Caracterización de señales producidas por dosímetros electrónicos mediante IA

Universidad Internacional de Valencia

Titulación:

Máster en Big Data y Ciencia de Datos

Alumno/a:

Larman Eizmendi, Iñigo

Convocatoria:

Curso académico:

2022 – 2023

Director/a de TFM:

Sánchez Goez, Sebastián

Fecha:

mayo de 2023

De:
Planeta Formación y Universidades

Índice

Universidad Internacional de Valencia

Índice de figuras

Índice de tablas

Resumen

Los dosímetros de radiación basados en diodos de tipo PIN se encuentran actualmente en fase de investigación y son dispositivos que sirven para medir el nivel de radiación al que está expuesto una persona. Presentan la ventaja de que son muy económicos de producir, sin embargo, son susceptibles a dar falsos positivos debido a fuentes externas como golpes, fuentes de calor o ruido electromagnético. En este proyecto se ha desarrollado un clasificador binario que sea capaz de distinguir entre series temporales generadas por radiación y ruido basado en algoritmos de inteligencia artificial.

Para poder entrenar los algoritmos, se han tomado mediciones de un dosímetro en un laboratorio usando cesio 137 como fuente de radiación. Se han realizado dos modelos diferentes, uno basado en redes neuronales convolucionales y el otro en transformers y se ha elegido el modelo campeón evaluando las métricas de rendimiento. Ambos modelos han sido capaces de distinguir las series temporales de ruido y radiación con una precisión del 99 %.

Palabras clave: Radiación, dosímetro, inteligencia artificial, clasificador binario, series temporales, redes neuronales convolucionales, transformers.

1 Introducción

La dosimetría se trata de un campo esencial dentro de la protección radiológica para poder medir la dosis de radiación ionizante recibida en los humanos. La radiación emitida por elementos externos se mide mediante dosímetros y estos se tienen que utilizar como parte del equipo de protección radiológica para todos los trabajadores que puedan tener contacto con la radiación. Existen diferentes tecnologías de los dosímetros para poder medir el fenómeno físico de la radiación y el que se va a usar en este proyecto está basado en diodos PIN.

Este tipo de dosímetros se encuentran todavía en fase de investigación y tienen la gran ventaja de ser muy económicos de fabricar. Sin embargo, uno de los problemas que tienen es que pueden dar falsos positivos debido eventos como golpes, fuentes de calor o ruido electromagnético. Para poder evitar estos falsos positivos, se va a entrenar un algoritmo de inteligencia artificial supervisado que sea capaz de distinguir las señales procedentes de fuentes de radiación reales del ruido producido.

Para poder conseguir los datos de entrenamiento, se va a realizar un montaje en un laboratorio en el que se medirán las señales de series temporales generadas por el diodo del dosímetro mediante un osciloscopio. Se aplicará una fuente de radiación de cesio 137 para generar las series temporales reales y después se registrarán falsos positivos mediante ruido eléctrico para obtener series temporales falsas. Estos datos se usarán para entrenar un clasificador binario basado en redes neuronales convolucionales y otro de transformers. Finalmente, se evaluarán los resultados de ambos algoritmos mediante las métricas de rendimiento y se elegirá el modelo campeón.

Debido a que la electrónica que constituye al dosímetro es muy simple y no se incluye un microprocesador para incluir un algoritmo de IA en él, los datos se van a procesar en una computadora externa.

2 Objetivos

El objetivo principal de este proyecto es poder discriminar las señales de ruido de aquellas generadas por la presencia de radiación generadas en un dosímetro tipo PIN mediante un clasificador binario basado en un algoritmo de inteligencia artificial.

Los objetivos específicos, que dan sustento al objetivo general del proyecto son los siguientes:

- Obtener datos de series temporales de ruido y radiación generados por un dosímetro de radiación en un laboratorio el cual es tratado en el apartado 4.3.1 donde se muestra cómo se ha realizado el montaje experimental y cómo se han tomado los datos.
- Analizar los datos obtenidos para contrastar las diferencias entre los datos de radiación y ruido, el cual se ha tratado en el apartado 4.3.2.
- Limpiar y procesar los datos de las mediciones. Este se ha tratado en el apartado 4.3.3 donde se resumen las operaciones realizadas en los datos.
- Entrenar una arquitectura basada en redes neuronales convolucionales (CNN) para la clasificación binaria de las series temporales. Se ha tratado en el apartado 4.3.4.1 y en este se ha resumido la arquitectura e hiperparámetros utilizados para el entrenamiento.
- Entrenar una arquitectura basada en transformers para la clasificación binaria de las series temporales. Se ha tratado en el apartado 4.3.4.2 y en este se ha resumido la arquitectura e hiperparámetros utilizados para el entrenamiento.
- Analizar el coste computacional de ambos modelos, donde se han medido el tiempo de inferencia para los datos de test usando la CPU y la GPU. Este se ha tratado en el apartado 4.4.3.
- Analizar las métricas de rendimiento de los modelos para elegir el modelo campeón, el cual se ha tratado en el apartado 4.4.4.

3 Marco teórico

En este apartado se va a describir el marco teórico correspondiente al proyecto. Se ha dividido en dos apartados diferentes: En el primer apartado se describe el fenómeno de la radiación y cómo se detecta. En el segundo apartado se describen los algoritmos de inteligencia artificial utilizados para la discriminación del ruido y radiación.

3.1 La radiación y su detección

En este apartado se recogen las nociones teóricas sobre qué es la radiación y cómo se detecta para poder comprender cómo se ha realizado la fase de mediciones en el laboratorio.

3.1.1 La radiación y los diferentes tipos

La radiación es la propagación de energía en forma de partículas u ondas a través del vacío o de un medio material. Puede clasificarse en dos tipos: la radiación no ionizante y la radiación ionizante.

La radiación no ionizante no tiene energía suficiente para ionizar átomos o moléculas. Algunos ejemplos de radiaciones no ionizantes son las ondas de radio, las microondas, la radiación infrarroja, la luz visible y la radiación ultravioleta. Aunque la radiación no ionizante no suele ser tan perjudicial como la ionizante, la exposición a niveles elevados de algunos tipos de radiación no ionizante, como la radiación ultravioleta del sol, puede causar daños en la piel y aumentar el riesgo de cáncer de piel [1].

Por otro lado, la radiación ionizante es la que tiene energía suficiente para ionizar átomos o moléculas, quitándoles uno o varios electrones. Algunos ejemplos de radiación ionizante son las partículas alfa, las partículas beta, los rayos gamma, los rayos X y los neutrones. Este tipo de radiación puede ser perjudicial para los organismos vivos porque la ionización puede causar cambios químicos en las células y el ADN, lo que provoca daños celulares y efectos biológicos potencialmente dañinos [2]. Los dosímetros se usan para detectar la radiación ionizante, pero en función de su tecnología son capaces de determinar diferentes tipos.

Existen diferentes tipos de radiación ionizante, los cuales se clasifican en función de sus propiedades y cómo interactúan con la materia (Figura 1):

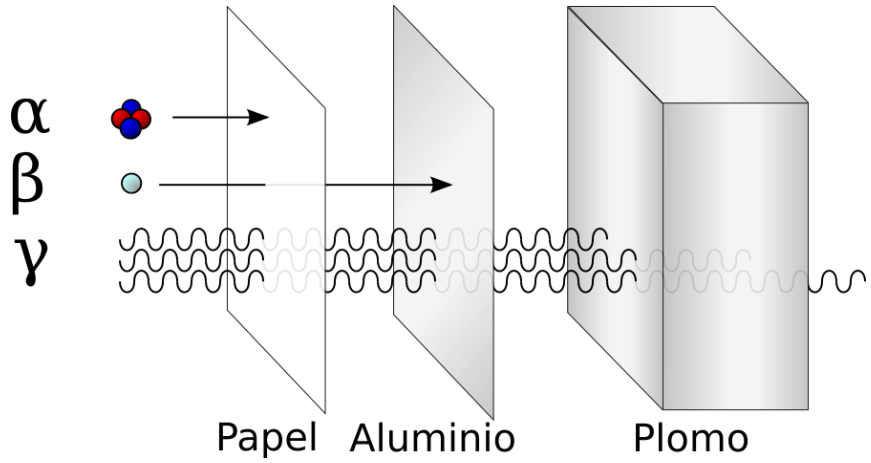

Figura 1. Capacidad de penetración de diferentes tipos de radiación ionizante en diferentes materiales. Fuente: [1]

- Partículas alfa: Son partículas cargadas positivamente que constan de dos protones y dos neutrones. Tienen un bajo poder de penetración y pueden ser detenidas por una hoja de papel o la capa externa de la piel, pero pueden ser muy dañinas si se inhalan o ingieren.
- Partículas beta: Son electrones de alta energía que se emiten desde el núcleo de un átomo radiactivo. Tienen mayor poder de penetración que las partículas alfa, pero pueden ser detenidas por una lámina de aluminio o varios milímetros de plástico.
- Rayos gamma: Son ondas electromagnéticas de alta energía emitidas por el núcleo de un átomo radiactivo. Tienen el mayor poder de penetración de todos los tipos de radiación ionizante y sólo pueden ser detenidos por varios centímetros de material denso, como el plomo o el hormigón.
- Neutrones: Son partículas neutras emitidas por el núcleo de un átomo radiactivo. Tienen un alto poder de penetración y pueden ser detenidas por varios centímetros de agua o unos centímetros de hormigón.

3.1.2 Interacción de la radiación con la materia

La radiación ionizante (fotones, neutrones, partículas cargadas etc.) se caracteriza por poder interaccionar y penetrar sobre la materia. Cuando se da esta interacción, la radiación pierde total o parcialmente su energía, transfiriéndosela al medio con el que está interactuando mediante diferentes mecanismos que dependen del tipo de radiación, su energía y las propiedades del medio material [3].

Los procesos de interacción de la radiación con la materia serán los que causarán principalmente los efectos biológicos en seres vivos y determinarán cómo se propaga la radiación en el medio material. El mecanismo de interacción de la radiación dependerá

principalmente de la carga eléctrica y de su masa, por lo que se detallarán los principales mecanismos que se dan en los fotones y los electrones:

3.1.2.1 Radiación de fotones

En la radiación de fotones (radiación gamma y rayos X) principalmente se dan el efecto fotoeléctrico y el efecto Compton.

Efecto fotoeléctrico:

El efecto fotoeléctrico es el fenómeno en el que se emiten electrones libres (fotoelectrones) cuando los átomos se exponen a una luz de cierta frecuencia o energía. Cuando los fotones interactúan con el átomo, los electrones absorben la energía de los fotones de la luz. Si la energía de los fotones es lo suficientemente alta, los electrones pueden superar las fuerzas de atracción que los retienen y estos son emitidos. (Figura 2).

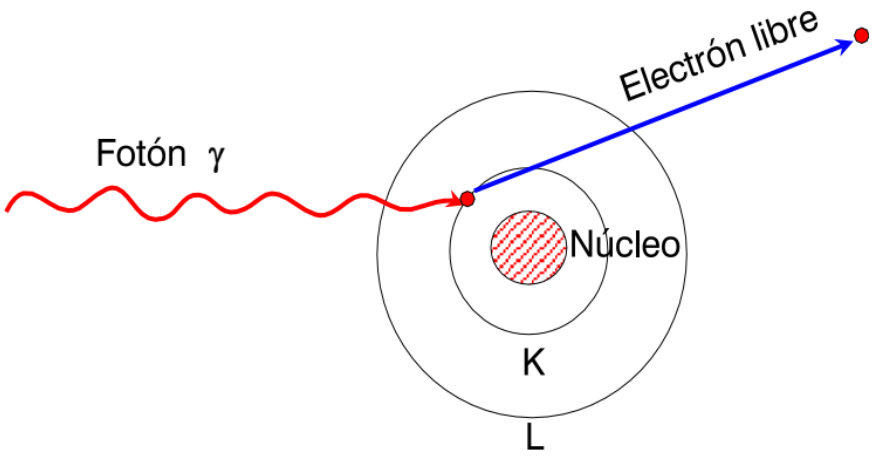

Figura 2. Representación gráfica del efecto fotoeléctrico. Fuente: [2]

El átomo resultante del efecto fotoeléctrico se trata de un ion positivo con hueco en una capa profunda (normalmente K) y a demás del fotoelectrón se generarán rayos X [3].

Efecto Compton:

El efecto Compton, también conocido como dispersión Compton, es un fenómeno que se produce cuando un fotón (normalmente de rayos X o gamma) colisiona con un electrón de un núcleo o libre, haciendo que el fotón pierda energía y cambie de dirección, mientras que el electrón se dispersa y gana energía cinética.

El efecto Compton puede explicar considerando la dualidad onda-partícula de la luz. Según la mecánica cuántica, la luz puede describirse como una onda y una partícula, conocida como fotón. Cuando un fotón colisiona con un electrón como se puede ver en la Figura 3, se comporta como una partícula y transfiere parte de su energía y momento al electrón. Esto hace que el fotón pierda energía y cambie su longitud de onda y dirección, mientras que el electrón se dispersa con un ángulo y velocidad diferentes.

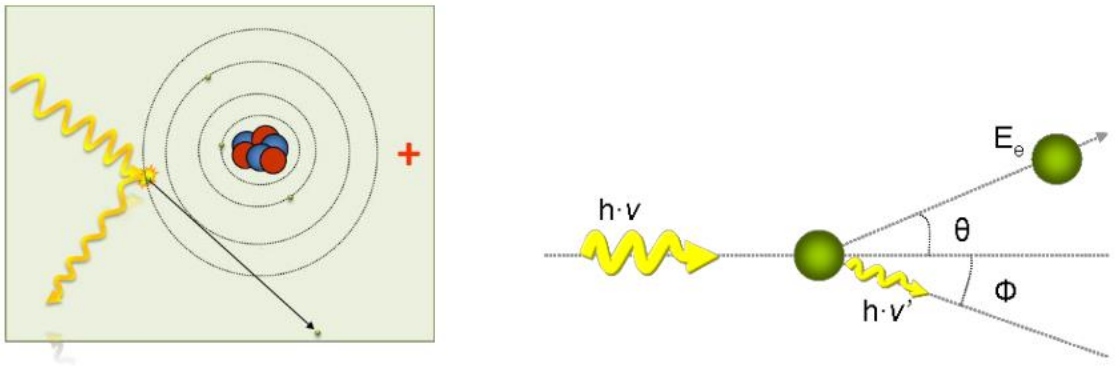

Figura 3. Representación gráfica del efecto Compton. Fuente: [2]

La cantidad de energía perdida por el fotón y ganada por el electrón en el efecto Compton depende del ángulo entre el fotón incidente y el fotón dispersado, así como de la masa y la velocidad del electrón. La longitud de onda del fotón dispersado es mayor que la del fotón incidente, y la diferencia de longitud de onda se conoce como desplazamiento Compton [4].

3.1.2.2 Radiación de electrones

En la radiación de electrones el efecto predominante que se produce es la dispersión Coulombiana, también conocida como la dispersión de Rutherford. La dispersión de Coulomb es un tipo de dispersión que se produce cuando las partículas cargadas interactúan a través de sus campos eléctricos. La dispersión surge de la fuerza de Coulomb, que es la fuerza de atracción o repulsión entre dos partículas cargadas que es proporcional al producto de sus cargas e inversamente proporcional a la distancia entre ellas.

En la dispersión de Coulomb, la trayectoria de una partícula cargada se ve alterada por la fuerza de Coulomb de otra partícula cargada o de un conjunto de partículas cargadas, lo que provoca que se disperse en una dirección diferente. Este efecto se puede ver en el experimento realizado por Rutherford (Figura 4) en el que se bombardea una lámina de oro con partículas cargadas positivamente y hay algunas que se desvían debido a las fuerzas de Coulomb entre las regiones positivas de los átomos de oro y las partículas cargadas [5].

