Máster en Ciencia de Datos

Trabajo fin de máster

Deep learning para aumentar la resolución temporal de modelos climáticos

Universidad de Cantabria

Predictia Intelligent Data Solutions, SL

Autor: Rubén Alonso Salas

Directores: Javier Díez Sierra y Daniel San Martín Segura

Septiembre 2024

Agradecimientos

En primer lugar, quiero dar las gracias a mis padres no solo por apoyarme y animarme en todo momento, sino también por darme la oportunidad de formarme académicamente para poder disfrutar de un futuro prometedor.

Agradecer a la empresa Predictia y a sus trabajadores la oportunidad de realizar las prácticas y el trabajo de fin de máster en un entorno tan familiar.

Agradecer a los profesores del máster en Ciencia de Datos por su tiempo, dedicación y por compartir sus conocimientos. Sus diferentes enfoques y perspectivas resultan claves en mi formación.

Por último, quiero dar las gracias a María, por apoyarme en los momentos menos buenos y confiar en mi en todo momento.

Resumen

Se ha desarrollado un modelo de aprendizaje profundo con el propósito de aumentar la resolución temporal de datos de precipitación, transformándolos de una escala diaria a horaria. Este avance resulta clave para mejorar la precisión en las proyecciones climáticas que requieren datos más detallados.

Para lograr este objetivo, se han ampliado y adaptado una serie de librerías de Aprendizaje Profundo, diseñadas e implementadas por la empresa Predictia.

El proyecto siguió varias fases clave. En primer lugar, se procesó y cargaron los datos, garantizando su correcta preparación para el entrenamiento. A continuación, se desarrolló el modelo de aprendizaje profundo UNet. Posteriormente, el modelo se integró en MLflow, lo que permitió generar predicciones. Estas predicciones fueron comparadas con observaciones reales, facilitando un análisis detallado de los resultados y la identificación de áreas para mejorar en futuros trabajos.

Palabras Clave: aprendizaje profundo, red neuronal, UNet, modelos climáticos, desagregación temporal, precipitación

Abstract

A deep learning model has been developed in order to increase the temporal resolution of precipitation data, transforming them from a daily to an hourly scale. This advance is key to improving the accuracy of climate projections that require more detailed data.

To achieve this objective, a series of Deep Learning libraries, designed and implemented by the company Predictia, have been expanded and adapted.

The project followed several key phases. First, the data was processed and loaded, ensuring its correct preparation for training. Next, the deep learning model UNet was depeloped. Subsequently, the model was integrated into MLflow, which enabled the generation of predictions. These predictions were compared with real observations, facilitating a detailed analysis of the results and identifying areas for improvement in future work.

Key words: deep learning, neural network, UNet, climate models, temporal downscaling, precipitation

Índice general

Capítulo 1

Introducción

1.1. Motivación

La correcta gestión de recursos y planificación urbana de las sociedades está ligada al conocimiento de los eventos climáticos y meteorológicos, puesto que la capacidad de desagregar datos a escalas temporales más finas, como horas o minutos, permite una evaluación más detallada y completa. En particular, los datos de precipitación son especialmente importantes para la gestión de inundaciones, el funcionamiento de la agricultura de precisión y muchos otros ámbitos relacionados con la riqueza y el desarrollo de las sociedades.

En los últimos años, se ha presenciado un incremento notable en la frecuencia e intensidad de fenómenos climáticos extremos. Estas condiciones adversas presentan un alto impacto en sectores críticos como la agricultura, la salud y la industria [1][2]. Esta tendencia ha provocado un creciente interés en la comunidad global por conocer la predicción meteorológica en detalle, ya que las consecuencias pueden ser catastróficas tanto a nivel económico como humano [3].

La agricultura, uno de los pilares fundamentales de la alimentación a nivel mundial, es especialmente vulnerable a las fluctuaciones climáticas. Sequías prolongadas, inundaciones repentinas y tormentas extremas pueden alterar gravemente los cultivos, modificando así tanto la producción como la disponibilidad de los alimentos [4]. Esta situación provoca un desafío para los agricultores, así como para los sistemas de distribución de los alimentos. Es por eso que tener datos de precipitaciones desagregados pueden guiar decisiones de riego más efectivas y adaptarse a condiciones climáticas cambiantes, maximizando la producción y minimizando pérdidas.

En el ámbito de la salud, las olas de calor pueden aumentar notablemente el riesgo de padecer afecciones relacionadas con las altas temperaturas, como la deshidratación y los golpes de calor, especialmente en poblaciones vulnerables como los niños pequeños y ancianos [5]. A su vez, los eventos de precipitación extrema y las inundaciones pueden no solo generar muertes directas, sino también muertes indirectas debido a la propagación enfermedades relacionadas con el agua, como la diarrea y el cólera [6]. La capacidad de predecir con exactitud la intensidad y frecuencia de las precipitaciones ayuda a diseñar infraestructuras de drenaje más eficientes y a planificar mejor los sistemas de abastecimiento de agua potable, lo que reduce el riesgo de inundaciones y la contaminación de fuentes de agua. Esto, a su vez, disminuye la probabilidad de brotes de

enfermedades transmitidas por el agua y mejora la capacidad de respuesta de los servicios de salud ante emergencias climáticas.

Por otro lado, el sector industrial también se ve afectado por las condiciones climáticas extremas. Eventos como tormentas extremas, huracanes e incendios forestales pueden alterar sustancialmente la cadena de suministro, la distribución de los productos y las infraestructuras. Estas perturbaciones pueden tener un impacto en la producción y la rentabilidad de las empresas [7]. La desagregación temporal de precipitaciones permite desarrollar planes de contingencia más efectivos y ajustados a las condiciones meteorológicas precisas. Igualmente optimiza la gestión de inventarios y la programación de la producción, reduciendo costos operativos y fortaleciendo la resiliencia general de la empresa ante eventos climáticos adversos.

Por todo lo anterior, en respuesta a estas crecientes amenazas, motivadas por el cambio climático [8], se vuelve indispensable disponer de proyecciones climáticas fiables, particularmente en el contexto de las precipitaciones. La disponibilidad de datos de precipitaciones detallados y desagregados, que incluyan resoluciones temporales finas como horarias, permite una evaluación más precisa de los eventos extremos como tormentas intensas, lluvias torrenciales y sequías prolongadas. A diferencia de las predicciones meteorológicas, que ofrecen información a corto plazo con alta resolución, la necesidad que motiva este trabajo es la de obtener datos de precipitación horaria a partir de datos con resolución diaria. Estos datos detallados son fundamentales para modelar y anticipar la dinámica de los eventos atmosféricos, facilitando la planificación y ejecución de estrategias de mitigación y respuesta.

Los métodos de aprendizaje automático, con su habilidad para capturar patrones complejos y aprender de grandes volúmenes de datos, ofrecen una oportunidad para superar las limitaciones de los enfoques tradicionales y proporcionar pronósticos más precisos y útiles. Al desarrollar estos modelos, se busca no solo mejorar la calidad de las predicciones, sino también contribuir a la toma de decisiones informadas y a la mitigación de riesgos asociados con eventos climáticos extremos, beneficiando así a comunidades y sectores económicos que dependen de una comprensión detallada del clima. Además, es conocido que en la sociedad actual existe una creciente demanda de información actualizada y precisa debido a la necesidad de tomar decisiones informadas en tiempo real.

En última instancia, invertir en la recopilación, análisis y difusión de información climática es clave para enfrentar los desafíos del cambio climático que debe afrontar la humanidad. Esto es aún más relevante en un contexto donde el aprendizaje automático está jugando un papel cada vez más crucial en el análisis de datos climáticos. Los avances en las técnicas permiten procesar y analizar grandes volúmenes de datos climáticos con una precisión sin precedentes, facilitando la identificación de patrones complejos y la creación de modelos predictivos más sofisticados.

1.2. Estado del arte

La desagregación temporal de datos de precipitación es una técnica empleada en el campo de la ciencia de datos y la modelización climática. Su objetivo es transformar series temporales de datos de precipitación, que generalmente tienen una resolución temporal baja (como diaria o mensual), en series de mayor resolución temporal (como horaria o incluso sub-horaria). Esta capacidad es fundamental para aplicaciones que requieren una resolución temporal más

fina, como la gestión de recursos hídricos, la planificación agrícola y la simulación de eventos extremos.

A la hora de predecir la precipitación, se pueden identificar dos enfoques principales. El primero se basa en la resolución de las ecuaciones físicas que gobiernan la dinámica de la atmósfera. Estas ecuaciones, conocidas como las ecuaciones de Navier-Stokes para fluidos en movimiento, junto con otras relacionadas con la termodinámica y la radiación, permiten modelar de manera detallada los procesos atmosféricos [9]. Sin embargo, este enfoque requiere una enorme cantidad de recursos computacionales, debido a la complejidad y escala de los cálculos necesarios para simular la atmósfera con precisión. Aunque los modelos físicos son potentes, no son capaces de predecir la precipitación de forma precisa en términos temporales y espaciales para determinadas tareas [10].

El segundo enfoque se basa en el desarrollo de modelos estadísticos, los cuales dependen del análisis de los datos históricos. Estos modelos no resuelven las complejas ecuaciones físicas de la atmósfera, sino que establecen relaciones estadísticas entre los datos observados para realizar predicciones. Este enfoque presenta la ventaja de ser menos costoso computacionalmente en términos de procesamiento y tiempo de ejecución, lo cual lo hace adecuado cuando los recursos computacionales son limitados. De esta forma, se establecen modelos estadísticos capaces de identificar correlaciones entre variables meteorológicas y generar pronósticos a partir de nuevos datos. Sin embargo, no todo son ventajas, ya que su precisión depende fuertemente de la calidad y cantidad de los datos proporcionados, tendiendo a ser menos fiables cuando se enfrentan a situaciones atmosféricas poco habituales o cambios climáticos [10].

En algunos casos, se pueden combinar ambos métodos, utilizando modelos híbridos que integren el conocimiento físico con técnicas estadísticas, mejorando así la precisión y la resolución de los pronósticos.

En cuanto a las bases de datos de precipitación se pueden destacar cuatro grandes tipos: redes de estaciones meteorológicas, datos satelitales, observaciones de radares terrestres y las desarrolladas a partir de modelos climáticos [11].

Las redes de estaciones meteorológicas utilizan instrumentos como los pluviómetros, que miden la cantidad de precipitación de manera puntual. Aunque son consideradas la fuente más confiable de datos de precipitación, su cobertura espacial y temporal puede ser limitada para determinadas tareas, especialmente en aquellas áreas donde no hay estaciones disponibles. Un ejemplo sería la red meteorológica de AEMET (Agencia Estatal de Meteorología).

Los satélites proporcionan datos de precipitación mediante la medición de radiación visible, infrarroja y de microondas. La principal ventajas de este tipo de base de datos frentes a las estaciones meteorológicas es que ofrecen una cobertura prácticamente global, alcanzado zonas donde no hay estaciones de medición terrestre.

Los radares terrestres estiman la precipitación midiendo la radiación de microondas reflejada por las partículas suspendidas en la atmósfera. Las principales desventajas de este tipo de base de datos son que no cubren toda la superficie terrestre y la incapacidad para atravesar diferentes barreras como edificios. A su vez, al igual que ocurre con las bases de datos satelitales, esta tecnología se encuentra operativa desde finales del siglo anterior, cubriendo por lo tanto una cobertura temporal reducida respecto a las redes de estaciones meteorológicas.

Los modelos climáticos globales (GCMs) simulan la interacción del océano, la atmósfera, el relieve terrestre y el hielo para generar datos de precipitación. Estos modelos se pueden usar tanto para reproducir datos históricos como para realizar proyecciones de futuros escenarios. Los GCMs se suelen utilizar como condiciones de contorno de los modelos climáticos regionales (RCMs), los cuales presentan una mayor resolución temporal y espacial.

Cada uno de estos tipos de bases de datos presenta fortalezas y limitaciones, y su uso combinado puede mejorar la precisión y cobertura de los datos de precipitación para diferentes aplicaciones.

La desagregación temporal de la precipitación es un proceso clave a la hora de realizar proyecciones climáticas, ya que permite convertir datos de precipitación con una resolución alta (estacional, mensual, semanal) en datos de intervalos más cortos (diarios, horarios, sub-horarios). Esta técnica resulta fundamental para mejorar la precisión y la utilidad de los modelos climáticos a nivel local y regional.

Un aspecto clave de este método es su capacidad para resolver eventos extremos, como tormentas intensas o sequías, que suelen ocurrir en escalas de tiempo cortas (horas o días), y los datos con resolución mensual o estacional no capturan su dinámica con precisión [12]. Estos eventos extremos pueden tener efectos devastadores sobre los ecosistemas y las infraestructuras, y no disponer de la información precisa para enfrentarlos puede subestimar sus riesgos y consecuencias.

Además, los modelos hidrológicos dependen de disponer de datos de precipitación de alta resolución temporal [13] para simular el comportamiento de los ríos, la saturación de los suelos, el escurrimiento superficial y la infiltración. Al disponer de información más precisa sobre la distribución de la lluvia, se puede modelar con mayor realismo la crecida de los ríos o la capacidad de infiltración del duelo durante un periodo de precipitación intensa, resultando en una gestión del riesgo más efectiva [14].

En el ámbito de la evaluación de riesgos climáticos, la ocurrencia de fenómenos como inundaciones o erosiones dependen en mayor medida de la intensidad y la distribución de la precipitación en periodos cortos de tiempo, que por la cantidad total de precipitación acumulada en un mes o una estación [15]. En áreas urbanas, donde los sistemas de drenaje son frecuentemente inadecuados para enfrentarse a las lluvias intensas, disponer de datos de precipitación con una mayor resolución puede ayudar a planificar de forma más eficiente la infraestructura de drenaje de la ciudad y diseñar estrategias para reducir el riesgos de inundaciones repentinas[16].

Asimismo, en la agricultura y en la gestión de los ecosistemas, los cultivos y los sistemas naturales responden de diferente forma a fenómenos de precipitación intensa que a lluvias menos intensas pero más prolongadas en el tiempo [17]. A su vez, el abastecimiento de agua potable y el uso eficiente del agua en sistemas de riego se ve influenciado por la distribución de lluvias, por lo que la desagregación temporal de precipitación resulta decisiva para mejorar la eficiencia de los sistemas de disponibilidad del agua.

