uc3m | Universidad Carlos III de Madrid

Grado Universitario en Ingeniería Informática
2022-2023

Trabajo Fin de Grado

"Predicción de la contaminación del aire en la ciudad de Madrid mediante redes LSTM"

Victoria Bueno Pérez

Tutor

José María Valls Ferrán

Leganés, septiembre de 2023

RESUMEN

La contaminación atmosférica representa uno de los problemas ambientales más importantes actualmente en la sociedad y su impacto en la salud pública y en el medio ambiente puede llegar a ser muy grave. Gracias a las redes neuronales artificiales y al Deep Learning (DL), podemos predecir de manera más o menos precisa el futuro. Es por ello por lo que redes como las Long Short-Term Memory (LSTM) se usan para realizar predicciones sobre series temporales.

La presente memoria recoge el estudio realizado sobre la predicción de la contaminación del aire, en específico del dióxido de nitrógeno (NO_2), en la ciudad de Madrid mediante el uso de redes LSTM. El estudio comprende la predicción de los 5 días siguientes de la calidad del aire. Además, puede ser utilizado por los usuarios gracias a la implementación de una aplicación web con acceso a los modelos de predicción creados.

Para realizar este estudio se han llevado a cabo varias fases de experimentación. La primera fase consiste en investigar el tema de estudio y estudiar proyectos similares. La segunda fase trata el estado del arte que rodea a este. La tercera fase se centra en buscar y obtener los datos necesarios, hacerles un preprocesado y limpieza completa para poder usarlos y, por último, analizar la serie temporal resultante. La siguiente fase, la cuarta, comprende una exhausta experimentación con modelos de redes LSTM, una evaluación de los resultados obtenidos y la generación de los modelos finales de predicción.

Además de la experimentación realizada, se ha creado una aplicación web para que los usuarios que lo deseen puedan hacer uso de los modelos de predicción generados. Dicha aplicación se ha creado mediante el framework Streamlit. En ella el usuario puede elegir una fecha y una estación de las tres posibles, Paseo de la Castellana, Plaza del Carmen y Retiro. Con estos datos la aplicación mostrará en pantalla un gráfico que contiene la predicción a 5 días vista y también los valores reales de los 14 días previos. La aplicación web puede utilizarse accediendo aquí.

Esta memoria también recoge el análisis del sistema, así como el diseño y la implementación, además de la planificación, el entorno socioeconómico, las conclusiones y los posibles trabajos futuros.

Palabras clave: Deep Learning, red LSTM, serie temporal, calidad del aire, NO_2, aplicación web.

ABSTRACT

Air pollution represents one of the most significant environmental issues currently in society, and its impact on public health and the environment can be extremely severe. Thanks to artificial neural networks and Deep Learning (DL), we can try to predict the future with some precision. This is why networks like Long Short-Term Memory (LSTM) are used to make predictions about time series data.

This document presents the study conducted on predicting air pollution, specifically focusing on nitrogen dioxide (NO_2), in the city of Madrid using LSTM neural networks. The study involves predicting the air quality for the next 5 days. Additionally, it can be used by users through a web application that provides access to the created prediction models.

Several phases of experimentation were carried out to conduct this study. The first phase involved researching the subject matter and studying similar projects. The second phase addressed the current state of the art in this field. The third phase focused on acquiring and preprocessing the necessary data, performing thorough cleaning to make it usable, and finally analyzing the resulting time series. The subsequent phase, the fourth one, encompassed extensive experimentation with LSTM network models, an evaluation of the achieved results, and the creation of the final prediction models.

In addition to the experimentation, a web application has been developed for users who wish to utilize the generated prediction models. This application has been created using the Streamlit framework. In it, the user can select a date and one of the three possible locations, Paseo de la Castellana, Plaza del Carmen, and Retiro. With this input information, the application will display a chart on the screen containing the 5-day forecast as well as the actual values from the previous 14 days. The web application can be used by accessing it here.

This document also covers system analysis, design, and implementation, as well as planning, the socio-economic context, conclusions, and potential future work.

Keywords: Deep Learning, LSTM network, time series, air quality, NO_2, web application.

DEDICATORIA

Con la realización de este trabajo finalizo la etapa más importante de mi vida académica. Estos cuatro años de universidad no hubieran sido lo mismo sin la gente que me ha acompañado durante ellos. Es por ello por lo que quiero dar las gracias a la gente maravillosa con la que me he cruzado durante mi recorrido en la Carlos III, que hoy en día tengo la suerte de llamar amigos, y que gracias a ellos estos cuatro años han sido mucho más amenos y divertidos.

También me gustaría agradecer el esfuerzo de mis padres por darme la oportunidad de irme a estudiar fuera durante mi tercer año de carrera, ya que fue un año de mucho crecimiento personal y el más increíble que he vivido hasta la fecha. Y por supuesto, darles las gracias por aguantarme durante estos años, que no han sido nada fáciles.

Para finalizar, agradecer a mi tutor, José María Valls, por el gran apoyo que me ha brindado y la gran confianza que tuvo en el proyecto desde el día que le comenté la idea que quería desarrollar.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1. Motivación

La contaminación atmosférica representa uno de los problemas ambientales más importantes actualmente en la sociedad y su impacto en la salud pública y el medio ambiente está más que comprobado [1].

De acuerdo con los datos de la Organización Mundial de la Salud (OMS) [2], un 99% de la población mundial respira aire que supera los límites recomendados por la misma Organización. Los contaminantes del aire como las partículas en suspensión pueden causar enfermedades respiratorias y otros trastornos, siendo una de las principales causas de enfermedad y muerte a nivel mundial. Además, la calidad del aire está muy relacionada con el clima global y los ecosistemas. Casi todas las fuentes de contaminación atmosférica también emiten gases de efecto invernadero, lo que contribuye al cambio climático. Por lo tanto, las políticas enfocadas en reducir la contaminación del aire tienen beneficios tanto para la salud como para la mitigación del cambio climático en el corto y largo plazo.

Una satisfactoria predicción de los valores de contaminación del aire podría ayudar a anticipar picos, y en base a ellos tomar decisiones sobre las regulaciones de tráfico, las actividades industriales y otras fuentes de emisión de contaminantes atmosféricos. Además, también podría ser útil para cualquier persona que desee comprobar cómo va a ser la calidad del aire en los próximos cinco días, ya sea por necesidad como podría ser por tener alguna enfermedad que se pudiese agravar con la inhalación de aire contaminado, o por tomar decisiones personales como salir a correr al aire libre o elegir si coger el coche o el transporte público en función a los resultados de realizar la predicción.

A nivel tecnológico este trabajo ofrece la posibilidad de poder realizar un crecimiento y aprendizaje personal sobre nuevos modelos de predicción, Deep Learning, ciencia de datos y la implementación correspondiente para su uso.

Es por todos estos motivos por lo que se ha decidido realizar el proyecto que se explica en la presente memoria.

1.2. Descripción del sistema y objetivos

Para este proyecto se plantea crear modelos de redes LSTM que sean capaces de predecir los niveles de NO_2 en ciertas estaciones de la ciudad de Madrid a cinco días vista y que el usuario pueda hacer uso de dichos modelos mediante una aplicación web para obtener una predicción en tiempo real.

Los objetivos que se pretenden alcanzar al finalizar este proyecto son:

- Aprendizaje sobre la contaminación del aire en la ciudad de Madrid.

- Obtención y limpieza de todos los datos necesarios.
- Análisis de las series temporales.
- Aprendizaje sobre las redes Long Short-Term Memory y modelos de predicción usados en series temporales.
- Análisis de modelos de predicción.
- Aprendizaje de frameworks para desarrollo de aplicaciones web.
- Diseño e implementación de una aplicación web interactiva para el usuario.

