Facultad de Ciencias

DeepScan4Failure

Trabajo de Fin de Máster
para acceder al

MÁSTER EN DATA SCIENCE

Autor: Javier A. Cuartas Micieces
Director\es: Diego García Saiz (UC)

Junio- 2023

ÍNDICE

Resumen en español

Los principales objetivos de este trabajo fueron el desarrollo de una herramienta útil de detección de anomalías aplicando un *autoencoder* en Python, utilizando la librería Pytorch y ejecutando su validación en un conjunto de datos, teniendo en cuenta todos los pasos de un ciclo de vida de los datos que involucra un estudio riguroso.

Considerando los requisitos de una tarea de detección de anomalías real, se seleccionaron los datos de la competición *VSB Line Fault Detection* de *Kaggle*, para el análisis de datos y validación de la herramienta. Este es un conjunto de datos no balanceado y ruidoso, a la que la solución de aprendizaje semi-supervisada del *autoencoder*, se ajusta.

El importante número de estudios previos sobre este conjunto de datos ha ayudado a escribir la sección de análisis preliminar, curación y varias partes del *Data Management Plan (DMP)* como aquellas relacionadas con los equipos de medida o las condiciones de gestión de los datos.

El código para la curación de la competición ha sido modificado y ejecutado para obtener un grupo de variables de cada serie temporal, teniendo en cuenta la eliminación de ruido y otros pasos de pretratamiento. Esto tomó similar cantidad de tiempo que la construcción de la herramienta del escáner incluyendo el código del modelo de autoencoder y las herramientas complementarias.

Dos ciclos de ajuste de hiperparámetros fueron ejecutados después de construir la herramienta, aunque la limitada velocidad de entrenamiento permitió la prueba con sólo una pequeña cantidad de combinaciones de parámetros y tomó más de 15 días de búsqueda bayesiana alcanzar una respuesta relativa.

Sin embargo, ambas, la herramienta de detección de anomalías y su validación, han sido ejecutadas y están disponibles como resultado de este trabajo.

Resumen en inglés

The main goals of this work were the development of a useful tool to detect anomalies by applying an *autoencoder* built in python, by using the Pytorch library, and by performing its validation on a dataset, taking into account all the data life cycle steps involved in a rigorous study.

Considering a real time series anomaly detection task requirements, VSB Line Fault Detection competition data from Kaggle, was selected for the data analysis and tool validation. It is a heavily imbalanced and noisy dataset where the semi-supervised learning approach of the *autoencoder* tool fits.

The important number of previous studies on this dataset has helped in the writing of the preliminary analysis section, curation and several Data Management Plan (DMP) parts such as those regarding the measuring equipment or the data management conditions.

Curation code from the competition was modified and run to get a group of variables from each time series taking into account denoising and other pretreatment steps. This took a similar amount of time than the building of the scanner tool including the *autoencoder* model code and the complementary tools.

Two hyperparameter tunning cycles were performed after the building of the tool, though the limited training speed allowed the trial of only a small number of parameter combinations and It took more than 15 days of bayesian search to reach a relative answer.

Nevertheless, both the anomaly detection tool development and its validation have been performed, and they are available as a result of this work.

1. Introducción

El presente trabajo versa sobre el desarrollo de de una herramienta de detección de anomalías aplicando un *autoencoder,* y su validación en un conjunto de datos, teniendo en cuenta los pasos de un ciclo de vida de los datos que involucra un estudio riguroso. Se seleccionaron los datos de la competición *VSB Line Fault Detection* de *Kaggle*, que involucraban una tarea de aprendizaje semi-supervisado, etiquetados según la presencia de un tipo específico de anomalía, la Descarga Parcial (*Partial Discharge,* PD), en conductores aislados de media tensión de la red eléctrica de República Checa.

El código para la curación de la competición ha sido modificado y combinado con abordajes más recientes, para obtener un preprocesado de cada serie temporal, teniendo en cuenta la eliminación de ruido y otros aspectos. Esto supuso un tiempo similar a la construcción del escáner incluyendo el código del modelo de autoencoder y los complementos para su uso. Ambas, la herramienta de detección de anomalías y su validación, están disponibles como resultados.

El trabajo se tutorizó hasta septiembre de 2022, a través del Observatorio anual de trabajos académicos de la empresa HP SCDS. Esta, afincada en León y dedicada al desarrollo de firmware y software para impresoras 2D y 3D, deseaba contar con una **herramienta de análisis de datos que facilitase la identificación de anomalías en datos provenientes de sensores de equipos de impresión, con objeto de identificar de forma dinámica y automática la necesidad de suministros o de servicios de mantenimiento** predictivo. El tema se justifica por tres motivos: la abundante disponibilidad de datos en series temporales, por parte de la empresa interesada; la identificación de anomalías en ellos como finalidad del trabajo; y el interés concreto de la compañía, por desarrollar un *autoencoder ad hoc*, al ser estado del arte. De este modo, los objetivos del presente trabajo pueden resumirse en:

1. **Implementación de una herramienta** en Pytorch, basada en *autoencoders*, que sea aplicable a la detección de **anomalías en series temporales,** provenientes de datos de sensores, con las suficientes métricas por ciclo de entrenamiento (*epoch*), y los suficientes parámetros ajustables, para poder ser adaptado en función del problema de interés.
2. **Validación mediante la contextualización de la herramienta en el marco de una solución de datos global, sobre unos datos reales, explorando e incluyendo los elementos complementarios** asociados a las etapas del ciclo de vida de los datos de esa solución global, necesarias y razonables, como la curación, de acuerdo con los datos disponibles y el estado del arte**.**

Por motivos de protección de datos no se pudo disponer de datos concretos de la empresa, **enfocándose el trabajo con referencia a un conjunto de datos ajenos**, **vigilando que el uso de la herramienta fuese extrapolable en la máxima medida posible**. Para ello, en la búsqueda de datos se vigiló:

- Carácter de serie temporal.
- Carácter multidimensional que implique interacciones multivariables.
- Presencia de anomalías complejas (contextuales, colectivas y puntuales).
- Carácter estacionario, característico en series de datos asociadas a sensores.
- Dimensión importante, con los retos de escalabilidad de datos reales e interacc.

El conjunto de datos elegido consistió en una serie de señales medidas mediante dispositivos experimentales desarrollados por el *ENET Centre* de *VSB T.U.* de *Ostrava*, publicadas en *Kaggle*, en una competición iniciada en 2019, que pretendía identificar los patrones asociados a un tipo de fallo en conductores aislados de líneas de media tensión, antes referido como PD, que se asocia con el deterioro en el tiempo de este tipo de infraestructuras. Así, el trabajo surge a partir de la reutilización de los resultados de iteraciones anteriores del ciclo de vida de los datos general que refleja la figura 1, que estructuró sucesivos trabajos anteriores (Mišák et al 2019, Vantuch 2018…). Esta labor, que integra diversas investigaciones del grupo del *Centre ENET VSB-TU Ostrava*, parece orientada a hacer posible la implantación de soluciones *SMART GRID* relativa a la detección de fallos y averías en líneas de media tensión mediante una inversión económicamente rentable, basada en un dispositivo que pretende competir con aparatos de medida existentes como la bobina de Rogowski (Hashmi et. al. 2010), habiendo trabajado el grupo, tanto en el desarrollo de dispositivos de diagnóstico, como en el análisis de los datos derivados de ellos, pretendiendo respaldar la implantación sistemas de monitorización de fallos que favorezcan un mantenimiento predictivo de las líneas de distribución eléctrica.

Figura 1: Ciclo de vida de los datos.[1]

Habiendo reflejado aquí el interés y objetivos del trabajo realizado y su potencial aplicación, el documento se estructura en los subsiguientes apartados, siguiendo las etapas del ciclo de vida de los datos, y siempre con la intención de seguir los principios FAIR del conocimiento generado (*Findability*, *Accessibility*, *Interoperability*, y *Reuse*):

- *Planning*: cubre los apartados del *Data Management Plan (DMP),* y el análisis preliminar, además de la figura 23, explicando el alcance y tiempo invertido.
- *Collecting, Processing and Analysing*: se continúa con un apartado dedicado a la curación de los datos y otro referido al diseño, desarrollo y validación de la herramienta, que concentran entre ambos, la mayor parte del esfuerzo práctico.
- *Sharing*, *Preserving*, *and Reusing*: Se incluye un apartado de metadatos y preservación, incluyendo el DMP más detalles. Los resultados son el presente trabajo escrito, el código y la documentación para su ejecución. Las conclusiones y limitaciones, se relacionan también con la reutilización y mejora.

[1] Economic and Data Research Council (2019).

3. Data Management Plan (DMP)

General Information: Administrative Information

Creators: Javier Alejandro Cuartas Micieces y Diego García Saiz.

Affiliation: Universidad de Cantabria.

Funder: Universidad de Cantabria.

Template: H2020 European Commission DMP.

Project abstract: Ver apartados: Resumen en Español y Resumen en Inglés.

Start Date: 09/12/2022.

End Date: 01/06/2023.

Copyright Information: *Creative Commons Licence* (CC), Reconocimiento (*by*).

2.1 Data description and collection or re-use of existing data

1a How will new data be collected or produced and/or how will existing data be re-used?

Tradicionalmente, la identificación del fenómeno de PD a través de señales de voltaje o intensidad se ha producido durante la interrupción del suministro, con equipos especializados y por personal experto, con inconvenientes como la abundancia de ruido, ausencia de sensores no intrusivos, o las limitaciones de detectar descargas en bajas frecuencias que perjudica la localización espacial de los fallos, frente a altas frecuencias, cuando por ejemplo, tanto Hashmi (2010) como Mišák y Pokorný (2014) discuten la influencia de la distancia del fallo, como determinante en la utilidad en campo de los instrumentos de medida. La detección sobre redes en funcionamiento tiene, además, ventajas como la no interrupción del suministro o la posibilidad de detectar anomalías temporales, aunque imponga retos como el mayor ruido de fondo, sincronización de los aparatos de medida entre subestaciones para la localización de los fallos, demandas técnicas de ancho de banda, o de sensibilidad.

La fase de recogida, limpieza y etiquetado de los datos crudos va más allá del alcance del presente trabajo. El grupo de investigación que lo abordó recogió señales de voltaje sobre líneas aéreas de conductores aislados de media tensión para la detección de anomalías de PD en varias localizaciones de la República Checa, con medidores experimentales resultantes de sus trabajos anteriores. Los instrumentos más habituales de medida se refieren en la tabla 1, y se basan en fenómenos que acompañan la PD: la corriente eléctrica, la radiación emitida por partículas excitadas, los sonidos ultrasónicos, el calor generado o las reacciones químicas producidas. Zhang et. al. (2021), distingue 3 tipos de sensores para detectarla a partir de fenómenos eléctricos: los basados en acoplamiento inductivo (campo magnético), en acoplamiento galvánico (caída de tensión en una resistencia intercalada en el conductor) y en acoplamiento capacitivo (campo eléctrico). El dispositivo asociado a los datos de este trabajo es del primer tipo.

En 2011, el grupo de Mišák et al., en una de las primeras versiones, combinaron un ordenador industrial ENA440 junto a una tarjeta de medida osciloscópica NI PCI5102, además de GSM y otros sensores de temperatura, humedad… leyendo las salidas a través del programa LabVIEW2009. Más adelante, en el artículo de 2019 relacionado

con los datos aquí utilizados, Mišák et al. describen un sistema que no detallan tanto, haciendo referencia en términos genéricos a una DAQ (*Data Adquisition Device*) que sería análoga al ordenador industrial del sistema de 2011, en este caso con un módulo de comunicación GSM y su sensor particular, asociado el conjunto a cada RTU o subestación (*Remote Terminal Unit*).

En Mišák y Pokorný en 2014, se describen experimentos sobre dicho elemento de medida propio, que comparan con la bobina de Rogowski, sugerida por ellos como menos eficiente en coste, a pesar de ser la solución más ampliamente utilizada. Esta bobina de Rogowski es un toroide que rodea un núcleo rígido hueco, no magnético, dispuesto entorno al conductor en el que se desea medir, que consta de dos bucles de cable conectados con polaridades opuestas, para prevenir la influencia de corrientes intensas en conductores próximos, y fijar la inducción mutua y la autoinducción. Permite ajustar el balance entre sensibilidad y ancho de banda de frecuencia cubierto mediante los parámetros que influyen sobre la impedancia (Z_{out}), como los diámetros (d_1, d_2, y d_{rc}) o el número de espiras, dado que cuando aquella disminuye, disminuye la sensibilidad, pero aumenta el ancho de banda (Hashmi GM, 2010).

Tipo de Sensor	Ventajas	Desventajas
AE (*Acoustic Emission*).	Inmune al ruido electromagnético.	Alta atenuación.
HFCT (High Frecuency Current Transformer).	Instalación no intrusiva y ancho de banda amplio.	Saturación del material por grandes corrientes. Se necesita lazo de corriente.
Bobina de Rogowski.	**Ligeros y de bajo coste comparados con los HFCT.**	**Pequeña banda de frecuencia. Se necesita lazo de corriente.**
Condensador de acomplamiento.	Gran sensibilidad, posibilidad de integrarlo en el cable.	El coste y tamaño puede ser problemático sobre la red.
UHF (*Ultra High Frecuency*)	Buen rendimiento anti-perturbaciones.	Fuerte atenuación. No puede ser calibrado. Efecto de blindaje electrostático.
Imágenes Ultravioletas.	Fácil de utilizar.	Sólo puede detectar efecto corona en los terminales del cable.

Tabla 1: Sensores habituales de PD, ventajas y desventajas.[2]

El aparato experimental del que se obtienen los datos (Misák y Pokorný., 2019), es un elemento de medida también basado en la ley de la inducción electromagnética de Faraday. Por un lado, cuenta con un elemento formado por un SLI (*Single Layer Inductor,* L_1 en la figura 2), por fase, conectado cada uno a un divisor de capacitancia formado por 2 condensadores (*C Divider* o CD, en inglés), que reduce la amplitud de la señal de voltaje medida, actuando la resistencia (R1 en la figura 2), como filtro *Low Pass* cuando el divisor se encuentra cargado. Por otro lado, la lectura de voltaje se obtiene midiendo capacidad de acoplamiento del elemento anterior (C_{coup}), frente a un elemento circular metálico (C_{coil} en la figura) acoplado al conductor medido.