En resumen, la desagregación temporal de precipitación no solo mejora la capacidad de los modelos climáticos para realizar proyecciones climáticas más precisas, sino que también es vital para reproducir fenómenos críticos y eventos extremos con mayor detalle. Este enfoque tiene implicaciones directas en la toma de decisiones en sectores como el manejo de agua, la agricultura, la planificación urbana y la mitigación de las consecuencias del cambio climático. Así, al

disponer de datos más precisos, es posible desarrollar e implementar una serie de estrategias de adaptación y mitigación más efectivas, mejorando la resiliencia contra eventos climáticos extremos.

Sin embargo, la desagregación temporal de datos meteorológicos ha sido un desafío debido a la falta de herramientas adecuadas para realizar estas conversiones de manera eficiente.

En este contexto, se ha desarrollado una herramienta eficaz para la desagregación temporal de series meteorológicas llamada MELODIST [18]. Este software de código abierto, escrito en Python, permite convertir series diarias en series horarias de manera eficiente. MELODIST es capaz de desagregar variables meteorológicas clave utilizadas en el modelado geocientífico, como la temperatura, la precipitación y la humedad, facilitando así un análisis más detallado de los datos.

MELODIST presenta tres métodos para desagregar los datos de precipitaciones diarias en intervalos horarios [18]. El primero se conoce como redistribución equitativa y es el método más sencillo, el cual consiste únicamente en dividir la precipitación diaria entre 24 para obtener las precipitaciones horarias. El segundo se conoce como modelo de cascada, es un método más avanzado y probabilístico que desagrega la precipitación diaria en intervalos más pequeños de tiempo. El modelo divide el tiempo en pasos sucesivos, duplicando la resolución en cada uno y ajustando la distribución de la precipitación de acuerdo a probabilidades específicas. Aunque este método conserva el total de la precipitación diaria, no siempre mantiene las relaciones temporales de los eventos de precipitación. Por último, se encuentra la redistribución basada en otra estación. Este método se utiliza cuando hay estaciones cercanas que registran datos horarios. Se toma la curva de distribución horaria de una estación con datos detallados y se ajusta para estimar las precipitaciones horarias en estaciones que solo registran datos diarios.

Esta herramienta se ha empleado para la desagregación de precipitación utilizando el método de cascada, obteniendo histogramas de las series desagregadas y observadas que muestran distribuciones similares. Aunque el modelo sobrestima la duración de los eventos, las características generales de la precipitación sub-diaria se capturan de manera precisa [18].

Asimismo, las Redes Generativas Antagónicas (GANs) representan un método prometedor para la desagregación temporal de precipitaciones mediante el uso de redes neuronales [19]. Este tipo de redes presentan dos componentes principales: un generador, que genera datos sintéticos de alta resolución temporal (como precipitaciones horarias a partir de datos diarios), y un discriminador, que evalúa la calidad de los datos generados realizando comparaciones con los reales. Mediante un proceso competitivo entre ambos modelos, el generador trata de mejorar la precisión de los datos desagregados, generando series temporales de precipitaciones que conservan las características estadísticas originales. Esta técnica es especialmente útil cuando los datos observados son escasos o incompletos, ya que las GANs pueden producir información precisa y coherente en escalas más finas. Aunque su implementación es más compleja y demanda mayor poder computacional, las GANs presentan un gran potencial para mejorar la resolución temporal de precipitaciones, capturando tanto la variabilidad como los eventos extremos con mayor exactitud [19].

Además, las GANs destacan por su habilidad para aprender patrones y relaciones complejas en los datos, lo que les permite generar simulaciones de precipitaciones que reproducen con mayor precisión la dinámica temporal de la precipitación observada. La habilidad de las GANs

para generar datos sintéticos que simulan las características de los datos reales es especialmente importante en zonas con datos meteorológicos limitados o inexistentes. La capacidad de las GANs para suministrar datos detallados ofrece una herramienta potente para incrementar la precisión en la desagregación temporal de precipitaciones, así como la comprensión detallada de fenómenos meteorológicos con mayor resolución.

En conclusión, la desagregación temporal de precipitaciones se ve beneficiada en gran medida de la diversidad de enfoques empleados por los métodos de aprendizaje automático. Cada técnica ofrece ventajas diferentes y conlleva desafíos específicos en la captura de patrones temporales y la generación de datos de alta resolución. Mientras que algunas, como las redes neuronales, destacan por su capacidad para modelar relaciones temporales complejas y generar con precisión datos sintéticos, otras, como MELODIST, ofrecen soluciones más simples con menor requerimiento computacional. La selección del método adecuado dependerá de la naturaleza del problema, la disponibilidad de medios y los objetivos específicos del proyecto. Combinar estos enfoques puede proporcionar una visión más completa y precisa de los eventos meteorológicos, mejorando así la gestión y planificación en diversos contextos climáticos.

1.3. Objetivos

El proyecto se enfoca en la creación de un modelo de aprendizaje automático especializado en la desagregación de datos de precipitación. Su objetivo principal es mejorar la resolución y precisión de las predicciones climáticas mediante el uso de técnicas avanzadas de downscaling. Estas técnicas están diseñadas para transformar datos de baja resolución en representaciones más detalladas, permitiendo un análisis más fino de los eventos de precipitación.

Para alcanzar este objetivo, se desarrollará un modelo basado en redes neuronales profundas, concretamente un UNet3D. Este tipo de modelo es adecuado para manejar datos tridimensionales y puede capturar patrones complejos en la variabilidad de la precipitación. La implementación del UNet3D permitirá una mayor precisión en la desagregación de datos y mejorará la calidad de las predicciones.

El proyecto también incluye una fase integral de evaluación, que abarca desde la generación de muestras a partir de datos brutos hasta el análisis detallado del desempeño del modelo. Se evaluará la efectividad del modelo en la mejora de las predicciones y se ajustará en función de los resultados obtenidos.

Finalmente, se buscará identificar áreas para optimizar el modelo y proponer mejoras continuas. Esto implicará analizar las limitaciones actuales y desarrollar recomendaciones para futuras investigaciones y aplicaciones. El objetivo es proporcionar herramientas más precisas y útiles para la predicción y gestión de eventos de precipitación, contribuyendo así a una mejor planificación y respuesta ante fenómenos climáticos.

1.4. Organización del trabajo

El trabajo se estructura en 4 capítulos, comenzando con la introducción, donde se detallan las motivaciones, el estado del arte y los objetivos del proyecto.

En el segundo capítulo del proyecto, se abordan las técnicas de downscaling utilizadas para mejorar la resolución de datos de precipitación, junto con los fundamentos del aprendizaje profundo. Se explican los conceptos clave de redes neuronales, incluyendo los distintos tipos de capas y su función en el procesamiento de datos. Además, se presenta el modelo desarrollado para el proyecto, detallando su arquitectura y cómo se aplica para desagregar datos de precipitación con mayor precisión. Este capítulo proporciona el contexto teórico y técnico necesario para comprender el enfoque del proyecto.

En el tercer capítulo, se aborda el ciclo de vida completo del proyecto, comenzando con la generación de muestras a partir de los datos en bruto. Se detalla el proceso de desarrollo del modelo, que incluye la elaboración y ajuste de las redes neuronales diseñadas para la desagregación de precipitación. Finalmente, se examina el análisis de los resultados obtenidos, evaluando el desempeño del modelo y su efectividad en la mejora de las predicciones de precipitación. Este capítulo ofrece una visión integral de cada etapa del proyecto, desde la preparación de los datos hasta la evaluación final del modelo.

En el cuarto capítulo, se presentarán las conclusiones del trabajo, destacando los logros alcanzados y los hallazgos más relevantes del trabajo. Finalmente, se abordarán las posibles mejoras que podrían implementarse para optimizar el modelo y sus resultados, ofreciendo recomendaciones para futuros trabajos en el contexto de desagregación de precipitación mediante técnicas de *deep learning*.

Capítulo 2

Metodología y técnicas

2.1. *Downscaling*

La información climática se obtiene fundamentalmente a partir de simulaciones, estando por lo tanto su desarrollo directamente ligado a los avances en computación. Los modelos de circulación general (GCMs) son simulaciones que emplean formulas matemáticas para recrear los procesos físicos que rigen el clima de la Tierra [20]. Estos modelos describen la Tierra como una rejilla tridimensional (latitud, longitud y altura) con una resolución comprendida entre los 100-400 km, aunque para algunas casos concretos se haya alcanzado una resolución inferior (25 x 25 km).

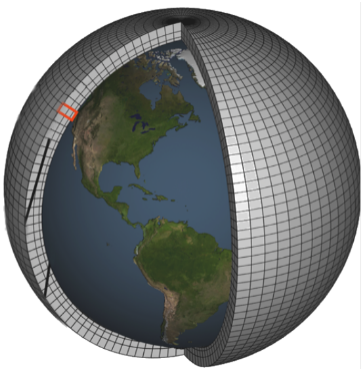

Figura 2.1: Representación en rejilla de la Tierra utilizada por los GCMs.

La aplicación de estos modelos no es directa, su baja resolución espacial provoca que no se consideren aspectos locales como la orografía, la distribución de la vegetación o el tipo de suelo [21]. A su vez, cobra especial interés capturar variaciones más detalladas en el tiempo, como patrones estacionales, diurnos o incluso intra-diurnos, los cuales son fundamentales para comprender fenómenos climáticos locales o regionales con mayor precisión [22]. Aquí es donde surge la necesidad de desarrollar una serie de técnicas de desagregación o *downscaling* que permitan incrementar tanto la resolución espacial como temporal de los modelos climáticos.

Las técnicas de *downscaling* se pueden dividir en dos grandes grupos: *Dynamical Downscaling*

(DD) y *Statistical Downscaling Methods* (SDM) [23]. La desagregación dinámica (DD) utiliza los modelos GCM y modelos climáticos regionales (RGM) para obtener resultados con una resolución fina y fuertemente ligados a los principios físicos subyacentes. Sin embargo, el coste computacional es elevado y son altamente susceptibles a los sesgos de los GCM [24]. Por otro lado, la desagregación estadística (SDM) combina los datos proporcionados por los GCM y datos históricos de estaciones locales con el objetivo de inferir relaciones entre la escala global y la escala a nivel local [25]. De esta forma, se consigue simplificar la complejidad del problema y se reduce considerablemente el coste computacional [26].

2.1.1. *Statistical Downscaling Methods*

La desagregación estadística o *statistical downscaling* es una técnica empleada para incrementar la resolución temporal o espacial de los modelos climáticos globales a partir de relaciones estadísticas entre variables climáticas a gran escala (predictores) y las variables locales de interés (predictando) [27].

A la hora de establecer estas relaciones, se consideran tres suposiciones implícitas: los predictores son variables relevantes y modelas por los GCMs, los predictores reflejan las variaciones que se producen en el clima y que la relación se mantiene estable bajo condiciones climáticas alteradas [28]. En meteorología se utilizan como predictores la temperatura, la presión, el geopotencial, la humedad relativa o específica y el viento entre otros.

Los métodos de desagregación estadística se pueden dividir en tres grandes grupos: la prognosis perfecta (PP), las estadísticas de salida del modelo (MOS) y los generadores de clima.

La técnica de prognosis perfecta (PP) se basa en establecer un modelo estadístico que relacione los predictores a gran escala con las variables locales de interés (predictandos) a partir de datos climáticos observados empíricamente. En este método cobra gran importancia la correcta selección de las variables predictoras, deben ser simuladas adecuadamente por los GCMs y representar la información climática relevante [29].

Por otro lado, el método de estadísticas de salida del modelo (MOS) adopta un enfoque diferente. Esta técnica establece una función de transferencia entre los datos del modelo y los observados, y luego se aplica esta función de transferencia para analizar posteriormente los datos del modelo. En el contexto del clima, el método MOS se puede considerar como una técnica de corrección de sesgos (*bias correction*), siendo los predictores y los predictandos la misma variable [30]. A su vez, esta técnica soluciona algunos de los problemas asociados con la PP, como no predecir correctamente eventos muy intensos o tener dificultades con la variabilidad en el tiempo y el espacio [31].

Por último, se encuentra la técnica de generadores de clima, que se basa en construir series temporales sintéticas mediante un generador de números aleatorios de manera que sus estadísticas coincidan con la de los datos disponibles. Sin embargo, es importante tener en cuenta que las series de tiempo generadas solo coinciden estadísticamente con las observaciones y pueden requerir técnicas adicionales, como el remuestreo, para ajustarse a intervalos de tiempo específicos [32]. A diferencia de los métodos deterministas, donde una entrada en un modelo siempre produce la misma salida, los métodos estocásticos se caracterizan por su aleatoriedad, requiriendo por lo tanto de un número elevado de ejecuciones para obtener resultados y conclusiones fiables.

2.2. Aprendizaje Profundo

El aprendizaje profundo, también conocido como *Deep Learning*, es una rama de la inteligencia artificial que ha revolucionado la forma en que las máquinas pueden aprender a partir de datos. Esta técnica se inspira en el funcionamiento del cerebro humano y su capacidad para procesar información de manera jerárquica y abstraer características complejas.

Los modelos de aprendizaje profundo emplean redes neuronales profundas para automáticamente adquirir representaciones cada vez más sofisticadas de los datos a medida que atraviesan las múltiples capas de la red. Esta organización jerárquica posibilita que los modelos sean altamente adaptables y capaces de identificar patrones complejos en datos estructurados y no estructurados, como imágenes, sonido y texto.

Las redes neuronales constan de estructuras más simples, las neuronas, las cuales se organizan formando capas. La capa de entrada recibe los datos iniciales, cada neurona realiza una suma ponderada cuyos pesos se ajustan durante el proceso de aprendizaje, y se transmite la información hacia la siguiente capa. Además, se añade un sesgo o *bias*. Hasta hora, únicamente se han realizado operaciones con funciones lineales, dando lugar a su vez a una función lineal. De forma que, para considerar el carácter no lineal de los datos es necesario el uso de una función de activación.