1.3. Marco regulador

Para la realización de este proyecto se ha hecho uso de un conjunto de datos proporcionados por el Ayuntamiento de Madrid en su portal de datos abiertos. La reutilización de estos datos está regulada por varias normas que tienen como objetivo el impulso de los mismos. Son las siguientes:

- "*Con carácter general, será reutilizable la información publicada sin necesidad de autorización previa y de forma gratuita*"(Artículo 27. Ordenanza de Transparencia de la ciudad de Madrid) [3].
- "*Mediante la reutilización de esta información generada por el sector público y la puesta a disposición de este gran volumen de información, se generan grandes beneficios económicos y sociales, así como de transparencia de las administraciones*." (Real Decreto-ley 24/2021, de 2 de noviembre, de transposición de la Directiva (UE) 2019/1024 de datos abiertos y reutilización de la información del sector público)

Con respecto a la contaminación en la ciudad de Madrid, la Junta de Gobierno de la Ciudad de Madrid aprobó en 2018 el Protocolo de Actuación para Episodios de Contaminación por Dióxido de Nitrógeno en la Ciudad de Madrid [4] que se activa en función de los niveles de contaminantes presentes en el aire, como el dióxido de nitrógeno (NO_2) y las partículas en suspensión (PM_{10} y $PM_{2.5}$). Este protocolo se divide en cuatro escenarios (escenario 1, 2, 3 y 4), y cada uno de ellos implica diferentes medidas de restricción del tráfico y de actividades para reducir la contaminación. Se establecen limitaciones a la velocidad de circulación en la M-30 y accesos a la ciudad, restricciones a la circulación de vehículos según su etiqueta ambiental y limitaciones en actividades industriales, entre otras medidas.

Se han establecido Áreas de Prioridad Residencial (APR) [5] en ciertas zonas de la ciudad para restringir el acceso al tráfico rodado en favor de los residentes, del transporte público, las bicicletas y los vehículos menos contaminantes. En estas áreas, el acceso a ciertos vehículos está regulado y, en algunos casos, se implementaron medidas de peatonalización.

La Dirección General de Tráfico creó etiquetas ambientales para clasificar los vehículos en función de su potencial contaminante en 2016 [6]. El Ayuntamiento de Madrid adoptó este sistema de etiquetado ambiental. Los vehículos son clasificados en función de sus emisiones y

eficiencia energética, recibiendo etiquetas de Cero, Eco, C, B o no etiqueta. Durante los episodios de alta contaminación, los vehículos con etiquetas más contaminantes podían ser restringidos en su circulación.

Por último, la ciudad cuenta con un Plan de calidad del aire y cambio climático denominado Plan A aprobado por la Junta de Gobierno en septiembre de 2017 y vigente en la fecha de realización de esta memoria [7]. Fue un conjunto de medidas implementadas por el Ayuntamiento de Madrid enfocado en reducir las emisiones de gases contaminantes y promover un entorno más sostenible y saludable para los ciudadanos. Entre las medidas se encuentran limitaciones de tráfico y circulación en la zona central de Madrid, mejora de la red de transporte público, promoción de la movilidad sostenible en bicicleta, fomento de la eficiencia energética en los edificios, y oferta de beneficios fiscales y ventajas para vehículos eléctricos.

1.4. Estructura de la memoria

Esta memoria queda estructurada de la siguiente manera, haciendo también referencia a qué contiene cada apartado:

- **Introducción**: incluye la motivación, la descripción del sistema, los objetivos que se desean alcanzar y el marco regulador.
- **Estado del arte**: se explican todos aquellos conceptos que se usan a lo largo del proyecto como lo son la contaminación atmosférica, las series temporales, las redes neuronales, en específico el perceptrón multicapa y las redes recurrentes, también las redes LSTM y otros conceptos teóricos relevantes en la predicción de series temporales. Además, se describen brevemente algunos trabajos similares que tratan sobre predicción de contaminación del aire y sobre redes LSTM. Por último, cuenta con una breve explicación de las herramientas utilizadas y un apartado de justificación de la solución.
- **Estudio experimental**: se explica la metodología a seguir durante la experimentación, además, incluye la obtención de los datos y su preprocesado, el análisis en profundidad de la serie temporal, los modelos y la experimentación con ellos y por último, una evaluación de los resultados.
- **Análisis del sistema**: incluye la especificación de los requisitos tanto funcionales como no funcionales y también los casos de uso del sistema, el diagrama y la descripción de estos, así como la matriz de trazabilidad entre ambos.

- **Diseño e implementación del sistema**: incluye una explicación de cómo se usan los modelos finales, el desarrollo de la aplicación web junto a su diseño, el despliegue y la URL donde se encuentra disponible para su uso.
- **Planificación y entorno socioeconómico**: desarrolla la planificación seguida en el proyecto y el diagrama de Gantt, además de una explicación detallada del presupuesto y los costes totales, y por último el entorno socioeconómico del proyecto.
- **Conclusiones y trabajos futuros**: incluye las conclusiones del proyecto, además de las personales, y los posibles trabajos futuros a desarrollar.

2. ESTADO DEL ARTE

2.1. Contaminación Atmosférica

La atmósfera es una capa gaseosa que rodea la Tierra. Esta capa se encarga de proporcionar los elementos necesarios para los seres vivos como el carbono, el oxígeno y el nitrógeno, sirve de protección frente a la radiación ultravioleta del sol y participa en la regulación del clima ya que se encarga del movimiento de las masas continentales y de aire caliente y frío de los océanos [8].

A su vez, la contaminación atmosférica es la presencia de sustancias perjudiciales para la salud y el medio ambiente en la atmósfera que provienen de la reacción de compuestos químicos. Los dos principales fenómenos a escala mundial que surgen de la contaminación atmosférica son el efecto invernadero, que se define como el aumento de la temperatura terrestre debido a las sustancias presentes en el aire, y la destrucción de la capa de ozono. Los contaminantes más importantes son el óxido de azufre ($SO_{x)}$), el óxido de nitrógeno (NO_x), el ozono troposférico (O_3), el monóxido de carbono (CO), el dióxido de carbono (CO_2), el plomo e hidrocarburos no metánicos. Por otro lado, la contaminación atmosférica urbana, que es aquella presente en las ciudades, viene de la emisión de contaminantes primarios como el óxido de azufre, óxido de nitrógeno, óxido de carbono, amoniaco y partículas sólidas, causados por la industria, la calefacción y el tráfico. Todas estas prácticas tienen en común la combustión de combustibles fósiles como el carbón o el petróleo que se lanzan al aire sin control. Estos elementos una vez en la atmósfera reaccionan con oxidantes y el sol y se convierten en contaminantes secundarios, siendo los más comunes el ácido nítrico, ácido sulfúrico y el ozono [9].

En Madrid, los contaminantes del aire más dañinos para la salud de los habitantes están causados por el tráfico rodado y son el dióxido de nitrógeno (NO_2), las partículas en suspensión ($PM_{2,5}$ y PM_{10} haciendo referencia a su diámetro en micras), y el ozono troposférico (O_3). En torno al 80% de NO_2 presente en la atmosfera de Madrid procede del tráfico, en específico de los motores diésel, y el resto surge de la actividad industrial. Las partículas en suspensión que se encuentran en la ciudad de Madrid son mayoritariamente procedentes del tráfico, calefacciones, industria, minería, etc. Las más pequeñas, aquellas con un diámetro de 2,5 micras, son las más nocivas ya que debido a su pequeño tamaño pueden pasar a la sangre desde los pulmones. Las partículas PM_{10} suelen ser de procedencia natural, y no consiguen pasar de los bronquios o fosas nasales por lo tanto son menos perjudiciales para la salud, pero favorecen a la aparición de asma en la población. Con respecto al ozono, existen dos tipos, el ozono estratosférico que se denomina ozono bueno y que se encarga de protegernos de los rayos ultravioleta, y el ozono troposférico o malo que se localiza en la capa responsable de proporcionar los gases necesarios para la vida en la Tierra. Este último, el malo, surge de reacciones químicas de automóviles, centrales térmicas y más procesos industriales, y se forma en proporción a los niveles de luz solar y temperatura presentes. Por lo tanto, en verano surge más concentración de este [10].

