doi.org/10.48550/arXiv.1912.12132.

[216] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems* (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds.), vol. 27, Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper_files/paper/2014/hash/f033ed80deb0234979a61f95710dbe25-Abstract.html.

[217] S. Ravuri, K. Lenc, M. Willson, D. Kangin, R. Lam, P. Mirowski, M. Fitzsimons, M. Athanassiadou, S. Kashem, S. Madge, R. Prudden, A. Mandhane, A. Clark, A. Brock, K. Simonyan, R. Hadsell, N. Robinson, E. Clancy, A. Arribas, and S. Mohamed, "Skilful precipitation nowcasting using deep generative models of radar," *Nature*, vol. 597, p. 672–677, 2021. DOI: https://doi.org/10.1038/s41586-021-03854-z.

[218] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Deep learning for precipitation nowcasting: a benchmark and a new model," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 5622–5632, Curran Associates Inc., 2017. DOI: https://dl.acm.org/doi/10.5555/3295222.3295313.

[219] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," *Advances in neural information processing systems*, vol. 28, 2015. DOI: https://dl.acm.org/doi/10.5555/2969239.2969329.

[220] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2022. DOI: https://doi.org/10.1109/TNNLS.2022.3160699.

[221] R. Rinehart and E. Garvey, "Three-dimensional storm motion detection by conventional weather radar," *Nature*, vol. 273, no. 5660, pp. 287–289, 1978. DOI: https://doi.org/10.1038/273287a0.

[222] W.-c. Woo and W.-k. Wong, "Operational application of optical flow techniques to radar-based rainfall nowcasting," *Atmosphere*, vol. 8, no. 3, 2017. DOI: https://doi.org/10.3390/atmos8030048.

[223] P. Novak, "The czech hydrometeorological institute's severe storm nowcasting system," *Atmospheric Research*, vol. 83, no. 2, pp. 450–457, 2007. DOI: https://doi.org/10.1016/j.atmosres.2005.09.014.

[224] J. Tang and C. Matyas, "A nowcasting model for tropical cyclone precipitation regions based on the trec motion vector retrieval with a semi-lagrangian scheme for doppler weather radar," *Atmosphere*, vol. 9, no. 5, 2018. DOI: https://doi.org/10.3390/atmos9050200.

[225] F. Sabah, Y. Chen, Z. Yang, M. Azam, N. Ahmad, and R. Sarwar, "Model optimization techniques in personalized federated learning: A survey," *Expert Systems with Applications*, vol. 243, p. 122874, 2024. DOI: https://doi.org/10.1016/j.eswa.2023.122874.

[226] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. DOI: https://doi.org/10.48550/arXiv.1412.6980.

[227] B. Kumar, H. Haral, M. Kalapureddy, B. B. Singh, S. Yadav, R. Chattopadhyay, D. Pattanaik, S. A. Rao, and M. Mohapatra, "Utilizing deep learning for near real-time rainfall forecasting based on radar data," *Physics and Chemistry of the Earth, Parts A/B/C*, vol. 135, p. 103600, 2024. DOI: https://doi.org/10.1016/j.pce.2024.103600.

[228] D. Han, Y. Shin, J. Im, and J. Lee, "Key factors for quantitative precipitation nowcasting using ground weather radar data based on deep learning," *Geoscientific Model Development Discussions*, vol. 2023, pp. 1–43, 2023. DOI: https://doi.org/10.5194/gmd-16-5895-2023.

[229] I. T. Jolliffe and D. B. Stephenson, *Forecast verification: a practitioner's guide in atmospheric science*. John Wiley & Sons, 2012. DOI: https://doi.org/10.1002/9781119960003.

[230] D. R. Stinson, *Cryptography: theory and practice*. Chapman and Hall/CRC, 2005. LC record available at: https://lccn.loc.gov/2018018724.

[231] A. Pedrouzo-Ulloa, *Homomorphic Lattice Cryptosystems for Secure Signal Processing*. PhD thesis, University of Vigo, Vigo, Spain, 2020. URL: http://hdl.handle.net/11093/1360.

[232] **J. Sáinz-Pardo Díaz**, A. Heredia Canales, I. Heredia Cachá, V. Tran, G. Nguyen, K. Alibabaei, M. Obregón Ruiz, S. Rebolledo Ruiz, and Á. López García, "Making federated learning accessible to scientists: The ai4eosc approach," in *Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '24, (New York, NY, USA),

p. 253–264, Association for Computing Machinery, 2024. DOI: https://doi.org/10.1145/3658664.3659642.

[233] HashiCorp, "Vault by HashiCorp." https://www.vaultproject.io/. [Accessed: 29-02-2024].

[234] AI4EOSC, "AI4OS extensions for the Flower framework (GitHub repository)." https://github.com/ai4os/ai4-flwr/tree/develop, 2025. [Accessed: 11-03-2025].

[235] AI4EOSC, "AI4EOSC/flower: Adaptations of the flower framework for use in AI4EOSC (GitHub repository)." https://github.com/AI4EOSC/flower/tree/credentials, 2025. [Accessed: 11-03-2025].

[236] X. Qiu, T. Parcollet, J. Fernandez-Marques, P. P. Gusmao, Y. Gao, D. J. Beutel, T. Topal, A. Mathur, and N. D. Lane, "A first look into the carbon footprint of federated learning," *Journal of Machine Learning Research*, vol. 24, no. 129, pp. 1–23, 2023. URL: https://www.jmlr.org/papers/v24/21-0445.html.

[237] B. Courty, V. Schmidt, Goyal-Kamal, MarionCoutarel, B. Feld, J. Lecourt, LiamConnell, SabAmine, inimaz, supatomic, M. Léval, L. Blanche, A. Cruveiller, ouminasara, F. Zhao, A. Joshi, A. Bogroff, A. Saboni, H. de Lavoreille, N. Laskaris, E. Abati, D. Blank, Z. Wang, A. Catovic, alencon, M. Stechly, C. Bauer, Lucas-Otavio, JPW, and MinervaBooks, "mlco2/codecarbon: v2.4.1," May 2024. DOI: https://doi.org/10.5281/zenodo.4658424.