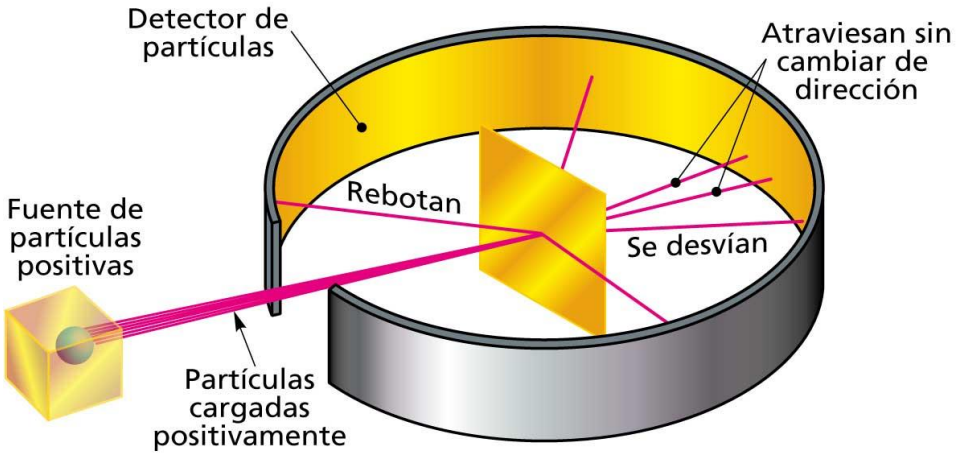

Figura 4. Experimento de Rutherford. Fuente: [3]

3.1.3 ¿Qué es la dosimetría personal?

La dosimetría personal es un método de medición y evaluación de la exposición de una persona a la radiación ionizante en el lugar de trabajo o en otros entornos en los que puede existir una exposición a la radiación. El objetivo de la dosimetría personal es controlar el nivel de exposición a la radiación de las personas que trabajan con fuentes de radiación ionizante o cerca de ellas, como los rayos X, los isótopos radiactivos o las centrales nucleares.

Figura 5. Ejemplo de dosímetro personal. Fuente: [4]

La dosimetría personal implica el uso de dosímetros, que son pequeños dispositivos que lleva la persona (Figura 5). Los dosímetros miden la cantidad de radiación a la que ha estado expuesto el individuo durante un periodo de tiempo, normalmente un día o una semana. Los datos recogidos por los dosímetros se analizan para determinar si la exposición de la persona a la radiación ha superado los límites de seguridad recomendados [6].

3.1.4 Tipos de dosímetros de radiación

Los dosímetros personales son dispositivos que se utilizan para medir y registrar la dosis de radiación recibida por un individuo durante un periodo de tiempo. Cada tipo de dosímetro personal tiene sus propias ventajas y limitaciones, y la elección del dosímetro dependerá de la aplicación específica y de los requisitos normativos [7].

Algunos de los tipos más comunes son:

- **Placas de película radiocrómica:** Las placas de películas radiocrómicas son uno de los tipos de dosímetros personales más sencillos. Consisten en un pequeño trozo de película sensible a la radiación, encerrado en un soporte de plástico. La película se expone a la radiación y el grado de oscurecimiento se utiliza para calcular la dosis de radiación.

Figura 6. Dosímetro de placas de película radiocrómica. Fuente: [5]

- **Dosímetros termoluminiscentes (TLD):** Los TLD utilizan un pequeño cristal, normalmente de fluoruro de litio, que es sensible a la radiación. Cuando el cristal se calienta, emite una luz proporcional a la cantidad de radiación que ha recibido. La intensidad de la luz se mide y se utiliza para calcular la dosis de radiación.

Figura 7. Dosímetro termoluminiscente. Fuente: [5]

- **Dosímetros de luminiscencia ópticamente estimulada (OSL):** Los dosímetros OSL utilizan un pequeño trozo de óxido de aluminio u otro material similar, que también es sensible a la radiación. Cuando se expone a la radiación, el material almacena energía, que se libera cuando es estimulado por la luz. La cantidad de luz emitida es proporcional a la cantidad de radiación recibida.

Figura 8. Dosímetro de luminiscencia ópticamente estimulada. Fuente: [5]

- **Cámaras de ionización de bolsillo:** Las cámaras de ionización de bolsillo son pequeños dispositivos portátiles que utilizan una cámara llena de gas para medir la radiación y pueden proporcionar mediciones en tiempo real.

Figura 9. Cámara de ionización de bolsillo. Fuente: [5]

- **Dosímetros personales electrónicos:** Estos dosímetros utilizan componentes electrónicos para medir y registrar la dosis de radiación. Pueden proporcionar lecturas de dosis en tiempo real, y algunos modelos también pueden almacenar los datos para su posterior análisis. Son los más utilizados en la actualidad y existen principalmente dos tipos en función del componente electrónico usado para detectar la radiación, los transistores MOSFET y los diodos [8]. En este proyecto se han utilizado los dosímetros basados en diodos de tipo PIN, por lo que se ha detallado su funcionamiento en el apartado 3.1.5.

Figura 10. Dosímetro personal electrónico. Fuente: [6]

3.1.5 El dosímetro electrónico de diodo

Un dosímetro de diodo es un tipo de detector de radiación que mide la cantidad de radiación ionizante que ha sido absorbida por un material. Funciona utilizando un diodo semiconductor, que es un tipo de componente electrónico que permite el flujo de corriente eléctrica en una sola dirección.

El dosímetro de diodo consiste en un diodo de unión p-n, que se fabrica dopando un material semiconductor con impurezas para crear una región con exceso de electrones (tipo n) y una región con déficit de electrones (tipo p). Cuando las dos regiones se unen, se forma una barrera de potencial en la unión, que permite que la corriente fluya en una dirección (Figura 11).

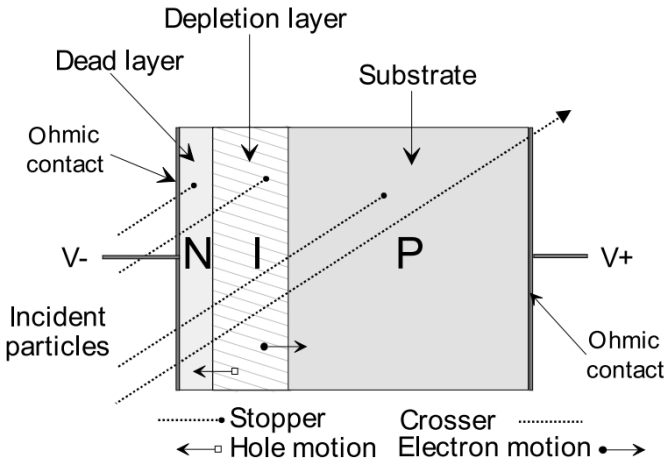

Figura 11. Detección de la radiación en un diodo PIN. Fuente: [7]

Para utilizar un dosímetro de diodo, éste se coloca en la trayectoria de la radiación que se desea medir. Cuando la radiación interactúa con el material semiconductor, crea pares electrón-hueco, que generan una pequeña corriente que fluye a través del diodo. La cantidad de corriente generada es proporcional a la cantidad de radiación absorbida por el diodo [8].

El uso de estos diodos para la detección de radiación se encuentra actualmente en fase de investigación y tienen la gran ventaja frente a las demás tecnologías de ser muy económicas de producir. Uno de los problemas que presentan este tipo de diodos es que pueden generar falsos positivos debido a fenómenos externos como las fuentes de calor, golpes o ruido electromagnético. Esto requiere de un procesamiento de las señales temporales de salida mediante un algoritmo que distinga entre las señales de radiación y las de ruido. Es por esta razón por la que se propone en este proyecto entrenar un algoritmo de inteligencia artificial que sea capaz de hacer esa clasificación binaria, los cuales se describirán teóricamente a continuación.

3.2 Algoritmos de inteligencia artificial

En este apartado se recogen las nociones teóricas sobre los algoritmos de inteligencia artificial utilizados en el clasificador binario. Se han evaluado los algoritmos más usados para la clasificación de series temporales, se explican las nociones básicas sobre las redes neuronales, redes neuronales convolucionales y transformers porque han sido las arquitecturas elegidas para el proyecto. Finalmente, se introduce la librería Keras usada para implementar estos algoritmos.

3.2.1 Algoritmos de inteligencia artificial usados para la clasificación de series temporales

En las últimas dos décadas, la clasificación de las series temporales se ha considerado como uno de los mayores retos en el ámbito del data mining. Debido al ordenamiento temporal de los datos, las series temporales están presentes en una gran cantidad de aplicaciones como dispositivos IoT, aparatos médicos de medición, sensores etc. Dada la necesidad de clasificar las series temporales, se están desarrollando algoritmos basados en deep learning que están obteniendo muy buenos resultados [9]. En la Figura 12 se puede ver cómo se realizaría la clasificación de series temporales mediante deep learning.

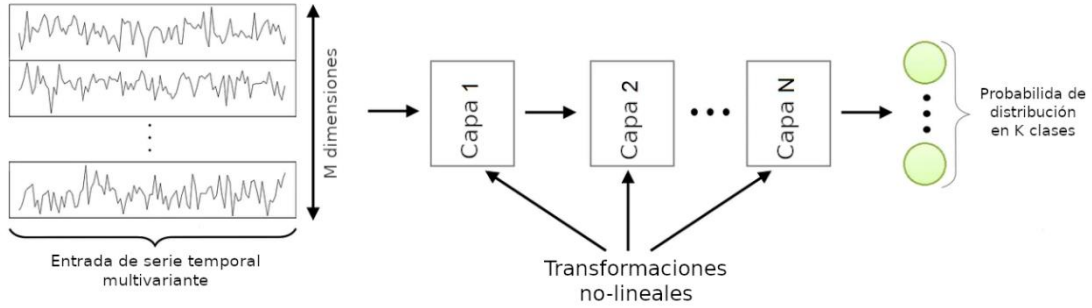

Figura 12. Clasificación de series temporales mediante deep learning. Fuente: [8]

Existen varios tipos de arquitecturas de redes neuronales que pueden utilizarse para clasificar series temporales en función de la complejidad del problema y de los requisitos específicos de la aplicación:

- **Redes neuronales recurrentes (RNN):** Las RNN son un tipo de red neuronal muy adecuada para procesar datos secuenciales como las series temporales. Tienen un bucle de retroalimentación que les permite retener información sobre entradas anteriores, lo que puede ser útil para predecir valores futuros. Una variante popular de las RNN es la red de memoria a largo plazo (LSTM), que puede captar dependencias a largo plazo.
- **Redes neuronales convolucionales (CNN):** Aunque suelen utilizarse para la clasificación de imágenes, las CNN también pueden emplearse para clasificar datos de series temporales. En este caso, se utiliza la arquitectura 1D-CNN, en la que la secuencia de entrada se trata como una señal 1D y se procesa con filtros convolucionales 1D para extraer las características relevantes. Los mapas de características resultantes pasan por capas totalmente conectadas para su clasificación.
- **Transformers:** Los modelos basados en transformers han ganado popularidad recientemente en tareas de procesamiento del lenguaje natural, pero también pueden utilizarse para la clasificación de series temporales. Estos modelos se basan en un mecanismo de atención para comprender selectivamente diferentes partes de la secuencia de entrada, lo que les permite manejar secuencias largas y capturar dependencias temporales.

- **Modelos de ensamble:** Los modelos basados en ensambles o la combinación de varios modelos, también pueden utilizarse para clasificar datos de series temporales. Esto implica entrenar múltiples modelos con diferentes arquitecturas e hiperparámetros y combinar sus predicciones para mejorar la precisión y la robustez del resultado final.

En este proyecto se han realizado dos modelos diferentes, uno mediante redes neuronales convolucionales y otro con transformers para conseguir un clasificador binario de series temporales.

3.2.2 Redes neuronales

En este apartado se explica el funcionamiento de los algoritmos de redes neuronales, la función de pérdida, la retropropagación y el descenso del gradiente.

3.2.2.1 Funcionamiento del algoritmo

Las redes neuronales son una clase de algoritmo de aprendizaje automático que se inspiran en la estructura y el funcionamiento del cerebro humano. Están formadas por capas de nodos interconectados, llamados neuronas, que procesan la información de forma paralela y distribuida [10]. En la Figura 13 se puede ver una representación de la arquitectura de una red neuronal totalmente conectada.

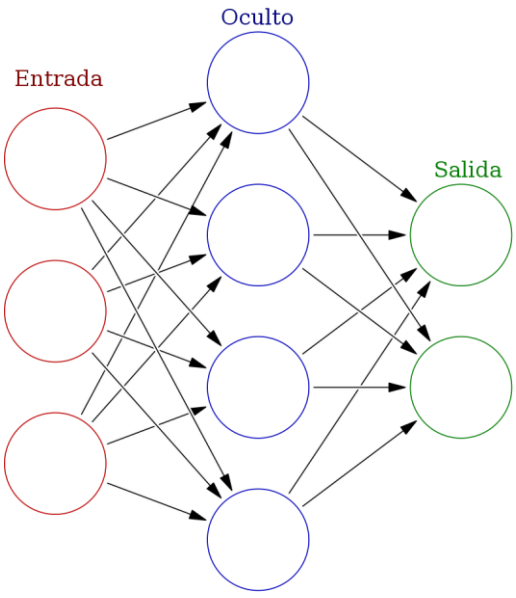

Figura 13. Representación de una red neural.
Fuente: [9]

El componente básico de una red neuronal es la neurona, que recibe información de otras neuronas o de una fuente externa, realiza un cálculo a partir de esa información y produce una salida que envía a otras neuronas. El cálculo que realiza una neurona suele ser una suma ponderada de sus entradas, seguida de una función de activación no lineal

que introduce la no linealidad en la red (Figura 14). Esta no linealidad permite a la red aprender patrones complejos de los datos.

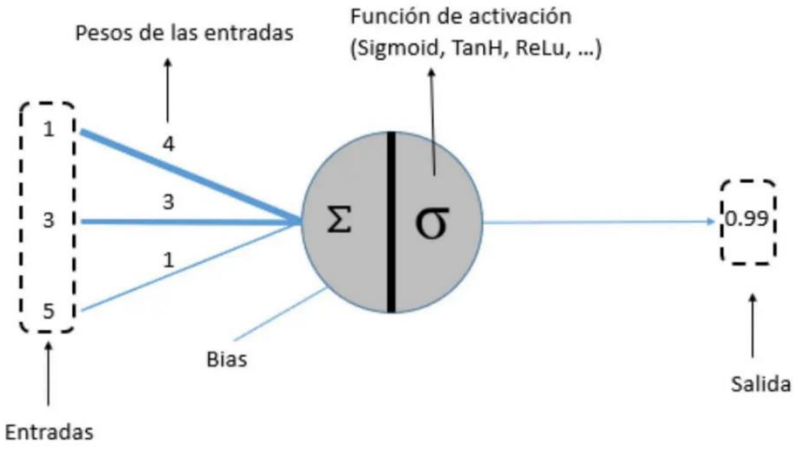

Figura 14. Funcionamiento de una neurona. Fuente: [10]

Las ecuaciones de ajuste lineal (1) aplicándole la función de activación (2) son las siguientes:

$$f(x) = \sum_i w_i x_i + b \tag{1}$$

$$g\big(f(x)\big) = g\left(\sum_i w_i x_i + b\right) \tag{2}$$

Siendo:

- w_i los pesos asociados al ajuste lineal.
- b correspondiente al sesgo (*bias*).
- f corresponde al ajuste lineal.
- g representa la función de activación.