Los principales problemas de salud que conlleva la contaminación atmosférica son las enfermedades respiratorias, enfermedades cardiovasculares, trastornos del desarrollo y hasta la

aparición de ciertos tipos de cáncer, presentando todos ellos un efecto a largo plazo. Un estudio realizado por el Instituto de Salud Carlos III con la colaboración de varias universidades madrileñas, la Agencia Estatal de Meteorología (AEMET) y el Consorcio de Investigación Biomédica en Red de Epidemiología y Salud Pública (CIBERESP) [11] sobre la relación entre el aumento de las hospitalizaciones y los niveles de contaminación en Madrid, atribuye un aumento de hospitalizaciones relacionadas con problemas respiratorios a niveles altos de NO_2 en el aire, y un aumento en problemas cardiovasculares con niveles altos de O_3.

Hay varios estudios que relacionan algunos factores meteorológicos con la presencia de contaminantes en la atmósfera [12]. En específico, hay tres factores con influencia directa, la temperatura, la precipitación y el viento. La cantidad o presencia de estos factores determinan de manera más o menos precisa los niveles de contaminación presentes en una zona específica. Es por ello por lo que se van a tener en cuenta a la hora de realizar el estudio de la predicción más adelante en este trabajo.

2.2. Series Temporales

Una serie temporal es una secuencia de observaciones ordenadas de una o más variables a lo largo de una secuencia de tiempo equiespaciado [13]. El objetivo de estudiar una serie temporal es conocer su patrón de comportamiento para intentar hacer predicciones futuras. Si la serie es determinista, es decir, podemos predecir el valor de la serie para un tiempo futuro determinado, el estudio no es de interés, como ocurre por ejemplo con una serie logística determinada por una ecuación. Es por ello por lo que las series temporales constan de fenómenos aleatorios que hacen que la predicción de valores no sea perfecta, sino una aproximación a la estructura probabilística que siguen. Las series temporales también reciben el nombre de procesos estocásticos, que son sucesiones de variables aleatorias que varían con el tiempo, por lo tanto, procesos que no se pueden predecir [13]. Como ejemplo de series temporales se encuentra el valor del IBEX35, la contaminación atmosférica de Madrid, el número de usuarios de un servicio, las ventas de una empresa, etc.

Figura 1: Evolución del IBEX35 [14].

El primer análisis de una serie temporal se realiza gráficamente, mediante el gráfico de secuencia, es decir, haciendo uso de su representación en un diagrama de líneas donde el eje de abscisas (x) representa el tiempo, y la variable de estudio el eje de ordenadas (y). Se pueden seguir dos enfoques diferentes a la hora de estudiar el comportamiento de las series temporales, la modelización por componentes y el enfoque Box-Jenkins.

La modelación por componentes se muestra en la Figura 2 y consiste en identificar los cuatro componentes en los que se puede dividir una serie temporal, aunque no siempre existen todas, son tendencia, estacionalidad, ciclos y ruido. La tendencia representa información a largo plazo y muestra cómo varía la variable a lo largo de él, pudiendo ser creciente o decreciente. La estacionalidad nos muestra las oscilaciones en periodos cortos de tiempo, es decir, se puede observar un patrón que se repite periódicamente ya sea anual, cuatrimestral, semanal, etc. Un claro ejemplo de estacionalidad semanal es el tráfico en las ciudades, ya que se cumple que los fines de semana el patrón de circulación es totalmente distinto al de los días laborables normales, y lo mismo pasa con los periodos de vacaciones como verano o navidad. Por otro lado, las variaciones cíclicas son fluctuaciones periódicas en los datos a lo largo del tiempo que no son causadas por factores estacionales o de tendencia, sino que son el resultado de patrones repetitivos a lo largo del tiempo, como pueden serlo las etapas de prosperidad económica o de depresión. Estas variaciones se producen en un rango de períodos que pueden ser cortos o largos, y pueden tener una amplitud constante o variable. Por último, el ruido o residuo es el factor que introduce azar o valores extraños a la serie temporal y es por tanto es el que se trata de explicar o predecir a la hora de obtener valores futuros [15].

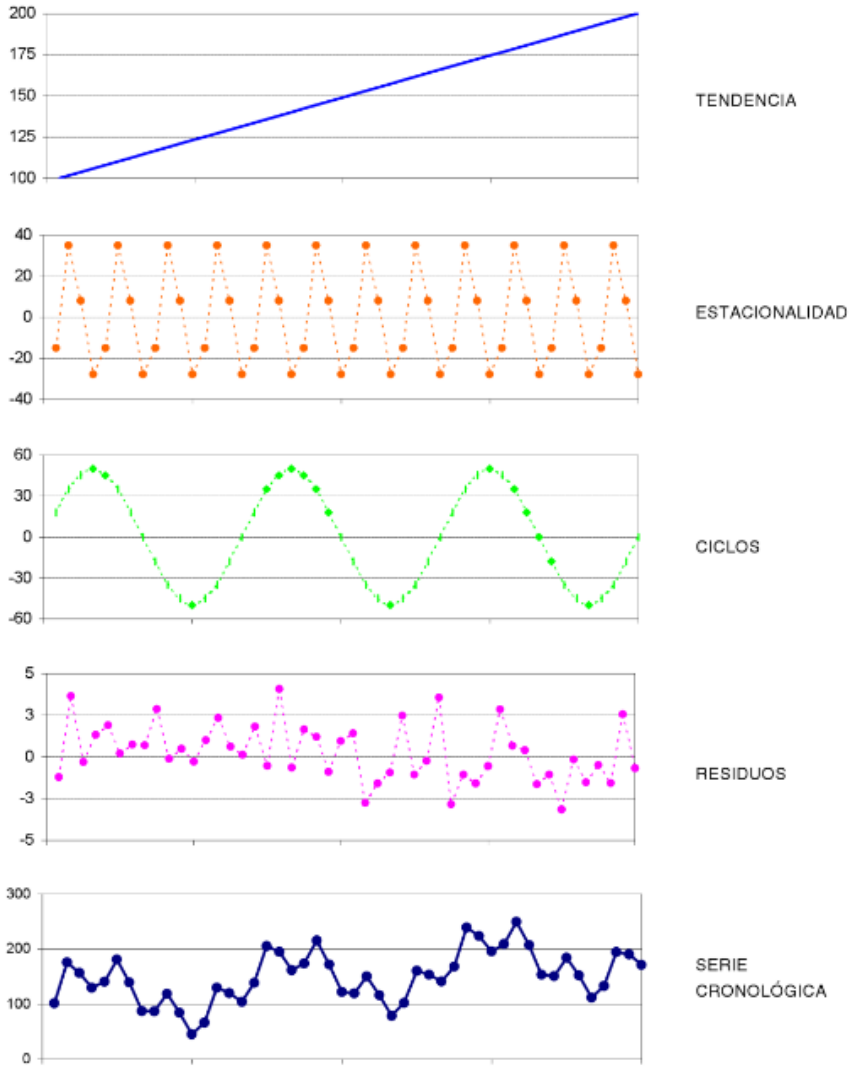


Figura 2: Componentes de una serie temporal [13].