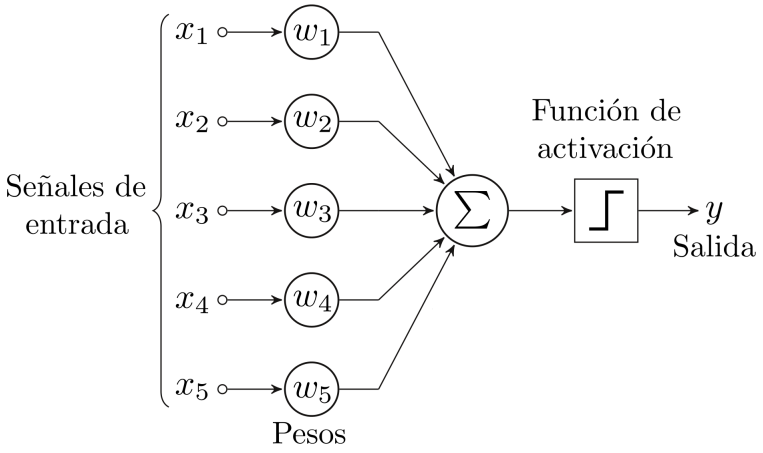

Figura 2.2: Esquema de un perceptrón que recibe como input cinco valores (x_1, x_2, x_3, x_4 y x_5), los cuales se ponderan mediante los pesos (w_1, w_2, w_3, w_4 y w_5), se añade el sesgo y el resultado se pasa a través de una función de activación para obtener una salida (y).

Existe una gran variedad de funciones de activación, el uso de una u otra va a depender del tipo de problema que se considere y el objetivo que se quiera alcanzar. A continuación, se describen las funciones de activación empleadas en este trabajo:

- Sigmoide [0,1]: proporciona una salida acotada entre 0 y 1, siendo 0 cuando la entrada es muy baja y 1 cuando es muy alta. Su principal aplicación es la clasificación binaria.

$$\sigma(x) = \frac{1}{1 + e^{(-x)}} \tag{2.1}$$

- Tangente hiperbólica [-1,1]: proporciona una salida acotada entre -1 y 1.

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.2}$$

- ReLU (*Rectified Linear Unit*): función no lineal a trozos que genera una salida igual a 0 cuando la entrada es negativa y una salida igual a la entrada cuando esta última es positiva. Soluciona fundamentalmente el problema de desvanecimiento del gradiente presente en las funciones sigmoide y tangente hiperbólica. Además, es más fácil de implementar computacionalmente en comparación con las otras dos funciones.

$$f(x) = max(0, x) \tag{2.3}$$

A partir del diagrama de una neurona presentado en la Figura 2.2, se construye una estructura como la presentada en la Figura 2.3. Se representa una red neuronal densa, donde cada neurona se encuentra conectada a todas las neuronas de la capa siguiente. El número de neuronas de cada capa y el número de capas se define al inicio del método, aumentando generalmente con la complejidad del problema considerado. Por ello, este tipo de aprendizaje se conoce como profundo.

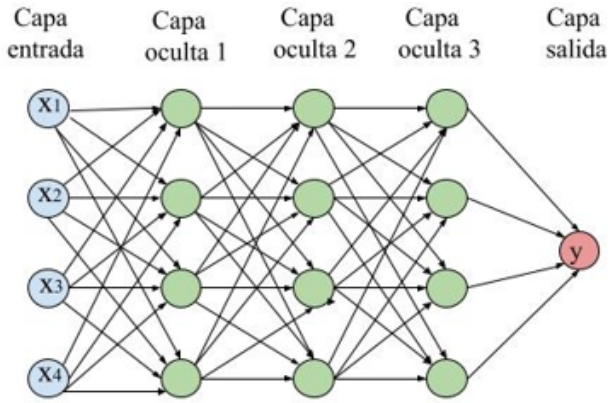

Figura 2.3: Esquema de una red neuronal densa con una capa de entrada con cuatro neuronas, tres capas ocultas con cuatro neuronas cada una y una capa de salida con una única neurona.

El entrenamiento de una red neuronal implica un proceso iterativo de optimización (ver Figura 2.4), donde cada iteración se conoce como un *epoch*. Durante este proceso, la red neuronal aprende a través de una función llamada función de pérdida. Esta función compara las predicciones de la red con los valores reales y evalúa la discrepancia entre ellos. Existe una gran variedad de funciones de pérdida. A continuación, se presentan las empleadas en este trabajo.

- Raíz error cuadrático medio (RMSE): mide el promedio de los errores al cuadrado.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{2.4}$$

- Error absoluto medio (MAE): mide el promedio de los errores en valor absoluto.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2.5}$$

- Función de pérdida de Chabonnier: es una variación suavizada de la función L1.

$$L(\hat{y}, y) = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(\hat{y}_i - y_i)^2 + \epsilon^2} \tag{2.6}$$

  Donde $\hat{y}_i$ son los valores obtenidos con el modelo, y_i los valores reales, N el número total de muestras y ϵ es un parámetro de suavización que evita la singularidad cuando la diferencia entre $\hat{y}_i$ e y_i es cercana a 0.

La red utiliza esta información para ajustar los pesos de las neuronas en todas las capas de la red de manera que la discrepancia se minimice. Este ajuste de los pesos es realizado por un optimizador (como *RMSprop* o *Adam*) que implementa el algoritmo de retropropagación o *Backpropagation*. El objetivo del optimizador es encontrar los valores óptimos de los pesos que reduzcan el valor de la función de pérdida, lo que permite a la red mejorar su capacidad de hacer predicciones precisas.

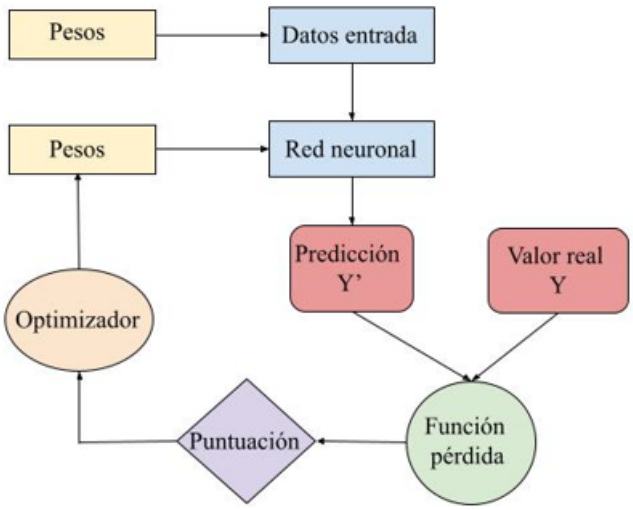

Figura 2.4: Esquema del proceso de aprendizaje de una red neuronal.

El algoritmo de *backpropagation* calcula el gradiente de la función de pérdida con respecto a los pesos de la red neuronal, utilizando el método del descenso del gradiente para ajustar los pesos y minimizar la pérdida. Durante este proceso, se multiplica el gradiente por un escalar conocido como tasa de aprendizaje o *learning rate*, que determina la magnitud de los ajustes de peso en cada iteración. Este hiperparámetro es de gran importancia, ya que un valor pequeño puede provocar que el proceso de aprendizaje dure demasiado tiempo, mientras que un valor elevado puede derivar en lo que se conoce como desvanecimiento del gradiente. A su vez, se propaga el error desde la capa de salida hacia atrás a través de la red, calculando gradientes parciales en

cada capa para actualizar los pesos de manera adecuada. Este proceso se repite iterativamente hasta que se alcanza una convergencia satisfactoria.

2.3. Capas

A continuación, se presentan los diferentes tipos de capas empleadas para el desarrollo del modelo de aprendizaje profundo.

2.3.1. Capa convolucional

Una convolución es una operación matemática ampliamente utilizada en el ámbito de la física y la ingeniería para el procesamiento de imágenes, señales y telecomunicaciones. Se define de la siguiente forma:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau \tag{2.7}$$

Obteniéndose una nueva función $(f*g)(t)$ que mide el grado de superposición entre las funciones f y g. Esto permite comprender cómo una función se ve influenciada por la otra a lo largo del tiempo.

Una capa de convolución es un tipo de capa fundamental en las redes neuronales convolucionales (CNNs), empleada principalmente para el procesamiento de datos con estructura cuadrada, como imágenes. Su funcionamiento se basa en aplicar una serie de filtros (kernels) sobre el mapa de características de entrada para aprender características locales, como bordes, patrones y texturas. Su principal ventaja frente a las capas densas, donde cada neurona está conectada a todas las neuronas de la capa anterior y de la capa siguiente, es la capacidad de generalización, puesto que estos patrones locales son invariantes a posibles traslaciones [33].

A parte de seleccionar el número de filtros y su tamaño (anchura y altura), se debe ajustar el paso o *stride* y el relleno o *padding*. Por un lado, el hiperparámetro *stride* indica cuántas posiciones avanza el filtro en cada paso tanto en la dirección de las filas como de las columnas. De forma que, si se aumenta el *stride* se reduce el tamaño del mapa de características resultante. Por otro lado, el *padding* consiste en agregar píxeles alrededor de la matriz de entrada de manera que la salida del filtro mantenga el mismo tamaño que el mapa de entrada original. Existen diferentes formas de agregar píxeles, que van desde la más simple como puede ser añadir 0, o técnicas de replicación y reflexión más sofisticadas. Además de con el objetivo de mantener el tamaño de la entrada, esta operación se realiza para evitar la infrarrepresentación de los píxeles que se encuentran en las esquinas, lo que se conoce como efectos de borde [34].

En la Figura 2.5 se muestra el funcionamiento de una red convolucional para un determinado tamaño de filtro, *stride* y sin usar *padding*.

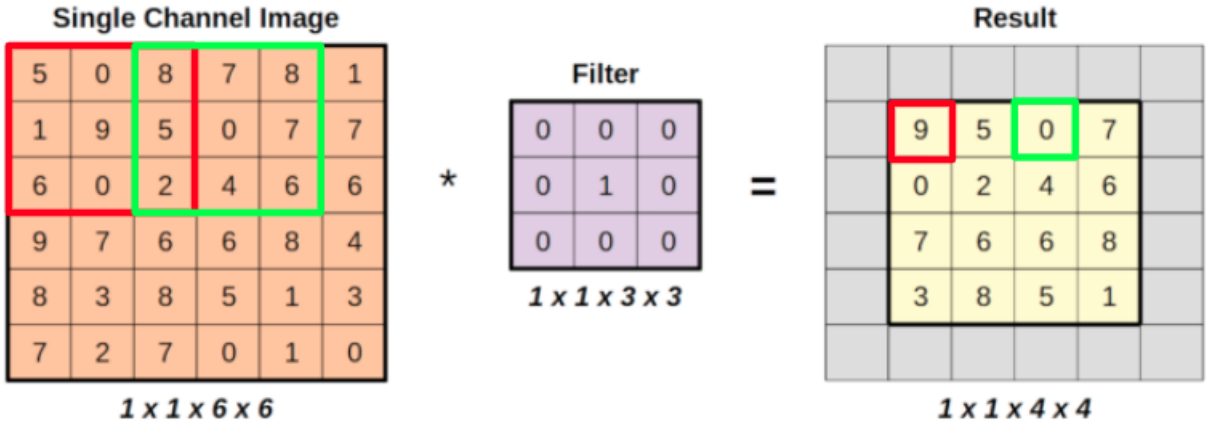

Figura 2.5: Esquema del funcionamiento de una capa convolucional. Se aplica un filtro unidad de tamaño 3×3 sobre una única imagen de 6×6, obteniéndose una matriz de salida de dimensiones 4×4. En este caso no se ha utilizado la técnica de *padding* y se ha establecido el *stride* en un valor de uno. Cada elemento se describe siguiendo la misma estructura: nºimágenes × nºcanales × píxeles fila × píxeles columna. Ref [35]

En la Figura 2.6 se muestra un esquema de la técnica de *padding*, eligiendo el método más sencillo donde se incluyen 0 en la matriz de entrada.

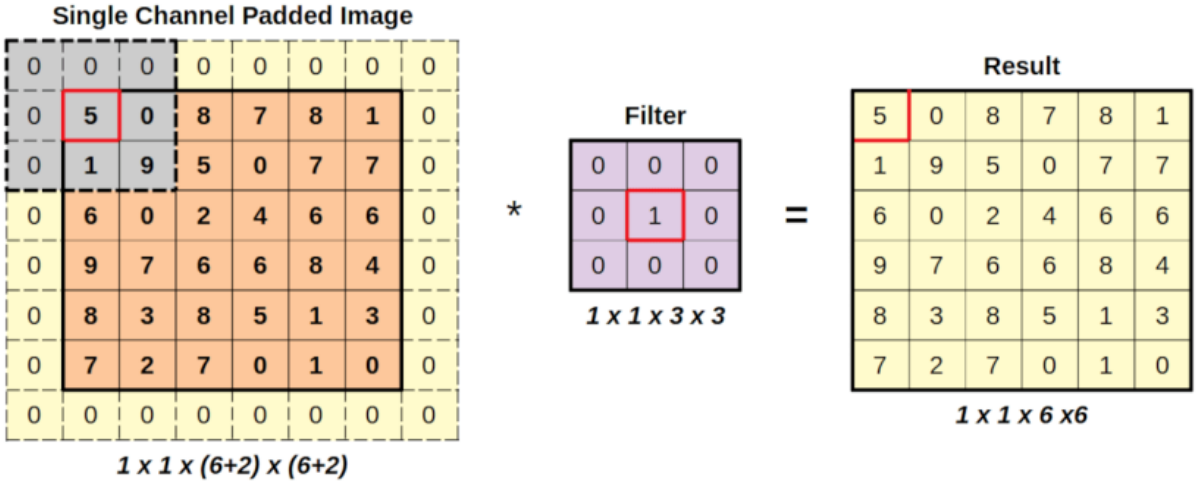

Figura 2.6: Representación de la técnica de *padding* en una capa convolucional. El paso toma un valor de 1, avanzando el filtro una posición para cada operación. El relleno es de 1, incremento en 2 unidades tanto el número de filas como el número de columnas. Ref [35].

Como se muestra en la Figura 2.6, inicialmente con el relleno se expande la imagen, se aplica un filtro y finalmente se obtiene una salida que conserva las dimensiones de la matriz original. Además de para preservar las dimensiones espaciales de la entrada original, esta técnica también se emplea con el objetivo de mejorar el rendimiento de los bordes, ya que las características en los bordes de la imagen reciben menos atención durante la convolución sin *padding*.

2.3.2. Capa convolucional transpuesta

Una capa de convolución transpuesta, también se conoce como degradada, es un tipo de capa empleada en las redes neuronales con el objetivo de aumentar la dimensión del mapa de características de entrada, incrementando así la resolución. Este tipo de capas son ampliamente utilizadas en diversas aplicaciones como mejorar la calidad de las imágenes o en segmentación semántica [36].

El funcionamiento de este tipo de capas es análogo al de las convoluciones estándar, pero de manera inversa, es decir, generan un mapa de características con una dimensión espacial mayor que la entrada original. Para ello, se utilizan los mismos hiperparámetros: *kernel*, *stride* y *padding*.