Una red neuronal suele constar de una capa de entrada, una o varias capas ocultas y una capa de salida. La capa de entrada recibe los datos brutos, como una imagen o una secuencia de texto, y los pasa a la primera capa oculta. Cada neurona de la capa oculta calcula una suma ponderada de sus entradas, aplica una función de activación y produce una salida que pasa a la siguiente capa. Este proceso se repite en cada capa oculta posterior hasta llegar a la capa de salida, que produce la salida final de la red.

Las redes neuronales pueden utilizarse para diversas tareas, como la clasificación de imágenes, análisis de regresión, el procesamiento del lenguaje natural y el

reconocimiento del habla. Han obtenido excelentes resultados en muchas pruebas comparativas y su uso está muy extendido en la industria y el mundo académico [11].

3.2.2.2 Función de pérdida

La función de pérdida o de coste es una función matemática que mide la diferencia entre la salida prevista y la salida real. La función de pérdida es una componente clave del proceso de entrenamiento porque guía al algoritmo de optimización para ajustar los pesos y sesgos de la red con el fin de minimizar la función de pérdida.

Durante el entrenamiento, los pesos y los sesgos de la red se ajustan para minimizar la función de coste mediante un algoritmo de optimización como el descenso del gradiente. El algoritmo de optimización calcula el gradiente de la función de coste con respecto a los pesos y sesgos, ajustándolos en la dirección que reduzca la función de coste [12].

Existen varios tipos de funciones de pérdida en las redes neuronales, y la elección de cuál utilizar depende del problema específico y del tipo de salida de la red que se esté buscando. Algunas de las funciones de pérdida más utilizadas son:

- **Error cuadrático medio (MSE):** Esta es una función de pérdida popular para problemas de regresión, donde el objetivo es predecir un valor numérico continuo. El MSE mide la media de las diferencias al cuadrado entre los valores predichos y los reales.

$$MSE = \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N} \quad (3)$$

  Siendo:
  - N: Número de muestras.
  - y_i: Valor actual.
  - $\hat{y}_i$: Valor predicho.

- **Entropía cruzada binaria:** Esta función de pérdida se utiliza normalmente en problemas de clasificación binaria en los que el resultado es 0 o 1. La entropía cruzada binaria mide la diferencia entre la probabilidad predicha y la etiqueta binaria real, penalizando al modelo cuanto más se aleje la probabilidad predicha de la etiqueta real.

$$Entropía\ cruzada\ binaria = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log\big(p(y_i)\big) + (1 - y_i) \cdot log\big(1 - p(y_i)\big) \quad (4)$$

  Siendo:
  - N: Número de muestras.
  - y_i: Indicador binario de clase (0, 1).
  - $p(y_i)$: Probabilidad predicha para la clase.

- **Entropía cruzada categórica:** Esta función de pérdida se utiliza para problemas de clasificación multiclase, en los que hay más de dos resultados posibles. La entropía cruzada categórica mide la diferencia entre la distribución de probabilidad predicha y la etiqueta real.

$$Entropíacruzada\ categórica = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} \cdot log\left(p(y_{ij})\right) \quad (5)$$

Siendo:

- N: Número de muestras.
- M: Número de clases.
- y_{ij}: Indicador binario de clase según one-hot encoding.
- $p(y_{ij})$: Probabilidad predicha para la clase.

- **Pérdida de bisagra:** Esta función de pérdida se utiliza habitualmente para máquinas de vectores de soporte (SVM) y otros clasificadores binarios. Mide la diferencia entre la probabilidad predicha y la verdadera, pero sólo penaliza las predicciones incorrectas si superan un determinado umbral.

$$\mathrm{l}(y) = \max(0, 1 - t \cdot y) \quad (6)$$

Siendo:

- t: Clase objetivo {-1, 1}.
- y: Clase predicha.

- **Pérdida de divergencia de Kullback-Leibler (KL):** Esta función de pérdida mide la diferencia entre dos distribuciones de probabilidad, como las distribuciones de etiquetas predicha y real.

$$KL(\mathrm{P}||Q) = -\sum_{x} P(x) \cdot \log(\frac{Q(x)}{\mathrm{P}(x)}) \quad (7)$$

Siendo:

- P, Q: Distribución de probabilidad de una variable aleatoria discreta real.
- Q: Distribución de probabilidad de una variable aleatoria discreta aproximada.

3.2.2.3 Retropropagación

La retropropagación es un algoritmo de aprendizaje utilizado en redes neuronales para actualizar los pesos y sesgos en función del error entre la salida de la red y la salida deseada. Se basa en la regla de la cadena y consiste en calcular el gradiente de la función de pérdida con respecto a los pesos y sesgos de cada neurona [13].

El proceso de retropropagación comienza con el paso hacia delante, en el que los datos de entrada pasan por la red neuronal y se obtiene la salida. El error entre la salida y la salida deseada se calcula utilizando una función de pérdida. El paso hacia atrás comienza calculando el gradiente de la función de pérdida con respecto a la salida de la última capa de la red. A continuación, este gradiente se propaga hacia atrás a través de la red, capa por capa, utilizando la regla de la cadena para calcular el gradiente de la función de pérdida con respecto a los pesos y sesgos de cada neurona. Posteriormente, se utiliza el algoritmo del descenso del gradiente para actualizar los pesos y los sesgos de las neuronas en función de los gradientes calculados. Esto implica ajustar iterativamente los pesos y sesgos en la dirección opuesta al gradiente,

multiplicado por una tasa de aprendizaje, que determina el tamaño de la actualización en cada iteración [14].

Mediante la repetición iterativa de los pasos hacia delante y hacia atrás, y la actualización de los pesos y sesgos de las neuronas, la red neuronal aprende gradualmente a producir salidas más precisas para una entrada dada. La retropropagación es un algoritmo de aprendizaje potente y ampliamente utilizado que ha permitido el desarrollo de redes neuronales profundas, capaces de aprender patrones complejos de los datos [15].

3.2.2.4 Descenso del gradiente

El descenso del gradiente es un algoritmo de optimización que se utiliza habitualmente para actualizar los pesos y los sesgos de las neuronas de una red neuronal durante el proceso de entrenamiento. El objetivo del descenso del gradiente es encontrar los valores de los pesos y los sesgos que minimizan la función de pérdida, que mide la diferencia entre la salida de la red y la salida deseada [15].

El algoritmo funciona ajustando iterativamente los valores de los pesos y los sesgos en la dirección opuesta al gradiente de la función de pérdida con respecto a los pesos y los sesgos. Esto se basa en la intuición de que, si nos movemos en la dirección del descenso más pronunciado de la función de pérdida, eventualmente alcanzaremos un mínimo local, que corresponde a los valores óptimos de los pesos y sesgos (Figura 15).

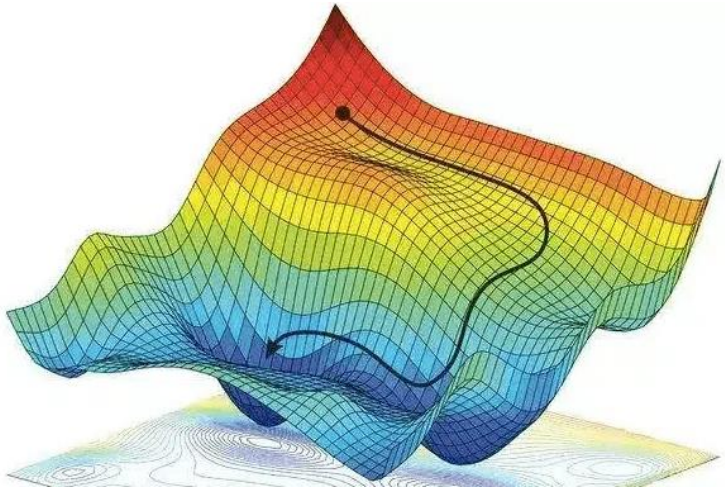

Figura 15. Ejemplo del descenso del gradiente para 3 dimensiones. Fuente: [11]

El gradiente se calcula utilizando la regla de la cadena, que consiste en calcular las derivadas parciales de la función de pérdida con respecto a cada peso y sesgo de la red. Este gradiente representa la dirección del ascenso más pronunciado de la función de pérdida, por lo que se actualizan los pesos y los sesgos en la dirección opuesta al gradiente para moverse hacia el mínimo. El tamaño de cada actualización viene determinado por la tasa de aprendizaje, que es un hiperparámetro que controla el tamaño del paso en la dirección del gradiente negativo (Figura 16). Si la tasa de aprendizaje es demasiado alta, el algoritmo puede sobrepasar el mínimo y no converger. Si la tasa de aprendizaje es demasiado pequeña, el algoritmo puede tardar demasiado en converger o quedarse atascado en un mínimo local subóptimo.

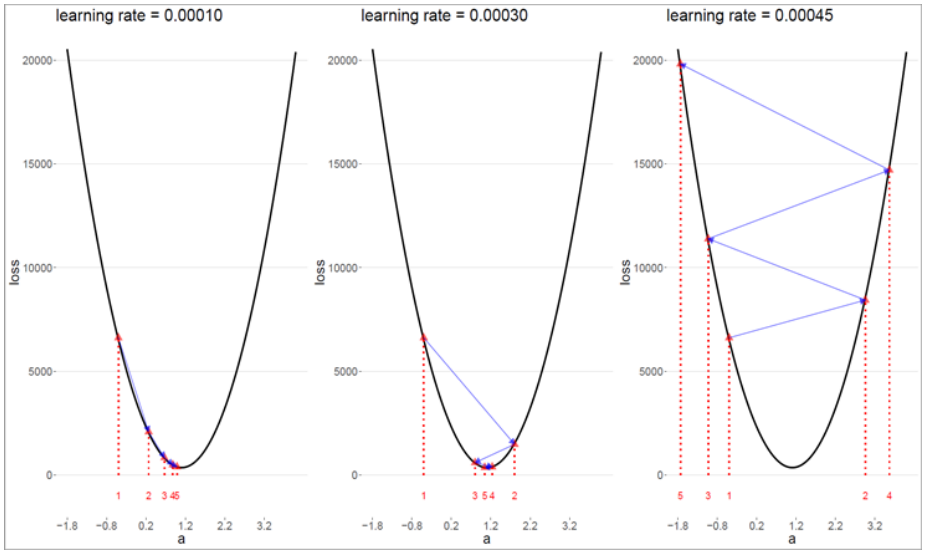

Figura 16. Ejemplo del descenso del gradiente con diferentes tasas de aprendizaje. Fuente: [12]

Existen diversas variantes del descenso del gradiente, las cuales introducen una complejidad adicional al algoritmo básico, pero a menudo pueden mejorar su

rendimiento y velocidad de convergencia. En la Figura 17 se pueden ver ilustrados algunos ejemplos y entre ellos se incluyen [16]:

- **Descenso del gradiente por lotes:** Este algoritmo implica calcular el gradiente de la función de pérdida con respecto a los parámetros para todo el conjunto de datos de entrenamiento. Esto significa que cada iteración del algoritmo actualiza los parámetros utilizando los gradientes de todo el conjunto de datos. El descenso del gradiente por lotes puede ser lento y consumir mucha memoria, pero puede converger a un mínimo global.
- **Descenso del gradiente estocástico (SGD):** Este actualiza los parámetros para cada ejemplo de entrenamiento individual, en lugar de para todo el conjunto de datos. Esto significa que el algoritmo puede converger más rápidamente que el descenso del gradiente por lotes, pero las actualizaciones pueden ser ruidosas y puede resultar difícil encontrar el mínimo global.
- **Descenso del gradiente por mini lotes:** Se trata de un compromiso entre el descenso del gradiente por lotes y el descenso del gradiente estocástico. Consiste en calcular los gradientes de un pequeño subconjunto de los datos de entrenamiento en cada iteración, denominado mini lote. Este enfoque puede ser más rápido que el descenso del gradiente por lotes y menos ruidoso que el descenso del gradiente estocástico.
- **Descenso del gradiente basado en el momento:** Utiliza un término de momento para acelerar la convergencia en el proceso de optimización. Consiste en actualizar los parámetros basándose en una media ponderada del gradiente actual y los gradientes anteriores. Esto ayuda a suavizar las actualizaciones y a reducir el impacto de gradientes ruidosos.
- **AdaGrad:** Es un tipo de descenso de gradiente que adapta la tasa de aprendizaje para cada parámetro basándose en el historial de gradientes para ese parámetro. Aumenta la tasa de aprendizaje para los parámetros que se actualizan con poca frecuencia y disminuye la tasa de aprendizaje para los parámetros que se actualizan con frecuencia.
- **Adam:** Es un algoritmo de optimización de la tasa de aprendizaje adaptativo que combina las ideas del descenso del gradiente basado en el momento y AdaGrad. Adapta la tasa de aprendizaje para cada parámetro basándose en el primer y segundo momento de los gradientes. Adam se utiliza habitualmente en aplicaciones de aprendizaje profundo.

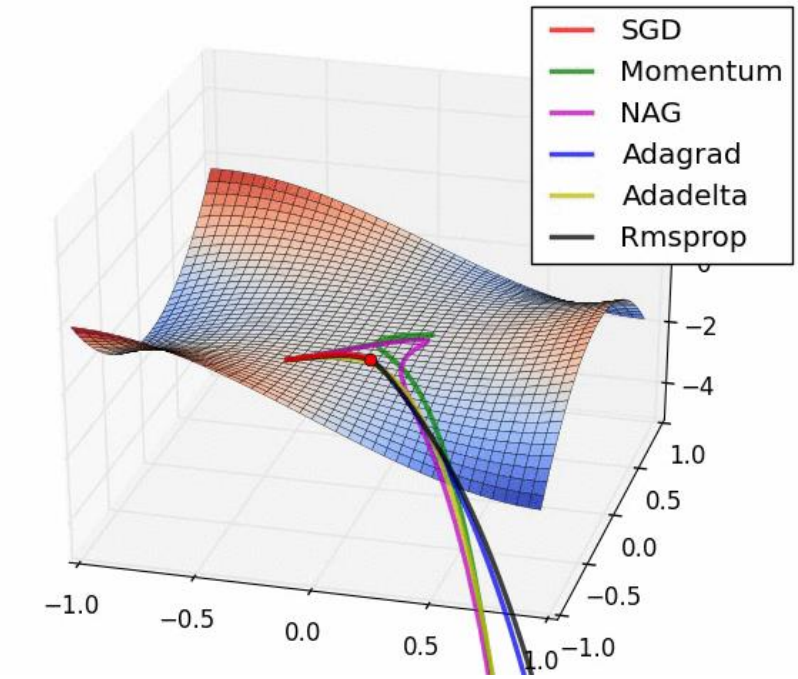

Figura 17. Ejemplo de diferentes variaciones del descenso del gradiente. Fuente: [13]

3.2.3 Redes neuronales convolucionales

Las redes neuronales convolucionales (CNN) se utilizan principalmente para el reconocimiento de imágenes, pero también pueden adaptarse para la clasificación de series temporales tratando los datos de las series temporales como una señal 1D [9].