El método clásico de análisis de series temporales dice que una serie temporal X_t queda definida como la función de los cuatro componentes mencionados tal que $X_t = f(C_t, T_t, S_t, E_t)$, donde T_t es la tendencia, C_t son los ciclos, S_t es la estacionalidad y E_t el ruido. Y que la función $f()$ suele unir los componentes de manera aditiva o multiplicativa. Comúnmente la descomposición estacional separa la serie en tendencia-ciclo, estacionalidad y ruido como muestra la Figura 3.

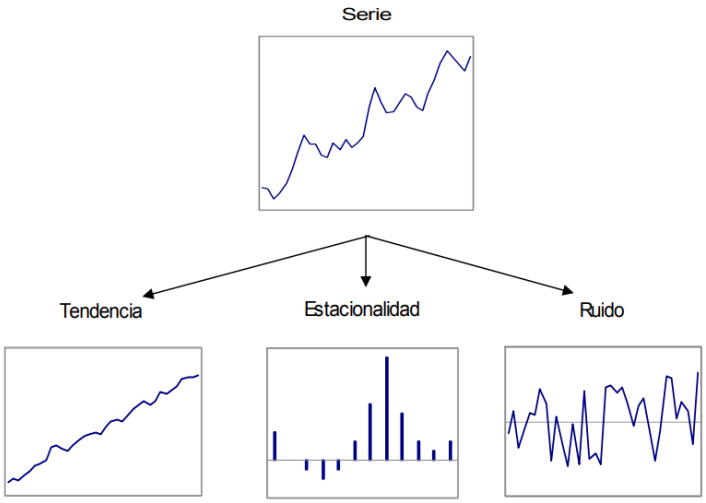


Figura 3: Descomposición estacional [15].

Por otro lado, está el enfoque de Box-Jenkins, que consiste en conseguir encontrar el modelo probabilístico estocástico que define la serie, teniendo más en cuenta los residuos, que son el componente aleatorio puro, que los otros componentes de la serie. Para ello hay que primero identificar el modelo, luego hacer una estimación de los parámetros y por último hacer una validación del modelo. Los conjuntos de modelos existentes son los de ruido blanco, medias móviles, autorregresivos, autorregresivos de medias móviles y los no estacionarios, haciéndose llamar modelos ARIMA [13].

Un proceso de ruido blanco en un proceso estocástico en el que los valores de la serie son aleatorios e independientes entre sí, sin tener ninguna correlación o patrón discernible. Cada valor de la serie se genera de manera independiente y con una distribución de probabilidad idéntica para cada observación. Por lo tanto, la media y la varianza son constantes en el tiempo. Estos procesos se denominan con el término blanco porque no tienen una ninguna frecuencia dominante.

Los procesos de medias móviles (MA) estiman el valor de la serie como una función lineal de los errores aleatorios o residuos pasados, siendo los residuos la diferencia entre los valores observados y las predicciones. Estos modelos se representan por MA (q) siendo q el orden de residuos pasados que se incluyen en el modelo.

En los procesos autorregresivos (AR) el valor actual de la serie depende de valores pasados de y un término aleatorio. Se representa por AR(p) donde p representa el número de valores pasados que se incluyen en el modelo. Estos procesos usan parámetros de autorregresión que indican la importancia relativa de los valores pasados en la predicción del valor actual de la serie.

Si se unen los procesos de medias móviles con los autorregresivos, se obtienen los modelos ARMA. En estos procesos, el valor actual de la serie temporal depende tanto de los valores pasados de la misma serie (AR) como de los errores aleatorios o residuos pasados (MA). Este

modelo se representa como ARMA (p, q) donde p y q son los órdenes de sus modelos predecesores. Estos modelos se utilizan para predecir los valores futuros de la serie temporal y para analizarla mediante la identificación de tendencias, patrones estacionales y fluctuaciones aleatorias.

Por último, están los procesos ARIMA que combinan los modelos AR y MA con el tiempo de los procesos integrados (I), eliminando la tendencia estacional de la serie. Se representan como ARIMA (p, r, q) donde p y q pertenecen a los modelos dichos y r hace referencia al grado del polinomio que representa el tiempo [13].

2.3. Redes de Neuronas Artificiales

El Machine Learning (ML) o aprendizaje automático es una rama de la inteligencia artificial, Figura 4, que se enfoca en aprender a partir de datos. Su objetivo es desarrollar algoritmos que puedan reconocer patrones, aprender de manera autónoma, mejorar su rendimiento a medida que se les proporciona más datos y hacer predicciones precisas.

El Deep Learning (DL) es una técnica de aprendizaje automático o ML, Figura 4, que se basa en la creación de modelos de redes neuronales artificiales con múltiples capas. Estas redes están diseñadas para imitar el funcionamiento del cerebro humano, y son capaces de aprender a partir de grandes cantidades de datos de manera no supervisada o supervisada. Las redes neuronales son la base del Deep Learning. Se inspiran en la estructura del cerebro humano, poseen neuronas y conexiones entre ellas, y se utilizan para resolver problemas de clasificación, reconocimiento de patrones y procesamiento de imágenes y voz.

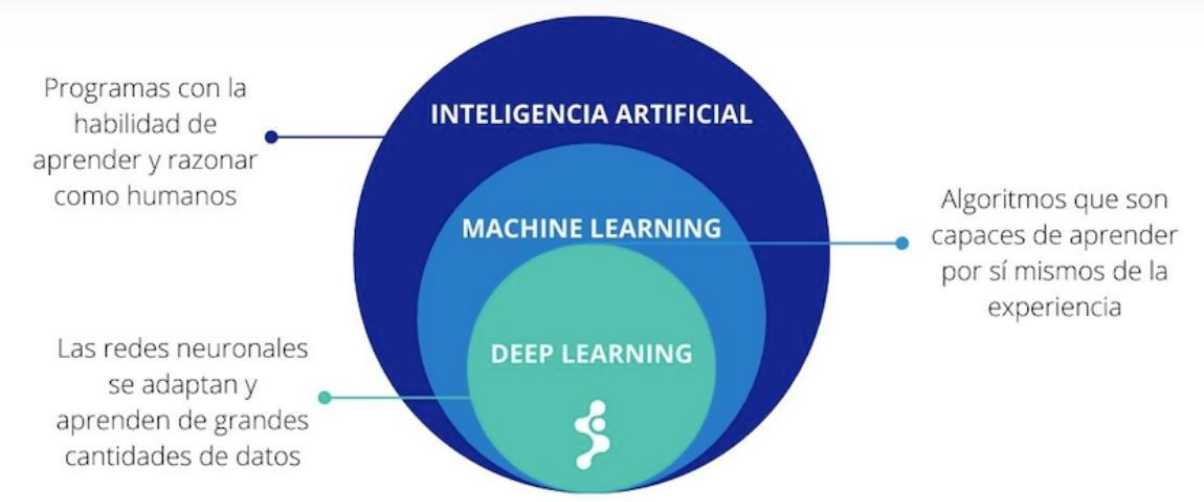


Figura 4: Inteligencia Artificial [16].

2.3.1. Perceptrón Multicapa

El perceptron multicapa, es una clase de red neuronal que nace de añadir funciones no lineales a las neuronas y retropropagación de los errores hacia las capas ocultas [18]. Estas redes son "full-connected" hacia delante, es decir, todas las neuronas de una capa están conectadas a todas las neuronas de la capa siguiente, también conocido como "feedforward", como se muestra en la Figura 5. El perceptrón multicapa tiene tres tipos de capas de neuronas con funcionalidades diferentes. La primera capa es la capa de entrada y posee una neurona por cada input que se introduce en la red, y su función es transmitir la entrada a la siguiente capa. La segunda capa contiene a su vez una o más capas que se hacen llamar capas ocultas. Estas capas ocultas propagan los datos de entrada, aplicando transformaciones no lineales a los datos que recibe cada neurona. La última capa es la capa de salida que se encarga de mostrar la salida obtenida de la red.