La capa de convolución transpuesta se puede describir en los siguientes pasos: inicialmente se insertan ceros entre los elementos del mapa de características de entrada, lo que incrementa la dimensión espacial. Posteriormente, se emplea el *padding*. Sin embargo, en este tipo de capas esta técnica no aumenta el tamaño de la matriz sino que lo reduce, eliminando el número de filas y columnas que se especifiquen. Por último, se aplica el filtro de forma convencional, obteniendo una salida de mayor tamaño que la matriz original.

2.3.3. Capa de *pooling*

Una capa de *pooling*, también conocida como de agrupación, es un tipo de capa empleadas en las redes convolucionales con el objetivo de reducir la dimensión de las características de una imagen, manteniendo la información de más relevancia.

Hay diferentes tipos de capas de pooling, pero la más común es la capa de pooling máximo (*max pooling*). En esta capa, se divide la imagen de entrada en regiones (por ejemplo, 2x2 o 3x3) y se toma el valor máximo de cada región como el valor de esa región en la capa de salida. Esto ayuda a conservar las características más importantes de la imagen, como bordes o texturas, mientras se reduce su tamaño.

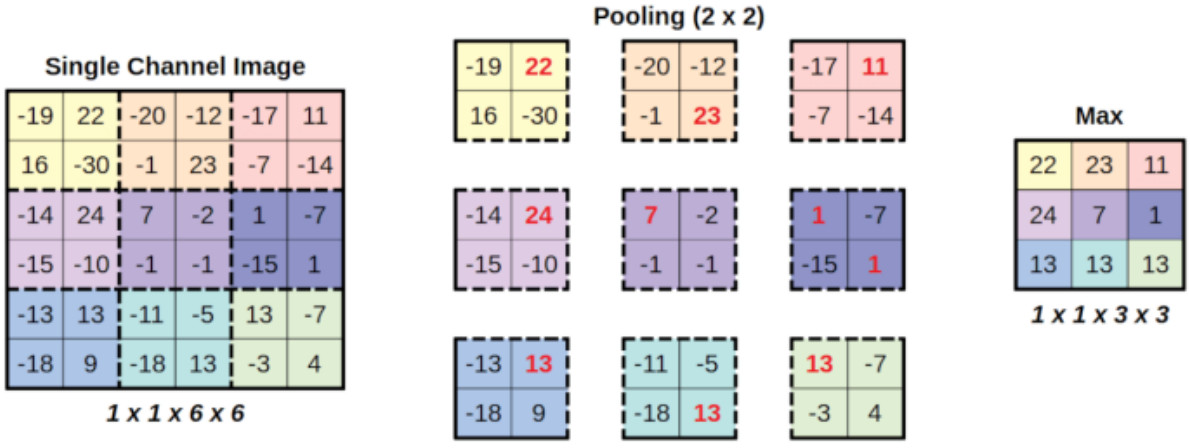

Figura 2.7: Representación de una capa de *max pooling*. Se aplica un *kernel* de tamaño 2×2 sobre una imagen 6×6, obteniéndose una matriz de salida 3×3. Se selecciona el valor máximo de cada región, reduciendo así el tamaño de la matriz resultante. Ref [35].

En la Figura 2.7 se puede observar como la matriz de salida presenta un tamaño inferior al mapa de características original. Cuanto mayor sea el tamaño del *kernel*, más reducida se va a ver la imagen de salida. Un filtro de tamaño dos reduce las dimensiones de la salida a la mitad, mientras que uno de tamaño tres lo reduciría a la tercera parte.

Existen otras operaciones de *pooling*, como podría ser el *average pooling*. En este caso, se divide la imagen de entrada en bloques y se realiza la media ponderada de los píxeles de esa región, en vez de seleccionar el valor máximo del bloque.

2.3.4. Capa de *batch normalization*

La normalización por lotes o *batch normalization* es una técnica ampliamente utilizada en el contexto de las redes neuronales con el objetivo de incrementar la estabilidad de la red y aumentar significativamente la velocidad de entrenamiento.

La idea principal de este método consiste en estandarizar los valores de activación de cada capa, fijando la media en cero y la varianza en uno. A su vez, en vez de calcular las variables estadísticas para el dataset de entrenamiento por completo, lo hace para cada mini-lote o *mini-batch*.

Para un mini-lote de entradas $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$, la media μ_B y la varianza σ_B^2 se calculan:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{2.8}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{2.9}$$

Normalización de las entradas para tener media 0 y varianza 1:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{2.10}$$

Escalado γ y desplazamiento β aprendido a lo largo del entrenamiento de la red neuronal:

$$y_i = \gamma\hat{x}_i + \beta \tag{2.11}$$

De esta forma, las variables predictoras presentan magnitudes comparables, evitando así no solo la sobrerrepresentación de una de ellas, sino también que gran parte del error se asocie con la variable de mayor magnitud.

2.4. U-Net

La estructura U-Net es una arquitectura de red neuronal originalmente empleada para la segmentación de imágenes en el contexto de la biomedicina en el año 2015 [37]. Su implementación provocó una mejora significativa respecto a los modelos utilizados en segmentación de imágenes médicas hasta esa fecha, siendo el entrenamiento del modelo sustancialmente más rápido [38].

La U-Net tiene una estructura en forma de "U" (ver Figura 2.8) que cuenta con dos partes principales: un camino de contracción (*encoder*) y un camino de expansión (*decoder*)

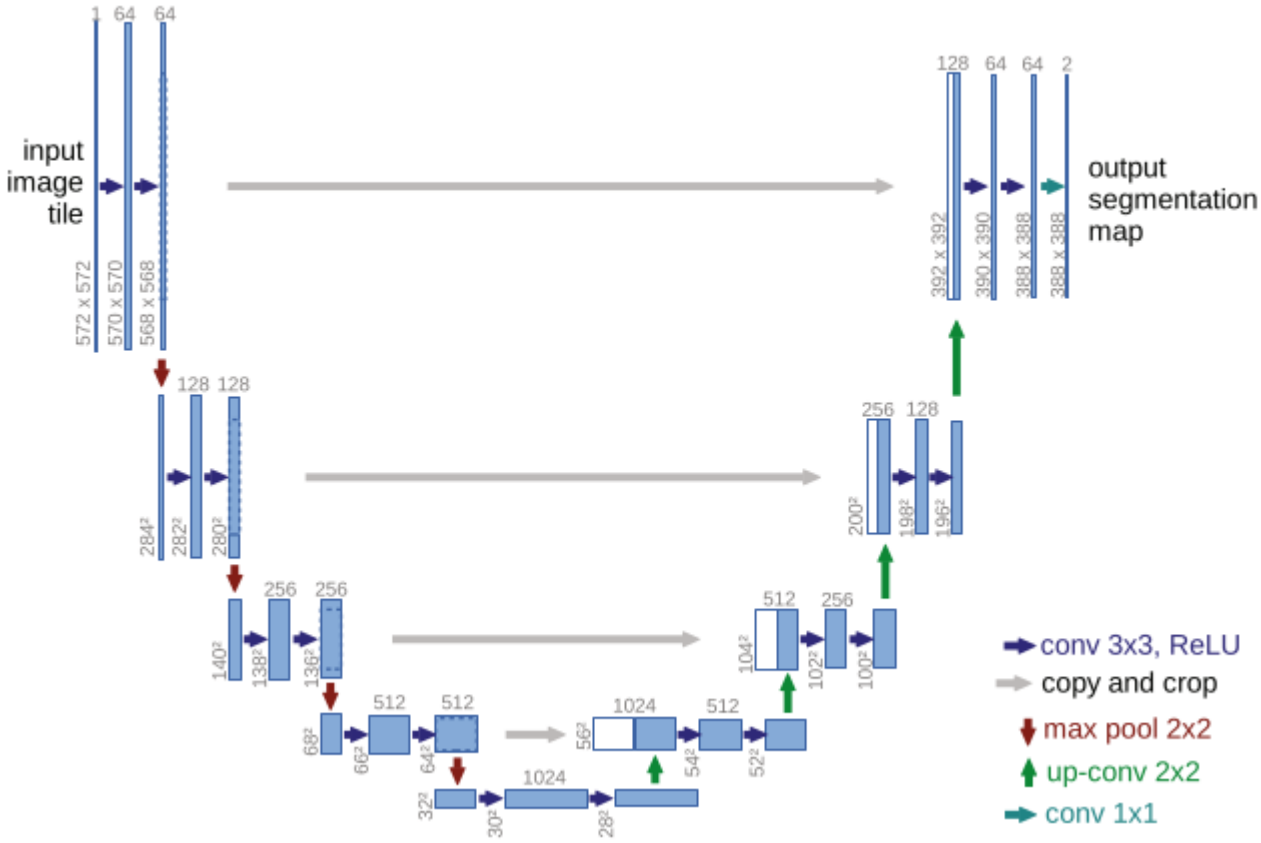

Figura 2.8: Arquitectura U-Net propuesta originalmente para la segmentación de imágenes médicas. El número de canales se indica en la parte superior de cada mapa de características (bloques azules). Las flechas definen las diferentes operaciones realizadas. Ref [37].

2.4.1. Camino de contracción (*Encoder*)

La parte de contracción o *encoder* es una red de convolución estándar que realiza una serie de operaciones de convolución, normalización y pooling para capturar información de la matriz de entrada. La función principal de esta parte es reducir la dimensión espacial de la imagen original a la vez que se aumenta la profundidad (canales), disminuyendo así el número de parámetros de la red.

El *encoder* de la arquitectura U-Net está diseñado con una estructura de doble convolución, que consiste en dos capas de convolución 3D consecutivas. Cada una de estas capas utiliza un

kernel de tamaño 3×3 y aplica un *padding* de valor 0.

Para introducir no linealidades en el modelo y mejorar su capacidad de aprendizaje, se emplea la función de activación ReLU (*Rectified Linear Unit*). Esta función transforma todas las activaciones negativas en cero, manteniendo las activaciones positivas sin cambios, lo que ayuda a acelerar el entrenamiento y a prevenir problemas de desvanecimiento del gradiente.

Además, se utiliza la técnica de normalización por lotes (batch normalization) para estabilizar y acelerar el proceso de entrenamiento. La normalización por lotes ajusta y escala las activaciones de cada capa, lo que contribuye a una convergencia más rápida y a una mejor generalización del modelo.

A continuación de las capas de convolución, se realiza una operación de *pooling* máximo (*max pooling*) con un tamaño de filtro de 2×2. Esta operación reduce las dimensiones espaciales de la imagen a la mitad, preservando las características más prominentes y relevantes mientras se disminuye la resolución espacial. El *pooling* máximo contribuye a reducir la carga computacional y a extraer características de alto nivel, facilitando así el proceso de aprendizaje del modelo.

En resumen, la estructura del *encoder* combina la convolución profunda, la activación no lineal, la normalización por lotes y el *pooling* máximo para transformar las imágenes de entrada en representaciones de características más compactas y significativas, preparándolas para el posterior proceso de decodificación y reconstrucción en el modelo U-Net.

2.4.2. Camino de expansión (*Decoder*)

La parte de expansión o *decoder* se encarga de reconstruir la imagen original a partir de las características extraídas y aprendidas en la parte de *encoder*. Esta parte de la red neuronal consta de una serie de operaciones de *upsampling* con el objetivo de incrementar la resolución espacial del mapa de características, generándose así una salida del mismo tamaño que la imagen inicial.

El camino de expansión en el modelo U-Net concluye con una capa de salida diseñada para ajustar el número de canales del resultado final de la red. Esta capa tiene la función de adaptar el número de canales de salida del modelo para que coincida con el número requerido por la tarea de segmentación o desagregación. De este modo, se asegura que la salida del modelo esté correctamente alineada con las especificaciones de la tarea.

2.4.3. U-Net 3D

La arquitectura de U-Net, originalmente desarrollada para la segmentación de imágenes médicas en 2D, ha sido adaptada para trabajar con datos volumétricos en 3D en el contexto de la desagregación de precipitación.

El modelo U-Net se configura a través de una serie de parámetros clave que determinan su arquitectura y funcionamiento. Entre estos parámetros, se encuentran las dimensiones de los datos de entrada y salida, especificadas por los valores de *input_shape* y *output_shape*. Estas dimensiones definen el tamaño volumétrico de los datos que el modelo procesará en las etapas de entrada y salida.

Asimismo, el modelo requiere la definición de los canales de entrada y salida, que están indicados por *input_channels* y *output_channels*. Estos parámetros determinan el número de canales presentes en las imágenes de entrada y salida, afectando la cantidad de información que puede ser capturada y procesada en cada capa.

El número de filtros aplicados en cada capa de convolución está especificado por el parámetro *filters*, el cual juega un papel crucial en la extracción de características a diferentes niveles de la red. El tamaño de las operaciones de *pooling*, regulado por el parámetro *pooling_size*, afecta la reducción de la dimensionalidad y la consolidación de la información a lo largo de las capas de la red.

A su vez, el método de interpolación utilizado durante el proceso de *upsampling* está determinado por *interpolation_method*. Este método es esencial para el reescalado de los datos, permitiendo que el modelo recupere y refine la resolución de los datos a medida que se realizan las operaciones de *upsampling*. El proceso de *upsampling* puede realizarse de diferente formas, destacando las capas de convolución transpuesta y métodos de interpolación bilineal, bicúbico o de vecinos cercanos. Se ha definido una variable booleana que en el caso de tomar el valor "True" realiza la interpolación correspondiente, mientras que para "False" realiza el *upsampling* con las capas de convolución transpuesta.

Estos parámetros trabajan en conjunto para definir la capacidad del modelo U-Net de procesar, extraer y reconstruir características complejas de los datos, haciendo de este un modelo potente y versátil en aplicaciones de desagregación de datos.

Capítulo 3

Resultados y análisis

En esta sección, se presentan los resultados obtenidos a lo largo de las distintas etapas del trabajo, desde la carga de datos hasta el análisis de los resultados finales. El objetivo es proporcionar una visión detallada del proceso y de los hallazgos clave. Cabe destacar que el trabajo ha consistido no en el desarrollo completo de las librerías de *Deep Meteo* implementadas por la empresa Predictia, sino en la extensión de estas para el problema de *temporal downscaling*.

Primero, se abordará el proceso de carga y procesamiento de datos utilizando la herramienta *Data-Toolkit*. Esta herramienta gestiona los datos desde su origen hasta su preparación para el análisis, detallando los procedimientos de selección, limpieza, transformación, y cualquier ajuste o estandarización aplicada. También se discutirá cómo se configuraron las características (features) y etiquetas (labels), y su impacto en la calidad y utilidad de los datos para el modelo.