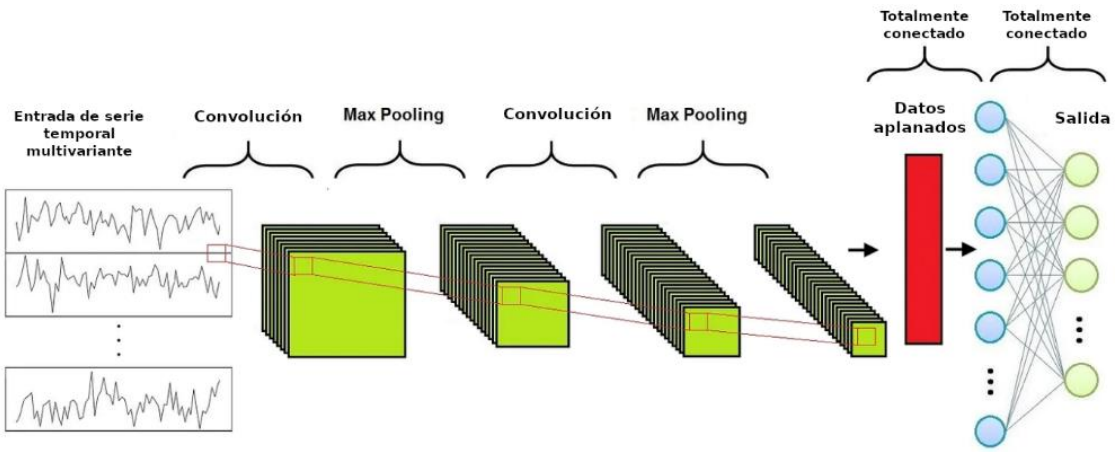

Figura 18. Arquitectura de un CNN para clasificar series temporales. Fuente: [8]

La arquitectura básica de una CNN consta de tres componentes principales (Figura 18): las capas convolucionales, las capas de agrupación (pooling) y las capas totalmente conectadas:

- **Capa convolucional:** La capa convolucional es el componente principal de la CNN. Aplica un conjunto de filtros a la secuencia de entrada y cada filtro aprende a detectar una característica concreta de los datos de la serie temporal. Los filtros se deslizan sobre la secuencia de entrada, realizando un producto escalar entre el filtro y una pequeña ventana de los datos de entrada en cada posición, generando un mapa de características.
- **Capa de agrupación (pooling):** La capa de agrupación se utiliza para reducir la dimensionalidad de los mapas de características generados por la capa convolucional, conservando la información más importante. El tipo más común de pooling utilizado en las CNN es el *Max Pooling*, que selecciona el valor máximo dentro de cada ventana del mapa de características.
- **Capa totalmente conectada:** La capa totalmente conectada toma la salida de las capas convolucionales y de agrupación, las aplana mediante una capa *flatten* y realiza la clasificación asignando las características a las clases de salida.

La salida de la última capa totalmente conectada se pasa a través de una función de activación *softmax* para producir las probabilidades de clase predichas. Durante el entrenamiento, los parámetros de la CNN, incluidos los filtros y los pesos de las capas totalmente conectadas, se optimizan mediante retropropagación para minimizar una función de pérdida elegida, como la entropía cruzada categórica.

Las CNN son eficaces para tareas de clasificación de series temporales, ya que pueden aprender automáticamente características relevantes a partir de los datos de entrada, reduciendo la necesidad de ingeniería de características. Además, las CNN pueden manejar secuencias de entrada de longitud variable, lo que las hace adecuadas para aplicaciones de series temporales que se encuentran en la práctica [9].

3.2.4 Transformers

Los modelos transformers son un tipo de arquitectura de red neuronal que se utiliza habitualmente en tareas de procesamiento del lenguaje natural (NLP) como la traducción de idiomas, el análisis de sentimientos y la generación de textos. Estos modelos se introdujeron en 2017 [17] y desde entonces, se han convertido en una componente fundamental de muchos sistemas de NLP de última generación [18].

A alto nivel, los modelos de transformers funcionan procesando secuencias de tokens de entrada (como palabras o caracteres) y produciendo secuencias de tokens de salida. La innovación clave de los modelos transformers es el mecanismo de atención, que permite al modelo centrarse en distintas partes de la secuencia de entrada al producir la secuencia de salida. El mecanismo de atención se basa en la idea de asignar pesos a las distintas partes de la secuencia de entrada, en función de su relevancia para el token de salida que se está generando. Estos pesos se calculan utilizando un conjunto

de parámetros aprendidos, que se actualizan durante el entrenamiento mediante retropropagación [17].

La arquitectura del transformer consiste en una pila de capas, cada una de las cuales contiene múltiples mecanismos de atención, así como redes neuronales feedforward que procesan la secuencia de entrada. Durante el entrenamiento, los parámetros de cada capa se actualizan mediante el descenso del gradiente, y todo el modelo se entrena de principio a fin para minimizar una función de pérdida que mide la discrepancia entre la secuencia de salida prevista y la secuencia de salida real.

Aunque los modelos de transformers se desarrollaron originalmente para tareas de procesamiento del lenguaje natural, también se han aplicado con éxito a problemas de clasificación de series temporales [19]. En este contexto, la entrada del modelo es una secuencia de medidas tomadas en intervalos regulares a lo largo del tiempo, y el objetivo es clasificar la serie temporal en una de varias categorías posibles.

Para utilizar un modelo transformer en la clasificación de series temporales, la secuencia de entrada se transforma primero en un conjunto de vectores de longitud fija mediante una técnica como la ventana deslizante o la convolución temporal. A continuación, estos vectores se introducen en el modelo transformer como si fueran tokens de una frase de lenguaje natural [20].

Una ventaja de utilizar modelos transformers para la clasificación de series temporales es su capacidad para capturar dependencias temporales complejas en la secuencia de entrada. Además, como el modelo transformer se entrena de principio a fin, puede aprender a extraer características relevantes de la secuencia de entrada sin necesidad de ingeniería de características manual [9].

3.2.5 Librería Keras

Keras es una biblioteca de redes neuronales de código abierto escrita en Python. Se desarrolló con el objetivo de ofrecer una experimentación rápida y una implementación sencilla de modelos de aprendizaje profundo. Keras fue desarrollado inicialmente por François Chollet en 2015 y desde entonces se ha convertido en una opción popular para construir modelos de aprendizaje profundo [21].

Figura 19. Logotipo de la librería Keras

Keras proporciona una API de alto nivel que facilita la definición, configuración y entrenamiento de una amplia gama de modelos de aprendizaje profundo. La biblioteca está construida sobre bibliotecas de computación numérica de bajo nivel, como TensorFlow, Theano o CNTK, que manejan los detalles de bajo nivel de la aceleración y optimización del hardware.

Una de las principales ventajas de Keras es su facilidad de uso. Ofrece una interacción sencilla e intuitiva que permite a los desarrolladores crear rápidamente prototipos de modelos de aprendizaje profundo sin necesidad de tener conocimientos de programación de bajo nivel o matemáticas complejas. Keras también incluye una serie de funciones integradas para tareas comunes como la carga de datos, el preprocesamiento y la evaluación, lo que simplifica aún más el proceso de desarrollo. Gracias a estas características y a la gran comunidad de desarrolladores que lo apoyan, actualmente es la segunda librería más usada en machine learning (Figura 20) [22].

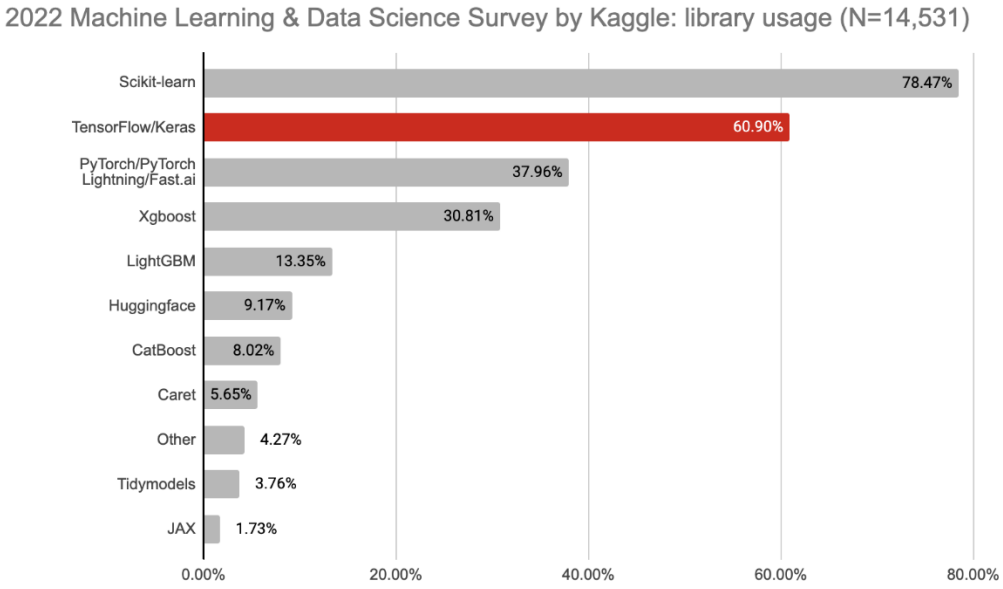

Figura 20. Tasa de popularidad de las librerías de machine learning. Fuente: [14]

Keras admite aceleración tanto en CPU como en GPU, lo que lo hace adecuado para entrenar modelos de aprendizaje profundo en una amplia gama de configuraciones de hardware. También incluye una serie de modelos y conjuntos de datos preentrenados, que pueden utilizarse como punto de partida para construir modelos más complejos.

Debido a las ventajas que ofrece esta librería para crear y entrenar modelos, esta ha sido la elegida para desarrollar el clasificador binario de series temporales.

4 Desarrollo del proyecto y resultados

En este apartado se va a describir la metodología seguida para desarrollar el clasificador binario para poder discriminar las señales de ruido generadas por un dosímetro de radiación, detallar cómo son los errores generados por los dosímetros, documentar el proceso completo hasta conseguir los modelos y finalmente se analizarán los resultados de los modelos.

4.1 Metodología CRISP-ML

En este apartado se va a describir la metodología CRISP-ML utilizada para el desarrollo del clasificador binario basado en modelos de machine learning.

La metodología CRISP-ML (Cross-Industry Standard Process for Machine Learning) es una metodología ampliamente utilizada y aceptada para proyectos de aprendizaje automático. Proporciona un enfoque estructurado para la gestión y ejecución de dichos proyectos, de principio a fin (Figura 21). El proceso es una variación de la metodología CRISP-DM para aplicarlo al machine learning e incluye seis fases [23]:

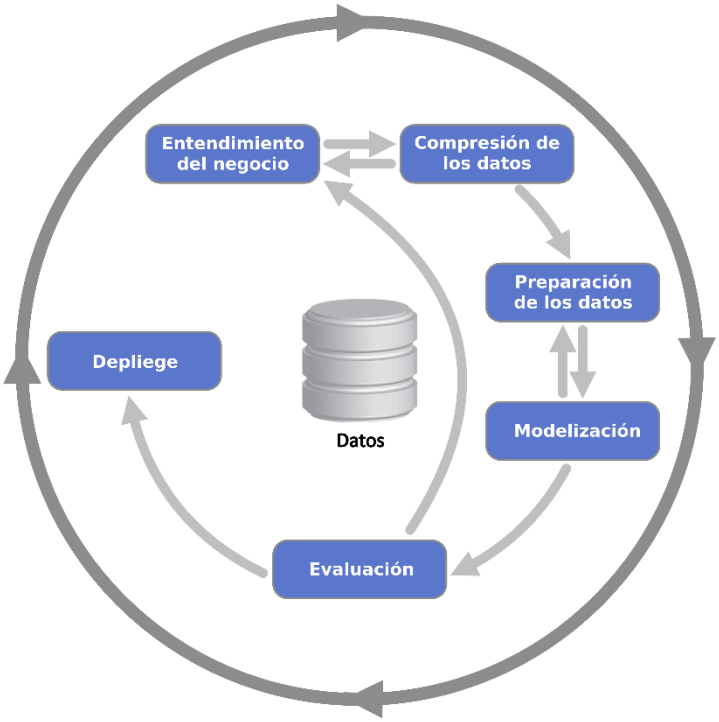

Figura 21. Fases de la metodología CRISP-ML. Fuente: [15]

1. **Entendimiento del Negocio:** En esta fase, las metas y objetivos del proyecto se definen mediante la comprensión del problema de negocio y la identificación de los objetivos del algoritmo.

2. **Comprensión de los datos:** En esta fase, los datos se recogen, revisan y exploran para entender su calidad, integridad, y las ideas iniciales que proporciona.
3. **Preparación de los datos:** Esta fase implica la limpieza de datos, transformación, selección de características, y la creación de nuevas variables que se utilizarán en la construcción de modelos.
4. **Modelización:** En esta fase, se aplican varios algoritmos de aprendizaje automático a los datos preprocesados para generar modelos capaces de predecir o clasificar la variable objetivo.
5. **Evaluación:** En esta fase, los modelos generados en la fase anterior se evalúan y comparan en función de las métricas de rendimiento y otros factores.
6. **Despliegue:** En esta fase final, se selecciona el modelo de mejor rendimiento y se despliega en el entorno de producción.

La metodología CRISP-ML se utiliza ampliamente en diversos sectores y se ha convertido en la norma de facto para los proyectos de minería de datos y aprendizaje automático. Su éxito puede atribuirse a su enfoque iterativo, que permite la retroalimentación y la modificación en cada etapa del proceso, lo que permite al equipo del proyecto ajustar y mejorar su trabajo según sea necesario [24].

En este proyecto se han realizado las fases 1-5 acabando en la evaluación de los modelos sin llegar a desplegarlo en un entorno real. El despliegue del modelo se ha dejado para trabajos futuros debido a que el dosímetro no puede procesar los algoritmos propuestos debido a su limitado hardware, siendo la única opción de hacerlo en un servidor mandando los datos a través de una conexión a internet.

4.2 Falsos positivos en los dosímetros

El dosímetro está formado por un diodo donde se puede medir la tensión que se produce cuando recibe radiación como se ha explicado en el apartado 3.1.5. La tensión que genera es proporcional a la cantidad de radiación recibida y gracias a este fenómeno se puede cuantificar el nivel de radiación al que está expuesto una persona.

El diodo está midiendo la radiación externa en todo momento y cuando se supera un umbral establecido, se detecta la radiación como se puede observar en la Figura 22. Esta se trata de una serie temporal obtenida durante la fase de experimentación con el dosímetro.

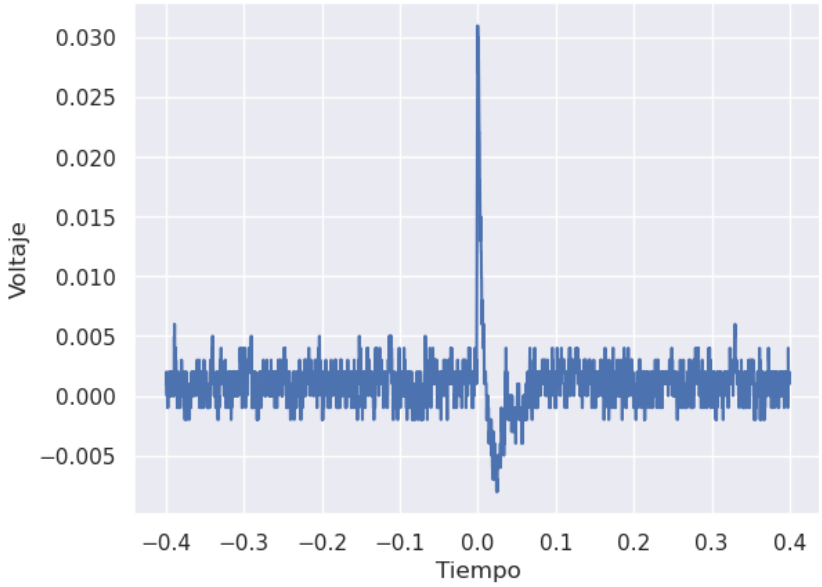

Figura 22. Serie temporal de radiación producida por el dosímetro

Uno de los problemas que presentan los dosímetros basados en diodos es que pueden dar falsos positivos debido a fenómenos externos que no sean radiación ionizante. Estos fenómenos externos producen señales como el de la Figura 23 que son erróneamente clasificadas como radiación. Se denominará como ruido las señales falsas positivas generadas por el dosímetro. Entre los fenómenos más habituales se encuentran:

- Ruido electromagnético.
- Fuentes de calor.
- Golpes e impactos físicos.