[2] Zhang et. al. 2021

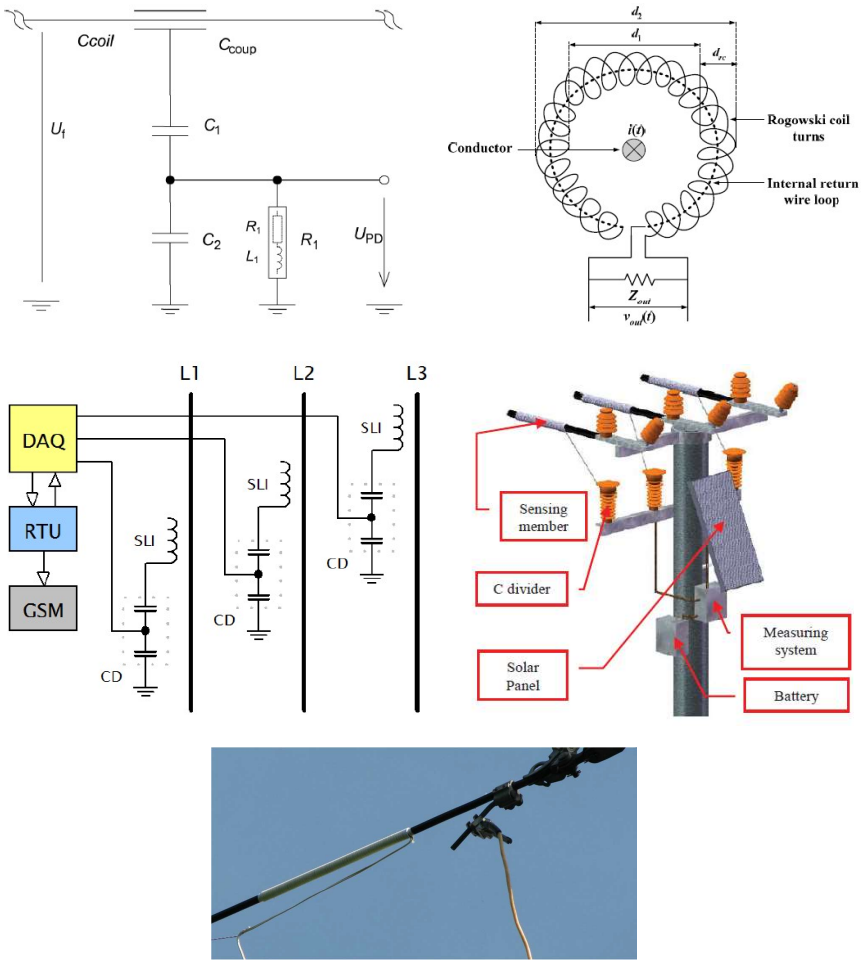

Figura 2: Construcción del elemento de medida basado en el SLI y el CD utilizado, construcción de una bobina de Rogowski y esquema de montaje del aparato de medida.[3]

El etiquetado de las lecturas como anómalas o no, por PD, fue realizado por un especialista, viajando el equipo in situ para comprobar cada caso, encontrándose falsos positivos debidos al mal funcionamiento del medidor o a interferencias por ruido de fondo (*EBN*), la mayor fuente de incertidumbre en la clasificación.

Los datos de señal de voltaje analizados son datos abiertos y provienen de la competición del *ENET Centre* de *VSB T.U.* de *Ostrava* publicada en *Kaggle* encontrándose recogidos en dos ficheros: uno de unos 4GB de tamaño en formato *parquet*, que contiene una serie temporal por cada fase, con 800.000 observaciones puntuales para cada una; y un segundo fichero en formato *csv*, que recoge las etiquetas para cada fase (2.904 grupos de 3 series, es decir, 8.712 series temporales). La competición incluye otros 9GB adicionales de datos, de *test,* no etiquetados, para aplicar sobre ellos los métodos entrenados sobre los datos antes mencionados y poder comparar los resultados con otras soluciones de otros competidores.

[3] Hashmi et al. 2010, Misàk y Pokorný 2014, Misàk et al. 2019 y Misàk et al. 2011.

Además de estos datos, se ha utilizado el código proveniente sobre todo, de la solución ganadora en la competición, publicada también en *Kaggle*, y según las normas de la competición con licencia de código abierto, para curar estos datos combinando la solución con ideas de trabajos relacionados, y obtener un conjunto reducido de indicadores relacionados, con el número y altura de los picos presentes en cada serie temporal, que son almacenados en un fichero en formato *h5py*, con el fin de ser posteriormente utilizados por la herramienta de detección de anomalías.

Asimismo, para el desarrollo del *autoencoder* se ha utilizado código proveniente de la documentación oficial de Pytorch (2022).

1b What data (for example the kind, formats, and volumes), will be collected or produced?

Los datos descargados a través de la competición de *Kaggle* son los siguientes, como se puede observar, la mayor parte son índices de tipo entero de 8 bytes que vinculan e identifican las señales, y representan una señal de voltaje medida durante 20ms, completando un ciclo de línea trifásica con una frecuencia de 50Hz.

- *metadata_test.csv y metadata_train.csv*: estos ficheros, como podemos ver en la siguiente tabla 2, en una representación del contenido del segundo de los mismos, contienen índices *signal_id* que asocian el contenido de cada una de las series temporales de 800.000 mediciones recogidas en los ficheros *test.parquet* y *train.parquet*, con los *id_measurement* que identifican cada terna de fases, distinguidas entre sí mediante la columna *phase,* y con el *target* que representa la presencia o no (1 ó 0 respectivamente), de patrones de Descarga Parcial (PD, el tipo de anomalía a detectar), en cada una de las señales.

signal_id	*Id_measurement*	*phase*	*target*
0	0	0	1
1	0	1	0
2	0	2	1
…	…	…	…
8.709	2.903	0	0
8.710	2.903	1	0
8.711	2.903	2	0

Tabla 2: Croquis de los ficheros *metadata_test.csv y metadata_train.csv.*

- *sample_submission.csv*: es un ejemplo de fichero de resultados de la aplicación de un hipotético algoritmo entrenado para detectar la descarga parcial y aplicado al conjunto de *test,* generando una columna *target* que indica la clasificación mediante 1 ó 0, y otra columna *signal_id,* que identifica unívocamente la serie de datos, como se ve en la tabla 3.

signal_id	*target*
8.712	
…	…
29.048	

Tabla 3: Croquis del fichero *sample_submission.csv.*

- *test.parquet y train.parquet*: en la tabla 4 se puede ver un ejemplo de representación del contenido de *train.parquet*, aunque la única diferencia para el caso del *test.parquet*, correspondería a que los índices de columna debiesen estar etiquetados desde el 8.712 al 29.048. Este fichero refiere los valores de la señal medida para cada una de las 8.712 series temporales de 800.000 observaciones cada una.

		signal_id		
		0	…	8.711
Observaciones temporales	0		…	
	…		…	
	800.000		…	

Tabla 4: Croquis de los ficheros *test.parquet y train.parquet.*

Por otro lado, los resultados derivados del trabajo se encuentran recogidos en los siguientes ficheros, compartiéndose la mayoría en los repositorios establecidos:

- *DeepScan4Failure.docx* y *DeepScan4Failure.pdf*, presente documento.
- Código del proyecto:
  - *DSF4-Pretreatment_example.ipynb* con las funciones de pretratamiento de los datos y su explicación.
  - *DSF4-Hyperparameter_tunning_example.ipynb* con un ejemplo de aplicación al problema de análisis aquí referido.
  - Módulo recogido en la carpeta /src/*ds4f*, con los ficheros *__init__.py*, *model.py*, *utils.py* y *ds4f.py*.
  - Ficheros *train.py* y *predict.py*, que permiten entrenar y utilizar los escáneres, eje del módulo en */src*, a través de *bash,* línea de comandos.
  - *README.md*
  - *LICENSE*
  - *requirements.txt*
- Datos obtenidos de la ejecución de los 2 notebook anteriores, no compartidos por su volumen y sencilla replicación a partir del código:

- *0-1500r_train_stnw.h5, 1500-3000r_train_stnw.h5, hasta 7500-8712r_train_stnw.h5* recogen conjuntos de variables preprocesadas descritos más adelante, a partir de ternas de señales originales, con una observación o señal (conjunto de variables) por cada *dataset*.
- *curatedset.h5*, a partir de la agregación de los ficheros del punto anterior, sigue la misma estructura con los datos de una señal por cada *dataset*.
- *scaled.h5* fichero resultante de la aplicación del escalado mediante el mínimo y máximo de cada variable, tomando como partida *curatedset.h5*.

2.2 Documentation and data quality

2a What metadata and documentation (for example the methodology of data collection and way of organising data) will accompany the data?

Los datos están actualmente disponibles a través de *Kaggle API* con el formato de metadatos correspondiente a la misma, permitiendo interaccionar con ella mediante código. Se ha utilizado también para la limpieza y pretratamiento de los datos, código de participantes de la competición que motivó la liberación de dichos datos, cuyos metadatos se encuentran estructurados según la misma API, pero que no se especifican aquí, aunque sean debidamente mencionados y especificados en el código derivado y en la bibliografía, encontrándose disponibles de forma abierta y sin necesidad de credencial alguna a través de la web de *Kaggle,* en el momento de la redacción de este trabajo.

Además, durante el desarrollo del proyecto, se han utilizado los nombres de los ficheros resultantes de algunas fases del proceso, como en el caso de la salida de las funciones de pretratamiento de los datos de entrada, como se señala en el apartado anterior 1b, contiene el código que identifica cada señal procesada, con objeto de que puedan ser agregadas correctamente en el fichero *curatedset.h5* y posteriormente *scaled.h5.*

En lo relativo al código resultante de la realización de este trabajo, se almacenará en un repositorio privado de Gitlab perteneciente a HP-SCDS acompañado de los metadatos recogidos según el esquema de la API de *Gitlab*, garantizando la accesibilidad e interoperabilidad por vía electrónica de los mismos, como refleja la figura 3.

Por otro lado, los metadatos del texto del proyecto en el repositorio Ucrea de la Universidad de Cantabria (2023), cumplen con diversos estándares y recomendaciones técnicas, como las de OAI-PMH, OpenAire… incluido Dublin Core, que es el vocabulario que se utiliza a continuación en un ejemplo orientativo para el texto a publicar, y que garantiza el cumplimiento de los principios FAIR en cuanto a la accesibilidad e interoperabilidad por vía electrónica de todos los proyectos de la Universidad:

- **dc.contributor.advisor:** García Saiz, Diego.
- **dc.contributor.author:** Cuartas Micieces, Javier Alejandro.
- **dc.contributor.other:** Universidad de Cantabria.
- **dc.date.accessioned:** Unknown.

- **dc.date.available:** Unknown.
- **dc.date.issued:** Unknown.
- **dc.identifier.uri:** Unknown.
- **dc.description.abstract:** Ver Resumen en español y Resumen en inglés.
- **dc.format.extent:** Unknown.
- **dc.language.iso:** spa
- **dc.rights:** Atribución España
- **dc.rights.uri:** https://creativecommons.org/licenses/by/4.0/
- **dc.subject.other:** Aprendizaje automático.
- **dc.subject.other:** Machine Learning.
- **dc.subject.other:** Anomaly Detection.
- **dc.subject.other:** *Autoencoder*.
- **dc.title:** DeepScan4Failure.
- **dc.type**: info:eu-repo/semantics/masterThesis.
- **dc.rights.accessRights**: embargoedAccess.
- **dc.description.degree**: Máster en Ciencia de Datos.

```
import gitlab

# Set the GitLab instance URL and your personal access token
gitlab_url = 'https://gitlab.com/'
access_token = '********************'

# Create a GitLab API client instance
gl = gitlab.Gitlab(gitlab_url, private_token=access_token)

# Authenticate with the GitLab API
gl.auth()
```

```
project_id = '30017802'
project = gl.projects.get(project_id)
project.attributes
```

```
{'id': 30017802,
 'description': 'Detección de comportamiento anómalo en impresoras HP mediante DL',
 'name': 'UC-DeepScan4Failure',
 'name_with_namespace': 'HP-SCDS / Observatorio / 2021-2022 / DeepScan4Failure / UC
 'path': 'uc-deepscan4failure',
 'path_with_namespace': 'HP-SCDS/Observatorio/2021-2022/deepscan4failure/uc-deepsca
 'created_at': '2021-09-29T19:14:47.780Z',
 'default_branch': 'main',
 'tag_list': [],
 'topics': [],
 'ssh_url_to_repo': 'git@gitlab.com:HP-SCDS/Observatorio/2021-2022/deepscan4failure
 'http_url_to_repo': 'https://gitlab.com/HP-SCDS/Observatorio/2021-2022/deepscan4fa
 'web_url': 'https://gitlab.com/HP-SCDS/Observatorio/2021-2022/deepscan4failure/uc-
 'readme_url': 'https://gitlab.com/HP-SCDS/Observatorio/2021-2022/deepscan4failure/
EADME.md',
 'forks_count': 0,
 'avatar_url': None,
 'star_count': 0,
 'last_activity_at': '2022-06-09T09:54:04.110Z',
```

Figura 3. Metadatos del proyecto accesibles a través de la API de Gitlab.

2b What data quality control measures will be used?

En cuanto a los datos crudos, al ser datos y código ya publicados, provenientes o utilizados por estudios académicos previos en medios relevantes, se ve respaldada su calidad en cuanto a la validación del método de detección de anomalías desarrollado. En lo relativo al presente documento, como trabajo para la obtención de un título académico oficial, será sometido junto con el código, a un tribunal de evaluación.

2.3 Storage and backup during the research process

3a How will data and metadata be stored and backed up during the research?

Al tratarse de datos abiertos no se prevén medidas de protección especial ni para el código en desarrollo ni para los datos utilizados tanto crudos como curados. Se ha trabajado parcialmente en Google Colab, como punto de partida recomendado por los tutores, además de en el ordenador personal del autor. No se prevé la publicación de los resultados del trabajo en otros repositorios distintos de la cuenta de Gitlab del propio autor para el código y la plataforma Ucrea de la Universidad de Cantabria para tanto el código como el trabajo escrito. Se han realizado copias semanales de seguridad en el disco local del autor y en un disco duro externo, durante el desarrollo del trabajo, para mitigar las consecuencias de incidentes indeseados.

3b How will data security and protection of sensitive data be taken care of during the research?

Al tratarse de datos abiertos, no se prevén medidas de protección especial para los mismos. En cuanto al código, contará con licencia MIT, también abierta, y se publicará con acceso público desde la cuenta en Gitlab del propio autor, además de en Ucrea. En cuanto al trabajo escrito, contará con las garantías propias de la plataforma Ucrea de la Universidad de Cantabria para el alojamiento de trabajos académicos.

2.4 Legal and ethical requirements, codes of conduct

4a If personal data are processed, how will compliance with legislation on personal data and security be ensured?

No serán procesados datos de tipo personal ni sujetos a ningún tipo de protección.