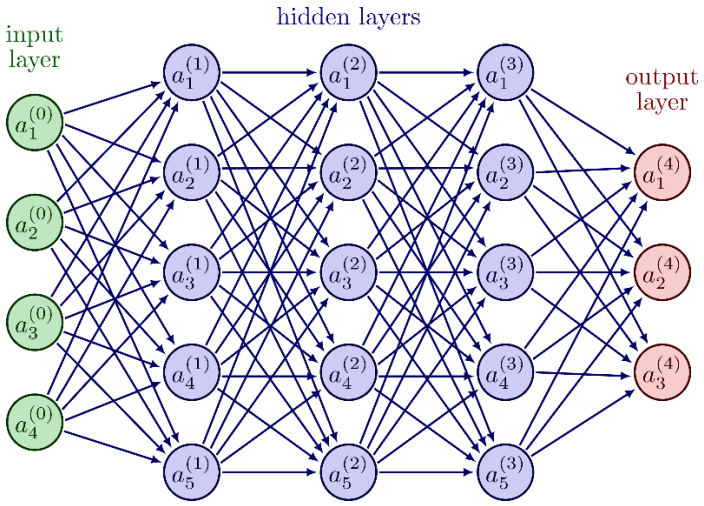


Figura 5: Arquitectura del Perceptron multicapa [19].

El elemento principal de las redes de neuronas artificiales es la neurona artificial. Estas neuronas son células que poseen un nivel de activación y una función llamada función de activación que hace cambiar la entrada que reciben. Cada neurona puede recibir información tanto del exterior como de otras neuronas. El estado de activación a_j de una neurona j, se calcula sumando todas las entradas x_i por los pesos w_i de las conexiones más el valor umbral θ de la propia célula,

$$a_j = F_j(x_1 w_1 + x_2 w_2 \ldots + \ x_n w_n + \theta_j)$$

La salida de la neurona j será el resultado de aplicar la función de activación F_j a la suma de las entradas y el umbral.

Las funciones de activación más utilizadas en las redes neuronales artificiales son la función sigmoidal, la función tangente hiperbólica y la función ReLU, Figura 6. La función sigmoidal

tiene un rango de valores de [0,1] y una función $f(x) = \frac{1}{1+e^{-x}}$. La función tangente hiperbólica cuenta con un rango mayor, [-1,1] y una expresión $f(x) = \frac{1-e^{-x}}{1+e^{-x}}$. Por último, la función ReLU que es la más utilizada actualmente $f(x) = \max(0, x)$. La función de activación de una red es la misma es todas las neuronas de las capas ocultas, y la misma o diferente en la capa de salida, y su elección la realiza el diseñador en función de los valores de activación que se quieran alcanzar.

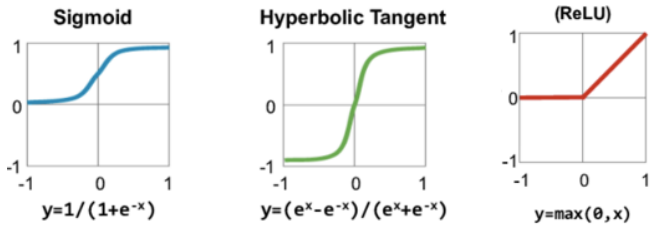


Figura 6: Funciones de activación más utilizadas [20].

El perceptrón multicapa es capaz de aprender gracias a su algoritmo de aprendizaje llamado algoritmo de RetroPropagación, Dicho algoritmo es supervisado, es decir, adapta y modifica los parámetros de la red de manera que el valor de las neuronas de la capa de salida se aproxime lo máximo posible a la salida esperada para cierta entrada [18]. El objetivo de este algoritmo es por lo tanto minimizar la diferencia entre la salida obtenida por la red y la salida esperada de todos los patrones o muestras que recibe como entrada. Se define la función de error como:

$$E = \frac{1}{N}\sum_{n=1}^{N} e(n)$$

siendo N el número de patrones o muestras disponibles y $e(n)$ el error con respecto a la salida esperada para el patrón n, obtenido por:

$$e(n) = \frac{1}{2}\sum_{i=1}^{N}(s_i(n) - y_i(n))^2$$

donde s_i corresponde al valor deseado del patrón i, e y_i al valor obtenido por la red. Para la minimización de esta función de error se suele utilizar el método del descenso del gradiente. El aprendizaje se realiza haciendo uso de métodos de gradiente estocástico, es decir, se realiza sobre los resultados de cada patrón en vez de sobre el error total E. Se utiliza la siguiente fórmula para cada patrón n,

$$w(n) = w(n-1) - \alpha\frac{\partial e(n)}{\partial w}$$

donde α es el valor de razón de aprendizaje y w representa cada parámetro (peso y umbrales) de la red. La razón de aprendizaje sirve para determinar la magnitud de los ajustes realizados a los pesos de las conexiones entre las neuronas durante el proceso de entrenamiento. Si la razón

del aprendizaje es demasiado alta, los ajustes serán grandes y la red puede no converger correctamente, lo que puede llevar a que el entrenamiento sea inestable o a que la red no logre aprender el patrón deseado. Por otro lado, si la razón del aprendizaje es demasiado baja, los ajustes serán pequeños y el entrenamiento puede volverse lento o quedar estancado en mínimos locales. Es por ello por lo que la razón de aprendizaje es un parámetro importante que debe ajustarse cuidadosamente durante el entrenamiento a base de realizar pruebas con la red.

El descenso de gradiente estocástico se realiza como se ha comentado para cada patrón, esto significa que para conjuntos de entrenamiento muy grandes se pierde eficiencia computacional. Pero, también existen otros tipos de descenso del gradiente, en concreto el descenso de gradiente por lotes o "batch" y el descenso del gradiente en mini-lotes o "mini-batch", que es el más utilizado [21]. El descenso del gradiente por lotes suma el error para cada patrón del conjunto de entrenamiento y calcula el cambio de pesos solo después de que todos los patrones hayan sido evaluados. Esta manera de calcular el descenso del gradiente reduce la cantidad de cálculo haciendo el proceso más eficiente, pero para aquellos conjuntos de entrenamientos más grandes el tiempo de espera se prolonga ya que necesita guardar en memoria todas las evaluaciones de los patrones existentes. Para lograr un equilibrio entre eficiencia y velocidad del descenso del gradiente existe el descenso de gradiente por mini-lotes que combina el estocástico y el de por lotes. El descenso del gradiente por mini-lotes se caracteriza por dividir el conjunto de entramiento en pequeños conjuntos o lotes, y realiza el cambio de pesos con las evaluaciones de los patrones de cada lote.

A este algoritmo se le llama retropropagación ya que el error de la salida se propaga por la red hacia atrás, asignando un valor de error para cada neurona.

Al igual que la razón de aprendizaje, el número de capas ocultas y neuronas por cada capa queda a la elección del diseñador. Hasta la fecha no hay ningún mecanismo que determine la mejor combinación de configuración de la red para obtener el menor error posible, por lo tanto, el proceso de encontrar la mejor configuración de la red se realiza por prueba y error.