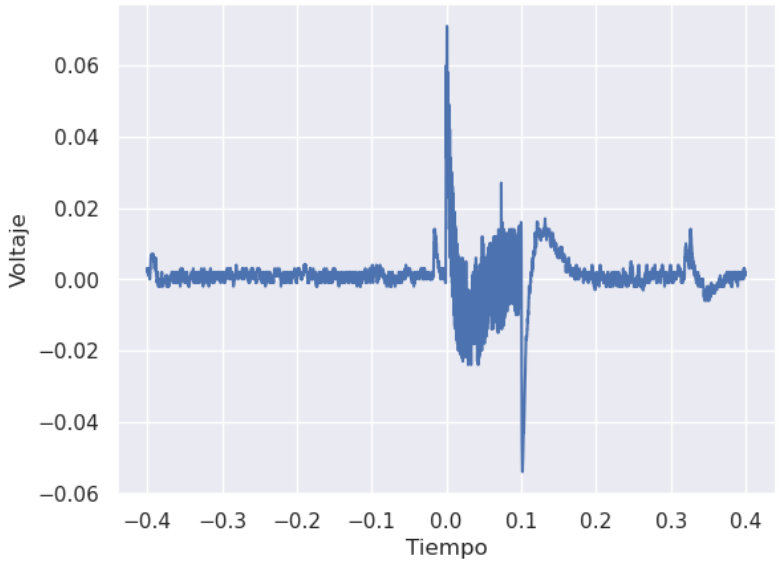

Figura 23. Medida de ruido producido por el dosímetro

Para poder filtrar este tipo de señales de ruido es necesario aplicar un algoritmo que sea capaz de analizar las mediciones del diodo. Es por ello por lo que se ha desarrollado en este proyecto un clasificador binario usando algoritmos de inteligencia artificial que sean capaces de distinguir entre las señales producidas por radiación y el ruido producido por fenómenos externos no deseados.

4.3 Desarrollo del proyecto

En este apartado se va a describir cómo se ha desarrollado el proyecto. Se va a explicar cómo se ha realizado la toma de datos en el laboratorio, el análisis de los datos obtenidos, la preparación de los datos y los modelos de deep learning realizados.

La parte del análisis y el código ha sido desarrollada en Python usando Jupyter notebook y los algoritmos se han entrenado en una instancia de Google Colab en un entorno con GPU. Todo el código y los datos se puede encontrar en el siguiente repositorio de GitHub: https://github.com/ilarman/Radiation-time-series-binary-classifier-using-CNNs-and-transformers

En la memoria no se va a incluir el código usado, se darán explicaciones de alto nivel de cómo se ha realizado el desarrollo y se mostrarán los resultados.

4.3.1 Toma de datos en el laboratorio

Debido a que no se tenían datos de las mediciones de radiación del dosímetro, estos se han tenido que medir en un laboratorio para poder crear un set de datos y poder entrenar los algoritmos de inteligencia artificial. En este apartado se va a explicar cómo se ha realizado el montaje experimental y cómo se han tomado los datos.

4.3.1.1 Montaje experimental

El diodo utilizado por el dosímetro es de tipo PIN y se trata del modelo BPW 34 FS de la marca Osram. En la Figura 24 se puede ver el dosímetro desmontado con el diodo expuesto para realizar las mediciones.

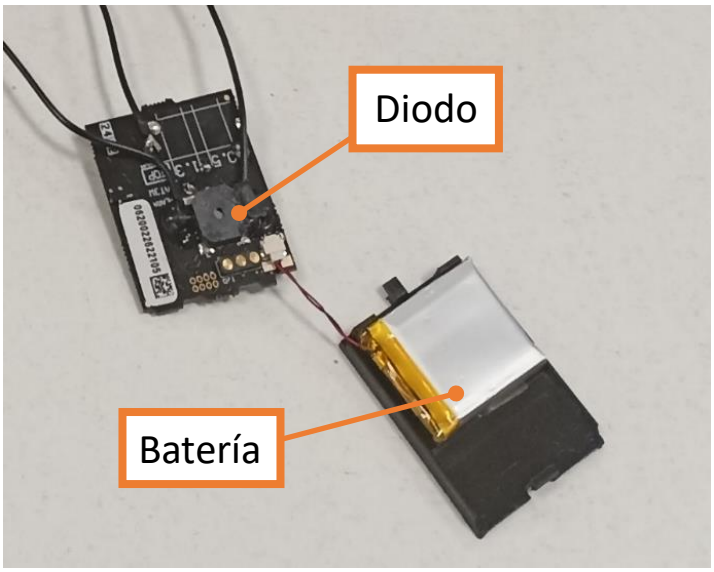

Figura 24. Dosímetro desmontado para las mediciones

Para poder medir la tensión generada por el diodo al recibir radiación, se ha usado el osciloscopio digital Hantek DSO1072E como se puede ver en la Figura 25. Este osciloscopio es capaz de guardar en una memoria externa las mediciones para poder exportar los datos.

Figura 25. Medición de la tensión del diodo con un osciloscopio

Como fuente de radiación se ha utilizado Cesio 137 (Figura 26). Este se trata de un isótopo radiactivo proveniente de productos de la fisión nuclear del uranio 235. El Cesio 137 decae a un estado metaestable del Bario 137 mediante la emisión de partículas beta de 0.512 MeV, el cual decae a su vez a un estado estable emitiendo rayos gamma de 0.6617 MeV [25].

Figura 26. Cesio 137 usado como fuente de radiación

Para medir la radiación con el diodo, se ha puesto el cesio 137 sobre el diodo como se puede ver en la Figura 27.

Figura 27. Medición de la radiación con cesio 137

4.3.1.2 Toma de datos

Para la toma de datos se ha seguido la siguiente secuencia:

1. Se ha realizado el montaje de la Figura 27.

2. Se ha configurado el osciloscopio para que guarde en memoria el intervalo de medición de ± 0.4 s en el momento que se supere 152 mV.
3. Se miden y guardan en memoria las series temporales producidas por la radiación.
4. Cada vez que el dosímetro detecta una alta radiación, hace sonar una alarma que lleva incorporada. Esta alarma genera ruido eléctrico y genera tensiones en el diodo que son erróneamente detectadas como radiación. Se miden y guardan en memoria estas series temporales de ruido.
5. El osciloscopio guarda cada serie temporal en un fichero CSV por separado. Se copian mediante un pendrive a un PC los CSVs para su posterior análisis y procesado.

4.3.2 Análisis de los datos

En este apartado se van a describir las conclusiones del análisis de los datos obtenidos.

4.3.2.1 Número de muestras

Durante la fase de toma de datos se han conseguido 402 series temporales de radiación y 409 correspondientes al ruido. Ambas clases están balanceadas, lo cual es favorable para la fase de entrenamiento de los algoritmos.

Sin embargo, debido a fallos humanos durante el guardado de los ficheros en el osciloscopio, se han duplicado algunas medidas y se han clasificado erroneamente algunas series temporales entre ruido y radiación. En la Tabla 1 se resume el número de series temporales afectadas.

	Radiación	Ruido
Nº de series temporales duplicadas	1	81
Nº de series temporales mal clasificadas	20	1
Total	21	82

Tabla 1. N.º de series temporales afectadas

Eliminando las series temporales duplicadas y mal clasificadas, se tienen 381 mediciones de radiación y 327 de ruido (Figura 28), por lo que puede considerarse que la muestra está balanceada en las clases.

Figura 28. Número de muestras sin duplicidad y correctamente clasificadas

4.3.2.2 Comparativa entre series temporales de radiación y ruido

En la Figura 29 se puede ver un ejemplo de las mediciones tomadas. La columna izquierda corresponde a las mediciones de radiación y la derecha las de ruido.

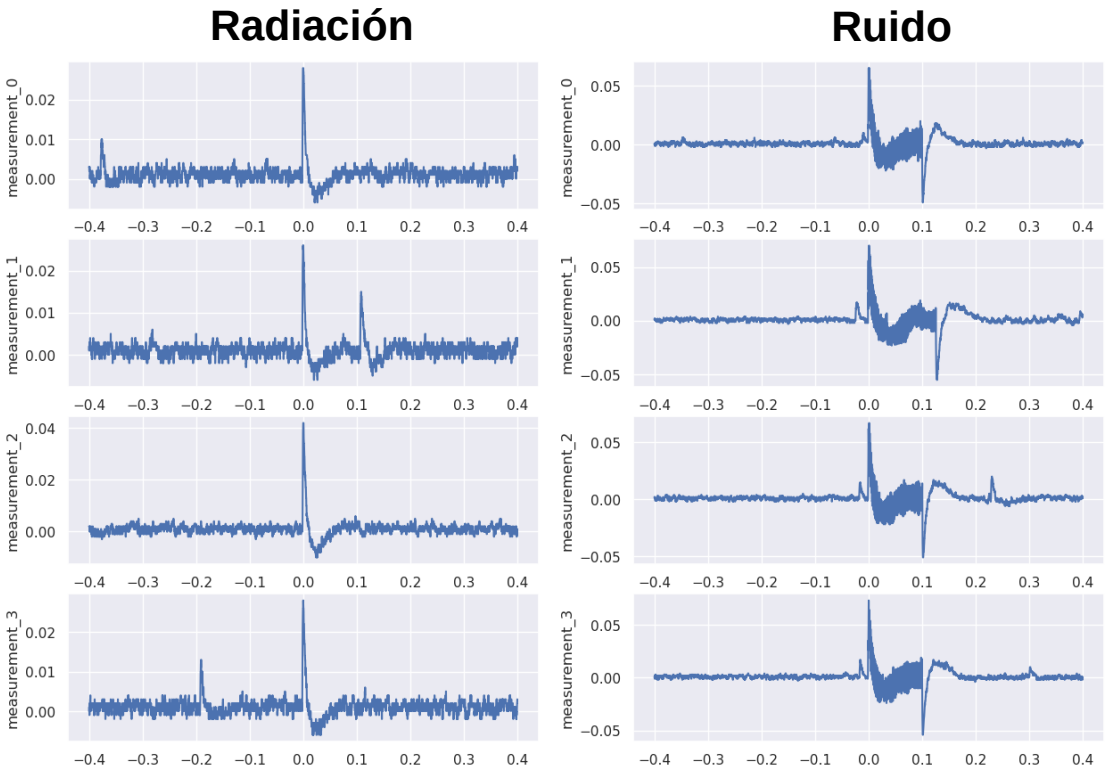

Figura 29. Muestras de series temporales de radiación y ruido

Se puede observar que existe una diferencia aparente en la forma de la señal entre las series temporales de ruido y radiación al visualizarlas. Superponiendo ambas mediciones en la Figura 30, se puede ver que la señal de ruido tiene una mayor diferencia entre los máximos y mínimos.

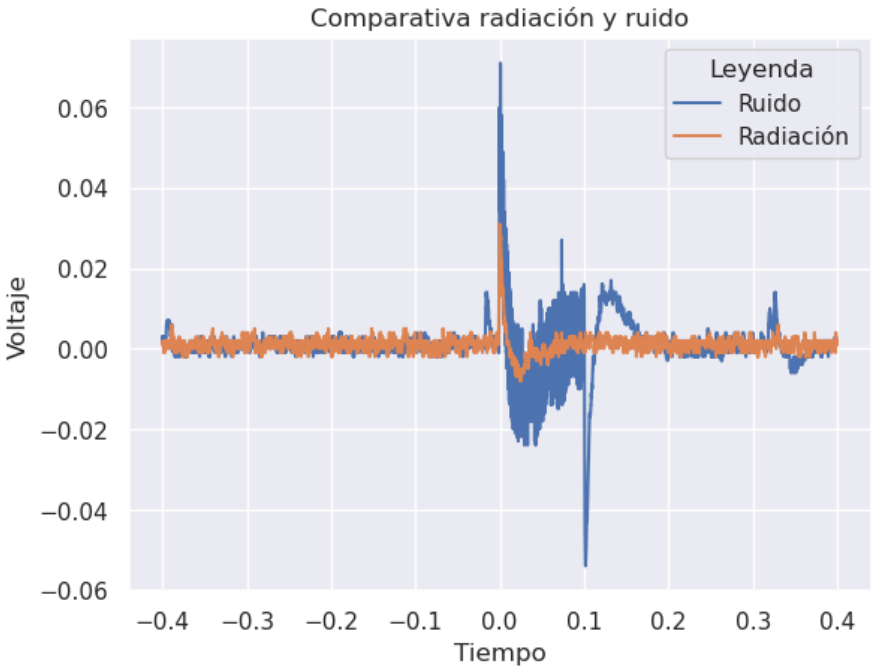

Figura 30. Comparativa entre radiación y ruido

4.3.2.3 Muestreo

Se puede observar en ambas señales que presentan mucha variación entre valores sucesivos al tener un muestreo de 4000 medidas en el intervalo de 0.8 segundos. Estas grandes variaciones pueden hacer que los algoritmos de clasificación presenten peores resultados, por lo que una opción para disminuir este efecto puede ser reducir el muestreo de las series temporales.

4.3.2.4 Información relevante

Se puede ver también que la mayoría de la información se encuentra centrada alrededor de t = 0 s donde ocurre el pico, siendo la información a partir de ±0.2 s poco relevante (Figura 31). Se podrían recortar todas las series temporales 0.2 s al principio y al final en la fase de preparación de datos para mantener la información más relevante.

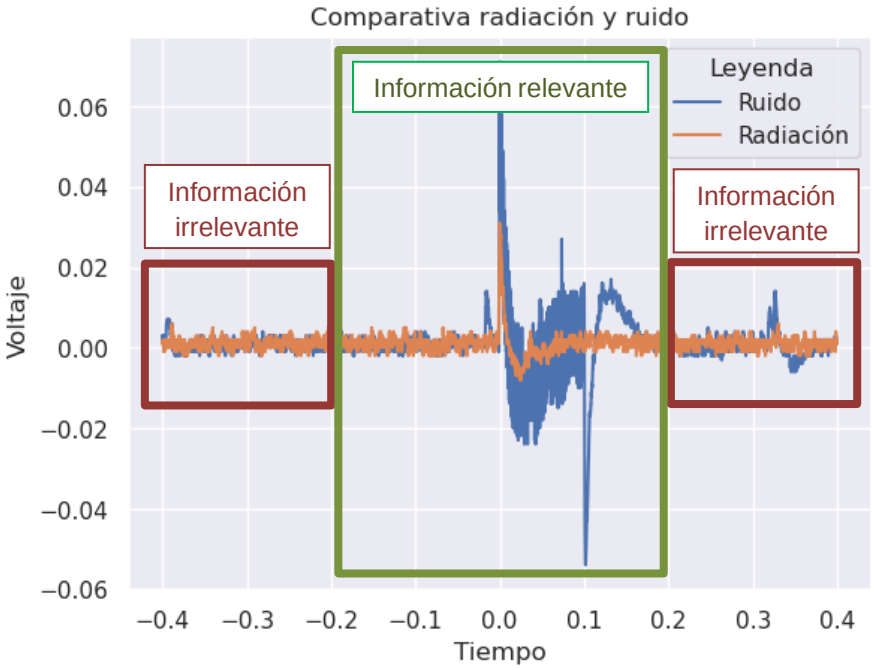

Figura 31. Información relevante en la serie temporal

4.3.3 Preparación de los datos

En este apartado se va a resumir la preparación y transformación aplicadas a los datos basado en la conclusión del análisis del apartado 4.3.2.

Se han realizado las siguientes transformaciones:

1. Eliminar las series temporales duplicadas.
2. Eliminar las series temporales mal clasificadas.
3. Eliminar las series temporales con valores faltantes.
4. Disminuir el muestreo de 4000 medidas a 400.
5. Recortar las series temporales 0.2 s al principio y al final para tener solamente información relevante.
6. Crear un conjunto de datos dividido en un 80 % para entrenamiento y un 20 % para validación.
7. Normalizar los datos.