4b How will other legal issues, such as intellectual property rights and ownership, be managed? What legislation is applicable?

Los derechos de los datos pertenecen al VSB Enet Centre y la Technical University of Ostrava. De acuerdo con las reglas de la competición publicada en *Kaggle*, fuente de los datos, se puede disponer de ellos libremente con el fin de investigaciones académicas, con fines educativos y no comerciales, siendo el acceso libre.

En cuanto al código utilizado proveniente de la competición, las reglas de la misma requieren que sea subido con licencia Open Source sin límites de uso comercial, y lo mismo se requiere de la solución ganadora.

En lo relativo a los resultados del presente trabajo, el código contará con licencia MIT, abierta, y se publicará en Ucrea y con acceso público desde la cuenta en Gitlab del propio autor. En cuanto al trabajo escrito, contará con las garantías propias de la plataforma Ucrea de la Universidad de Cantabria para el alojamiento de trabajos

académicos, previéndose para el mismo una *Creative Commons License* abierta, limitada al reconocimiento.

4c What ethical issues and codes of conduct are there, and how will they be taken into account?

Los principales problemas éticos a tener en cuenta respecto de los datos utilizados son los relacionados con la propiedad intelectual de los datos y del código utilizados, así como del trabajo derivado de ellos. El principal código de conducta, son las reglas de la competición y según estas, cualquier reclamación surgida de o relacionada con ellas, estará sujeta a las leyes de la República Checa, salvo por conflicto de leyes.

En lo relativo a la protección intelectual de los resultados del presente trabajo, la normativa de protección de la propiedad intelectual en España va desde el Código Civil y Ley Orgánica del Código Penal, hasta normativas sobre el funcionamiento, comunicación… de la Comisión de la Propiedad Intelectual como organismo relacionado, aunque la que detalla mejor el alcance de los derechos de autor aplicables a trabajos académicos y lo que denominan como "programas de ordenador", de aplicación para nuestro caso, probablemente sea la Ley de Protección Intelectual cuyo texto refundido fue aprobado por el Real Decreto Legislativo 1/1996 y el Reglamento para la ejecución de la Ley de 10 de enero de 1879 sobre propiedad intelectual en lo que no contravenga la anteriormente mencionada ley de 1996, de acuerdo con la información de la Agencia Estatal Boletín Oficial del Estado (2022).

2.5 Data sharing and long-term preservation

5a How and when will data be shared? Are there possible restrictions to data sharing or embargo reasons?

En cuanto al código, contará con licencia MIT, abierta, con permiso para la reutilización incluyendo fines comerciales, exigiéndose el reconocimiento, y se publicará con acceso público desde la cuenta en Gitlab del propio autor, además de en Ucrea, compartiéndose también en un repositorio de Gitlab de acceso restringido del observatorio HP SDCS, que fue primero en recoger la versión definitiva del código, aunque sólo estuviese involucrado en la tutorización del trabajo hasta septiembre del 2022. En cuanto al trabajo escrito, contará con las garantías propias de la plataforma Ucrea de la Universidad de Cantabria para el alojamiento de trabajos académicos, con una licencia abierta.

5b How will data for preservation be selected, and where data will be preserved long-term (for example a data repository or archive)?

No se almacenarán datos, pero el código podrá replicar todas las fases del ciclo de vida de los datos desarrolladas en el presente trabajo.

5c What methods or software tools are needed to access and use data?

En cuanto a reutilización de los datos crudos (fuentes utilizadas), puede utilizarse cualquier método que permita interaccionar con ficheros en formato *csv* y *parquet,* incluyendo la *Kaggle API* que permite descargarlos directamente mediante Python u otros lenguajes con soporte. En el caso de la lectura de los ficheros *csv*, la variedad de soluciones de software para acceder es amplia, y va desde Microsoft Excel, librerías habituales de Python como Pandas… En el segundo caso, la solución utilizada por el tamaño del archivo *train.parquet*, ha sido Python con su librería Pandas, habiéndose requerido al trabajar en el entorno local de un equipo propio, instalar otras librerías complementarias como *Pyarrow* o *Fastparquet*, a pesar de incluirse con el propio Pandas. En este caso, se limita la carga de datos a unos cuantos registros cada vez

(series temporales de 800.000 observaciones) por el importante tamaño. Si se desea trabajar con todo el conjunto de datos de una sola vez, también se ha probado a utilizar con éxito *Pyspark*, que orientó el desarrollo durante los tres primeros meses hasta que se sugirió la solución de Pandas, más sencilla, eficiente y adecuada al problema.

En lo relativo al resultado del presente trabajo, el documento podrá accederse a través de la interfaz web de Ucrea y podrá accederse a los metadatos que carguen los responsables de la Universidad de Cantabria, que asignan a cada trabajo un URI (*Uniform Resource Identifier*) que lo identifica, a través de un intérprete de Python u otros lenguajes que provean librerías para explorar metadatos o herramientas diseñadas para tal fin. Igualmente, el código se almacenará en Ucrea y contará con acceso público desde la cuenta de Gitlab del propio autor, indexable mediante sus propios metadatos asociados a la API de Gitlab como se ve en la figura 3.

2.6 Data management responsibilities and resources

6a Who (for example role, position, and institution) will be responsible for data management (i.e. the data steward)?

No se trata de una figura requerida en base al carácter que los confiere la documentación de la competición.

6b What resources (for example financial and time) will be dedicated to data management and ensuring that data will be FAIR (Findable, Accessible, Interoperable, Re-usable)?

Reiterando lo anteriormente comentado, el documento podrá accederse a través de la interfaz web de Ucrea y podrá accederse a los metadatos que carguen los responsables de la Universidad de Cantabria, que asignan a cada trabajo un URI (Uniform Resource Identifier) que lo identifica y que el código contará con acceso desde la cuenta en Gitlab del propio autor de forma indexada mediante sus propios metadatos asociados a la API de Gitlab, con lo que se considera que los recursos resultado del presente trabajo son encontrables y accesibles de acuerdo a los principios FAIR.

Se ha dedicado el tiempo necesario una vez terminada la implementación, a documentar el proyecto adecuadamente para hacerlo accesible en Ucrea y a través de la cuenta en Gitlab del propio autor, añadiendo el fichero *requirements.txt* asociado a los paquetes necesarios, licencias y detalles de funcionamiento asociados, que garantizan los principios de interoperabilidad y de reusabilidad, siendo también en parte garante la *Gitlab API*, puesta a disposición por dicha plataforma y el respeto del repositorio Ucrea por la interoperabilidad en los datos y metadatos derivados de los trabajos como el presente.

Más allá de todo esto, es digno mencionar que no está prevista la publicación de los datos crudos de origen junto con el código, ni tampoco los resultantes del preprocesado, por el importante espacio en disco requerido.

3. Análisis Preliminar

En primer lugar, la literatura básica define anomalía como *"Una observación que se desvía tanto de las otras observaciones como para despertar la sospecha de que fue generada por un mecanismo diferente"* (Hawkins 1980, citado por Hu-Sheng 2016), aunque quizás Cook (2019), ilustre mejor el concepto en el presente ámbito del IoT como *"las consecuencias medibles de un cambio inesperado en el estado de un sistema que se encuentra fuera de su norma global o local"*. Los datos (consecuencias medibles) en este caso son una colección de series temporales, es decir, secuencias temporales de registros ordenables cronológicamente, multivariables, cuya granularidad temporal es la misma y se corresponde en una relación uno a uno; el sistema sería la red de de distribución eléctrica mediante conductores aislados de media tensión (CC *Covered Conductors*); y el cambio inesperado, la Descarga Parcial (PD), son "*descargas localizadas que puentean sólo parcialmente el aislamiento entre conductores*" (IEC 60270), causadas por campos eléctricos fuertes de diversa naturaleza y que pueden derivarse de defectos en el aislamiento, contactos con ramas… En la señal son anomalías colectivas que detectar en el seno del conjunto de secuencias disponible.

Existen varios tipos de descarga parcial (Zhang et. al. 2021), aunque en nuestro caso se refiere a los dos, asociados a la referencia de Mišák y Pokorný (2014), al contacto de árboles o ramas sobre el conductor, o a la caída del propio conductor al suelo (ver figura 4):

- Efecto corona, debido a la ionización del aire en torno a un electrodo, se discute que no está tan relacionada con los conductores como con sus terminales.
- Descargas superficiales, ocurren en la interfaz entre dos materiales, en la superficie de un material dieléctrico. En los conductores a menudo se utilizan varias capas de estos materiales y es la principal causa de fallo.
- Descargas internas, generadas en el interior de dieléctricos debilitados por contaminación, rotura, huecos, contacto con ramas…

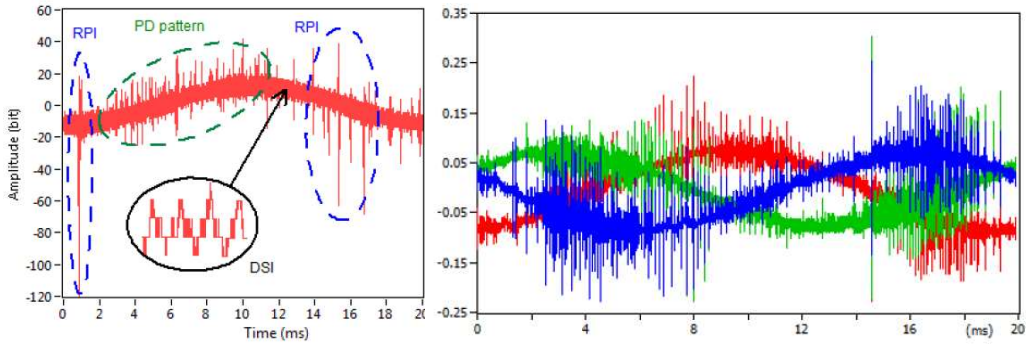

Figura 4. Tipos de ruido en la señal de una línea trifásica y patrón de PD en series temporales.[4]

Misák et. al 2017, resumen citando a Zhang et. al. 2007, las fuentes del ruido en la señal de voltaje que acompañan sus datos, de igual naturaleza, aunque menor resolución que los de este trabajo (400.000 observaciones por señal):

[4] Misák et. al 2017.

- *Discrete Spectral Interference* (DSI) debida a emisiones de radio. Las fuentes de DSI pueden reconocerse mediante una FFT (*Fast Fourier Transform*), basándose en la modulación, aunque durante el día su señal es variable, durante la noche son más significativas por el efecto de la ionosfera y pueden ser muy abundantes en las señales obtenidas (aunque aquí no contamos con los horarios de las mediciones).
- *Repetitive Pulses Interference*, debida a la electrónica de potencia cercana. En el caso estudiado en Misák et. al 2017 y Vantuch 2018, era fuente de pulsos de ruido independiente debidos a la emisora jde radio *Solec Kujawski*.
- *Random Pulses Interference* (RPI), debida a rayos, efecto corona…
- Ruido ambiente y amplificaciones.

El ruido parece algo mayor en las señales con fallo, pero es difícil la identificación a partir de la visualización como TRPD (*Time Resolved Partial Discharge*, es decir, el tiempo como eje de abscisas), como se puede ver en la figura 5, lo que se evidencia en la necesidad del equipo de recogida, de acudir al campo para confirmar los fallos de etiquetado por falsos positivos.

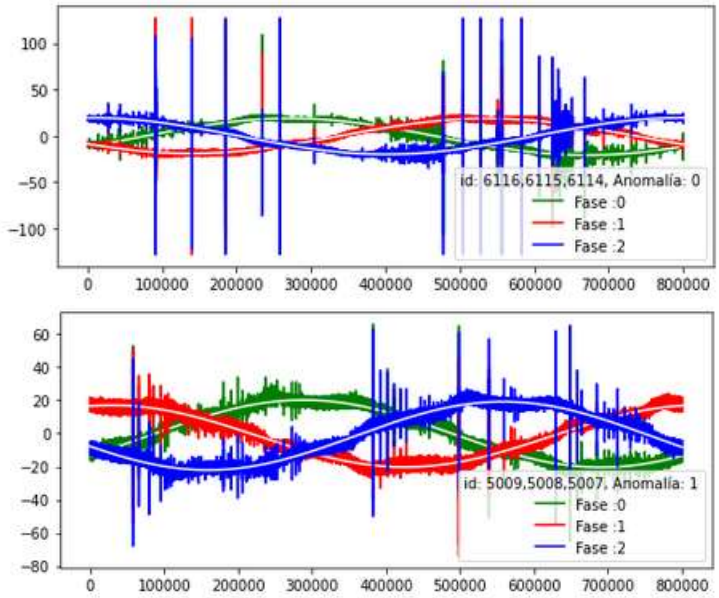

Figura 5. Ejemplo del parecido visual entre numerosas señales con PD y sin PD.

La PD aparece fundamentalmente con una longitud de onda en la escala de los microsegundos (aquí las señales abarcan 20ms), requiriendo datos de alta frecuencia de varias decenas de MHz (frecuencia de muestreo 40MHz), ocurriendo sólo unos cuantos pulsos por periodo de la frecuencia de suministro de 50Hz.

Por otro lado, es digno mencionar la existencia de 8 grupos de 3 fases cada uno, sin PD, que parecen proceder de sensores averiados por su carácter plano no senoidal, pero al ser clasificados como no anómalos, son fáciles de detectar utilizando un umbral sobre la desviación típica de los valores de la serie que puede ser implementado independientemente, han sido eliminadas del conjunto de entrenamiento para evitar introducir sesgos indeseados en el modelo (ver muestra en la figura 6).

Volviendo sobre el marco del trabajo, Cook (2019) hace mención del concepto de *Smart Grid*, como concepto al que deben servir herramientas como la que estructura este trabajo. Además, en la publicación se mencionan los elementos fundamentales de los datos IoT, a considerar en caso de pretender validar su utilidad en el marco del IoT.

Estos elementos son cinco, empezando por la información contextual de los datos, mencionando la dimensión espacial y externa que aquí no ha sido posible conseguir. Sí se ha tenido en cuenta un problema multidimensional y con una rica información contextual de tipo temporal, con 3 conductores que aportan cada uno, una serie temporal o dimensión distinta. Otro elemento de los cinco es el ruido, que se aborda en el próximo apartado, de curación. Por último, nos encontramos con el carácter estacionario de las señales, que se define como el estado en el que la media, varianza y autocorrelación de la señal no varía con el tiempo, y cuya ausencia invalida numerosos métodos de identificación de anomalías. Aquí, como se ve en las figuras 7 y 8, para una de las señales, el filtro aplicado tiene efecto en su corrección.