A continuación, se describirá la elaboración del modelo con *Modeling-Toolkit*, incluyendo la selección de algoritmos, configuración de parámetros y validación del rendimiento. Se detallarán las técnicas de optimización empleadas y se abordarán los desafíos encontrados durante esta fase.

Posteriormente, se explicará la implementación del *Deployment Toolkit*, que facilita el desarrollo de modelos de aprendizaje automático mediante la descarga de modelos entrenados y su integración en MLflow. La herramienta genera datos de entrada que reflejan fielmente el problema objetivo y realiza inferencias, produciendo predicciones almacenadas en un archivo NetCDF. Este formato permite una gestión eficiente de datos y facilita la comparación entre las predicciones y los valores reales, mejorando la evaluación y ajuste del modelo para futuras aplicaciones.

Finalmente, se presentará un análisis detallado de las predicciones del modelo. El objetivo de este análisis es evaluar el rendimiento del modelo en la predicción de datos de precipitación, comparando los resultados obtenidos con observaciones reales extraídas del conjunto de datos ERA5, un referente ampliamente utilizado en estudios climáticos.

3.1. *Data Toolkit*

El repositorio denominado *Data Toolkit* es una herramienta especializada en la carga y transformación de datos meteorológicos, y se encuentra integrado dentro de un conjunto de librerías más amplio llamado *Deep Meteo*. Estas librerías están diseñadas específicamente para aplicar técnicas avanzadas de aprendizaje profundo en el ámbito del análisis climático, un campo en el que la predicción y el análisis preciso de fenómenos meteorológicos son de suma importancia.

Los datos originales provienen de *ERA5*, un conjunto de datos climáticos de reanálisis producido por el *Centro Europeo de Predicciones Meteorológicas a Medio Plazo (ECMWF)*. *ERA5* ofrece información meteorológica detallada, tanto a nivel superficial como a diferentes altitudes, de diversas variables, como la temperatura, presión atmosférica, humedad, radiación solar y precipitación, siendo esta última la variable de interés en este trabajo. Los datos cuentan con una resolución temporal de una hora y cubren el período desde 1940 hasta el presente, proporcionando un registro climático extenso y continuo. En cuanto a la resolución espacial, es de 0.25^{o} en longitud y latitud. Estos datos se encuentran disponibles en formato NetCDF, un formato estándar en meteorología, diseñado para manejar grandes volúmenes de datos multivariables en dimensiones espaciales y temporales.

El objetivo central de esta herramienta es facilitar el procesamiento de datos meteorológicos que, en su forma original, suelen estar almacenados en el formato NetCDF. A través de *Data Toolkit*, estos ficheros son convertidos en tensores, lo que permite que los datos sean introducidos directamente en modelos de aprendizaje automático y profundo. Este paso es crucial, ya que los tensores son estructuras de datos que permiten representar información multidimensional, adaptándose perfectamente a las necesidades de los modelos de aprendizaje profundo, que requieren un formato específico para poder procesar grandes volúmenes de datos de manera eficiente.

En el núcleo de este repositorio se encuentra la clase principal denominada *DeepMeteoDataset*, que encapsula toda la lógica común y las funcionalidades necesarias para abordar los diferentes problemas típicos dentro del marco de *Deep Meteo*. Estos problemas suelen ser de aprendizaje supervisado, donde se trabaja con variables predictoras (*features*) y variables objetivo o de salida (*labels*), que el modelo intenta predecir. El uso de este enfoque supervisado es fundamental para muchos de los problemas que se abordan en meteorología, ya que se cuenta con un conjunto de datos etiquetados, donde las *features* representan variables como la temperatura, presión o velocidad del viento, mientras que las *labels* representan los valores que se quieren predecir, como el estado futuro de dichas variables.

Dentro del ámbito de *Deep Meteo*, algunos de los problemas más relevantes que se abordan son los relacionados con la predicción del clima a futuro, como el *forecasting* (pronóstico), y los procesos de desagregación, tanto espacial como temporal. La desagregación espacial (*spatial downscaling*) busca aumentar la resolución espacial de los datos meteorológicos, mientras que la desagregación temporal (*temporal downscaling*) tiene como objetivo mejorar la resolución temporal de estos datos, permitiendo realizar predicciones más precisas en marcos de tiempo más reducidos. Estos procesos son esenciales para mejorar la calidad y resolución de los modelos meteorológicos, lo que a su vez puede tener un impacto directo en la precisión de las predicciones climáticas.

En este trabajo se presenta principalmente la implementación de la herramienta *Data Toolkit* en el contexto de la desagregación temporal (*temporal downscaling*). Para ello, se ha desarrollado la clase *TemporalDownscalingDataset*, que extiende la funcionalidad de la clase base *DeepMeteoDataset*. Además, se ha implementado la clase *TemporalAggregation*, la cual define dos atributos clave: la resolución temporal (por ejemplo, '1H' para una hora) y el método de agregación (como 'sum' o 'mean'). Estos atributos se especifican en un archivo de configuración en formato YML, donde se recoge el esquema global del problema a resolver.

A partir de la clase *TemporalAggregation*, se crea un objeto que incorpora el proceso de agregación tanto en las variables predictoras (*features*) como en las variables objetivo (*labels*). Además, dentro de esta clase, se ha definido el método *apply*, que permite aplicar la agregación temporal a un conjunto de datos de entrada en función de los parámetros de resolución y método previamente definidos en el archivo YML.

En la función *load grid data*, encargada de la carga y preprocesamiento de los datos, se realiza el *resample* adecuado tanto para las *features* como para las *labels*, asegurando así que los datos se ajusten a la resolución temporal especificada. Asimismo, se ha añadido una estructura que permite considerar, además del archivo actual, los ficheros anterior y posterior, en caso de que existan. Esta funcionalidad es especialmente útil en situaciones donde la información temporal adyacente puede proporcionar contexto adicional relevante para el modelo.

A continuación, se presenta un archivo de configuración en formato YML con el objetivo de detallar las distintas secciones y componentes que conforman la herramienta *Data-toolkit*. Este archivo sirve como una guía integral para comprender cómo está estructurada la configuración y cómo cada parámetro influye en el funcionamiento de la herramienta.

```yaml
problem:
  type: temporal_downscaling
  configuration:
    number_steps_backward: 3
    number_steps_forward: 0

data_splits:
  train:
    temporal_coverage:
      start: 1990-01
      end: 2015-01
      frequency: MS
  valid:
    temporal_coverage:
      start: 2015-01
      end: 2018-01
      frequency: MS
  test:
    temporal_coverage:
      start: 2018-01
      end: 2020-12
      frequency: MS

features_configuration:
  variables:
  - name: pr_None
    data_location: /predictia-nas2/Data/temporal-downscaling/subset/santander/pr/None/
  source_dataset: era5
  spatial_coverage:
    lat_dim_name: "lat"
    lon_dim_name: "lon"
    longitudes: [-4.0, -3.0]
    latitudes: [43.0, 44.0]
  temporal_aggregation:
    resolution: "24H"
    method: "sum"
  standardization:
    to_do: false

label_configuration:
  variables:
  - name: pr_None
    data_location: /predictia-nas2/Data/temporal-downscaling/subset/santander/pr/None/
  source_dataset: era5
  spatial_coverage:
    lat_dim_name: "lat"
    lon_dim_name: "lon"
    longitudes: [-4.0, -3.0]
    latitudes: [43.0, 44.0]
  temporal_aggregation:
    resolution: "1H"
    method: "sum"
  standardization:
    to_do: false
```

Figura 3.1: Ejemplo de fichero de configuración empleado para la herramienta de carga de datos *Data-Toolkit*.

En el fichero de configuración mostrado en la Figura 3.1, se definen tanto el tipo de problema, especificado como *temporal_downscaling*, como los parámetros de configuración *number_steps_backward*

y *number_steps_forward*. Estos parámetros se han implementado con el objetivo de generar muestras que incluyen tanto los días anteriores como los posteriores al día de estudio, proporcionando un contexto temporal más amplio para el análisis posterior. De esta forma, para un valor de *number_steps_backward* igual a 3, la herramienta *Data-toolkit* genera una muestra que cuenta con el día de estudio y los tres anteriores. Asimismo, se presenta la división del conjunto de datos en conjuntos de entrenamiento, validación y prueba, especificando tanto el inicio como el fin de cada bloque, así como la frecuencia de los datos, que en este caso es mensual.

A su vez, también se establece la configuración tanto de las *features* como de las *labels*, definiendo cuidadosamente varios aspectos clave del procesamiento de los datos. En primer lugar, se selecciona la ruta de origen de los datos, garantizando que los archivos provengan de las fuentes correctas.

A continuación, se detallan las variables espaciales, como la longitud y la latitud, que permiten localizar geográficamente las muestras. Los puntos seleccionados en la malla abarcan una región geográfica definida por una longitud que varía entre [-4.0, -3.0] y una latitud entre [43.0, 44.0] grados decimales, con una resolución de 0.25°. Dicha área está aproximadamente centrada en un región cercana a la ciudad de Santander, cuyas coordenadas geográficas aproximadas son -3.8º de longitud y 43.5º de latitud. Por lo tanto, los puntos seleccionados reprensentan las posiciones geográficas más próximas a Santander.

Posteriormente, se evalúa si es necesario aplicar una estandarización a los datos. La estandarización de los datos es un paso común en el procesamiento previo de magnitudes que presentan diferentes magnitudes o escalas, transformando los datos con una media de 0 y desviación estándar de 1. Este proceso facilita la convergencia del modelo y evita que los valores elevados dominen el entrenamiento.

Finalmente, se define la agregación temporal adecuada para las distintas variables del modelo, diferenciando entre las *features* y las *labels*.

Por un lado, se encuentran las *features* que presentan una resolución diaria. Para ello, se utiliza el método de agregación *sum* que suma la precipitación correspondiente a cada hora para obtener la precipitación diaria. De esta forma, se generan muestras con resolución de 24h (diaria) adecuadas para el entrenamiento del modelo.

Por otro lado, las *labels* tienen una resolución horaria. Esto implica que los valores que el modelo busca predecir se registran en intervalos de una hora. La razón de esta diferencia en la resolución se debe a que se quiere evaluar la capacidad del modelo para desagregar los datos diarios en datos horarios, es decir, analizar cómo puede el modelo tomar información de entrada diaria y generar predicciones más detalladas a nivel horario.

3.2. *Modeling Toolkit*

El repositorio denominado *Modeling Toolkit* es una herramienta integral diseñada para gestionar y facilitar todo el proceso de modelización en proyectos de aprendizaje automático. Esta librería abarca desde la definición y desarrollo de las arquitecturas de los modelos hasta su entrenamiento, optimización y ajuste de hiperparámetros. Además, se encarga de aspectos críticos como el seguimiento de experimentos y el registro de modelos, asegurando una trazabilidad y gestión adecuada durante el ciclo de vida de los proyectos.

Una de las características destacadas del *Modeling Toolkit* es su integración directa con *MLflow*, una plataforma de código abierto ampliamente utilizada en el sector. *MLflow* permite gestionar de manera eficiente el ciclo de vida completo de un proyecto de *machine learning*, abarcando todas las etapas clave: desde la experimentación inicial, donde se prueban diferentes configuraciones y arquitecturas, hasta la fase de producción, en la que se despliegan los modelos en entornos reales. Esto incluye la capacidad de versionar modelos, registrar métricas y parámetros, y realizar el seguimiento de los resultados de los experimentos.

En este contexto, se ha implementado el modelo U-Net para el desarrollo del proyecto, haciendo uso de la librería de *PyTorch* y la biblioteca *Transformers*. Esta última se utiliza para la gestión de configuraciones, heredando de *PretrainedConfig* y empleando *PreTrainedModel* para la construcción y el entrenamiento del modelo.

En primer lugar, se define la configuración del modelo, especificando aspectos clave como las dimensiones del modelo, el número de canales de entrada y salida, los filtros utilizados en las capas de convolución, el tamaño del *kernel* y la normalización de los datos. Esta configuración asegura que el modelo esté adecuadamente ajustado a las necesidades del proyecto y optimiza su rendimiento.

Posteriormente, se implementa la arquitectura del modelo U-Net. Esta arquitectura se caracteriza por sus operaciones de *downsampling*, que reducen la dimensión espacial de las características y permiten captar información de contexto a nivel más alto, y operaciones de *upsampling*, que aumentan la dimensión espacial para recuperar la resolución original. La combinación de estas operaciones permite que el modelo capture y reconstruya características a múltiples escalas, mejorando la calidad de las predicciones y la capacidad de generalización. Finalmente, se ha desarrollado una capa de convolución final para ajustar el número de canales de salida.

Además, el modelo está diseñado para incorporar covariables adicionales, tanto en las características de entrada como en las etiquetas de salida. Para lograr esto, se realizan una serie de concatenaciones que permiten procesar y concatenar de manera eficiente los tensores correspondientes, integrando estas covariables adicionales en el flujo de datos del modelo.

Asimismo, el modelo está equipado con funcionalidades para la normalización por instancias, lo que permite ajustar la distribución de las características de entrada de manera que cada instancia sea tratada de manera homogénea. Esto mejora la estabilidad y el rendimiento del modelo durante el entrenamiento. Además, se han implementado técnicas de interpolación y escalado para ajustar la resolución de los datos, asegurando que tanto las características como las etiquetas estén alineadas correctamente y sean compatibles con la resolución requerida por el modelo.

Estas características adicionales no solo aumentan la flexibilidad del modelo, sino que también permiten una adaptación más precisa a diferentes tipos de datos y tareas, mejorando así la capacidad del modelo para generalizar y realizar predicciones más precisas.

En la Figura 3.2 se presenta el fichero de configuración empleado para la herramienta de *Modeling Toolkit*.

```
model_configuration:
  class_name: Unet3D
  kwargs:
    filters: [64, 128]
    pooling_size: [2, 2]
    instance_normalization: true
    embed_instance_stats_as_channel: true
  features:
  - pr_None
  labels:
  - pr_None
optimizer:
  class_name: Adam
  kwargs:
    lr: 0.0001
metrics:
- CustomMetrics
loss:
  class_name: ConservativeLoss
  kwargs:
    eps: 0.001
    alpha: 1.0
    beta: 1.0
training_parameters:
  num_epochs: 100
  batch_size: 4
  gradient_accumulation_steps: 1
  lr_warmup_steps: 750
  early_stopping_patience: 700
  early_stopping_min_delta: 0.01
  val_monitor: "val_conservation"
  mixed_precision: "fp16"
  num_workers_dataloader: 8
  sampler: weighted_by_value
  device: cuda
  output_dir: "/predictia-nas2/Data/temporal-downscaling/results/santander"
  save_image_epochs: 1
  save_model_epochs: 5
  limit_train_batches: 800
  limit_valid_batches: 100
  limit_test_batches: 100
```

Figura 3.2: Ejemplo de fichero de configuración empleado para la herramienta de *Modeling Toolkit*.