A continuación, se van a describir brevemente las trasformaciones realizadas.

4.3.3.1 Eliminar las series temporales duplicadas

Se eliminan las series temporales duplicadas según se ha analizado en el apartado 4.3.2.1.

4.3.3.2 Eliminar las series temporales mal clasificadas

Se eliminan las series temporales mal clasificadas según se ha analizado en el apartado 4.3.2.1.

4.3.3.3 Eliminar las series temporales con valores faltantes

Se eliminan todas las series temporales que contengan algún valor faltante. En la Tabla 2 se recoge el número de series temporales eliminadas para cada caso.

Tipo de serie temporal	Número de series temporales con faltantes
Radiación	1
Ruido	20
Total	2

Tabla 2. Número de series temporales con datos faltantes

4.3.3.4 Disminuir el muestreo de 4000 medidas a 400

Se disminuye el muestreo de 4000 medidas a 400 para cada serie temporal. Para ello, se ha elegido el valor máximo cada 10 medidas. En la Figura 32 y Figura 33 se puede ver una comparativa entre la serie temporal original y la de muestreo disminuido para las mediciones de radiación y ruido.

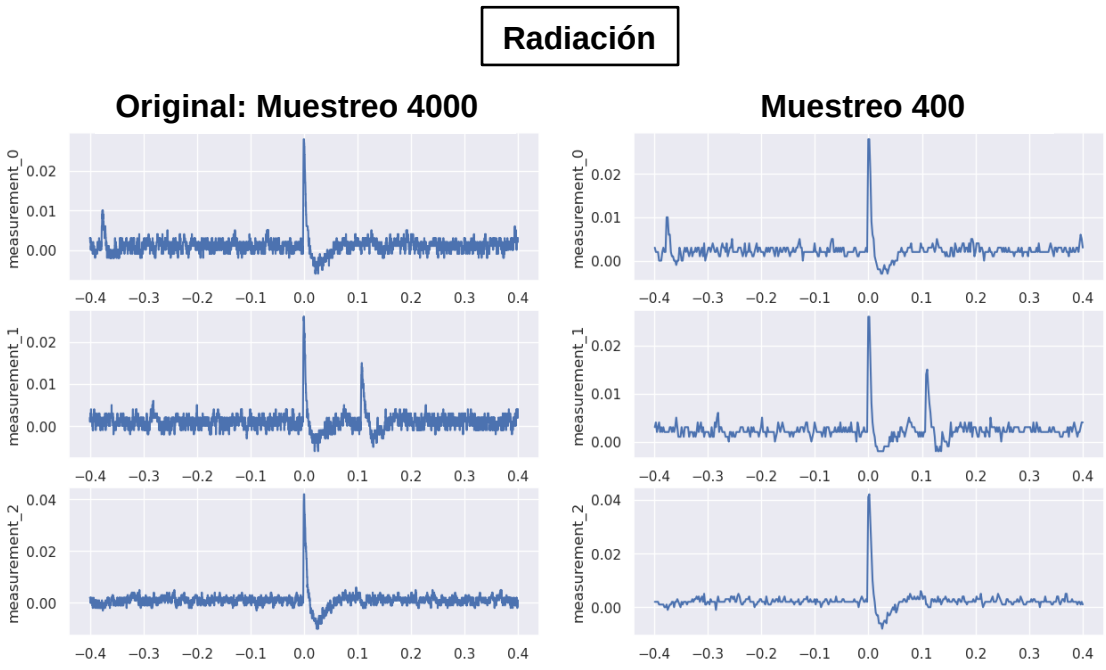

Figura 32. Comparativa del muestreo para series temporales de radiación

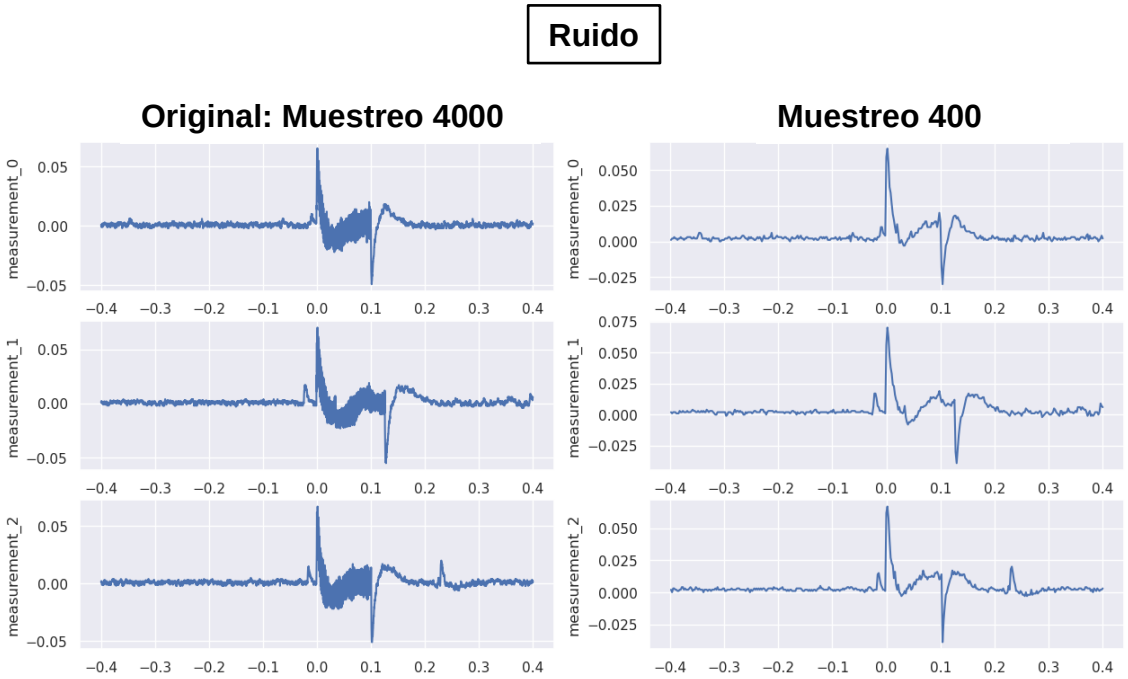

Figura 33. Comparativa del muestreo para series temporales de ruido

Se puede observar que para ambos casos la señal es mucho más limpia y no existe pérdida de información.

4.3.3.5 Recortar las series temporales 0.2 s al principio y al final para tener solamente información relevante

Se van a recortar todas las series temporales 0.2 segundos al principio y al final según lo analizado en el apartado 4.3.2.4.

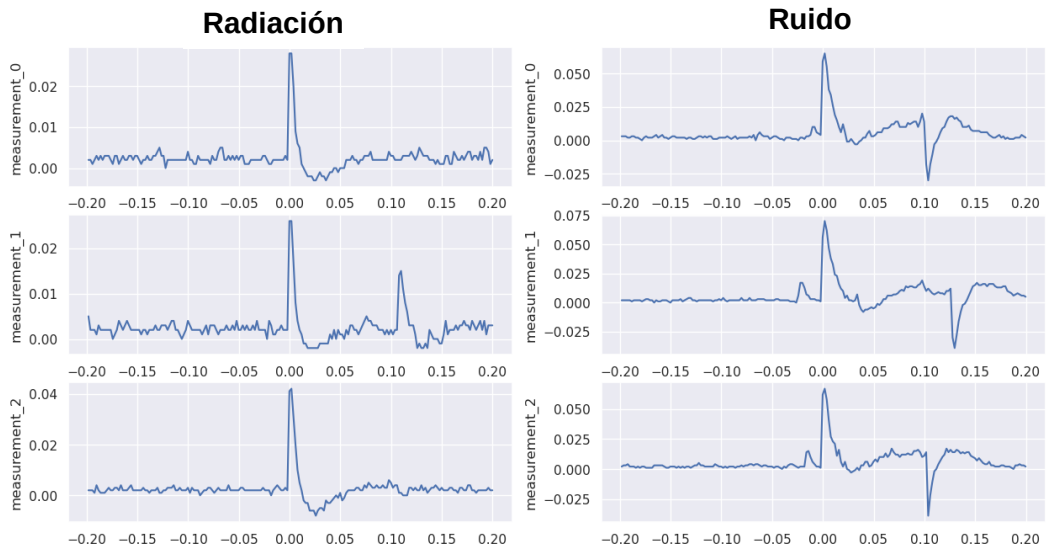

Figura 34. Series temporales recortadas ±0.2 s

En la Figura 34 se puede ver que se sigue conservando la información relevante.

4.3.3.6 Crear un conjunto de datos dividido en un 80 % para entrenamiento y un 20 % para validación

Se dividen el conjunto de datos en dos subconjuntos, el de entrenamiento consistiendo el 80 % de las series temporales y el de validación, el 20 % restante.

4.3.3.7 Normalizar los datos

Se normalizan los valores de las tensiones aplicando una interpolación lineal en el rango [0,1] para realizar el entrenamiento de los algoritmos de redes neuronales.

4.3.4 Modelización

En este apartado se van a describir cómo se han realizado los modelos basados en redes neuronales para conseguir un clasificador binario que sea capaz de distinguir entre las series temporales de radiación y las de ruido generadas por el dosímetro. Se han desarrollado dos modelos, uno basado en redes neuronales convolucionales y otro en transformers.

4.3.4.1 Modelo basado en redes neuronales convolucionales

Se ha usado la guía [26] de Keras como referencia para realizar el modelo basado en redes neuronales convolucionales. La arquitectura del modelo está basada en el artículo científico [27], en este se propone un modelo para poder clasificar series temporales sin requerir de un alto preprocesamiento de los datos en bruto y la recomiendan como buen punto de partida gracias a su buena capacidad de generalizar en usos para el mundo real.

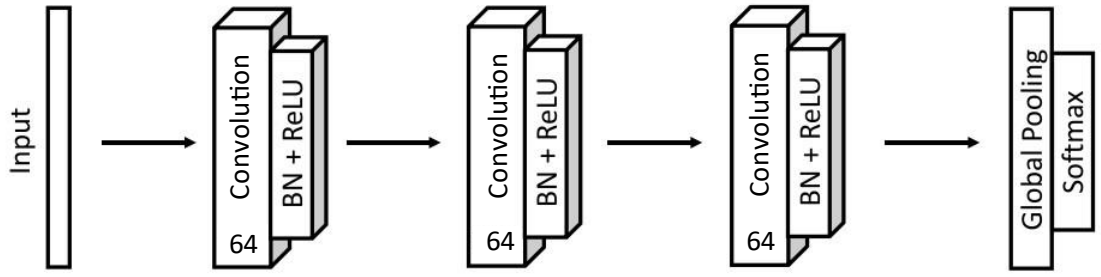

Figura 35. Arquitectura usada basada en redes neuronales convolucionales. Fuente: [16]

La arquitectura (Figura 35) consiste en tres bloques básicos formados por una capa convolucional, seguido de una capa batch normalization y una capa de activación ReLU. La operación de convolución se realiza por tres kernels 1D de tamaño 3 aplicando padding y se ha usado un tamaño de filtro de 64 para los tres bloques. El objetivo de aplicar la capa de batch normalization es aumentar la velocidad de convergencia y

mejorar la generalización. Tras aplicar los bloques convolucionales, se aplica una capa global average pooling, el cual reduce mucho el número de pesos comparada con una capa totalmente conectada y finalmente se realiza la clasificación mediante una capa softmax.

Para la fase de entrenamiento se han utilizado los mismos hiperparámetros que en la guía [26]:

- Nº de épocas: 500 (Se aplica un EarlyStopping cuando la función de pérdida deja de mejorar en 20 épocas).
- Batch size: 32
- Optimizador del descenso del gradiente: Adam
- Learning rate: $1e^{-3}$
- Función de pérdida: Sparse categorical crossentropy
- Métrica de precisión: Sparse categorical accuracy
- División datos de validación: 20 %

4.3.4.2 Modelo basado en transformers

Se ha usado la guía [28] de Keras como referencia para realizar el modelo basado en transformers. La arquitectura del modelo está basada en el conocido artículo científico llamado *Attention is all you need* [17], en este se propone la arquitectura de los

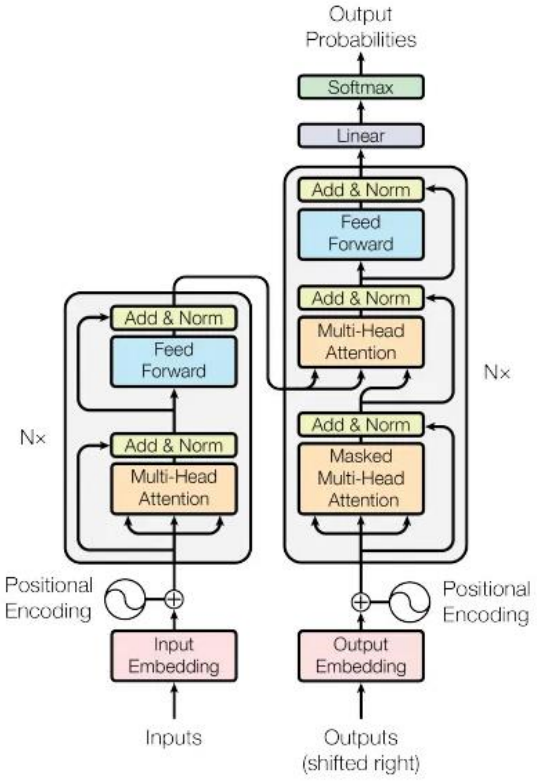

Figura 36. Arquitectura del transformer. Fuente: [17]

transformers basado en el mecanismo de atención, donde no se aplican recurrencias o convoluciones.

La arquitectura del transformer (Figura 36) consta de un codificador y un descodificador. El codificador toma una secuencia de entrada y produce una secuencia de representaciones ocultas. El descodificador toma la salida del codificador y genera la secuencia de salida.

El componente clave de la arquitectura transformer es el mecanismo de autoatención. Este mecanismo permite a la red calcular una suma ponderada de las representaciones ocultas de toda entrada. El mecanismo de autoatención se utiliza tanto en el codificador como en el decodificador. Además de la autoatención, el transformer también utiliza conexiones residuales y capas de normalización para mejorar la estabilidad del entrenamiento y la convergencia. A la salida del decodificador se incluye una capa lineal totalmente conectada y finalmente una capa softmax asigna las probabilidades del vector producido por la capa lineal.

Para la fase de entrenamiento se han utilizado los mismos hiperparámetros que en la guía [28]:

- Nº de épocas: 200 (Se aplica un EarlyStopping cuando la función de pérdida deja de mejorar en 10 épocas).
- Batch size: 64
- Optimizador del descenso del gradiente: Adam
- Learning rate: $1e^{-4}$
- Función de pérdida: Sparse categorical crossentropy
- Métrica de precisión: Sparse categorical accuracy
- División datos de validación: 20 %

4.4 Resultados

En este apartado se recogen los resultados obtenidos del entrenamiento y validación para los modelos de redes neuronales convolucionales y los transformers. Se ha realizado también una comparativa del coste computacional de ambos modelos y finalmente se ha analizado cuál es el modelo campeón.

4.4.1 Resultado de la red neuronal convolucional

Para evaluar el rendimiento del algoritmo se ha utilizado la métrica *sparse categorical accuracy* (Figura 37). El entrenamiento ha terminado en 294 épocas debido al early stopping y se ha conseguido una precisión de un 99.30 % para los datos de validación.