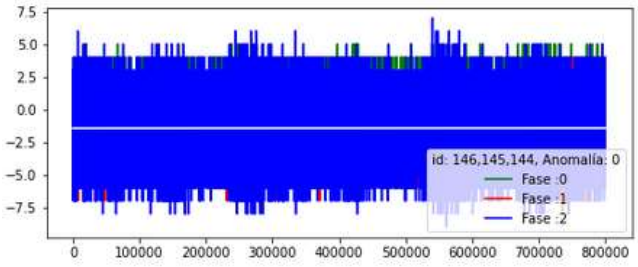

Figura 6. Ejemplo de señal sin PD, de carácter plano no senoidal, que se ha valorado eliminar.

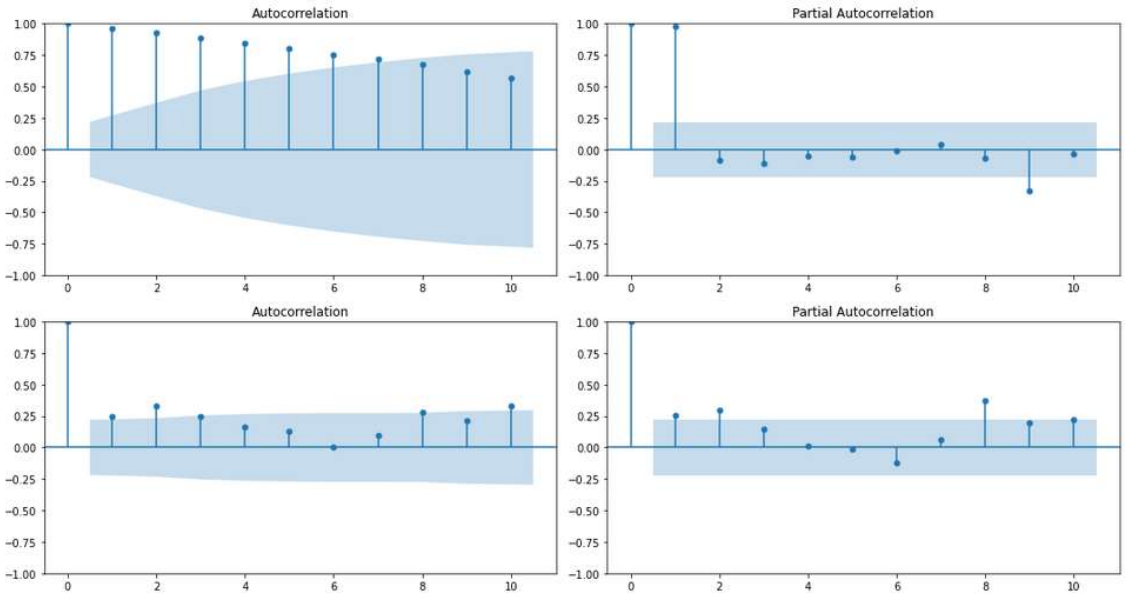

Figura 7. Autocorrelación y autocorrelación parcial de muestra cada 10.000 observaciones de la señal 0, antes (arriba) y después de la aplicación del filtro de *Savitzky-Golay* de ventana 4 y orden 3 (abajo), con 10 lags, intervalo de confianza del 95%.

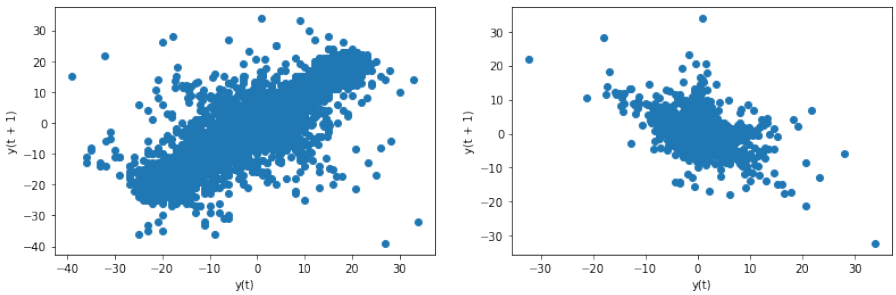

Figura 8. Ejemplo de lag plot de la señal 0, antes (izquierda) y después de la aplicación del filtro de filtro de *Savitzky-Golay* de ventana 99 y orden 3 (derecha).

4. Curación

El proceso de curación de los datos disponibles para todas las publicaciones revisadas basadas en ellos es en términos genéricos, el mismo que se recoge ya por Vantuch (2018) y la publicación asociada de Mišák et al. 2017, en el primer abordaje del problema, aunque variando los métodos empleados en cada fase. De hecho, los objetivos de este primer trabajo estaban mayoritariamente orientados a la curación y la ingeniería de características ha demostrado constituir el principal elemento de influencia en los resultados de las soluciones propuestas en la competición y en los estudios posteriores. En el presente análisis se ha realizado un proceso de curación intermedio entre los enfoques abordados por los dos estudios con más éxito en la identificación de anomalías, Elmastry y Waldy (2022) y Chen et al. (2021), que se justifica en la tabla resumen número 5 y en el ranking de la tabla 6.

	Fases	**Abordaje**	**Resultado**
(Mišák et al. 2017 y Vantuch 2018)	Filtrado de ruido y eliminación de extremos	Sincronización del seno, *thresholding* de bajas frecuencias con DWT *(Discrete Wave Transform* con *db4* y otros *wavelets* probados*)* y filtro de *Butterworth.* Eliminación de pulsos simétricos y extremos	Herramientas de análisis disponibles y extracción de características
	Extracción de características	Sintetización mediante regresión simbólica, *autoencoders*… ó extracción de pulsos calculando su ancho, altura… Después se puede proceder con la reducción de dimensionalidad (PCA, SVD…) y distintos posibles empleos del *clustering,* antes de alimentar el clasificador.	
	Modelo de clasificación/otros	ANN, SVM… y optimización de hiperparámetros GSO, PSO, SOMA …	
	Fases	**Abordaje**	**Resultado**
(Chen et. al 2021)	Filtrado de ruido y eliminación de extremos	Sincronización del seno mediante DFT. Filtro de *Savitzky-Golay* para eliminar bajas frecuencias (seno)	Identificación de PD **(MCC: 0.77)**
	Extracción de características	Extracción de pulsos no demasiado concentrados, verificando que son máximos en su entorno, el nivel de ruido… Después, tras el *clustering* del entorno de los pulsos, se obtiene su número, media y desviación típica de la altura, así como el *Root Mean Squared Error* (RMSE) respecto de plantillas resultado del *clustering* de entrenamiento, unas 165 variables.	
	Modelo de clasificación/otros	lightGBM	

	Fases	Abordaje	Resultado
(Michau et al. 2021)	Filtrado de ruido y eliminación de extremos	Mediante un filtro de *Butterworth* se eliminan bajas frecuencias, eliminando ruido y ayudando a extraer los pulsos.	Identificación de PD y extracción de características **(MCC: 0.78)**
	Extracción de características y modelo de clasificación/otros	Una red neuronal convolucional de dimensión 1, con 2 bloques de 2 capas convolucionales y una *Max Pooling* cada uno, con una *Global Average Pooling* (GAP) de regularización final, que también permite obtener el PAM (*Pulse Activation Map*), terminando con una capa *Fully Connected* y un clasificador binario (STL+SVM ó LSTM).	
	Fases	**Abordaje**	**Resultado**
(Elmasry y Waldy, 2022)	Filtrado de ruido y eliminación de extremos	Eliminación de extremos por encima y debajo de 3 veces el IQR (*Interquartile Range*). División en trozos.	Identificación de PD **(MCC: 0.90)**
	Extracción de características	Diversos estadísticos del valor de la señal (media, desviación típica, mínimo, máximo, 5 percentiles, altura, 7 percentiles relativos y límites superior e inferior de cada trozo).	
	Modelo de clasificación/otros	Se normalizan los datos y se aplica PCA y OC-SVM con PSO para optimización	

Tabla 5: Abordaje de la curación y objetivos detallados en publicaciones anteriores, a partir de los mismos datos.[5]

Como uno de los ejes de la curación, la publicación de Cook (2019) menciona tres motivos fundamentales de la no estacionariedad de series temporales, que es necesario corregir:

- La deriva de concepto: cambio de distribución estadística con el tiempo.
- La estacionalidad: caso especial del anterior, con una periodicidad menor que la frecuencia de muestreo.
- Puntos de cambio: modificaciones permanentes del estado normal monitoreado.

La curación incluirá en el apartado, medidas para subsanar al menos los dos primeros motivos anteriores. Por ejemplo, la estacionalidad marcada por la frecuencia de la red de suministro se aborda mediante un filtro; o la presencia de distintas fuentes de ruido antes mencionadas, como RPI, que generan picos de voltaje, se filtran mediante un umbral estadístico para subsanar los casos más exagerados, pudiéndose cubrir la tercera mediante reglas programadas en la aplicación práctica del algoritmo, aunque

[5] Mišák et al. (2017), Vantuch (2018), Chen et al. (2021), Michau et al. (2021) y Elmasry y Waldy (2022).

excede el ámbito del trabajo al ser conveniente una adicional recogida de muestras en operación, tras la implementación en un entorno real.

4.1 Corrección de fase

En el presente trabajo se ha optado por ajustar una curva senoidal sobre los datos, a partir de código compartido por Egan J (2019), que obtuvo medalla de plata en la competición, para calcular la distancia entre el cero y el corte con el eje x, añadiendo la medida como variable, en vez de proceder con una sincronización del seno inicial, que no se menciona para la solución que mejores resultados afirma obtener, de Elmasry y Wadi (2022). Esta solución sí divide la señal para su análisis, encontrando los autores que, a mayor número de fragmentos, obtenían mejores resultados en la identificación de anomalías. En este trabajo se ha optado por dividir las señales en el máximo número de fragmentos considerado por Elmasry y Wadi, en este caso introduciendo una variable *One-Hot Encoded,* que determina en base a los cortes con las x de la curva senoidal ajustada, qué fases abarca cada fragmento. Así, se pretende dotar a la salida de la información de posición respecto de la frecuencia más baja, con la que contarían los datos si se encontrasen en PRPD, conjuntamente con información de precedencia temporal propia del PRTD, a través del orden de los bloques de variables, e información relativa a la geometría de los pulsos y su entorno inmediato, con la que contarían las señales en formato PRPS (*Phase Resolved Pulse Sequence*).

4.2 Filtrado de ruido y eliminación de extremos

El segundo paso tras el cálculo del desfase de cada señal fue la eliminación de los valores de cada señal, que se encuentran más allá del umbral calculado a partir del IQR, del mismo modo que Elmasry y Wadi (2022), que se puede ver en la figura 9.

$$L_b = Q_1 - 3 \cdot IQR$$
$$U_b = Q_1 - 3 \cdot IQR$$

Figura 9: Identificación de *outliers* mediante rango intercuartílico.[6]

También se aplica en el presente trabajo el filtro de *Savitzky-Golay,* que tan buen resultado dio a Chen et al. (2021), integrándolo junto con algunos pasos adicionales, en dos de las funciones para procesar los datos y obtener pulsos a partir de ellas, que compartió en su solución *mark4h* (2019)*,* con medalla de oro en la competición. El filtro, permite eliminar el seno característico de las frecuencias más bajas, constituyendo un paso fundamental en la eliminación del ruido mediante el ajuste de una polinomial, en este caso de orden 3 con un tamaño de ventana de 99, al sustraerse la curva ajustada de los datos crudos. Si bien, se discuten en algunas publicaciones los problemas que presenta en los extremos de la señal (Schmid et al. 2022), sus beneficios respetando los pulsos en altas frecuencias son reconocidos en la literatura y muy deseados para la limpieza de las señales con PD, como reconocen estos críticos, aunque detallen recomendaciones adicionales sobre su empleo.

Tras la aplicación de este filtro, se almacena el valor absoluto del vector resultante y se aplica la función *get_peaks* según prosigue el código de *mark4h*, calculándose los máximos locales, que después se ordenan de mayor a menor, almacenando un número de ellos establecido por la variable *knee_x*. Esta, se trata de la posición a partir de la que se estabiliza la magnitud de las medidas ordenadas, calculándose primero el

[6] Elmasry y Waldy (2022).

gradiente y aplicando una convolución rectangular de longitud 9 y magnitud 1/9 para suavizar los gradientes.

También se ha reescrito la función *calculate_peak_features* para calcular las variables asociadas a cada pulso, recorriéndose en un bucle cada uno de los fragmentos en que se ha dividido la señal, terminando con un recorrido sobre la señal completa. Estas variables se basan en estadísticas relacionadas con la altura de los pulsos (máximo menos mínimo de la ventana considerada, como consideran Elmasry y Waldy en 2022), o su magnitud (altura sobre el origen de referencia, *h0*), añadiendo algunas adicionales que pretenden condensar información sobre la forma de la señal en las inmediaciones de los pulsos, y emular en alguna medida la función del *clustering* en el abordaje de Chen et al. (2021). En este último caso, se agrupaban los entornos de los pulsos identificados en la muestra de entrenamiento, formando varias plantillas de referencia (tomando una muestra de entre todos los pulsos identificados por señal, para evitar permitir que las señales con más pulsos influyesen más en el análisis). Así, se calculaba para los datos tratados, el grupo al que correspondía cada pulso mediante el RMSE frente a cada plantilla y se calculaba el número de pulsos, media y desviación típica de sus alturas, para cada grupo, a lo que se añadían los mismos estadísticos, pero calculados sobre distintos intervalos de fase de la señal global. A diferencia del enfoque antes mencionado, en el presente trabajo se intenta sustituir el *clustering,* por una batería de variables que pretenden caracterizar a través de medidas estadísticas, la geometría del entorno de cada pulso, en global para todo el fragmento: 7 percentiles, media, desviación típica, mínimo y máximo.

Así, tras dividir las medidas dentro de una ventana establecida en 25 unidades hacia cada lado de cada pico identificado (total de 51 observaciones incluyendo el pulso), se procede a multiplicar por el signo del pulso (para que el pulso y su serie siempre tengan la misma orientación, como hace *mark4h,* 2019) y después a extraer los máximos y mínimos locales de la ventana (cambios en la dirección ascendente o descendente). Estos darán lugar a los vectores *pksd* (referente a los mínimos y máximos de la señal filtrada en el entorno del pulso) y *pksdp* (referente a sus posiciones), que serán una de las fuentes principales de las que se extraigan las variables finales, del mismo modo que las diferencias entre los mínimos y máximos consecutivos de *pksd*, recogidas en la variable *prat*. En el presente caso se ha optado por dos versiones de estos vectores y las variables consideradas, basada la última de ellas en el abordaje de *mark4h* (2019) para el *clustering* respecto de las plantillas, y se trata de una versión con los valores reales de los vectores (variables terminadas en r) y otra realizando la división de las medidas del vector entre la magnitud del pulso, considerando así de forma insensible a amplificaciones o distorsiones, las diferencias entre extremos locales consecutivos. Además, se fuerza el signo positivo de los pulsos.