2.3.2. Redes de Neuronas Recurrentes

Dentro de las redes de neuronas artificiales encontramos también las redes de neuronas recurrentes, RNN por su significado en inglés Recurrent Neural Networks. Son un tipo especializado de red neuronal que posee bucles en las neuronas, denominado conexiones recurrentes [22]. Estos bucles pueden ser de una neurona con conexiones a ella misma, conexiones entre neuronas pertenecientes a la misma capa, o conexiones de una capa con la capa anterior. Con la presencia de estos bucles, el estado de activación de una neurona no depende solo de las neuronas de la capa anterior como pasaba en el caso del perceptron multicapa, sino que depende de toda neurona conectada a ella. Estas redes incluyen una variable de tiempo en la activación de las neuronas que se representa de la siguiente manera:

$$a_i(t+1) = f_i\left(\sum_j w_{ji} a_j(t)\right)$$

donde j es el índice de la neurona conectada a la neurona i. Gracias a la incorporación de la variable tiempo, estas redes pueden modelar las relaciones entre los elementos de una secuencia y pueden capturar información contextual para hacer predicciones más precisas que las redes de neuronas tradicionales, ya que la información se propaga a través del tiempo, Figura 7. Por ello, se suelen utilizar para procesar datos secuenciales, como lo son el lenguaje natural, el habla, la música y las series temporales. Además, estas redes son especialmente útiles para procesar secuencias de longitud variable, ya que la red puede seguir procesando la entrada hasta que se alcanza un símbolo especial de final de secuencia. Esto significa que pueden trabajar con entradas de longitud variable, como oraciones de diferentes longitudes.

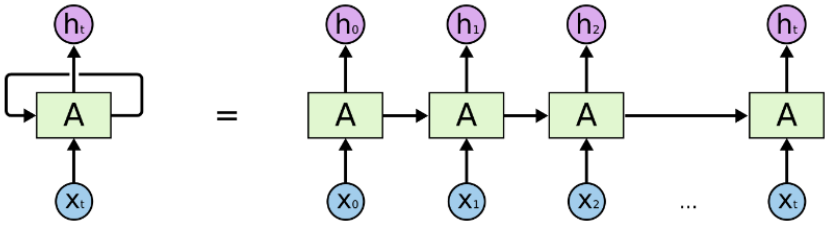


Figura 7: Arquitectura de una RNN [23].

Aunque las RNN son eficaces para modelar datos secuenciales, tienen un problema conocido como "desvanecimiento del gradiente". Este problema se refiere a la disminución de la magnitud del gradiente a medida que se retropropaga el error por las capas de la red, que resulta en gradientes que se hacen cada vez más pequeños y finalmente desaparecen. Esto significa que el impacto del error en las capas iniciales de la red disminuye significativamente y se vuelva casi insignificante, provocando que las capas iniciales no se actualicen adecuadamente y tengan un impacto mínimo en el proceso de aprendizaje. Como consecuencia las redes recurrentes son especialmente problemáticas en secuencias largas, donde los gradientes se propagan a través de múltiples pasos de tiempo. Como solución a este problema se crearon arquitecturas de redes más avanzadas que controlan este problema, como las redes LSTM (Long Short-Term Memory) o GRU (Gated Recurrent Unit).

2.4. Long Short-Term Memory

Las redes LSTM (Long Short-Term Memory) fueron presentadas por Hochreiter & Schmidhuber en 1997 [24]. Se crearon con la intención de resolver el problema memoria a largo plazo, “long-term memory”. Mientras que las redes recurrentes cuentan con una sola capa en el módulo recurrente A, como se muestra en la Figura 8, las redes LSTM cuentan con cuatro capas que interactúan entre ellas, Figura 9.

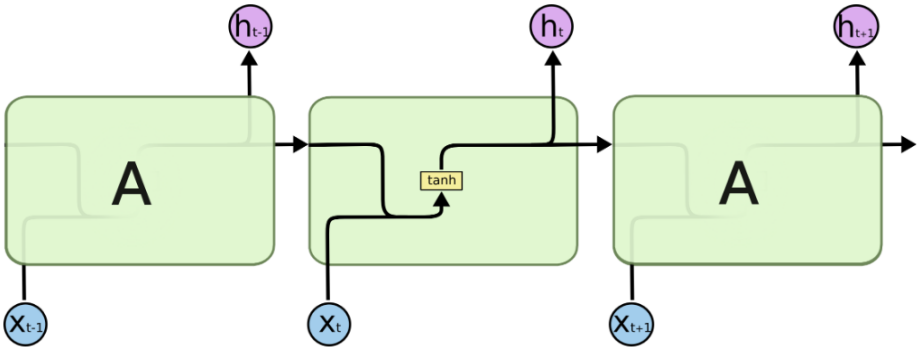


Figura 8: Estructura del módulo recurrente en las RNN [23].

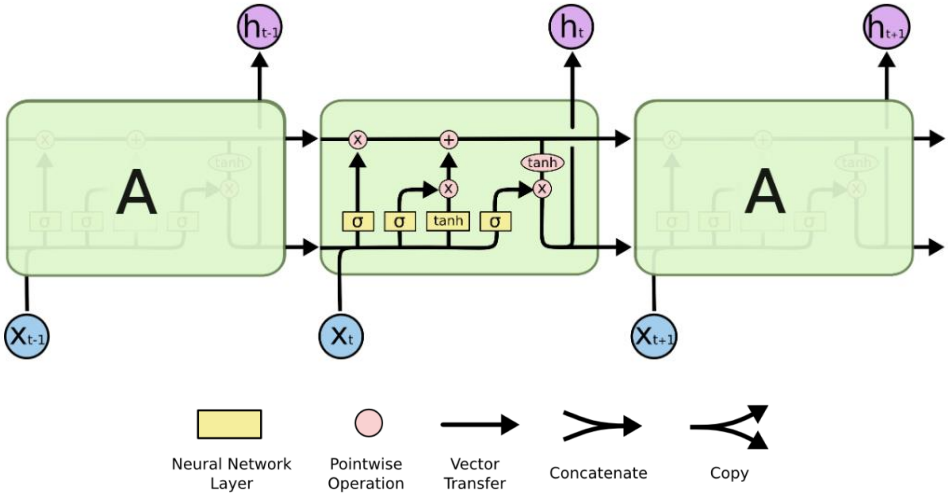


Figura 9: Estructura del módulo recurrente en las LSTM [23].

En la Figura 8 y 9, las líneas negras representan un vector completo, cada cuadrado amarillo representa una capa aprendida de la red con sus correspondientes funciones de activación, los puntos rosas definen operaciones vectoriales como la suma de vectores, las líneas que se fusionan representan una concatenación de dos vectores, y las líneas que bifurcan significa que el contenido del vector se copia a las dos salidas.

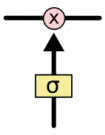
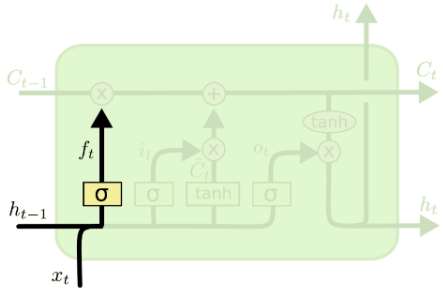


Figura 10: Puerta lógica en las células de memoria de una red LSTM [23].

Las células de memoria de las redes LSTM son capaces de aprender cuándo es importante olvidar información antigua y cuándo es importante recordarla para predecir la salida, y esto lo consiguen gracias al uso de puertas lógicas. Estas puertas están compuestas por una capa de

neuronas que poseen la función sigmoidal, seguido de una operación de multiplicación para controlar la información que continúa en la célula de memoria A, como se puede ver en la Figura 10. La función sigmoidal como produce números entre 0 y 1 permite indicar cuanta cantidad de información se quiere conservar para cada valor del vector, siendo 0 nada y 1 todo. Las redes LSTM implementan una arquitectura que consta de tres puertas: la puerta de entrada, la puerta de olvido y la puerta de salida.