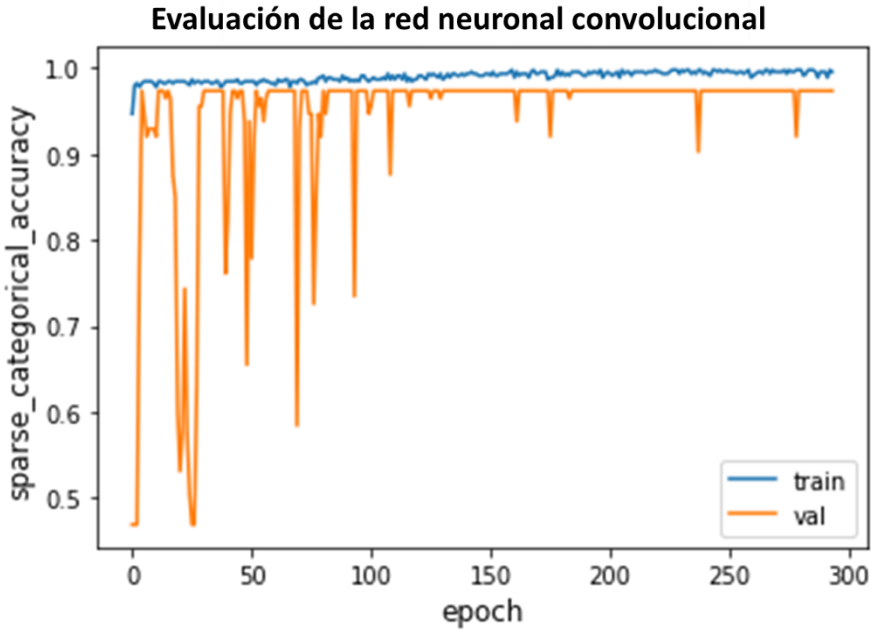

Figura 37. Resultados de la evaluación de la red neuronal convolucional

En la Figura 37 se puede observar que hasta la época 115 los resultados de entrenamiento son inestables, pero a partir de esa época se estabilizan. Se puede ver que no existe sobreaprendizaje y el algoritmo es capaz de distinguir entre las señales de radiación y ruido.

La matriz de confusión de los datos de test es la siguiente:

		Clase predicha	
Clase real		55	0
		2	85

Tabla 3. Matriz de confusión para el CNN

La presión conseguida es de un 98.59 %.

4.4.2 Resultado de los transformers

Para evaluar el rendimiento del algoritmo se ha utilizado la métrica *sparse categorical accuracy* (Figura 38). El entrenamiento ha terminado en 181 épocas debido al early stopping y se ha conseguido una precisión de un 98.67 % para los datos de validación.

En la Figura 38 se puede observar que a partir de la época 15 se alcanza el rendimiento máximo para los datos de validación y se mantiene estable durante todas las épocas sucesivas. Los resultados de la validación son estables a lo largo de todo el entrenamiento y se puede ver que no existe sobreaprendizaje. Se puede concluir que el algoritmo es capaz de distinguir entre las señales de radiación y ruido.

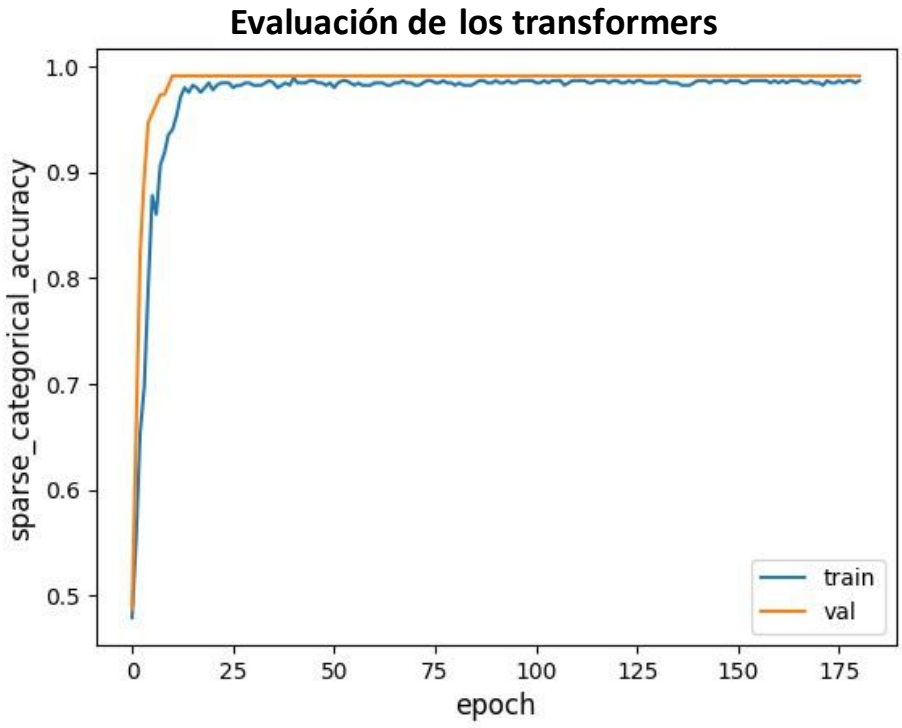

Figura 38. Resultados de la evaluación de los transformers

La matriz de confusión de los datos de test es la siguiente:

		Clase predicha
Clase real	54	1
	2	85

Tabla 4. Matriz de confusión para el transformer

La presión conseguida es de un 97.89 %.

4.4.3 Coste computacional de los modelos

Para evaluar el coste computacional de los modelos basados en redes neuronales convolucionales y transformers se ha medido el tiempo para realizar la inferencia sobre las 142 series temporales usadas para test y se ha dividido el tiempo entre las 142 muestras para calcular el tiempo de inferencia por serie temporal. Para tener en cuenta la variación debido a eventos estocásticos, se ha realizado la inferencia de las 142 muestras 10 veces.

El hardware utilizado para los ensayos ha sido los que proporciona la instancia gratuita de Google Colab:

- CPU: 1 núcleo de Intel(R) Xeon(R) CPU @ 2.30GHz.
- GPU: Nvidia Tesla T4.

Se han ejecutado la inferencia utilizando la GPU, CPU y se han comparado los tiempos entre ambas.

4.4.3.1 Ejecución en GPU

En la Figura 39 se puede ver el tiempo de ejecución de una serie temporal para ambos algoritmos para las 10 iteraciones.

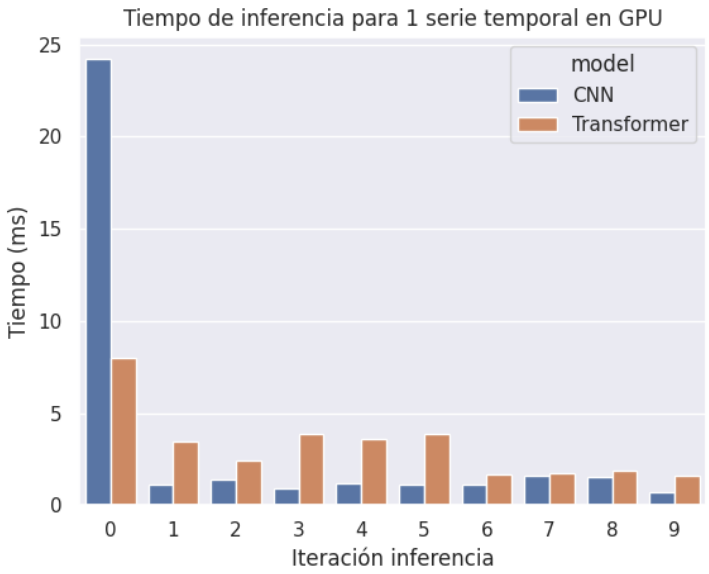

Figura 39. Tiempo de inferencia para una serie temporal en GPU

Se puede observar que para la primera inferencia (iteración 0) el tiempo de ejecución es hasta 20 veces superior que para las siguientes iteraciones en el caso de la CNN. Esto es debido a que cuando la GPU no se está utilizando, este entra en un estado de bajo consumo y apaga ciertos componentes y subsistemas. Cuando un programa requiere del uso de la GPU, este tendrá que cargar el driver o inicializarla, haciendo que tarde mucho más durante la primera ejecución [29]. Debido a este efecto, si ignoramos la primera iteración (Figura 40) se puede ver que no existe tanta variación.

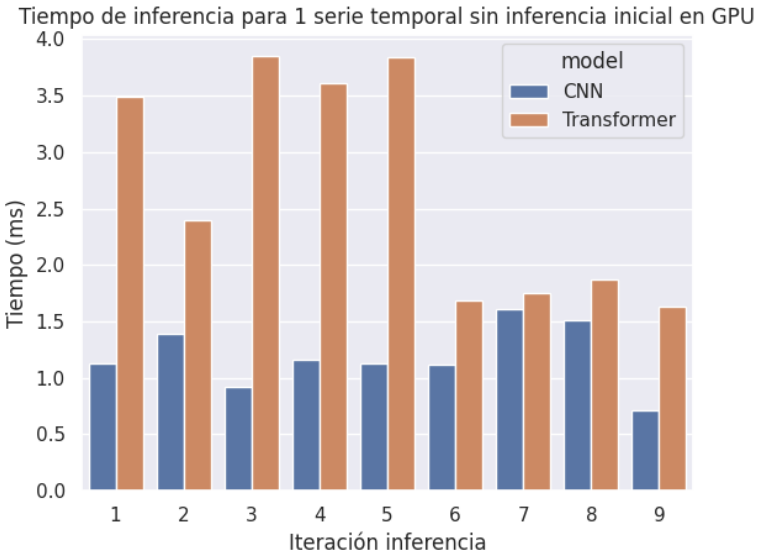

Figura 40. Tiempo de inferencia para una serie temporal sin primera inferencia en GPU

Como se puede observar en el gráfico de barras y el de bigotes (Figura 41), el modelo CNN tiende a ejecutarse en la mitad de tiempo y tiene una menor variación respecto al transformer.

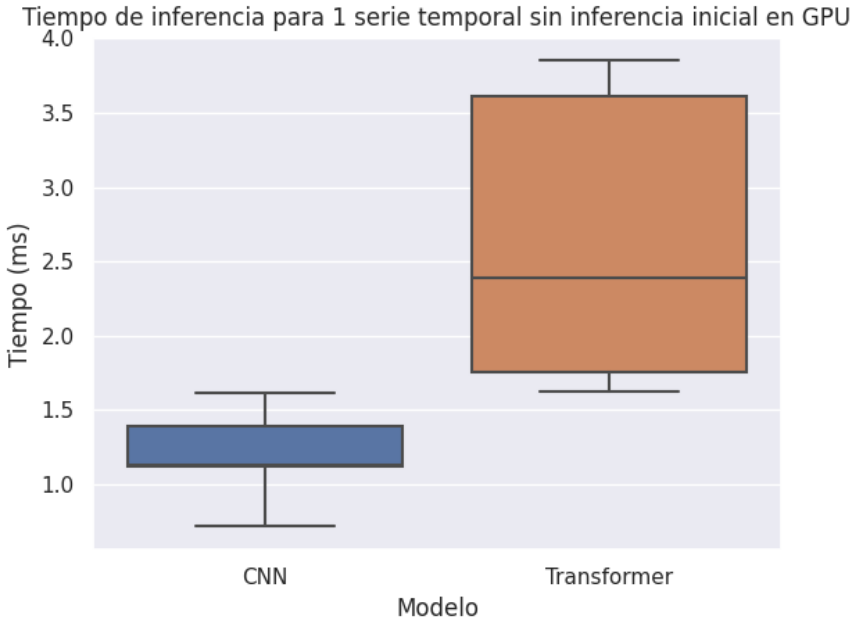

Figura 41. Distribución del tiempo de inferencia para una serie temporal sin primera inferencia en GPU

4.4.3.2 Ejecución en CPU

Ejecutando ambos algoritmos usando la CPU, se obtienen los siguientes tiempos de ejecución por serie temporal (Figura 42 y Figura 43):

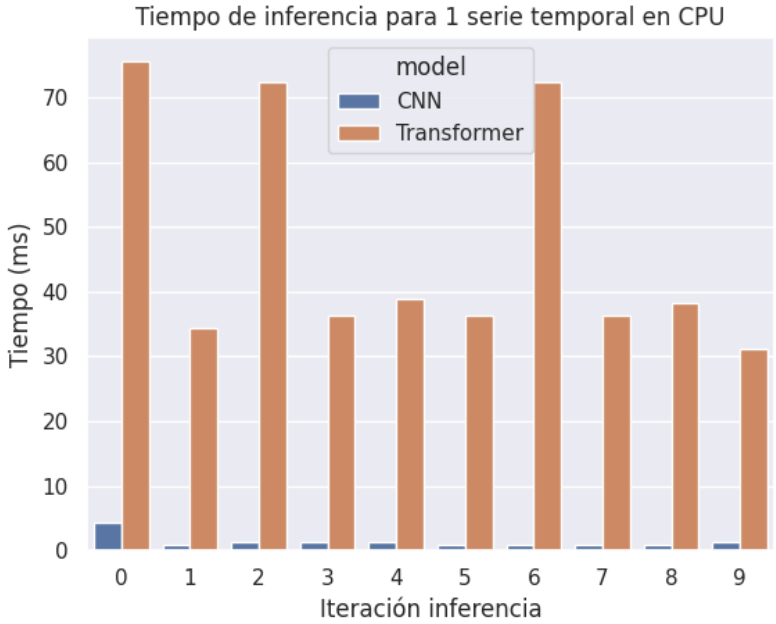

Figura 42. Tiempo de inferencia para una serie temporal en CPU

En el caso de la CPU no existe el fenómeno que ocurre en la GPU durante la inicialización y hay menos variación entre la primera y siguientes ejecuciones. Se puede

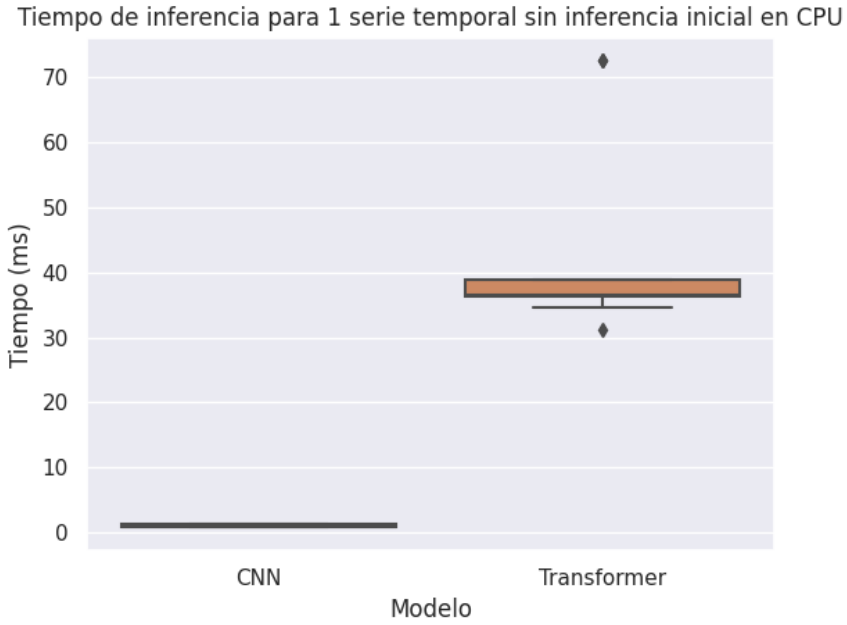

Figura 43. Distribución del tiempo de inferencia para una serie temporal en CPU

ver que el tiempo de ejecución de los transformers es entorno a 30 veces superior respecto a las CNN.