También se han encontrado fragmentos de señal en los que no se identifican pulsos sobre los que después poder analizar la presencia de PD, que han sido introducidos de todas formas en el análisis.

4.3 Extracción de características

Por último, al igual que los otros estudios, antes de entrenar o alimentar el algoritmo que ha de distinguir los casos anómalos de los que no lo son, las numerosas observaciones en TRPD de los datos crudos que constituyen cada señal, se condensan en una sola matriz que representa un grafo de características con la siguiente información, en este caso almacenada en una sola fila de un fichero en formato *h5* por la dimensionalidad de los datos de partida.

Son de interés los resultados de Vantuch 2018 que se resumen en la tabla 6, a pesar de haber calculado aquel, medidas de entropía, dimensión fractal y otros no considerados aquí, son tres características sencillas, la altura, el ancho y el número de los pulsos, las variables más correlacionadas con la PD identificada en las señales, en términos de la MI (*Mutual Information*) calculada según el algoritmo de Kraskov. Se utilizan además la altura y el número de pulsos, como eje de las soluciones con mejores resultados, y justificadas en la citada fuente, también se mantienen para el presente trabajo.

Pese a la selección de estas variables, probablemente sea el espacio latente del *autoencoder*, utilizado más adelante como método probabilístico de identificación de anomalías, el que constituiría el espacio de salida real del proceso de curación, al tener este tipo de técnicas un uso habitual en etapas previas de la curación, relativas a la reducción de la dimensionalidad y extracción de características, y por lo tanto un efecto seguro en etapas de procesado propias de la curación de la entrada, como ocurre con Michau et al. (2021), en aquel caso refiriéndose a una red convolucional, o como discute Vantuch (2018), que analiza el desempeño de un *autoencoder* en la fase de extracción de características.

Más allá de esto, cabe citar dos particularidades que han sido consideradas mediante reglas condicionales en el proceso de depuración:

- Para los fragmentos de señal en los que no existe forma senoidal y por lo tanto, no hay fases que identificar, simplemente se rellena el vector de 8 elementos binarios con ceros, para todos los fragmentos en los que se divida la señal y existe una última variable en la fase, que identifica con 1 uno aquellos casos que pertenecen a esta categoría y con un 0 aquellos que no. Estos fragmentos se retiran del análisis al no incluirse en los argumentos con las series de *Pandas* que contienen los id de entrenamiento y validación.
- Para los fragmentos de señal en los que no se identifican pulsos mediante la función *get_peaks* tras la aplicación del filtro de *Savitzky-Golay*, los valores de todas las variables correspondientes al fragmento son 0, en el caso de que la aplicación del filtro no devuelva ningún valor.

Las variables de salida del proceso de curación, por cada señal, susceptibles aun así de tratamiento adicional antes de alimentar el algoritmo de identificación de anomalías, han sido las de la tabla 7.

4.4 Normalización

Son diversas las posibilidades a este respecto (Kandanaarachchi 2019), destacando de forma general dos, la normalización a través de la media y desviación típica de las diferentes variables por un lado, y el escalado a través del mínimo y el máximo de los valores de cada una, por otro. En la publicación anteriormente citada, se refería como superior para la mayoría de los diversos métodos probados, el escalado mediante mínimo y máximo, y es este el que se ha aplicado aquí.

4.5 Reducción de la dimensionalidad

En la figura 11, se representan las distribuciones de algunas de las variables obtenidas tras las fases previas de curación, tanto para los casos con PD, como para los que no manifiestan dicho fenómeno. Esto pretende ayudar en la toma de decisiones sobre las variables a emplear, a través de la visualización de los datos disponibles, a lo que también contribuyen los valores del ranking de MI que se muestra en la figura 10, para cada una de las variables de entrada correspondientes a cada señal.

En este caso, la única estrategia que involucra una reducción de dimensionalidad se trata del propio *encoder* correspondiente a la arquitectura del *Autoencoder*, aunque aplicar otros métodos sería una segura apuesta para mejorar los resultados del trabajo.

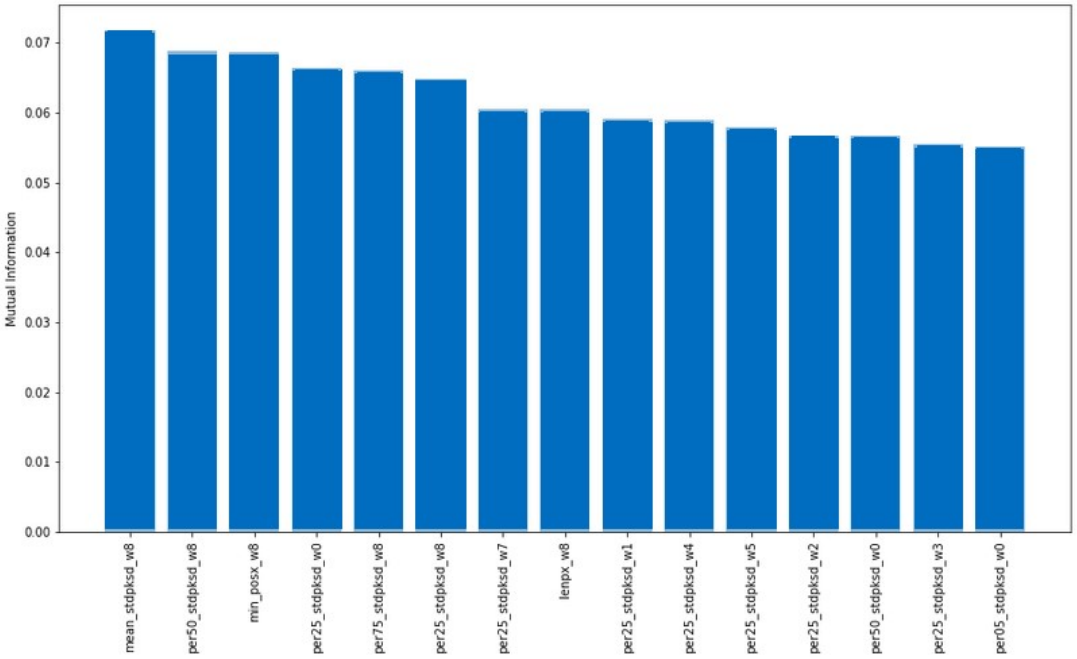

Figura 10. 15 variables más representativas en cuanto a la Información Mutua respecto del carácter de PD de las señales.

En la figura 11, se representan, asimismo, diversos histogramas de densidad de frecuencia para las variables, que se han ayudado a reconocer visualmente al menos de manera, cómo algunas variables presentan distribuciones de parámetros señaladamente distintos en función de la identificación de PD en ellas o no, mientras que en otros casos se superponen las distribuciones de ambas categorías.

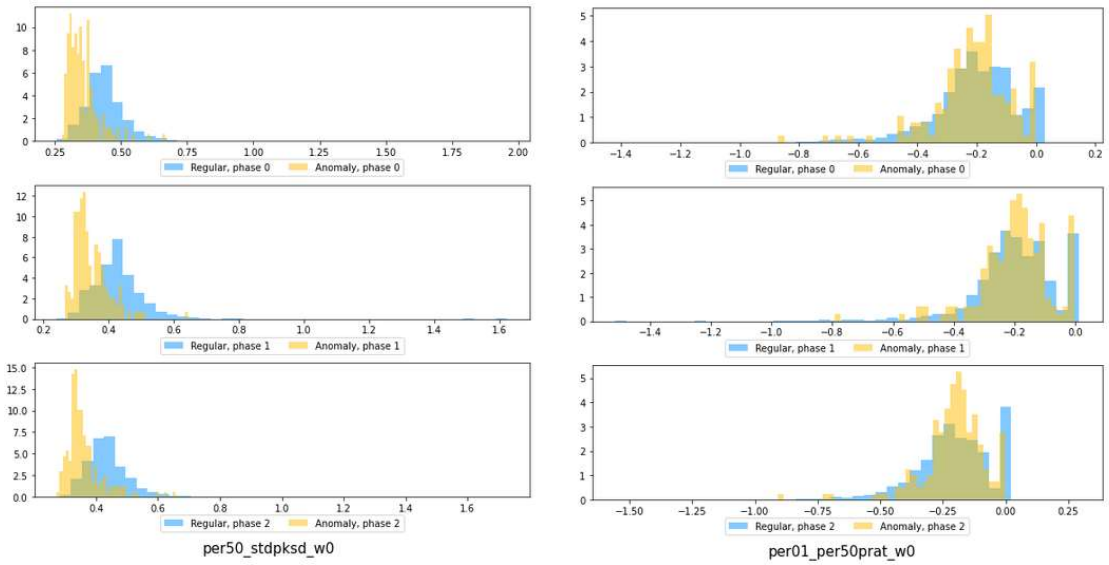

Figura 11. Percentil 50 de la desviación típica del número de extremos locales en las inmediaciones de los picos identificados en la ventana 0 como predictor prometedor (a la izquierda). Percentil 1, del percentil 50 de la diferencia entre extremos locales consecutivos en las inmediaciones de los picos identificados en la ventana 0, menos prometedor (a la derecha).

Nombre	MI
Media	0.3787
Desviación Típica	0.0516
Simetría	0.0483
Curtosis	0.0311
Entropía de la señal	0.0708
Energía de descomposición	0.0708
Entropía de los coeficientes de detalle	0.0721
Dimensión Fractal	0.0820
Número de pulsos	1.7676
Anchura media de los pulsos	1.8311
Altura media de los pulsos	0.7818
Anchura máxima de los pulsos	1.9384
Altura máxima de los pulsos	0.9907
Anchura mínima de los pulsos	1.8822
Altura mínima de los pulsos	0.9474

Tabla 6. Análisis de MI, cuanto más alto, más correlación existe con la variable objetivo.[7]

Nº	Variables extraídas del entorno de cada pulso (51 medidas)	Variables sobre cada fragmento de señal	Variables globales sobre la señal completa
64	-	-	8 x 8: 8 **variables de fase binarias *One-Hot Encoded*** para 8 fragmentos de señal)
1	-	-	1: **Posición del cero** de la curva senoidal
9	-	9: 8 fragmentos y señal global. **Número de pulsos**.	-
99	**Valor real de la magnitud del pulso** con su signo	11 x 9: 8 fragmentos y señal global. Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de la columna a la izquierda	-
99	**Número de extremos locales**	11 x 9: 8 fragmentos y señal global. Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de la columna a la izquierda	-
99	**Número de extremos locales** por debajo de 0 tras corregir el signo de los pulsos negativos	11 x 9: 8 fragmentos y señal global. Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de la columna a la izquierda	-

[7] Vantuch (2018).

Nº	Variables extraídas del entorno de cada pulso (51 medidas)	Variables sobre cada fragmento de señal	Variables globales sobre la señal completa
99	**Número de extremos locales** por encima de 0 tras corregir el signo de los pulsos negativos	11 x 9: 8 fragmentos y señal global. Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de la columna a la izquierda	-
792	Media, desviación típica, máximo y mínimo de las **diferencias entre extremos locales consecutivos**, en su magnitud real y divididos entre la magnitud del pulso asociado a la ventana	11 x 9 x 4 x 2: 8 fragmentos y señal global. Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de la columna a la izquierda	-
1386	Percentiles 1,5,25,50,75,95 y 99 de las **diferencias entre extremos locales consecutivos**, en su magnitud real y divididos entre la magnitud del pulso asociado a la ventana	11 x 9 x 7 x 2: 8 fragmentos y señal global. Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de la columna a la izquierda	-
396	Media y desviación típica del valor de los **extremos locales** con su signo corregido, en su magnitud real y divididos entre la magnitud del pulso asociado a la ventana	11 x 9 x 2 x 2: 8 fragmentos y señal global. Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de la columna a la izquierda	-
198	**Diferencia entre el máximo y el mínimo** valor de los extremos locales (altura del pulso) con signo corregido, en su magnitud real y divididos entre la magnitud del pulso asociado a la ventana	11 x 9 x 2: Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de las magnitudes de la columna a la izquierda	-
99	**Posición del pulso** en el fragmento	11 x 9: Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de las magnitudes de la columna a la izquierda	-
99	Desviación típica de las **posiciones de los extremos locales**	11 x 9: Media, desviación típica, mínimo, máximo y percentiles 1,5,25,50,75,95 y 99 de las magnitudes de la columna a la izquierda	-

Tabla 7. Batería de características extraídas tras el proceso de curación, por cada señal.

5. Análisis y Procesado

Se refiere aquí la documentación de acuerdo con el ciclo de vida del software de las fases de análisis, planificación, codificación y prueba, de la herramienta de detección de anomalías desarrollada, quedando excluida la última fase de implantación y mantenimiento en un entorno real, que excede el alcance del presente trabajo.

5.1. Diseño y metodología

El resultado del método aplicado para la detección de anomalías, como señalan Cook et al. (2019), más allá del modo en que se categorice que puede cambiar en función de la literatura consultada, habitualmente provee como resultado, siempre una puntuación y a menudo una etiqueta en función de un umbral, que discrimina estas frente a los casos normales. El enfoque aquí decidido es probabilístico, buscando obtener además de la discriminación de la anomalía, una probabilidad derivada de la distribución a posteriori implícita en el método elegido, que ayudará a valorar objetivamente el nivel de riesgo asociado a la misma. Esta publicación también hace referencia a los *autoencoders* como método competitivo en datos de series temporales tanto univariadas como multivariables, en el ámbito del IoT.

En esta misma línea, el abordaje a través de *autoencoders* del problema particular de detección de PD, tampoco es difícil de encontrar en la literatura, aunque es más habitual su aplicación en la curación y extracción de características, que, en la propia identificación de las anomalías, como hizo Vantuch (2018) y también Nussbaum (2019), medalla de plata en la competición. Volviendo sobre la revisión de la literatura técnica, con datos próximos a los utilizados en el presente estudio, destaca por ejemplo la publicación de Li et al. (2019), que recogieron las señales mediante GIS (*Gas Insulated Switchgear*), y utilizaron redes neuronales convolucionales (CNN en inglés) junto con un *autoencoder* para la extracción de características en datos en PRPD (*Phase Resolved Partial Discharge*). Asimismo, existen ejemplos distantes en el tiempo como el de Martinelli et al. (2004) que aplicaron un *autoencoder* a datos de subestaciones, abordando una detección de anomalías sin carácter predictivo y con otro tipo de información (datos medidos sobre transformadores), orientada a detectar a cambios de topología como cortes de suministro, o incrementos inusuales de la demanda, a través de un módulo con este algoritmo integrado, por cada subestación, con el RMSE como función objetivo del *autoencoder*, es decir, sin el enfoque probabilístico.