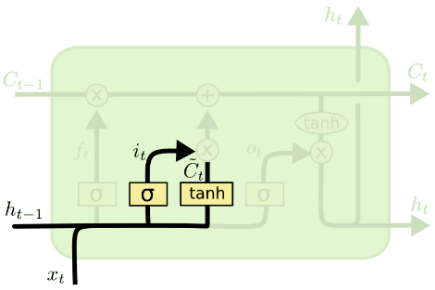
El primer paso que realizan estas redes es decidir cuánta información antigua eliminan, Figura 11. Para ello utilizan la puerta de olvido que coge los vectores de h_{t-1} (output de la célula anterior) y x_t (vector entrada de la red), y le asigna un valor entre 0 y 1 con la función sigmoidal (σ) a cada valor del vector.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figura 11: Puerta de olvido en las redes LSTM [23].

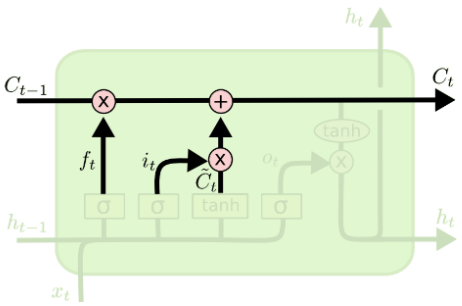
El siguiente paso de la red es decidir qué nueva información se va a guardar en la célula de memoria y cuanta importancia tendrá dicha información. Para esto se implican dos capas, la primera donde actúa la función sigmoidal sobre h_{t-1} y x_t , llamada puerta de entrada, que decide que valores se van a modificar, y una segunda donde la función de activación tangente-hiperbólica (tanh) crea un vector nuevo denominado $\tilde{C}_t$ que posee información nueva que podría añadirse a C_t, Figura 12.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figura 12: Puerta de entrada en las redes LSTM [23].

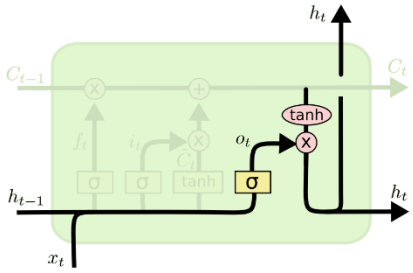
La siguiente acción es realizar el cálculo del nuevo valor de C_t. Para ello se multiplica el vector C_{t-1} y f_t para así eliminar la información a olvidar, y a su vez se multiplica el vector $\tilde{C}_t$ con i_t para indicar la nueva información a guardar. Finalmente, se suman ambos valores y se obtiene el nuevo vector de estado, Figura 13.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figura 13: Actualización del vector de estado de la célula de memoria en el instante t [23].

Por último, encontramos la puerta de salida que decide qué información sale como output de la célula, Figura 14. Dicha salida se compone de información del vector de estado C_t en el porcentaje que decide la capa sigmoidal con resultado O_t. Para ello, el vector de estado pasa primero por la función tanh que asignar a cada valor de este un número entre -1 y 1, y luego se multiplica con el vector O_t para así indicar cuanta cantidad de cada valor del vector se convierte en output de la célula, designado como h_t.



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figura 14: Puerta de salida en las redes LSTM [23].

El proceso definido se realiza dentro de cada célula de memoria (A) de la red, pudiendo tener la red tantas células como el diseñador desee. Normalmente esta elección se realiza mediante experimentación.

2.5. Transformada de Fourier

La Transformada de Fourier es una herramienta matemática que se utiliza para analizar las frecuencias de una función que depende del tiempo. Permite descomponer una serie temporal en componentes de diferentes frecuencias y obtener información sobre la contribución relativa de cada una de ellas.

En el contexto de las series temporales, la transformada de Fourier se aplica para analizar la estacionalidad ya que permite identificar las frecuencias dominantes presentes en la serie y su amplitud relativa pudiendo identificar patrones periódicos o la presencia de componentes oscilatorios en la misma [25]. Además, también permite representar series temporales mediante frecuencias, haciendo uso de funciones como seno y coseno. Con la Transformada de Fourier

también se puede identificar cambios o picos en el espectro de frecuencias, lo que puede indicar anomalías o eventos significativos en las series. Es por todo esto por lo que se usa en las series temporales ya que puede aportar información relevante para el análisis y procesamiento de estas series.

Usar la Transformada de Fourier en una serie temporal diaria de contaminación atmosférica puede ser útil para encontrar variaciones semanales, mensuales, estacionarias o anuales que ayuden a la predicción de la red neuronal. Para este tipo de series temporales se usa la Transformada Discreta de Fourier ya que el input son valores discretos, un valor por cada paso de tiempo, y no continuos. La salida u output de esta transformada es un valor de amplitud para cada posible frecuencia, representados gracias a los Coeficientes de Fourier, y cuanta mayor amplitud tenga una frecuencia mayor importancia tendrá sobre la serie temporal. La fórmula de la nueva serie representada en frecuencias es la siguiente:

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{i2\pi}{N}kn}$$

donde X_k es la nueva serie de números imaginarios, x_n es la serie original y N su longitud.

2.6. Herramientas utilizadas

A continuación, se describen algunas de las herramientas usadas a lo largo de este proyecto:

- **Python**: Es un lenguaje de programación eficiente, sencillo de aprender y fácil de leer, además, cuenta con una amplia variedad de librerías con código reutilizable por los programadores gracias a la enorme comunidad activa que tiene detrás. Se utiliza hoy en día sobre todo en desarrollo de software, de aplicaciones web y en la ciencia de datos y machine learning [26].
- **TensorFlow**: Es una plataforma de código abierto que permite a los usuarios crear modelos de aprendizaje automático. Sus API están basadas en Keras para definir y entrenar redes neuronales, lo cual permite la creación y evaluación de prototipos y modelos rápidamente y fáciles de usar [27]. Esta plataforma incluye las redes LSTM usadas en el proyecto.
- **pandas**: Es un módulo o librería de Python que proporciona estructuras de datos rápidos y adaptables que facilitan el trabajo con datos relacionales. Se utiliza para realizar análisis de datos prácticos y es la herramienta perfecta para la manipulación de los mismos ya que es muy potente y flexible [28]. Sus dos estructuras de datos principales son las *Series* (1 dimensión) y los *DataFrames* (2 dimensiones). En este caso se han utilizado los *DataFrames* para realizar el proceso de preprocesado y limpieza de datos, así como el análisis de la serie temporal.
- **NumPy**: Es una librería de Python que ofrece el uso de un *array* multidimensional (n dimensiones) llamado *ndarray*. Además, ofrece un gran número de funciones pasa usar con los *ndarrays* desde selección, ordenación y manipulación de los mismos hasta operaciones básicas y algunas más complejas [29]. El tipo de array que ofrece esta

librería se utiliza para entrenar los modelos de Keras disponibles en TensorFlow por lo tanto es necesaria para poder realizar los entrenamientos de los modelos.

- **Streamlit**: Es un framework open source que permite crear aplicaciones web interactivas basadas en machine learning y datos usando como lenguaje de programación Python [30].

Aunque las descritas son estas debido a que son las que han tenido más importancia a lo largo del proyecto, no son las únicas que se han utilizado en él.

2.7. Estudios similares

Existen varios trabajos ya realizados en el marco de la predicción de los valores de contaminación. Muchos de ellos comparan modelos estadísticos de aproximación como ARIMA con modelos de redes neuronales.