En el fichero de configuración presentado en la Figura 3.2 se puede observar como en primer lugar se define la configuración del modelo, especificando tanto el tipo de modelo (Unet3D), como sus principales argumentos (filtros, tipo de pooling y normalización) y la variable de estudio (precipitación).

Por otro lado, se define el optimizador (*Adam*), junto con su argumento *learning_rate*. A su vez, se seleccionan las métricas y la función de pérdida. Se ha implementado una función de pérdida denominada *Conservative Loss* que cuenta con dos términos, combina la *Charbonnier Loss* con un término adicional destinado a la conservación de precipitación. El término de conservación de precipitación busca asegurar que la suma total de las predicciones y los valores objetivos coincidan, lo que es útil en contextos donde la conservación de ciertas propiedades (como la cantidad total de precipitación) es importante. El peso de los términos de *Charbonnier Loss* y el término de conservación de la precipitación se controla mediante los parámetros *alpha* y *beta* en el fichero de configuración. Se puede observar como en este caso se ha establecido el mismo peso para ambos términos.

A continuación se describen los parámetros utilizados para el entrenamiento del modelo. El proceso se lleva a cabo durante 100 épocas (*num_epochs*), con un tamaño de lote de 4 (*batch_size*). Esto implica que el modelo realiza 100 ciclos completos sobre los datos y que en cada lote se procesan 4 muestras. El modelo acumula gradientes durante un paso (*gradient_accumulation_steps*) antes de aplicar una actualización a los pesos.

Para un ajuste más eficiente de la tasa de aprendizaje, se emplea un mecanismo de calentamiento (*lr_warmup_steps*) durante los primeros 750 pasos, lo que permite que el modelo incremente gradualmente la tasa de aprendizaje.

Con el fin de optimizar el rendimiento y evitar el sobreajuste, se implementa un esquema de detención temprana (*early_stopping*), el cual interrumpirá el entrenamiento si no se observan mejoras en la métrica de validación tras 700 pasos consecutivos, siempre que el cambio mínimo en dicha métrica sea inferior a 0.01.

Los parámetros *limit_train_batches*, *limit_valid_batches* y *limit_test_batches* definen el número máximo de lotes (*batches*) que se procesarán en las etapas de entrenamiento, validación y *test*, respectivamente.

Se ha implementado un parámetro llamado *sampler*, el cual toma el valor de *weighted_by_value*, que tiene como objetivo crear un muestrador ponderado en *PyTorch* basado en los valores de precipitación presentes en los datos. Esta implementación es especialmente útil para conjuntos de datos desequilibrados, como en problemas meteorológicos, donde la precipitación no ocurre con la misma frecuencia (frecuentemente es cero). El objetivo es que el modelo preste mayor atención a los ejemplos con precipitación significativa. El código que se ha desarrollado genera un muestrador que otorga mayor importancia a las muestras con una mayor precipitación promedio. De este modo, si una muestra presenta un valor elevado de precipitación, será seleccionada con mayor probabilidad durante el proceso de muestreo, como en la fase de entrenamiento del modelo. Esto ayuda a equilibrar el impacto de las diferentes muestras en el entrenamiento, reduciendo la influencia de aquellas con baja o nula precipitación (las cuales podrían recibir un peso menor o incluso ser ignoradas con un peso de 0).

El parámetro *save_model_epochs* define la frecuencia con la que se guarda una copia del modelo

entrenado. Por ejemplo, si se establece *save_model_epochs: 5*, el modelo se guardará automáticamente cada 5 épocas. Esto es útil para prevenir la pérdida de progreso en caso de interrupciones y para reanudar el entrenamiento desde un punto intermedio si es necesario.

El parámetro *save_image_epochs* especifica la frecuencia con la que se guardan imágenes o resultados intermedios del entrenamiento. Con *save_image_epochs: 1*, se guardarán imágenes del progreso del entrenamiento cada 1 época, lo que permite monitorear visualmente el rendimiento del modelo.

A su vez, *output_dir* define el directorio donde se almacenarán todos los resultados del entrenamiento, incluidos los modelos guardados y las imágenes.

El parámetro *device*, como en este caso con el valor *cuda*, especifica el hardware empleado para la ejecución del modelo, indicando que el entrenamiento y la evaluación se realizan en una GPU.

Para poder realizar este proceso de forma eficiente y escalable, se ha utilizado Kubernetes como plataforma para gestionar contenedores. Kubernetes ha permitido gestionar y automatizar el despliegue del entorno de entrenamiento, asegurando así la disponibilidad de los recursos y facilitando la integración de GPU para poder realizar el entrenamiento del modelo.

Además, se ha empleado una imagen de Docker personalizada que contiene todas las dependencias necesarias para el entrenamiento del modelo, como los repositorios de *Deep Meteo* (*Data Toolkit*, *Modeling Toolkit*), controladores de GPU y cualquier otro software requerido.

3.3. *Deployment Toolkit*

El repositorio denominado *Deployment Toolkit* es una herramienta creada para optimizar el proceso de implementación de modelos de aprendizaje automático. Su propósito principal es facilitar la descarga de modelos previamente entrenados, lo que permite a los usuarios integrar fácilmente modelos de inteligencia artificial en sus aplicaciones sin necesidad de reentrenar desde cero. Además, el *Deployment Toolkit* genera datos de entrada que imitan con precisión las características y el formato del problema objetivo, asegurando que las pruebas y la inferencia se realicen en condiciones que reflejen fielmente el entorno real.

Una vez que los datos de entrada han sido preparados, la herramienta lleva a cabo la generación de predicciones utilizando el modelo descargado, produciendo predicciones basadas en los datos proporcionados. Los resultados de estas predicciones son luego almacenados de manera ordenada y accesible, permitiendo una posterior revisión y análisis. Esta solución integral no solo agiliza el proceso de despliegue, sino que también garantiza que los resultados sean coherentes y útiles para la toma de decisiones.

El primer paso consiste en la descarga del modelo de aprendizaje automático registrado en MLflow junto con sus artefactos asociados (ficheros de configuración y datos estáticos) desde un servidor de MLflow, usando criterios de búsqueda como el nombre del modelo, la versión y posibles etiquetas adicionales.

A continuación, se ha implementado un método *get_input_data* específicamente para preparar los datos de entrada necesarios para obtener predicciones con el modelo desarrollado. Un aspecto crucial de este método es la consideración de los días previos al estudio, parámetro que se define como *number_steps_backward* en el fichero de configuración de *Data Toolkit* (ver Figura 3.1). Este aspecto asegura que se generen muestras que incluyan datos históricos adicionales, lo que permite al modelo aprovechar información temporal previa para realizar predicciones más precisas y confiables. De este modo, se obtienen muestras diarias de precipitación para los días incluidos en el período de test, abarcando los puntos de la malla seleccionada con una resolución de 0.25^{o}.

Una vez que se han preparado los datos de entrada, se ha desarrollado un método denominado *generate_predictions* que se encarga de ejecutar el modelo de aprendizaje automático para generar predicciones a partir de dichos datos. El método sigue una lógica específica basada en el nivel de precipitación diaria registrado. Por un lado, si la precipitación diaria es inferior a 0.1 mm, el método establece una precipitación nula para las 24 horas de cada día, evitando así problemas con días de precipitación mínima. Por otro lado, si la precipitación diaria supera 0.1 mm, los valores de precipitación se obtienen a partir de las predicciones del modelo desarrollado.

Finalmente, los resultados obtenidos se almacenan en un archivo NetCDF, un formato de datos ampliamente utilizado en ciencias climáticas debido a su capacidad para manejar grandes volúmenes de datos multidimensionales de manera eficiente. Este archivo NetCDF actúa como la estructura de datos principal para los resultados generados por el modelo, manteniendo la coherencia con el formato de los datos en bruto originales.

El uso de NetCDF no solo asegura una gestión adecuada y accesible de los datos, sino que también proporciona un marco estructurado para facilitar la comparación entre las predicciones

realizadas por el modelo y los valores reales observados durante el período de tiempo correspondiente. Esta capacidad de comparación es fundamental para evaluar el rendimiento del modelo y realizar ajustes necesarios, garantizando así la precisión y utilidad de las predicciones en estudios y aplicaciones futuras.

Las predicciones del modelo se almacenan en un único archivo NetCDF, que contiene datos de precipitación con resolución horaria para los puntos de la malla definidos por latitud y longitud, cubriendo el periodo de test especificado en el archivo de configuración del *Data Toolkit* (ver Figura 3.1). En este caso, la región geográfica seleccionada corresponde a Santander, y el periodo de test abarca desde enero de 2018 hasta diciembre de 2020.

3.4. *Test*

En esta sección se presenta un análisis detallado de las predicciones generadas para el modelo desarrollado. El objetivo de este análisis es evaluar el rendimiento del modelo en la predicción de datos de precipitación, comparando los resultados obtenidos con observaciones reales extraídas del conjunto de datos ERA5, un referente ampliamente utilizado en estudios climáticos. Para ello, se ha desarrollado un programa en *Python* que automatiza este análisis.

En primer lugar, se ha tenido en cuenta la distinta naturaleza de los datos de predicción, recogidos en único dataset con resolución horaria para todo el periodo de *test* comprendido entre enero de 2018 hasta diciembre de 2020, con los datos de ERA5, que se presentan con resolución horaria pero se agrupan por meses. De forma que, a la hora de realizar la carga de observaciones de ERA5, se han concatenado en un único dataset, facilitando de esta forma su análisis.

Con el objetivo de contar con un marco de referencia para comparar el rendimiento del modelo, se ha definido un *benchmark* basado en los datos de precipitación diaria. Este *benchmark* se construye dividiendo el total de precipitación diaria entre las 24 horas del día, asignando ese valor constante a cada hora. De esta manera, se obtiene una estimación horaria de la precipitación que sirve como referencia básica para evaluar la capacidad predictiva del modelo.

3.4.1. Santander

Los puntos seleccionados en la malla, en el fichero de configuración de *Data Toolkit*, abarcan una región geográfica definida por una longitud que varía entre [-4.0, -3.0] y una latitud entre [43.0, 44.0] grados decimales, con una resolución de 0.25°. Dicha área está aproximadamente centrada en un región cercana a la ciudad de Santander, cuyas coordenadas geográficas aproximadas son -3.8º de longitud y 43.5º de latitud. Por lo tanto, los puntos seleccionados reprensentan las posiciones geográficas más próximas a Santander.

Para evaluar de manera efectiva el desempeño del modelo en comparación con este marco de referencia, a continuación se presentan las métricas estadísticas asociadas a ambos, el modelo y el *benchmark*.

Métrica	**Predicciones**	***Benchmark***
RMSE	0.322	0.282
MAE	0.137	0.115

Tabla 3.1: Tabla comparativa de métricas de error (RMSE y MAE) correspondientes a las predicciones del modelo y al marco de referencia (*benchmark*).

En cuanto a las métricas de error de la Tabla 3.1, se puede observar como el *benchmark* presenta valores inferiores tanto de la raíz del error cuadrático medio (RMSE) como del error medio absoluto (MAE). Esto sugiere que el *benchmark* tiene un mejor desempeño en términos de precisión en comparación con las predicciones del modelo. Los valores más bajos de RMSE y MAE del *benchmark* indican que este método tiene menores errores en promedio y es más preciso en la estimación de las precipitaciones en comparación con el modelo evaluado.

Los resultados presentados en la Tabla 3.1 reflejan la superioridad del *benchmark* en términos de precisión respecto al modelo desarrollado.

A la vista de estos resultados, se han seleccionado determinadas muestras aleatorias del periodo de *test* para observar visualmente la capacidad del modelo de desagregar precipitación diaria a horaria frente a las observaciones de ERA5.

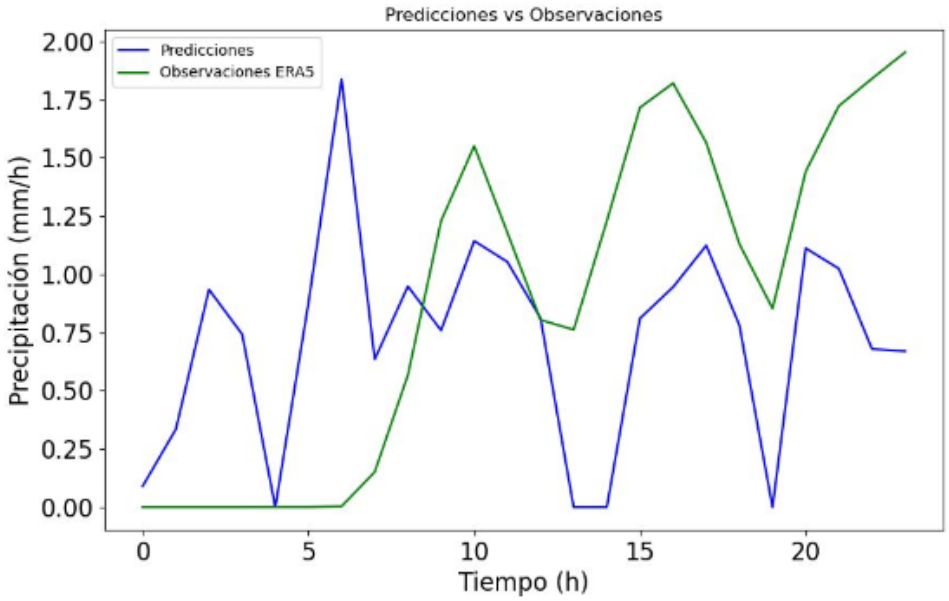

Figura 3.3: Predicciones (azul) y observaciones de ERA5 (verde) de la precipitación horaria en función de las horas de un día en Santander.