Merece la pena destacar alguna revisión actualizada del estado del arte en el área específica de PD que parece haber sido ya intensamente abordado, aunque sigue despertando interés por la rápida evolución reciente de los métodos de aprendizaje automático gracias a la de la infraestructura que los ejecutan, siendo un ejemplo representativo el de la publicación de Lu et al. (2020). Es de destacar su comentario sobre las bondades del formato TRPD de la señal de partida que evita el problema de superposición de información de dominio de fase que presentan las señales en PRPD o PRPS, calculándose en este caso, como ya hemos visto en el apartado de curación, información relativa a estos dos últimos conceptos para considerar un enfoque híbrido lo más amplio posible, como partida. El *autoencoder*, se encuentra explicado en la revisión de métodos de Deep Learning susceptibles ser utilizados, mencinando diversos casos recientes en la literatura de diversas variantes probadas, resultando como ganador en comparativas respecto de otros métodos tradicionales como PCA… Sin embargo, en el texto menciona además algunos retos a tener en cuenta en la

aplicación de los *autoencoders* como método no supervisado, que se expresan a continuación junto con los señalados en la publicación de Baradaran (2021):

- Identificación de las señales más significativas para el etiquetado (como la eliminación de las líneas planas sin la característica forma senoidal, que son etiquetadas normales al no presentar PD, pero se dan con muy poca frecuencia en los datos).
- Identificación del tamaño de la muestra mínimo para el entrenamiento (en este caso se considera abordado por los responsables de la recogida en iteraciones anteriores del ciclo de vida de los datos).
- Interpolación en la generación a partir del espacio latente, teniendo en cuenta la posible discontinuidad de este (se muestrean varios puntos por cada vez que se alimenta el modelo).
- Selección del mejor grado de compresión del *autoencoder* (se utiliza la selección de hiperparámetros para seleccionar la dimensión del espacio latente).
- Tendencia al desvanecimiento de gradiente (se utiliza la activación *leaky_relu* y capas *BatchNorm1d* en la arquitectura).
- Distorsión de la información de entrada en su reconstrucción, sufriendo de sesgos de memorización y por tamaño de entrenamiento insuficiente, sin considerar cierta información de alto nivel, ni poder identificar todas, o adecuadamente, las categorías de eventos en los datos (limitada mediante ingeniería de características y el número de capas en la arquitectura, según Radhakrishnan, 2019, ver figura 12, asumiéndose en parte).
- Desbalance de información en cada capa (limitación asumida de asimetría en su arquitectura, que otras arquitecturas como *U-Net* considerarían).

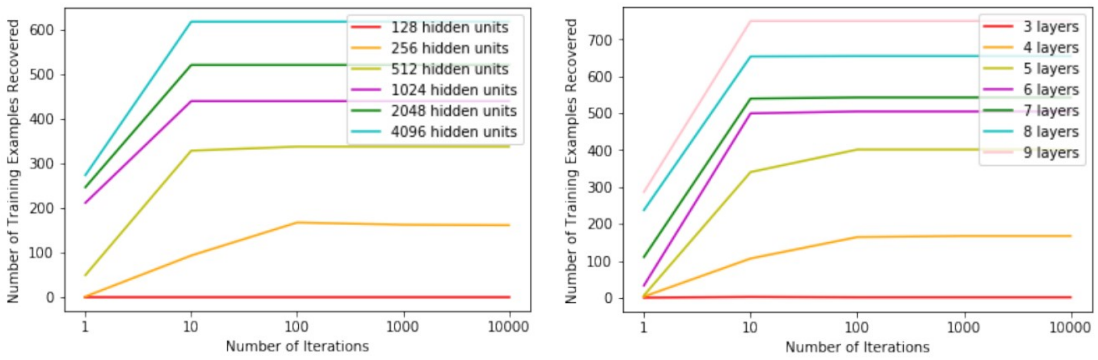

Figura 12. *Autoencoders* con distintos tamaños de capa oculta (izquierda) y profundidades (derecha), fueron entrenados con 1000 muestras del MNIST.[8]

En este caso, la mayor parte de las capas son una combinación de *Linear* y *BatchNorm1d*, como se puede ver en la figura 13, tras las que se aplica la función de activación *leaky_relu*, salvo en el caso de la última capa antes del espacio latente y la capa anterior a la de salida, en la que se utiliza *sigmoid*.

[8] Radhakrishnan (2019).

```
--------------------------
ARCHITECTURE
--------------------------
******************************
ModuleList(
  (0): Linear(in_features=10320, out_features=10320, bias=True)
  (1): BatchNorm1d(10320, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (2): Linear(in_features=10320, out_features=5282, bias=True)
  (3): BatchNorm1d(5282, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (4): Linear(in_features=5282, out_features=245, bias=True)
  (5): BatchNorm1d(245, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (6): Linear(in_features=245, out_features=245, bias=True)
  (7): BatchNorm1d(245, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (8): Linear(in_features=245, out_features=245, bias=True)
  (9): BatchNorm1d(245, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (10): Linear(in_features=245, out_features=245, bias=True)
  (11): BatchNorm1d(245, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (12): Linear(in_features=245, out_features=5282, bias=True)
  (13): BatchNorm1d(5282, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (14): Linear(in_features=5282, out_features=10320, bias=True)
  (15): BatchNorm1d(10320, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (16): Linear(in_features=10320, out_features=10320, bias=True)
  (17): Linear(in_features=10320, out_features=10320, bias=True)
)
******************************
```

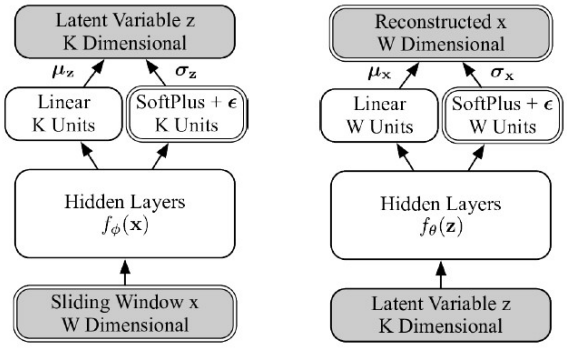

Figura 13. Arquitectura del mejor modelo y diagrama explicativo de la red variacional o *encoder,* a la izquierda y la red generativa o *decoder* a la derecha. La *Reconstructed x W Dimensional* de salida, son 2 bloques, la media y el logaritmo de la desviación típica correspondientes a p(x/z).[9]

En este caso, se pretende utilizar un enfoque que aporte información probabilística de la que acompañar la información discriminatoria del método. Para ello se ha elegido la función de coste de Higgins et al. (2017), con un término de Kullback-Leiber que asume una distribución latente gausiana (*prior distribution,* p(z/x)) de media 0 y desviación típica 1, que aporta una regularización, reforzada con las capas *BatchNorm1d* y controlada a través de un hiperparámetro, *beta*. También cuenta además como segundo término basado en la derivación del *Evidence Lower BOund* (ELBO), la media en q(z) del logaritmo de p(x/z), que constituye el error de reconstrucción para esta solución. La función de coste se resume así, en la figura 14.

$$\mathcal{F}(\theta, \phi, \beta; \mathbf{x}, \mathbf{z}) \geq \mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}, \beta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta\, D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \simeq \frac{1}{2}\sum_{j=1}^{J}\left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + \frac{1}{L}\sum_{l=1}^{L}\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)})$$

Figura 14. Función objetivo.[10]

[9] Xu H et al. (2018)

[10] Higgins et al. (2017) y Kingma y Welling (2013).

La herramienta consta de diversos parámetros ajustables, relacionados con el porcentaje de la muestra de validación utilizado en cada reporte, hiperparámetros de la arquitectura como profundidad y tamaño de capa, número de lecturas de validación por epoch… y una estructura que pretende hacerlo cómodo de utilizar, como se puede apreciar en la figura 15.

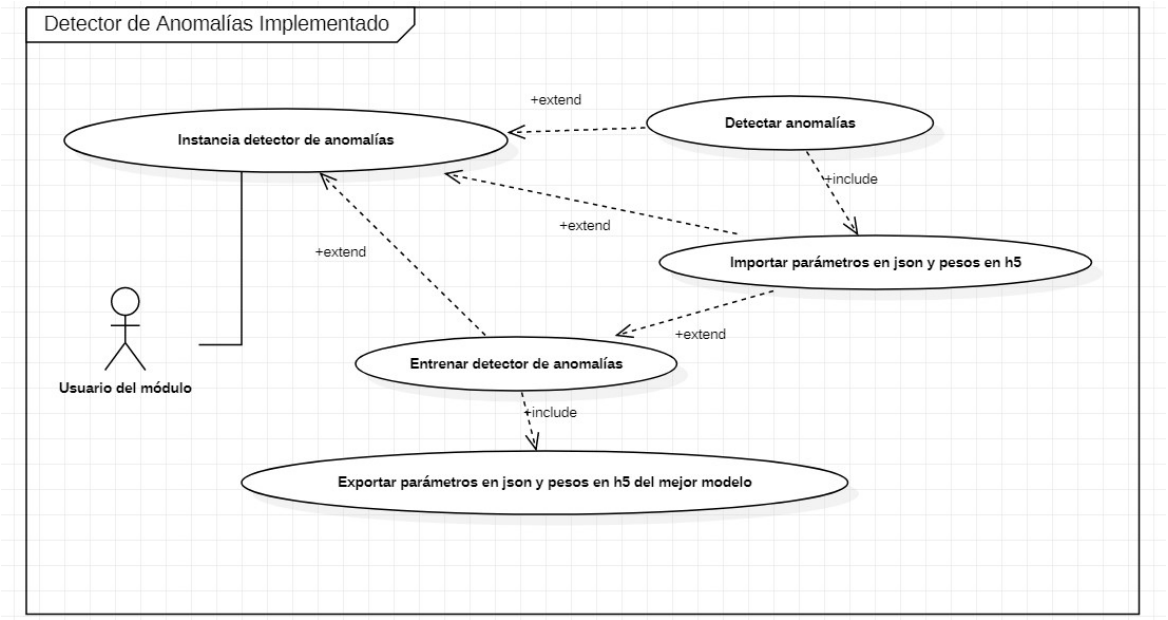

Figura 15. Diagrama de casos de uso.

5.2. Implementación

Si la última figura, número 22, trataba de identificar los principales agentes y funcionalidades que se han desarrollado en la solución sobre la que gira este trabajo, la tabla de la figura número 23, refleja las tareas llevadas a cabo para su elaboración, incluyendo el análisis y procesado, dentro del ciclo de vida de los datos, en forma de *Backlog*.

Sprint	Requisito	Tarea	Estimación (semanas)
1	**Redacción de la memoria:** *Autoencoders* y detección de anomalías.	Aprendizaje y recopilación de bibliografía sobre *autoencoders* y detección de anomalías en IoT.	4
05/01/2022			
2-3	**Aprendizaje y primer prototipo:** Pytorch.	Desarrollo de prototipo básico de *autoencoder* con Pytorch basado en RMSE como función de coste, con métricas de evolución del entrenamiento en Tensorboard.	8
03/05/2022			

4	**Datos para la validación**.	Búsqueda de un conjunto de datos sobre el que trabajar.	1
4	**Curación**: Lectura de los datos y análisis preliminar.	Análisis preliminar, lectura y visualización mediante pyspark, de los datos en formato parquet.	2
		Selección de los conjuntos de entrenamiento y validación, cálculo de medias móviles y normalización de las series mediante pyspark	2
06/07/2022			
5	**Arquitectura**: *autoencoder* variacional, con su ciclo de entrenamiento.	Adaptación de la arquitectura original, incorporación de una función de coste más sofisticada y una arquitectura de ancho y profundidad regulables, y adaptación del ciclo de entrenamiento.	3
5	**Ciclo de entrenamiento y validación:** Desarrollo base, control de eficiencia y desempeño.	Adaptación del ciclo de validación junto al entrenamiento e introducción de métricas internas. Subsanación de errores relacionados. Introducción de criterios de eficiencia para la validación, en cuanto a tiempo (máximo 5% del tiempo de entrenamiento) y control del uso de memoria.	4
5	**Estructuración del código para la entrega**.	Diseño de una estructura usable respetuosa con normas de estilo para favorecer el posterior uso y la revisión en la tutorización.	1
5	**Validación y reingeniería:** para la optimización de tiempos y subsanación de errores.	Optimización de los tiempos de ejecución, mediante la sustitución de pyspark en la importación de los datos y tratamiento preliminar por pandas. Subsanación de errores relacionados. Sin éxito en cuanto a desempeño.	2
15/09/2022			
6	**Documentación y redacción de la memoria:** Estructura del trabajo.	Revisión del análisis preliminar ante resultados insatisfactorios del entrenamiento, estructuración de la memoria y documentación de uso.	1
6	**Estructuración del código para la entrega, ciclo de entrenamiento y validación:** en cuanto a los parámetros del circuito de entrenamiento y validación, para facilitar una validación con desempeño aceptable.	Nuevas funcionalidades sobre exportación del modelo y parámetros de diseño como las funciones de optimización, profundidad… de la arquitectura, o tamaño de los bloques de entrenamiento / validación buscando mejores resultados y un diseño versátil para el entrenamiento.	3

6	**Validación y reingeniería:** nuevas funcionalidades y parámetros para facilitar una validación con desempeño aceptable.	Subsanación de errores y mejoras en el circuito de entrenamiento y validación, como el balanceo de casos normales /anómalos en la validación, parametrización del tiempo de validación/entrenamiento… Segundas pruebas con datos reducidos por medias móviles y eliminación, con desbordamiento de memoria o desempeño pobre (figura 17).	4
14/11/2022			
7	**Aprendizaje:** curación y análisis de series temporales y PD.	Revisión bibliográfica del estado del arte, metodología aplicada en la competición…	4
7	**Redacción de la memoria**.	DMP, Análisis preliminar, Curación y Análisis y procesado.	4
16/01/2023			
8	**Curación:** Aplicación en base a trabajos relacionadas de un proceso de curación.	Según la bibliografía, subsanando errores por desbordamiento de memoria en numpy en cálculos y similares.	4
8	**Validación y reingeniería:** selección de variables.	Selección y evaluación de las variables resultado de la curación.	4
13/02/2023			
9	**Estructuración del código para la entrega, arquitectura, ciclo de entrenamiento y validación:** Completando funcionalidades anteriores en base a la curación e información probabilística de salida de la herramienta.	Sustitución de parquet por h5, formato resultado de la curación, al resultar más familiar y manejable al autor. Reingeniería de la herramienta completa para facilitar la importación de modelos y parámetros, y salida de un resultado probabilístico, con métricas AUC-ROC y la MCC utilizada en la competición.	4
9	**Validación y reingeniería:** obtención de un modelo competitivo.	Perfeccionamiento a través de selección de hiperparámetros.	4
15/04/2023			
10	**Redacción de la memoria**.	Finalización de la memoria y de la defensa, elaboración de documentación, y ajuste de hiperparámetros.	4

Tabla 8. *Sprint Backlog*.[11]

[11] Basado en Schwaber y Sutherland (2020) y Yazyi S. (2011).