4.4.3.3 Comparativa de ejecución entre GPU y CPU

Comparando el tiempo de ejecución medio de ambos modelos entre la GPU y la CPU se han obtenido los siguientes resultados (Figura 44):

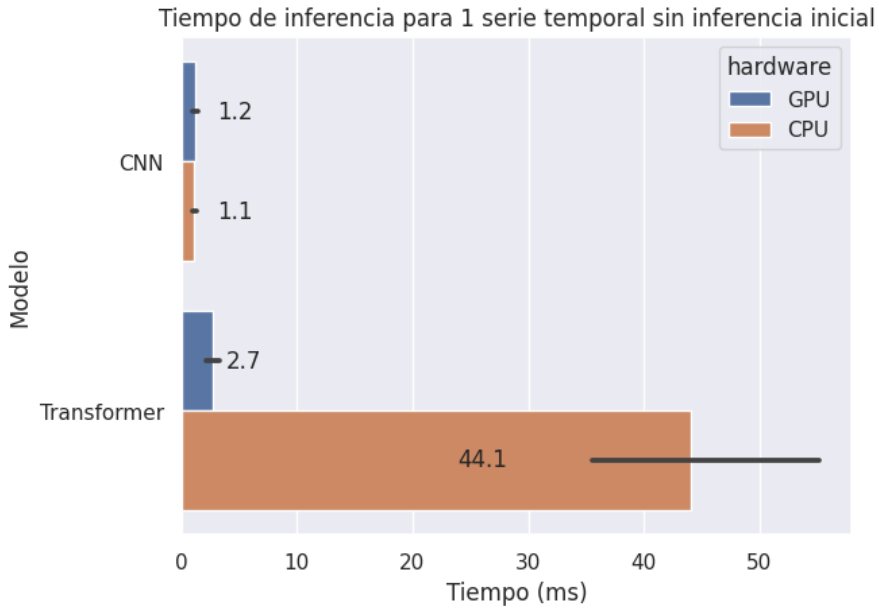

Figura 44. Tiempo de inferencia medio para una serie temporal. Comparativa entre GPU y CPU

Se puede ver que los modelos basados en transformers tienen un rendimiento medio 16 veces peor al ser ejecutados en la CPU y no disponer del paralelismo que ofrece una GPU. El tiempo de ejecución de los transformers es 2 veces superior que en los CNN usando la GPU y sorprendentemente para el caso de las CNN, el tiempo de inferencia es similar entre la GPU y CPU.

4.4.4 Modelo campeón

Comparando los resultados de la evaluación entre el modelo de redes convolucionales y los transformers (Figura 45), se puede concluir que los transformers son el modelo campeón. Aunque para ambos casos se ha conseguido una precisión del 99 %, el entrenamiento de los transformers ha sido estable en todas las épocas y solamente ha necesitado 15 épocas para alcanzar ese rendimiento. Como desventaja de los transformers como se ha visto en el apartado 4.4.3.3, este tiene un tiempo de inferencia 2 veces superior frente a los CNN usando una GPU.

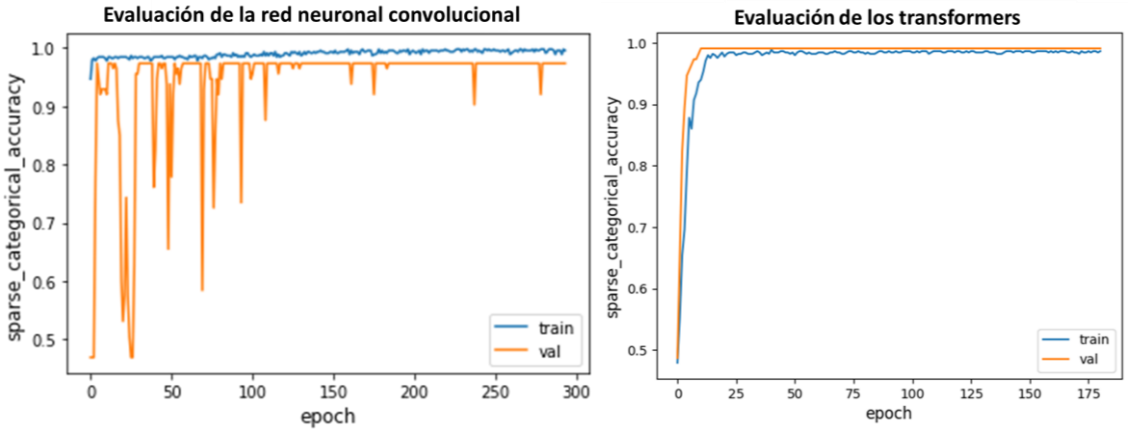

Figura 45. Comparativa entre la evaluación de las CNN (izquierda) y transformers (derecha)

5 Conclusiones y trabajos futuros

En este proyecto se ha conseguido desarrollar con éxito un clasificador binario que sea capaz de distinguir entre las series temporales de radiación y ruido generados por un dosímetro. Para poder entrenar los algoritmos, se ha conseguido generar un set de datos midiendo las series temporales producidas por el dosímetro en un laboratorio mediante un montaje experimental, distinguiendo las series entre las señales temporales producidas por cesio 137 y las de ruido eléctrico producidas por el zumbador que lleva incorporado.

La preparación de los datos realizada ha servido para poder descartar las series temporales duplicadas y mal clasificadas debidas al error humano durante la fase de mediciones. Así mismo, gracias a disminuir el muestreo de 4000 a 400 medidas aplicando el valor máximo en los intervalos, conservar solamente las series temporales en el intervalo de tiempo ±0.2 s y normalizar las series temporales, se ha conseguido disminuir el tiempo de procesamiento significativamente sin tener un impacto negativo en el rendimiento del clasificador binario.

Se ha conseguido entrenar un modelo basado en redes neuronales convolucionales y otro en transformers, obteniendo muy buenos resultados en ambos. El modelo basado en CNN ha conseguido una precisión de un 99.30 % para los datos de test y realizando una matriz de confusión se han conseguido 55 verdaderos negativos, 85 verdaderos positivos, 0 falsos positivos y 2 falsos negativos. El modelo basado en transformers ha conseguido una precisión de un 98.67 % para los datos de test y realizando una matriz de confusión se han conseguido 54 verdaderos negativos, 85 verdaderos positivos, 1 falso positivo y 2 falsos negativos.

Se ha evaluado también el coste computacional de ambos modelos para realizar inferencias usando la GPU y la CPU. En los resultados se ha visto que usando una GPU, el modelo basado en CNN tiende a procesar la inferencia en la mitad de tiempo que el modelo basado en transformers. En cuanto a los resultados de las inferencias usando la CPU, el tiempo de inferencia de los transformers es 30 veces superior a las CNN, mostrando la necesidad que tienen los transformers del paralelismo de la GPU.

Finalmente, comparando las métricas de resultados de ambos modelos, se ha elegido a los transformers como modelo campeón debido a que el entrenamiento de las CNN no ha sido estable y ha tenido mucha variación durante las épocas mientras que los transformers han alcanzado su rendimiento máximo en la época 15 y se ha mantenido estable durante todo el entrenamiento. En los resultados se ha podido ver que ambos modelos ofrecen buenos resultados para esta aplicación usando como base las arquitecturas propuestas y no requieren de un procesamiento complejo de los datos originales.

Los datos y el código utilizados para procesar y entrenar los algoritmos se encuentran disponibles en el siguiente repositorio de GitHub: https://github.com/ilarman/Radiation-time-series-binary-classifier-using-CNNs-and-transformers

Como trabajos futuros, se podrían tomar datos de fuentes de ruido adicionales como los debidos a golpes o fuentes de calor y entrenar los algoritmos con estos datos para que el modelo sea más robusto a la hora de clasificar ruido en un entorno real. Adicionalmente, se podrían expandir las posibilidades del clasificador binario para que además de distinguir entre el ruido y la radiación, consiga cuantificar el nivel de radiación al que está expuesta la persona.

Finalmente, otro aspecto en el que se podría trabajar en el futuro es el de conseguir implementar este algoritmo en un entorno de producción para que pueda ser usado en tiempo real. El hardware de procesamiento que lleva actualmente el dosímetro ensayado es muy limitado y no es capaz de procesar los algoritmos propuestos, por lo que habría que desarrollar al dosímetro la capacidad de poder conectarse a internet para que pueda enviar los datos a un servidor, este ejecute el algoritmo de clasificación y le responda con los resultados al dosímetro.

6 Referencias

[1] «Non-Ionizing Radiation - Overview | Occupational Safety and Health Administration». https://www.osha.gov/non-ionizing-radiation (accedido 1 de mayo de 2023).

[2] «Radiation Studies: Ionizing Radiation | CDC». https://www.cdc.gov/nceh/radiation/ionizing_radiation.html (accedido 1 de mayo de 2023).

[3] «Curso de SUPERVISORES de instalaciones radiactivas (IR) MÓDULO BÁSICO © CSN-2013 TEMA 2: INTERACCIÓN DE LA RADIACIÓN CON LA MATERIA», Consejo de Seguridad Nuclear, 2013. Accedido: 8 de abril de 2023. [En línea]. Disponible en: https://csn.ciemat.es/MDCSN/recursos/ficheros_md/764096047_157200911241 1.pdf

[4] «Compton effect | Definition, Formula, & Facts | Britannica». https://www.britannica.com/science/Compton-effect (accedido 1 de mayo de 2023).

[5] P. E. R. F.R.S., «LXXIX. The scattering of α and β particles by matter and the structure of the atom», *https://doi.org/10.1080/14786440508637080*, vol. 21, n.º 125, pp. 669-688, may 2009, doi: 10.1080/14786440508637080.

[6] «Personal Radiation Dosimeter - Radiation Safety - Safety, Health, Environment, and Risk Management - UTHealth Houston». https://www.uth.edu/safety/radiation-safety/personal-radiation-dosimeter.htm (accedido 1 de mayo de 2023).

[7] «What Is a Dosimeter?» https://www.flyability.com/dosimeter (accedido 9 de abril de 2023).

[8] J. Barthe, «Electronic dosimeters based on solid state detectors», *Nucl Instrum Methods Phys Res B*, vol. 184, n.º 1-2, pp. 158-189, 2001, doi: 10.1016/S0168-583X(01)00711-X.

[9] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, y P. A. Muller, «Deep learning for time series classification: a review», *Data Min Knowl Discov*, vol. 33, n.º 4, pp. 917-963, jul. 2019, doi: 10.1007/S10618-019-00619-1/FIGURES/16.

[10] «Explained: Neural networks | MIT News | Massachusetts Institute of Technology». https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414 (accedido 16 de abril de 2023).

[11] J. Zou, Y. Han, y S. S. So, «Overview of artificial neural networks», *Methods in Molecular Biology*, vol. 458, pp. 15-23, 2008, doi: 10.1007/978-1-60327-101-1_2/COVER.

[12] «Loss Functions and Their Use In Neural Networks | by Vishal Yathish | Towards Data Science». https://towardsdatascience.com/loss-functions-and-their-use-in-neural-networks-a470e703f1e9 (accedido 16 de abril de 2023).

[13] I. Goodfellow, Y. Bengio, y A. Courville, *Deep Learning*. MIT Press, 2016. [En línea]. Disponible en: https://www.deeplearningbook.org/contents/

[14] H. Robbins y S. Monro, «A Stochastic Approximation Method», *https://doi.org/10.1214/aoms/1177729586*, vol. 22, n.º 3, pp. 400-407, sep. 1951, doi: 10.1214/AOMS/1177729586.

[15] «Deep Learning in a Nutshell: History and Training | NVIDIA Technical Blog». https://developer.nvidia.com/blog/deep-learning-nutshell-history-training/ (accedido 16 de abril de 2023).

[16] «Gradient Descent - Gradient descent - Product Manager's Artificial Intelligence Learning Library». https://easyai.tech/en/ai-definition/gradient-descent/ (accedido 16 de abril de 2023).

[17] A. Vaswani *et al.*, «Attention Is All You Need», *Adv Neural Inf Process Syst*, vol. 2017-Decem, pp. 5999-6009, jun. 2017, Accedido: 17 de abril de 2023. [En línea]. Disponible en: https://arxiv.org/abs/1706.03762v5

[18] T. Wolf *et al.*, «Transformers: State-of-the-Art Natural Language Processing», pp. 38-45, nov. 2020, doi: 10.18653/V1/2020.EMNLP-DEMOS.6.

[19] Q. Wen *et al.*, «Transformers in Time Series: A Survey», feb. 2022, Accedido: 17 de abril de 2023. [En línea]. Disponible en: https://arxiv.org/abs/2202.07125v4

[20] «Multivariate Time Series Forecasting with Transformers | by Jake Grigsby | Towards Data Science». https://towardsdatascience.com/multivariate-time-series-forecasting-with-transformers-384dc6ce989b (accedido 2 de mayo de 2023).

[21] «About Keras». https://keras.io/about/ (accedido 17 de abril de 2023).

[22] «Why choose Keras?» https://keras.io/why_keras/ (accedido 17 de abril de 2023).

[23] S. Studer *et al.*, «Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology», Accedido: 2 de abril de 2023. [En línea]. Disponible en: https://arxiv.org/abs/2003.05155

[24] G. Mariscal, Ó. Marbán, y C. Fernández, «A survey of data mining and knowledge discovery process models and methodologies», *Knowl Eng Rev*, vol. 25, n.º 2, pp. 137-166, jun. 2010, doi: 10.1017/S0269888910000032.

[25] D. Delacroix, J. P. Guerre, P. Leblanc, y C. Hickman, «Radionuclide and radiation protection data handbook 2nd edition (2002)», *Radiat Prot Dosimetry*, vol. 98, n.º 1, pp. 1-168, ene. 2002, doi: 10.1093/OXFORDJOURNALS.RPD.A006705.

[26] «Timeseries classification from scratch». https://keras.io/examples/timeseries/timeseries_classification_from_scratch/ (accedido 23 de abril de 2023).

[27] Z. Wang, W. Yan, y T. Oates, «Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline», *Proceedings of the International Joint Conference on Neural Networks*, vol. 2017-May, pp. 1578-1585, nov. 2016, doi: 10.1109/IJCNN.2017.7966039.

[28] «Timeseries classification with a Transformer model». https://keras.io/examples/timeseries/timeseries_classification_transformer/ (accedido 23 de abril de 2023).

[29] «The Correct Way to Measure Inference Time of Deep Neural Networks - Deci». https://deci.ai/blog/measure-inference-time-deep-neural-networks/ (accedido 30 de abril de 2023).

[30] «Dosimeter and Radiation Detection | Med-Pro». https://med-pro.net/dosimeter-radiation-detection-problems-solved/ (accedido 9 de abril de 2023).

[31] «Radiation - Wikipedia». https://en.wikipedia.org/wiki/Radiation (accedido 8 de abril de 2023).

[32] «ROLscience: ¿Qué es el experimento de Rutherford?» https://www.rolscience.net/2020/11/que-es-el-experimento-de-rutherford.html (accedido 9 de abril de 2023).

[33] «Dosimeter - Wikipedia». https://en.wikipedia.org/wiki/Dosimeter (accedido 7 de abril de 2023).

[34] «Time Series Classification with Deep Learning | by Marco Del Pra | Towards Data Science». https://towardsdatascience.com/time-series-classification-with-deep-learning-d238f0147d6f (accedido 16 de abril de 2023).

[35] «Red neuronal artificial - Wikipedia, la enciclopedia libre». https://es.wikipedia.org/wiki/Red_neuronal_artificial (accedido 16 de abril de 2023).

[36] «Redes Neuronales artificiales: Qué son y cómo se entrenan». https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i (accedido 16 de abril de 2023).

[37] «learning-rate in keras». https://livebook.manning.com/concept/keras/learning-rate (accedido 16 de abril de 2023).

[38] «Cross-industry standard process for data mining - Wikipedia». https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining (accedido 10 de abril de 2023).