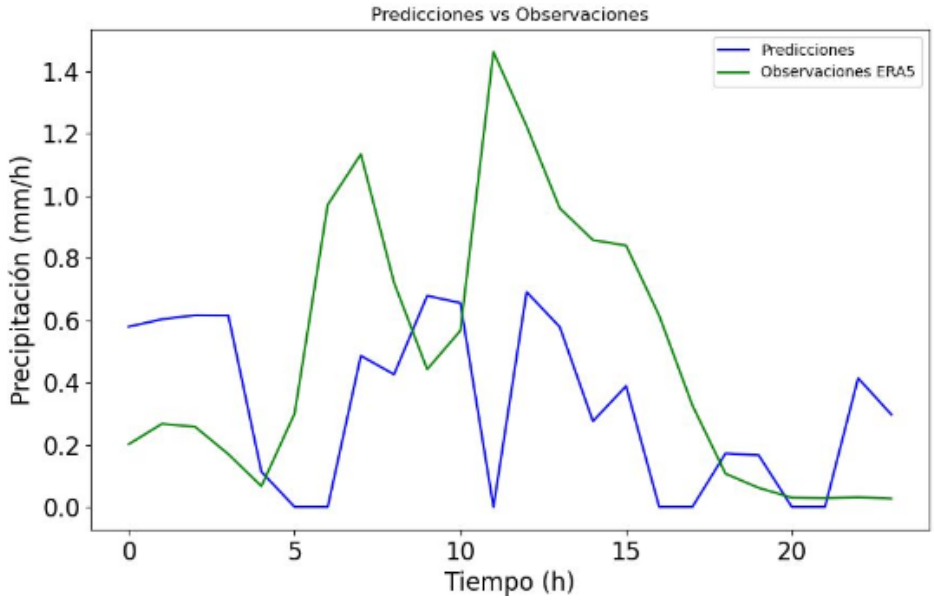

Figura 3.4: Predicciones (azul) y observaciones de ERA5 (verde) de la precipitación horaria en función de las horas de un día en Santander.

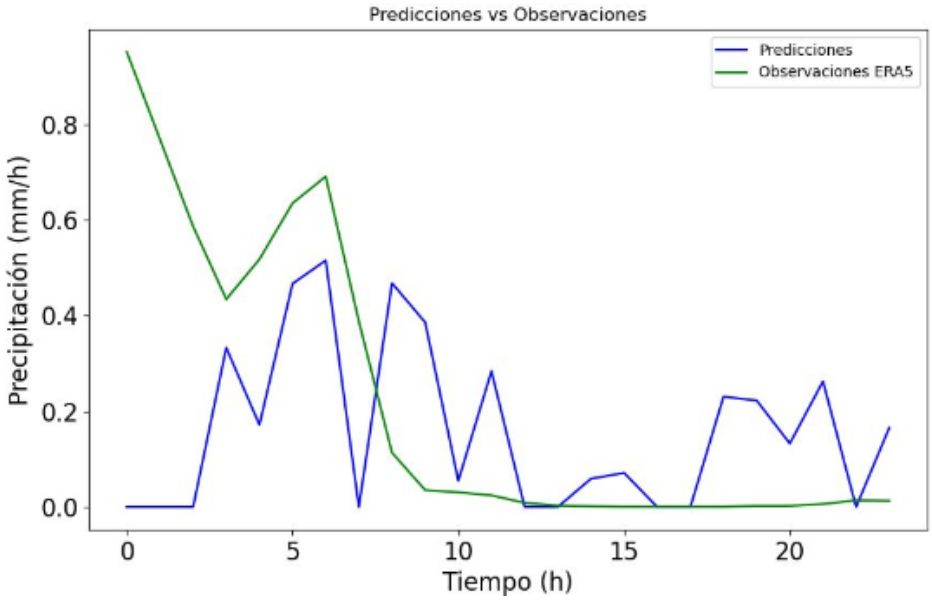

Figura 3.5: Predicciones (azul) y observaciones de ERA5 (verde) de la precipitación horaria en función de las horas de un día en Santander.

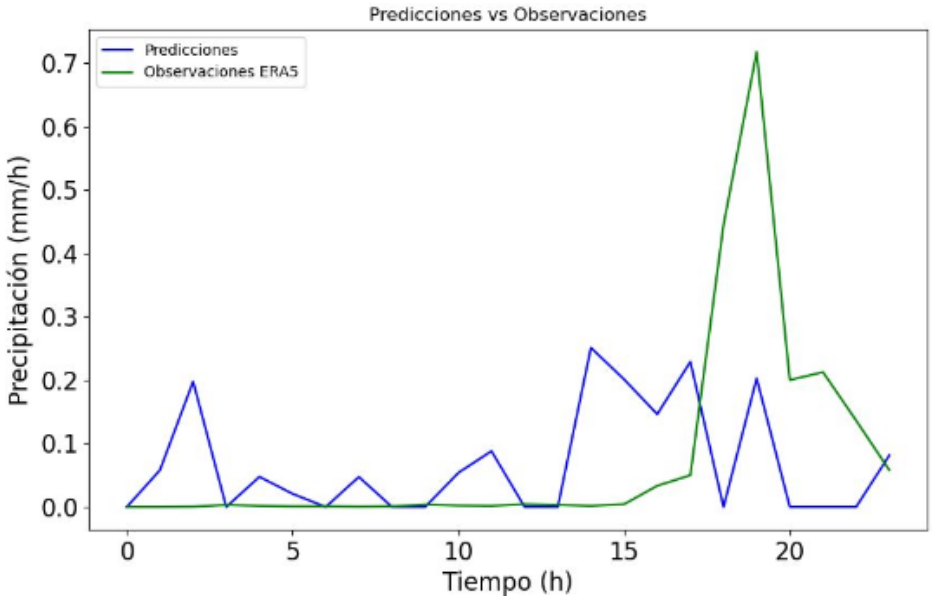

Figura 3.6: Predicciones (azul) y observaciones de ERA5 (verde) de la precipitación horaria en función de las horas de un día en Santander.

En las Figuras 3.3 y 3.4, se aprecia una ligera similitud en la dinámica de las funciones que describen tanto las predicciones como las observaciones. No obstante, hay periodos en los que las observaciones no registran precipitaciones, mientras que las predicciones sí indican la ocurrencia de lluvia en esos mismos intervalos. Además, se evidencian diferencias significativas en las cantidades de precipitación para la mayoría de los rangos horarios. Esta discrepancia indica que, aunque las predicciones capturan ciertas tendencias generales, persiste una falta considerable

de concordancia en cuanto a la ocurrencia e intensidad de la precipitación en comparación con los datos observados

Por otro lado, las Figuras 3.5 y 3.6 resaltan un problema recurrente: la incapacidad del modelo para representar con precisión los periodos sin precipitación. En múltiples ocasiones, cuando las observaciones no muestran presencia de lluvia, el modelo predice incluso máximos relativos y absolutos de precipitación.

Dada esta situación, se ha optado por ajustar los hiperparámetros establecidos en el fichero de configuración de *Modeling Toolkit* (ver Figura 3.2). A pesar de estos ajustes, las métricas de validación y los gráficos de desagregación no mostraron mejoras significativas. En consecuencia, se ha decidido explorar una solución alternativa: la modificación del modelo. Esta decisión busca abordar de manera más efectiva las deficiencias observadas y mejorar el rendimiento general del sistema.

El modelo UNet ha sido modificado para aprender los residuos de la interpolación uniforme, que actúa como el *benchmark*. En este nuevo enfoque, para cada muestra se selecciona el día de estudio, se divide la precipitación total diaria entre las 24 horas del día, y se asigna este valor a cada hora. Posteriormente, este valor se suma a la salida de la última capa de convolución de la red. Este ajuste tiene como objetivo guiar al modelo utilizando los valores del *benchmark*, mejorando así la precisión de las predicciones al capturar las diferencias entre la interpolación uniforme y los datos reales.

A continuación, se presentan las métricas estadísticas correspondientes al marco de referencia, las predicciones originales del modelo y las predicciones del modelo guiado por el *benchmark*.

Métrica	**Predicciones**	**Predicciones guiadas**	***Benchmark***
RMSE	0.322	0.281	0.282
MAE	0.137	0.114	0.115

Tabla 3.2: Tabla comparativa de métricas de error (RMSE y MAE) correspondientes al marco de referencia, las predicciones originales del modelo y las predicciones del modelo guiado por el *benchmark* para Santander.

En la Tabla 3.2 se puede observar como las predicciones del modelo guiado por el marco de referencia presentan prácticamente valores idénticos al *benchmark*. Ante esta situación, se presenta gráficamente los resultados de la desagregación de precipitación de escala diaria a escala horaria mediante el modelo guiado.

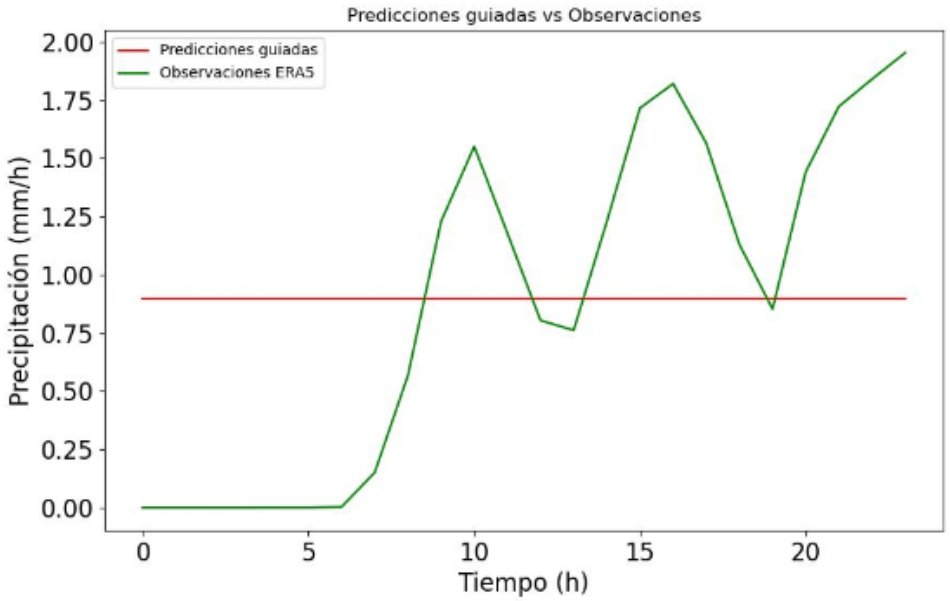

Figura 3.7: Predicciones del modelo guiado (rojo) y observaciones de ERA5 (verde) de la precipitación horaria en función de las horas de un día en Santander.

En la Figura 3.7 se observa que las predicciones del modelo guiado por el *benchmark* replican simplemente los resultados del mismo. Esto se debe a que asigna a cada hora del día un valor de precipitación correspondiente a la división de la precipitación acumulada diaria entre 24. Aunque este enfoque muestra métricas de error más bajas en comparación con el modelo original, dichas mejoras no reflejan una correcta desagregación de la precipitación. En consecuencia, se ha comprobado que guiar al modelo con el *benchmark* no ha proporcionado un resultado satisfactorio, ya que no capta adecuadamente las variaciones horarias de la precipitación.

A modo de comparación, se presentan a continuación los resultados correspondientes para otra región espacial.

3.4.2. Alicante

Los puntos seleccionados en la malla, en el fichero de configuración de *Data Toolkit*, abarcan una región geográfica definida por una longitud que varía entre [-1.0, 0.0] y una latitud entre [38.0, 39.0] grados decimales, con una resolución de 0.25°. Dicha área está aproximadamente centrada en un región cercana a la ciudad de Alicante, cuyas coordenadas geográficas aproximadas son -0.5º de longitud y 38.3º de latitud. Por lo tanto, los puntos seleccionados reprensentan las posiciones geográficas más próximas a Alicante.

Métrica	**Predicciones**	***Benchmark***
RMSE	0.342	0.290
MAE	0.084	0.061

Tabla 3.3: Tabla comparativa de métricas de error (RMSE y MAE) correspondientes a las predicciones del modelo y al marco de referencia (*benchmark*) en Alicante.

De igual forma que sucedía en la región de Santander, para Alicante el *bechmark* ofrece valores inferiores tanto de la raíz del error cuadrático medio (RMSE) como del error medio absoluto (MAE). Nuevamente el marco de referencia ofrece un mejor desempeño en términos de precisión en comparación con las predicciones del modelo.

A la vista de estos resultados, se ha seleccionado una muestra aleatoria del periodo de *test* para observar visualmente la capacidad del modelo de desagregar precipitación diaria a horaria frente a las observaciones de ERA5.

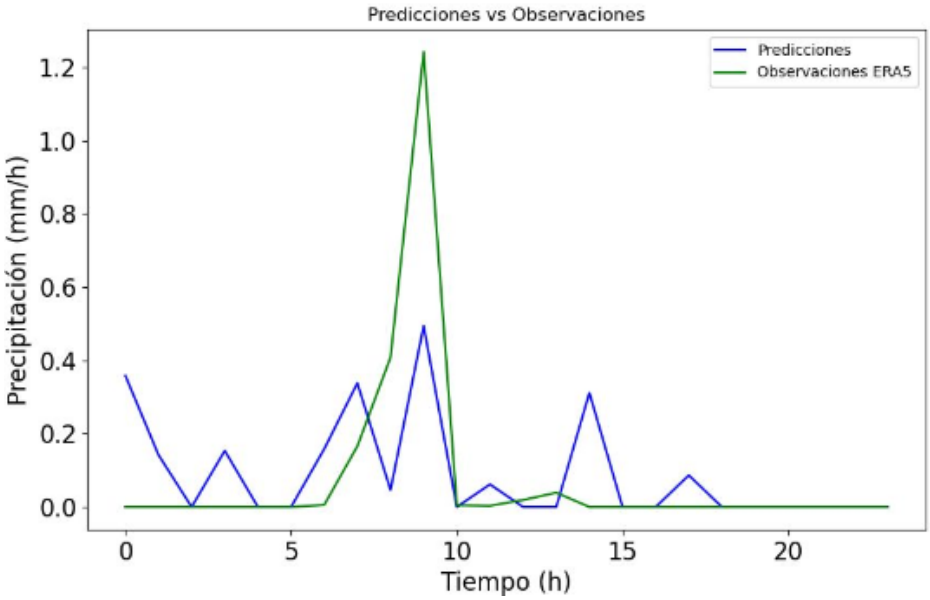

Figura 3.8: Predicciones (azul) y observaciones de ERA5 (verde) de la precipitación horaria en función de las horas de un día en Alicante.

En la Figura 3.8 se puede observar, al igual que sucedía en Santander, diferencias significativas entre las observaciones y las predicciones del modelo. No solo se reflejan discrepancias en cuanto a la cantidad de precipitación para la totalidad de los rangos horarios, sino que también se manifiestan periodos en los que las observaciones no registran precipitaciones, mientras que las predicciones sí indican la ocurrencia de lluvia en esos mismos intervalos. De esta forma, se resalta un problema recurrente: la incapacidad del modelo para representar con precisión los periodos sin precipitación.