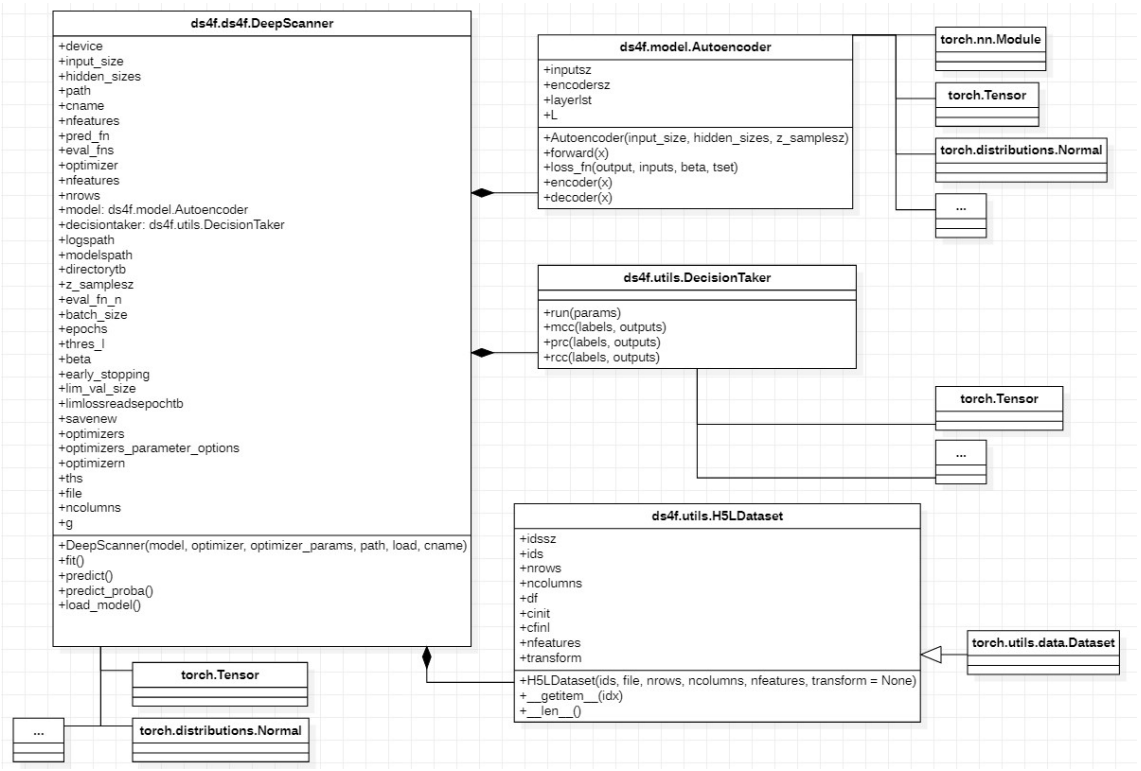

Figura 16. Diagrama de clases.

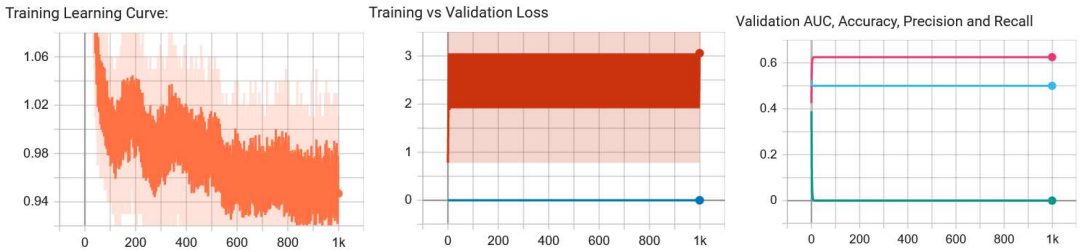

Figura 17. Salida de *tensorboard* sprint 6.

5.3. Validación y Resultados

A través del método de optimización bayesiana de hiperparámetros elegido, probando dos librerías de Python, se han realizado pruebas de validación reflejadas en el *notebook DS4F-Hyperparameter_tunning_example.ipynb*. En primer lugar, con una de las librerías, GpyOpt, se han entrenado 3 modelos para cada una de las 3 combinaciones de hiperparámetros muestreadas, de los siguientes intervalos, prolongándose la ejecución unos 9 días (resultados en la tabla 9):

- Número de neuronas intermedias: 5-1000.
- Profundidad de la red: 3 y 5 como tamaños posibles de *encoder*.
- Parámetro *beta* que regula la influencia del término de Kullback-Leiber: 0.5-10.
- *learning rate*: 0.00001-0.01 (habiendo seleccionado de entre los dos métodos de optimización incluidos en el detector de anomalías, heredados de la librería *torch*, el *Adams*).

Caso	Beta	Learning Rate	Encoder Size	Latent Size	MCC medio	MCC desviación típica	MCC mejor	AUC mejor	Precision mejor	Recall mejor
0	2.53	0.0001	5	135	0.173	0.015	0.19	0.57	0.52	1.00
1	0.65	0.00009	3	245	0.345	0.027	0.38	0.70	0.86	0.38
2	6.83	0.001	3	200	0.320	0.116	0.39	0.69	0.72	0.62

Tabla 9. Ajuste de hiperparámetros 1. El "mejor" seleccionado se refiere al model con el valor del MCC más elevado. Se entrenan 3 modelos por cada caso.

Para el caso 2 encontramos los valores más altos de MCC y AUC para alguno de los modelos, mayores incluso que los señalados en la tabla con valores de 0.39 y 0.72 respectivamente (model_8trainedScanner_20230425_212437.pt), pero comprometen de forma importante el *recall*, un fenómeno que se detecta también en las otras configuraciones.

En una segunda iteración de búsqueda de hiperparámetros con otra de las librerías, *bayesian-optimization*, reflejada en la tabla 10, tras corregir algunos errores en las curvas de entrenamiento de *tensorboard*, se ha probado un intervalo más limitado de valores para las mismas variables dejando sólo 2 a ajustar, durante 5 días, con el fin de afinar el desempeño del modelo:

- Número de neuronas intermedias: 20-250.
- Parámetro *beta* que regula la influencia del término de Kullback-Leiber: 1-10.

Caso	Beta	Learning Rate	Encoder Size	Latent Size	MCC medio	MCC desviación típica	MCC mejor	AUC mejor	Precision mejor	Recall mejor
0	9.60	0.0005	3	201	0.120	0.056	0.20	0.55	0.55	0.85
1	9.62	0.0005	3	63	0.201	0.082	0.30	0.62	0.67	0.59
2	5.60	0.0005	3	131	0.128	0.009	0.13	0.51	0.51	0.99
3	9.02	0.0005	3	47	0.140	0.028	0.18	0.52	0.56	0.76
4	7.68	0.0005	3	68	0.156	0.007	0.27	0.59	0.64	0.60

Tabla 10. Ajuste de hiperparámetros 2. El "mejor" seleccionado se refiere al model con el valor del MCC más elevado. Se entrenan 3 modelos por cada caso. Se mantienen constantes la profundidad de la red y la *learning rate*.

Por último, se ha ejecutado una tercera prueba, de la tabla 11, al considerar que quizás los resultados tan pobres pudieran deberse al valor de 0.0005 de *learning rate,* utilizándose para esta prueba el valor 0.0001, y manteniéndose la misma configuración que la anterior prueba, para el resto de hiperparámetros. En esta ejecución, a diferencia de las anteriores, no se han entrenado 3 modelos por cada configuración de parámetros, sino solamente un modelo.

Caso	Beta	Learning Rate	Encoder Size	Latent Size	MCC medio	MCC desviación típica	MCC mejor	AUC mejor	Precision mejor	Recall mejor
0	4.62	0.0001	3	78	0.15	0	0.15	0.51	0.55	0.79
1	2.26	0.0001	3	140	0.12	0	0.12	0.52	0.51	0.99
2	1.96	0.0001	3	156	0.13	0	0.13	0.52	0.54	0.81
3	1.90	0.0001	3	246	0.17	0	0.17	0.55	0.57	0.68
4	4.92	0.0001	3	231	0.14	0	0.14	0.52	0.51	0.99
5	4.18	0.0001	3	247	0.09	0	0.10	0.48	0.53	0.70
6	9.61	0.0001	3	163	0.10	0	0.10	0.53	0.51	0.99
7	1.09	0.0001	3	37	0.07	0	0.15	0.50	0.55	0.79

Tabla 11. Ajuste de hiperparámetros 3. El “mejor” seleccionado se refiere al model con el valor del MCC más elevado.Se entrena 1 modelo por cada caso.

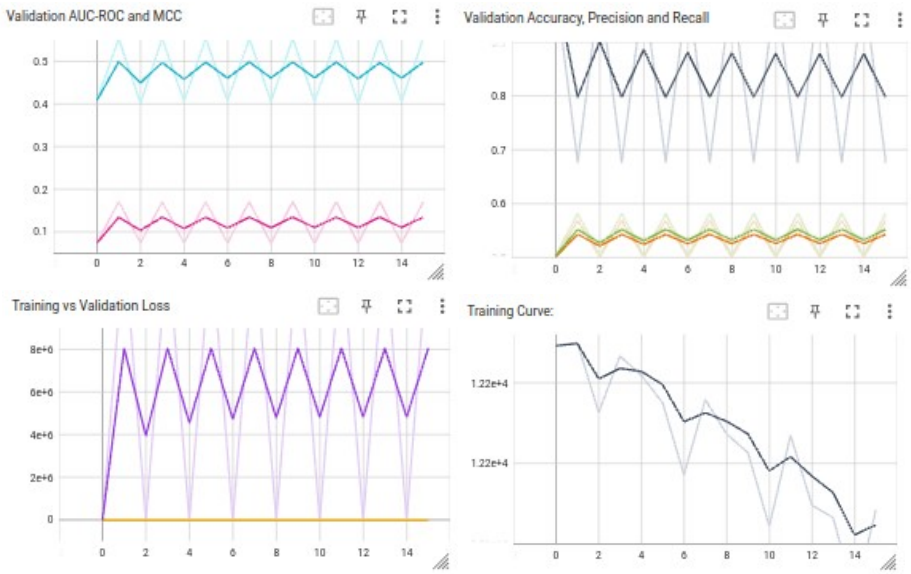

Figura 18. Salida de *tensorboard* sprint 9.

Como se puede observar en la figura 18, los resultados tampoco mejoraron en ningún caso los resultados, que en base a la curva de aprendizaje (*Training vs Validation loss*) del mejor modelo correspondiente a la última prueba ejecutada, presentan problemas de sesgo. Los resultados demandan así, una reflexión más profunda sobre el posible excesivo número de variables, que inicialmente se consideró útil por el costoso proceso de extracción de características, pero que una vez obtenidas, requieren posiblemente algún método de reducción de la dimensionalidad que culmine la curación previamente descrita, dada la presencia de variables que parecen indicar de un modo bastante definido, una frontera entre casos normales y anómalos (ver figura 16).

6. Metadatos y Plan de Preservación

Se puede ver los ficheros resultantes de la ejecución del código en el apartado 2.1 1b. De estos, los resultados finales del ajuste del proceso de curación (ficheros h5 denominados *processed* y *scaled*), del ajuste de hiperparámetros (ficheros en la carpeta logs, runs y models) y el modelo final entrenado con todo el conjunto de entrenamiento proporcionado por la competición de *Kaggle,* no se facilitan por los motivos desarrollados en el apartado 2.4 4b, aunque su obtención es replicable a través del código compartido.

El resto de documentación y recursos resultantes se encuentran en tres repositorios diferentes, acompañados de los metadatos detallados en el apartado 2.2 2a:

- En el repositorio de Ucrea de la Universidad de Cantabria se encuentran almacenados todo el código y los documentos en formato docx y pdf del presente trabajo escrito.

- La cuenta de Gitlab de HP SDCS (que tuteló hasta septiembre de 2022 el trabajo) y la del presente autor, se refieren aquí como una sola por contener la misma información. Almacenan y comparten de forma pública el código de la herramienta en diversos directorios, en el que los datos del directorio /*data* sirven como input del código del directorio /*src* (el módulo *ds4f,* que puede ser llamado desde los notebooks o desde los ficheros *train.py* y *predict.py* a través de línea de comandos). Este código genera las salidas de los directorios */runs, /logs*, y */models*, correspondientes a los *logs* de entrenamiento los dos primeros y a los modelos y configuración del escáner señalado, el último.

Se limita así este apartado, a recoger de forma concisa referencias a otros lugares del documento, que desarrollan la información clave reflejada en su título, al no haberse cubierto la fase de implementación en campo de la herramienta, que sí implicaría medidas adicionales de preservación y metadatos asociados a un flujo de datos más complejo desde los aparatos de medida hasta los servidores de procesado, desde donde se lancen las alarmas correspondientes a las anomalías.

7. Herramientas Utilizadas

El trabajo ha sido redactado utilizando la suite Microsoft Office, que también ha servido al examinar parte de los datos en formato *csv.* Más allá de esta herramienta, destaca el uso del software *StarUML,* para elaborar los diagramas de casos de usos y diagrama de clases que se recogen en las figuras 15 y 16.

Asimismo, aparte del uso de *Google Colab* durante la primera mitad del trabajo, se ha trabajado también en un equipo local utilizando fundamentalmente Jupyter Notebook dentro de entornos de *Conda* para Windows, lo que ha permitido instalar y restaurar librerías instaladas fácilmente, optando por otras nuevas en función de las problemáticas encontradas en su puesta en funcionamiento, como fue el caso con las 2 que se han utilizado para ajuste de hiperparámetros (*GpyOpt* y *bayesian-optimization*).