Al comparar los resultados obtenidos para las dos regiones geográficas, Santander y Alicante, no se han observado diferencias significativas en las métricas de error. En ambas localidades, se presentan los mismos problemas en la desagregación temporal de la precipitación, específicamente en cuanto a la cantidad y la distribución de la precipitación a lo largo de las horas del día.

Capítulo 4

Conclusiones y futuros trabajos

En este capítulo, se presentan las principales líneas de mejora identificadas a lo largo del desarrollo del trabajo, así como las conclusiones finales. A lo largo del documento se han abordado distintas problemáticas y desafíos relacionados con el tema de estudio, y en esta sección, se presentan las opciones que podrían ayudar a optimizar los resultados y ampliar el rendimiento del trabajo para futuras investigaciones.

4.1. Conclusiones finales

Se ha desarrollado un modelo de aprendizaje profundo con el objetivo de incrementar la resolución temporal de datos de precipitación, transformándolos de una escala diaria a horaria. Para ello, se ha llevado a cabo la carga y procesamiento de los datos, el desarrollo del modelo UNet, su integración en aplicaciones para generar predicciones y el análisis de los resultados obtenidos.

Las discrepancias entre los patrones de precipitación de observaciones reales y las predicciones del modelo evidencian su incapacidad tanto para ajustar la cantidad de precipitación como su distribución a lo largo de las horas del día.

Se ha establecido un marco de referencia para comparar el modelo, el cual se construye dividiendo el total de precipitación diaria entre las 24 horas del día y asignando ese valor a cada hora. Este *benchmark* ofrece mejores resultados en las métricas estadísticas RMSE y MAE que el modelo.

Además, se pone de manifiesto un error recurrente: la incapacidad del modelo para representar con precisión los periodos sin precipitación. En la gran mayoría de muestras, cuando las observaciones no muestran presencia de lluvia, el modelo predice incluso máximos de precipitación.

Asimismo, se han comparado los resultados del modelo en dos regiones geográficas, Santander y Alicante, presentando métricas de error sin diferencias significativas entre sí y observándose los mismos problemas en la desagregación temporal de precipitación.

A su vez, el modelo UNet ha sido modificado para ver si se observaba una mejora en la precisión. Para ello, se ha sumado el marco de referencia a la última capa de convolución de la red,

con el objetivo de guiar al modelo utilizando los valores del *benchmark*. Las predicciones de este modelo guiado son idénticas al marco de referencia. De forma que, guiar al modelo no ha proporcionado un resultado satisfactorio, ya que nuevamente no se captan las variaciones horarias de la precipitación.

De cara al futuro del proyecto, se propone llevar a cabo nuevas simulaciones del modelo integrando información adicional. Estas simulaciones podrían incorporar variables predictoras adicionales, como la presión y el geopotencial, así como un contexto espacial más amplio y covariables relevantes, como la máscara de suelo. Además, se sugiere incrementar la complejidad del modelo y explorar diferentes valores de hiperparámetros para afinar su rendimiento.

Otra dirección prometedora para la investigación es la evaluación de arquitecturas alternativas a la UNet. Investigar estas nuevas arquitecturas podría potencialmente mejorar el rendimiento del modelo en el contexto específico del proyecto, ofreciendo así una oportunidad significativa para optimizar los resultados obtenidos.

4.2. Líneas de mejora

A continuación, se presentan una serie de líneas de mejora que podrían optimizar los resultados presentados en este trabajo.

En primer lugar, podría ser recomendable incrementar la complejidad del modelo, el actual cuenta con en torno a 1 millón de parámetros. Desarrollar un modelo más complejo podría ayudar a una mayor captura de relaciones complejas en los datos, como patrones no lineales y correlaciones sutiles.

Una posibilidad interesante para mejorar el rendimiento del modelo sería proporcionar predictores adicionales, que proporcionen al sistema una visión más completa sobre la situación a modelar. Por ejemplo, disponer de información espacial podría ayudar a diferenciar distintas zonas geográficas, ajustando así los patrones temporales en función de la localización.

Además, se podrían incluir covariables como la máscara de suelo, lo cual ayudaría a diferencia entre distintas características geográficas que pueden influir en los resultados del modelo. Este aspecto podría ser interesante principalmente en aquellas zonas donde las características del terreno juegan un papel fundamental.

A su vez, incluir predictores adicionales como la presión o el geopotencial podrían aportar una perspectiva más amplia sobre las condiciones globales del sistema. Estas variables pueden influir en la presencia de tormentas, aportando así un contexto más amplio para la predicción.

Otra mejora posible es ajustar la función de coste, añadiendo algún término que penalice fuertemente los errores en los patrones horarios o la falta de correlación en la serie temporal.

Asimismo, se propone realizar una búsqueda más exhaustiva de los valores óptimos de los diferentes hiperparámetros utilizados en el entrenamiento del modelo. Un caso relevante, sería el *learning rate*, ya que un valor inapropiado, bien demasiado bajo o demasiado alto, puede afectar negativamente a la convergencia del modelo. Mientras que un valor demasiado bajo puede ralentizar el proceso de aprendizaje, uno demasiado alto podría impedir que el modelo

alcance un mínimo adecuado. Por lo tanto, la optimización de este y otros hiperparámetros podría mejorar el rendimiento final del modelo.

Finalmente, una opción más radical sería explorar el uso de arquitecturas alternativas a la UNet. Modelos como LSTM (*Long short-term memory*), Conv-LSTM o *Temporal Fusion Transformer* podrían proporcionar una mejora significativa en la gestión de las relaciones temporales en los datos, lo que podría resultar en un rendimiento superior para este contexto.

Estas propuestas ofrecen una base sólido para futuras investigaciones, abriendo nuevas oportunidades para mejorar y optimizar la precisión de los resultados obtenidos. Implementar estas sugerencias podría potenciar el rendimiento del modelo y proporcionar mejores resultados.

Bibliografía

[1] C. Aydinalp y M.S. Cresser. "The effects of global climate change on agriculture". En: *American-Eurasian Journal of Agricultural & Environmental Sciences* 3.5 (2008), págs. 672-676.

[2] A. Haines, R.S. Kovats, D. Campbell-Lendrum y C. Corvalán. "Climate change and human health: impacts, vulnerability and public health". En: *Public health* 120.7 (2006), págs. 585-596.

[3] K.L. Ebi, J. Vanos, J.W. Baldwin, J.E. Bell y D.M. Hondula. "Extreme weather and climate change: population health and health system implications". En: *Annual review of public health* 42.1 (2021), págs. 293-315.

[4] A. Cogato, F. Meggio, M. De Antoni Migliorati y F. Marinello. "Extreme weather events in agriculture: A systematic review". En: *Sustainability* 11.9 (2019), pág. 2547.

[5] Z. Xu, P.E. Sheffield, H. Su, X. Wang, Y. Bi y S. Tong. "The impact of heat waves on children's health: a systematic review". En: *International journal of biometeorology* 58 (2014), págs. 239-247.

[6] G. Greenough, M. McGeehin, S.M. Bernard, J. Trtanj, J. Riad y D. Engelberg. "The potential impacts of climate variability and change on health impacts of extreme weather events in the United States." En: *Environmental health perspectives* 109.suppl 2 (2001), págs. 191-198.

[7] S.A. Changnon, D. Changnon, E.R. Fosse, D.C. Hoganson, R.J. Roth Sr y J.M. Totsch. "Effects of recent weather extremes on the insurance industry: major implications for the atmospheric sciences". En: *Bulletin of the American Meteorological Society* 78.3 (1997), págs. 425-436.

[8] Oscar Brown Manrique, Yurisbel Gallardo Ballat, Amaury Correa Santana y Sergio Barrios García. "El cambio climático y sus evidenciasen las precipitaciones". En: *Ingeniería hidráulica y Ambiental* 36.1 (2015), págs. 88-101.

[9] G.D. Robinson. "Weather and climate forecasting as problems in hydrodynamics". En: *Monthly Weather Review* 106.4 (1978), págs. 448-457.

[10] V. Lucarini, R. Blender, C. Herbert y F. Ragone. "Mathematical and physical ideas for climate science". En: *Reviews of Geophysics* 52.4 (2014), págs. 809-859.

[11] J.D. Sierra. "Predicción, desagregación y cambio climático en la precipitación para aplicaciones hidrológicas". Tesis doct. Universidad de Cantabria, 2019.

[12] H. Tabari, S.M. Paz, D. Buekenhout y P. Willems. "Comparison of statistical downscaling methods for climate change impact analysis on precipitation-driven drought". En: *Hydrology and Earth System Sciences* 25.6 (2021), págs. 3493-3517.

[13] D. Maraun, F. Wetterhall, A.M. Ireson, R.E. Chandler y E.J. Kendon. "Precipitation downscaling under climate change: Recent developments to bridge the gap between dynamical models and the end user". En: *Reviews of geophysics* 48.3 (2010).

[14] Y.B. Dibike y P. Coulibaly. "Hydrologic impact of climate change in the Saguenay watershed: comparison of downscaling methods and hydrologic models". En: *Journal of hydrology* 307.1-4 (2005), págs. 145-163.

[15] P. Willems y M. Vrac. "Statistical precipitation downscaling for small-scale hydrological impact investigations of climate change". En: *Journal of hydrology* 402.3-4 (2011), págs. 193-205.

[16] Z. Zahmatkesh, M. Karamouz, E. Goharian y S.J. Burian. "Analysis of the effects of climate change on urban storm water runoff using statistically downscaled precipitation data and a change factor approach". En: *Journal of Hydrologic Engineering* 20.7 (2015).

[17] A.W. Robertson, A.V.M. Ines y J.W. Hansen. "Downscaling of seasonal precipitation for crop simulation". En: *Journal of Applied Meteorology and Climatology* 46.6 (2007), págs. 677-693.

[18] K. Forster, F. Hanzer, B. Winter, T. Marke y U. Strasser. "An open-source MEteoroLOgical observation time series DISaggregation Tool (MELODIST v0. 1.1)". En: *Geoscientific Model Development* 9.7 (2016), págs. 2315-2333.

[19] S. Scher y S. Pessenteiner. "Temporal disaggregation of spatial rainfall fields with generative adversarial networks". En: *Hydrology and Earth System Sciences* 25.6 (2021), págs. 3207-3225.

[20] D. Randall, R. Wood y Bony S. "Climate models and their evaluation". En: *Climate change 2007: The physical science basis. Contribution of Working Group I to the Fourth Assessment Report of the IPCC (FAR)*. Cambridge University Press, 2007, págs. 589-662.

[21] F. Giorgi y L.O. Mearns. "Approaches to the simulation of regional climate change: a review". En: *Reviews of geophysics* 29.2 (1991), págs. 191-216.

[22] T. Lee y C. Jeong. "Nonparametric statistical temporal downscaling of daily precipitation to hourly precipitation and implications for climate change scenarios". En: *Journal of Hydrology* 510 (2014), págs. 182-196.

[23] J. Boé, L. Terray, F. Habets y E. Martin. "Statistical and dynamical downscaling of the Seine basin climate for hydro-meteorological studies". En: *International Journal of Climatology: A Journal of the Royal Meteorological Society* 27.12 (2007), págs. 1643-1655.

[24] Y. Sun, K. Deng, K. Ren, J. Liu, C. Deng e Y. Jin. "Deep learning in statistical downscaling for deriving high spatial resolution gridded meteorological data: A systematic review". En: *ISPRS Journal of Photogrammetry and Remote Sensing* 208 (2024), págs. 14-38.

[25] R. Tareghian y P.F. Rasmussen. "Statistical downscaling of precipitation using quantile regression". En: *Journal of hydrology* 487 (2013), págs. 122-135.

[26] G. Dayon, J. Boé y E. Martin. "Transferability in the future climate of a statistical downscaling method for precipitation in France". En: *Journal of Geophysical Research: Atmospheres* 120.3 (2015), págs. 1023-1043.

[27] J.M. Eden y M. Widmann. "Downscaling of GCM-simulated precipitation using model output statistics". En: *Journal of Climate* 27.1 (2014), págs. 312-324.

[28] P. Coulibaly, Y.B. Dibike y F. Anctil. "Downscaling precipitation and temperature with temporal neural networks". En: *Journal of Hydrometeorology* 6.4 (2005), págs. 483-496.

[29] R.L. Wilby, S. P Charles y E. Zorita. "Guidelines for use of climate scenarios developed from statistical downscaling methods". En: *Supporting material of the Intergovernmental Panel on Climate Change, available from the DDC of IPCC TGCIA* 27 (2004).

[30] D. Maraun y M. Widmann. *Statistical downscaling and bias correction for climate research*. Cambridge University Press, 2018.

[31] M. Turco, P. Quintana-Seguí, M Llasat, S. Herrera y J.M. Gutiérrez. "Testing MOS precipitation downscaling for ENSEMBLES regional climate models over Spain". En: *Journal of Geophysical Research: Atmospheres* 116.D18 (2011).

[32] Uwe Haberlandt, Yeshewatesfa Hundecha, Markus Pahlow y Andreas H Schumann. "Rainfall generators for application in flood studies". En: *Flood risk assessment and management: how to specify hydrological loads, their consequences and uncertainties* (2011), págs. 117-147.

[33] S. Albawi, T. A. Mohammed y S. Al-Zawi. "Understanding of a convolutional neural network". En: *2017 international conference on engineering and technology (ICET)*. Ieee. 2017, págs. 1-6.

[34] F. Alrasheedi, X. Zhong y P Huang. "Padding module: Learning the padding in deep neural networks". En: *IEEE Access* 11 (2023), págs. 7348-7357.

[35] D. Godoy. *Deep learning with pytorch step-by-step: A Beginner's Guide*. 2022.

[36] A. Paszke, A. Chaurasia, S. Kim y E. Culurciello. "Enet: A deep neural network architecture for real-time semantic segmentation". En: *arXiv preprint arXiv:1606.02147* (2016).

[37] O. Ronneberger, P. Fischer y T. Brox. "U-net: Convolutional networks for biomedical image segmentation". En: *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer. 2015, págs. 234-241.

[38] N. Siddique, S. Paheding, C.P. Elkin y V. Devabhaktuni. "U-net and its variants for medical image segmentation: A review of theory and applications". En: *Ieee Access* 9 (2021), págs. 82031-82057.