En lo relativo a las librerías de Python utilizadas, han sido muy diversas. Pytorch fue la opción considerada desde el principio como condicionante, para implementar la herramienta del escáner. Más allá de esta, se ha utilizado, por ejemplo, Scikit-learn en las métricas de validación, aunque se haya optado en algunos casos por implementarlas manualmente por problemas en situaciones concretas relacionadas con el carácter no balanceado de los datos. Matplotlib y Seaborn han sido las opciones utilizadas para la mayor parte de las representaciones gráficas, aunque Tensorboard fue utilizado también para permitir tener un registro del desempeño según avanza el entrenamiento de la red y Statsmodels fue la opción utilizada para representar los diagramas de autocorrelación y autocorrelación parcial durante el análisis preliminar. Pyspark fue utilizado durante los primeros meses por el gran volumen de observaciones de cada serie de datos, pero fue descartado después en todo el trabajo y sustituido por Pandas combinado con H5py, como librerías para manejo de los datos, al resultar más eficientes y rápidas para el problema abordado en cuestión.

8. Conclusión

8.1. Conclusiones

El trabajo ha sido una oportunidad para dotarme de herramientas como científico de datos. Por ejemplo, el esfuerzo realizado en la curación me ha provisto de importantes herramientas, aprendiendo diversas técnicas disponibles relacionadas con el tratamiento de señales y series temporales, novedosas para mí, además de haberme enfrentado al uso de una librería tan popular en el mundo del *deep learning* como Pytorch. También he podido crecer desde los problemas iniciales, aprendiendo a realizar una apropiada gestión de versiones con Git, o a vigilar la apropiada documentación y estructura del código compartido.

En lo relativo a la herramienta, implementada y recogida en el módulo dsf4, permite entrenar un *autoencoder* con datos alimentados desde un fichero en formato HDF5 y varios *dataframes* identificando la partición entre datos de entrenamiento y validación, a través de un método *fit()*. Después, se puede volver a cargar el modelo entrenado a través de *load_model()* para continuar el entrenamiento o bien para realizar predicciones probabilísticas o binarias sobre nuevos datos mediante los métodos *predict_proba()* y *predict()*, respectivamente. Para ello se vale de diversos métodos como *nfeatures* que hace referencia al número de series temporales que corresponden a una sola observación del conjunto de datos, ya sea de entrenamiento o de validación.

La salida probabilística es uno de los factores diferenciadores frente a otras implementaciones observadas en otros repositorios de *github* o *kaggle*. También lo es la configuración del modelo, que admite como entrada un diccionario con tres valores: el tamaño de la capa de entrada, el número de muestras del espacio latente para cada ciclo de entrenamiento y una lista con los distintos tamaños de capa desde la de entrada hasta el espacio latente. Esta configuración ha supuesto el uso del módulo de Pytorch ModuleList y puede resultar farragosa en su lectura, pero permite tratar el número de capas y su tamaño, como hiperparámetros configurables a través de una sola variable, como se ha hecho en el apartado 6.

En el ámbito de la identificación de la PD como tipo de anomalía concreto, podemos observar una utilidad marginal, al menos para el preprocesado y el ajuste de hiperparámetros descritos en este trabajo, sobre los que se proponen nuevas ideas en el apartado siguiente de limitaciones, dado que la solución queda lejos de lo que se logró en trabajos anteriores con la misma finalidad (ver figura 13).

Sin embargo, se puede afirmar que la implementación y validación marcados como objetivos del presente trabajo han sido satisfactoriamente alcanzados, pudiéndose extrapolar su uso a datos de distinto ámbito o bien mejorar el pretratamiento de los aquí utilizados para mejorar los resultados.

8.2. Limitaciones y trabajos futuros

La primera y principal limitación a mencionar es el escaso éxito obtenido en la detección del tipo de anomalía buscado, las señales con PD, en comparación con otras técnicas estado del arte. Además, podríamos señalar algunas áreas de mejora adicionales que pudieran abordarse en trabajos futuros:

- Posibilidad de mejorar los resultados aplicando técnicas de reducción de la dimensionalidad o realizando selecciones de variables más exhaustivas.

También al respecto del pretratamiento, sería interesante probar a aplicar *Discrete Wavelet Transform* (DWT) para mantener el desarrollo temporal de las variables, filtrando aquellos pulsos que no estén en la banda de la PD, con objeto de reducir el ruido y poder buscar variables prometedoras (figura 16) en distintas bandas de tiempo, fase y frecuencia.

- Posibilidad de introducir un intervalo de confianza para la probabilidad obtenida mediante el método *predict_proba()* del escáner, al basarse en una muestra del espacio latente a partir de la que se aplica una red neuronal determinística de la que se obtiene una estimación de la media y desviación típica correspondientes a la p(x/z) a través de una media.

- Prueba de métodos de calibración a la salida de *predict_proba()*, no habiéndose tenido en cuenta en el seno de la herramienta dado que las librerías con implementaciones de métodos de aprendizaje automático como *sklearn*, no lo integran habitualmente, requiriéndose una fase posterior al respecto.

- Necesidad de ejecutar una selección de hiperparámetros durante más tiempo, al resultar muy limitados los intentos ejecutados, siendo otro potencial aspecto de mejora relacionado, el tiempo de ejecución del entrenamiento.

9. Bibliografía

- An J, Cho S. “Variational Autoencoder based Anomaly Detection using Reconstruction Probability”. *Special Lecture on IE*. 2015. Vol. 2. No. 1. pp. 1–18.

  https://www.semanticscholar.org/paper/Variational-Autoencoder-based-Anomaly-Detection-An-Cho/061146b1d7938d7a8dae70e3531a00fceb3c78e8?p2df [Accesed: Feb. 2023].

- Baradaran M. “A Critical Study on the Recent Deep Learning Based Semi-Supervised Video Anomaly Detection Methods”. *arXiv:2111.01604v1* [cs.CV]. Nov. 2021. [Online]. Available:

  https://arxiv.org/abs/2111.01604v1 [Accesed: Jun. 2022].

- Chen K, Vantuch T, Zhang Y, Hu J, He J. “Fault Detection for Covered Conductors With High-Frequency Voltage Signals: From Local Patterns to Global Features”. *IEEE Transactions on Smart Grid*. Vol. 12. No. 2. pp. 1602 – 1614. Oct. 2021. [Online]. [Accesed: Jun. 2022].

- Cook A, Misirli G, Fan Z. “Anomaly Detection for IoT Time-Series Data: A Survey”. *IEEE Internet of Things Journal*. Vol. 7. No. 7. pp. 6481 – 6494. Dec. 2019. [Online]. [Accesed: Jun. 2022].

- Digital Curation Centre and UC3 team at the California Digital Library. “Public DMPs”. *dmponline.dcc.ac.uk*. [Online]. Available:

  https://dmponline.dcc.ac.uk/public_plans [Accessed Sep. 2022].

- Economic and Data Research Council, United Kingdom. *Research Data Lifecycle*. 2019. Format: Vídeo. [Online]. Available:

  https://ukdataservice.ac.uk/learning-hub/research-data-management/ [Accessed Sep. 2022].

- Egan J. *VSB Power Line Fault Detection Approach*. [Competition Notebook]. Kaggle: Power Line Fault Detection Competition of Enet Centre, VSB – TU of Ostrava. Egan J. 2019.

  https://www.kaggle.com/code/jeffreyegan/vsb-power-line-fault-detection-approach [Accessed Sep. 2022].

- Elmastry W, Wadi M. “Enhanced Anomaly-Based Fault Detection System in Electrical Power Grids”. *International Transactions on Electrical Energy Systems*. Vol. 2022. 1970136. Feb. 2022. [Online]. [Accesed: Jun. 2022].

- Hashmi GM, Lehtonen M, Nordman M.“Modeling and Experimental Verification of On-line PD Detection in MV Covered-conductor Overhead Networks”. *IEEE Transactions on Dielectrics and Electrical Insulation*. Vol. 24. No. 2. pp. 167-179. Apr. 2017. [Online]. [Accesed: Dec. 2022].

- Hawkins DM. *Identification of outliers*. Chapman and Hall. 1980. [Accesed: Jun. 2022].

- Higgins I, Matthey L, Pal A, Burgess C, Glorot X, Botvinick M, Mohamed S, Lerchner A. “beta-VAE: Learning Basic Visual Concepts with a Constrained

Variational Framework". *International Conference on Learning Representations (ICLR).* 2017. [Online]. Available:

https://openreview.net/forum?id=Sy2fzU9gl [Accessed: Feb. 2021].

- Hu-Sheng W. "A survey of research on anomaly detection for time series". *13th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE. UESTC. 2016. [Online]. [Accessed: Dec. 2021].
- International Electrotechnical Comission (IEC). "High-voltage test techniques - Partial discharge measurements". *IEC 60270:2000*. Dec. 2000. [Accesed: Dec. 2021].
- Institute of Electrical and Electronics Engineers (IEEE) Dataport. *How to Cite References: IEEE Documentation Style*. [Online]. Available:

  https://ieee-dataport.org [Accesed: Dec. 2021].
- Kandanaarachchi S, Muñoz MA, Hyndman RJ. "On normalization and algorithm selection for unsupervised outlier detection". Data Mining and Knowledge Discovery. Vol. 34. No. 309–354. Nov. 2019. [Accessed: Sep. 2022].
- Kingma DP, Welling M. "Auto-Encoding Variational Bayes". *arXiv:1312.6114v11 [stat.ML].* Dec 2013. [Online]. Available at:

  https://arxiv.org/abs/1312.6114 [Accessed: Dec. 2022].
- Li S, Man Y, Zhang C, Fang Q, Li S, Deng M. "PRPD data analysis with Auto-Encoder Network". *E3S Web of Conferences WREM 2018*. Vol. 81. No. 01019. 2019. [Online]. [Accessed: Dec. 2021].
- Lu S, Chai H, Sahoo A, Phung BT. "Condition Monitoring Based on Partial Discharge Diagnostics Using Machine Learning Methods: A Comprehensive State-of-the-Art Review". IEEE Transactions on Dielectrics and Electrical Insulation. Vol. 27. No. 6. pp. 1861 – 1888. Dec. 2020. [Accessed: Dec. 2021].
- *Mark4h*. *VSB VSB_1st_place_solution*. [Competition Notebook]. Kaggle: Power Line Fault Detection Competition of Enet Centre, VSB – TU of Ostrava. *Mark4h*. 2019.

  https://www.kaggle.com/code/mark4h/vsb-1st-place-solution [Accessed Sep. 2022].
- Martinelli M, Tronci E, Dipoppa G, Balducelli C. "Electric Power System Anomaly Detection Using Neural Networks". In *Knowledge-Based Intelligent Information and Engineering Systems. KES 2004*. *Lecture Notes in Computer Science*, Negoita MG, Howlett RJ, Jain LC, Eds. Vol. 3213. Berlin. Heidelberg: Springer. 2004. [Online]. [Accessed: Dec. 2021].
- Michau G, Hsu CC, Fink O. "Interpretable Detection of Partial Discharge in Power Lines with Deep Learning". *Sensors*. Vol. 21. 2154. Mar. 2021. [Online]. [Accesed: Sep. 2022].
- Mišák S, Fulnecek J, Vantuch T, Buriánek T, Jezowicz T. "A complex classification approach of partial discharges from covered conductors in real environment". *IEEE Transactions on Dielectrics and Electrical Insulation*. Vol. 24. No. 2. Apr. 2017. [Online]. [Accesed: Jun. 2022].

- Mišàk S, Fulneček J, Vantuch T, Prokop L. “Towards the Character and Challenges of Partial Discharge Pattern Data Measured on Medium Voltage Overhead Lines”. *20th International Scientific Conference on Electric Power Engineering (EPE)*. IEEE. May. 2019. [Online]. [Accesed: Jun. 2022]

- Mišàk S, Hamacek S, Bilík P, Hofinek M, Petvaldský P. “Problems Associated With Covered Conductor Fault Detection”. *11th International Conference on Electrical Power Quality and Utilisation*. IEEE. Jan. 2011. [Online]. [Accesed: Jun. 2022].

- Mišák S, Pokorný V. “Testing of a Covered Conductor’s Fault Detectors”. *IEEE Transactions on Power Delivery*. Vol. 30. No. 3. Jun. 2015. [Online]. [Accesed: Jun. 2022].

- Nussbaum P. *VSB Power using Autoencoding V09*. [Competition Notebook]. Kaggle: Power Line Fault Detection Competition of Enet Centre, VSB – TU of Ostrava. Nussbaum P. 2019.

  https://www.kaggle.com/code/pnussbaum/vsb-power-using-autoencoding-v09 [Accessed Sep. 2022].

- Radhakrishnan A, Yang K, Belkin M. “Memorization in Overparameterized Autoencoders”. *arXiv:1810.10333v3 [cs.CV]*. Sep. 2019. [Online]. Available:

  https://arxiv.org/abs/1810.10333 [Accessed: Feb. 2023].

- Schmid M, Rath D, Diebold U. “Why and How Savitzky–Golay Filters Should Be Replaced”. ACS Measurement Science. Vol. 2. No. 2. pp. 185-196. Feb. 2022. [Online]. Available:

  https://pubs.acs.org/doi/10.1021/acsmeasuresciau.1c00054 [Accessed: Sep. 2022].

- Schwaber K, Sutherland J. The Definitive Guide to Scrum: The Rules of the Game. Nov. 2020. [Online]. Available:

  https://scrumguides.org/download.html [Accessed Sep. 2022].

  https://www.scrum.org/resources/scrum-guide [Accessed Sep. 2022].

- Universidad de Cantabria. *Qué es Ucrea*. [Online]. Available:

  https://repositorio.unican.es/xmlui/themes/unican/lib/que_es_ucrea.pdf [Accesed: Apr. 2023].

- Vantuch T. “Analysis of Time Series Data”. PhD Thesis. Technical University of Ostrava. Ostrava. 2018. [Online]. [Accesed: Jun. 2022].

- Xu H, Chen W, Zhao N, Li Z, Bu J, Li Z, Liu Y, Zhao Y, Pei D, Feng Y, Chen J, Wang Z, Qiao H.” Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs inWeb Applications”. arXiv:1802.03903v1 [cs.LG]. Feb 2018. [Online]. Available:

  https://arxiv.org/abs/1802.03903 [Accessed: Feb 2023].

- Zhang X, Pang B, Liu Y, Liu S, Xu P, Li Y, Liu Y, Qi L, Xie Q. “Review on Detection and Analysis of Partial Discharge along Power Cables”. Energies. Vol. 14. No. 22. 7692. Jun. 2021. [Online]. Available:

  https://www.mdpi.com/1996-1073/14/22/7692 [Accessed: Dec. 2021].

- Zhang H, Blacjbuirn TR, Phung BT, Sen D. “A novel wavelet transforms technique for on-line partial discharge measurement”. *IEEE Transactions on Dielectrics and Electrical Insulation*. Vol. 14. No. 1. pp. 3-14. Feb. 2007. [Accessed: Jun. 2022].