Un ejemplo de proyecto únicamente realizado con ARIMA lo realizó María Medina con un estudio de la predicción de la contaminación del aire en Madrid [31]. María centró su estudio en la predicción de CO_2 apoyándose de datos meteorológicos. Obtuvo una precisión de 87.98% en las predicciones a 24 horas, 87.36% para las de 48 horas y 77.97% en las de 7 días.

Hoy en día casi todos los estudios que salen comparan el comportamiento de ciertos modelos de redes de neuronas artificiales en la predicción de contaminación. Un ejemplo de estudio de este tipo es el realizado por S. Kumari y S. K. Singh [32]. En dicho estudio analizan el resultado que obtienen los modelos ARIMA, SARIMAX, Holt-Winters, linear regression, random forest regression, y LSTM en la predicción de NO_2 en India. En él concluyen que la red LSTM es la que mejor predicción realiza con un horizonte de 24 horas frente a los otros modelos, en términos de MAPE (Mean Absolute Percentage Error) llegando a alcanzar la red un error de solo un 3.101%. Otro estudio semejante fue realizado sobre la calidad del aire en las principales ciudades de China [33], donde se trabajó con una muestra que comprendía datos horarios de todos los días desde 2015 a 2019 donde se usaron diferentes métodos de predicción, los cuales incluyeron Back Propagation Neural Network (BPNN), Convolutional Neural Networks (CNN), Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), y Bi-directional Long Short-Term Memory (BiLSTM). En este estudio se estudiaron 7 contaminantes diferentes y con ellos intentaron predecir el Índice de Calidad del Aire. Los errores que estudiaron fueron el Root Mean Square Error (RMSE), Mean Square Error (MSE), Coefficient of Determination (R2), y Mean Absolute Error (MAE). El estudio concluyó que el modelo con los mejores resultados fue el BiLSTM, seguido del BPNN con los segundos mejores resultados.

También hay algunos trabajos que combinan varios tipos de modelos neuronales para realizar las predicciones de series temporales. Un ejemplo de esto es el trabajo realizado con una red CNN-LSTM que basándose en los 24 últimos datos horarios predice la concentración de contaminación de la hora actual [34]. El modelo cuenta con una capa de red CNN (Convolutional Neural Network), seguida de dos capas de red LSTM, una capa de regularización y una cada de perceptrón multicapa full-connected. El input del modelo son valores de concentración horarios de PM_{10}, datos meteorológicos y datos de estacionalidad y

tiempo. Para evaluar el modelo se usaron las medidas de RMSE y MAE, obteniendo finalmente un valor de 11,1 y 5,4 respectivamente. Por último, se comparó con modelos de Linear Regression, Random Forest Regression, Support Vector Regression y todos obtuvieron peores resultados que el modelo propuesto de Deep Learning.

Otro ejemplo de trabajo que mezcla varios modelos neuronales es el realizado por Jun Luo y Yaping Gong en el cual la predicción de calidad del aire se basa en un modelo ARIMA-WOA-LSTM [35]. En este modelo, ARIMA se encarga de extraer la parte lineal de la serie temporal, mientras que el modelo WOA-LSTM se encarga de realizar la predicción de la parte no lineal. La parte de WOA (Whale Optimization Algorithm) busca los mejores hiperparámetros para más tarde entrenar la red LSTM. Con este modelo consiguen mejores resultados para RMSE y MAE que usando ARIMA-LSTM o CEEMDAN-WOA-LSTM.

El estudio más similar al realizado en esta memoria es el trabajo de fin de máster de Gabriel Villalba [36]. En dicho trabajo el autor realiza un estudio de ciertos algoritmos de predicción para la contaminación en Madrid, MLP (Multilayer Perceptron), LSTM, CNN (Convolutional Neural Network) y SVM (Support Vector Machines) en función del RMSE (Root Mean Square Error), aunque en este estudio la predicción solo se realiza a día vista, es decir con un horizonte de 24 horas.

2.8. Justificación de la solución

Tras leer los estudios citados y ponderando otras alternativas se ha elegido para realizar este trabajo la generación de modelos LSTM ya que parecen los más adecuados dada su sencillez de uso y tienen excelentes resultados en predicción de series temporales. Además se evaluarán teniendo en cuenta los errores MSE, RMSE y MAE ya que son los más usados en los estudios citados.

A su vez, para trabajar con estas redes se ha elegido usar el lenguaje de programación Python ya que dispone de librerías muy útiles y sencillas para trabajar y realizar un estudio completo con estas redes como lo son TensorFlow, pandas, NumPy, matplotlib, statsmodels y sklearn entre otras. Además su uso es mayoritario en el sector de la ciencia de datos.

Para el desarrollo de la aplicación web se ha decidido utilizar Streamlit. Este framework es open source y permite crear aplicaciones web interactivas basadas en machine learning y datos usando como lenguaje de programación Python [30]. Como el proyecto desarrollado son modelos de machine learning y necesitan que el usuario seleccione ciertos parámetros, Streamlit se convierte en un framework perfecto para realizar un desarrollo rápido y sencillo de la aplicación web.

3. ESTUDIO EXPERIMENTAL

Para estudiar si es posible una predicción aproximada de la calidad del aire se ha hecho uso de redes LSTM. La variable que se ha intentado predecir en este estudio es el nivel de NO_2 (dióxido de nitrógeno) medio diario en diferentes estaciones de la ciudad de Madrid. Para ello ha sido necesario obtener datos para realizar el entrenamiento de la red. Principalmente los datos necesarios son registros de calidad del aire diarios. Además, como se menciona en el punto 2.1 de esta memoria, hay artículos que mencionan una posible correlación entre los valores meteorológicos con los valores de NO_2 en un mismo día, es por ello por lo que también se han obtenido valores meteorológicos para estudiar si la red es capaz de disminuir el error gracias a estos datos auxiliares. A continuación, se explica la metodología que se va a seguir, el proceso realizado de obtención y limpieza de datos, análisis de la serie temporal, experimentación y entrenamiento de la red y por último la evaluación de los resultados obtenidos.

3.1. Metodología

La metodología seguida para realizar el estudio experimental de los modelos de predicción consta de varias fases que se van a explicar en profundidad a continuación.

Para realizar el entrenamiento de los modelos se ha decidido separar los datasets en entrenamiento, validación y test (train-val-test) con un porcentaje de 80%, 10% y 10% en cada uno de ellos. El conjunto de entrenamiento es más grande ya que se utiliza para entrenar el modelo, la red neuronal ajusta sus parámetros (pesos y umbrales) en función de estos datos para minimizar el error en las predicciones. Por otra parte, el conjunto de validación sirve para ajustar los hiperparámetros más adecuados para la red neuronal y para controlar el rendimiento durante el entrenamiento. Además, ayuda a detectar el sobreajuste u “overfitting”, que ocurre cuando el modelo se adapta demasiado a los detalles específicos del conjunto de entrenamiento y no generaliza bien a nuevos datos. Y por último, el conjunto de test se utiliza para evaluar el rendimiento del modelo final una vez que se ha completado el entrenamiento. Proporciona una evaluación imparcial del rendimiento del modelo en datos que nunca ha visto antes, por lo tanto es fundamental que este conjunto no se utilice durante el proceso de entrenamiento, ya que puede llevar a la selección de modelos que se desempeñan bien solo en los datos de test.

Una vez que se ha estudiado cuál es la configuración de hiperparámetros que menor error obtiene en el conjunto de validación se junta el conjunto de entramiento y el de validación para entrenar el modelo final con los hiperparámetros elegidos y se evalúa el modelo con el conjunto de test. Para un mayor entendimiento se añade la Figura 15 que muestra el proceso experimental, y la Figura 16 con la distribución de cada dataset.

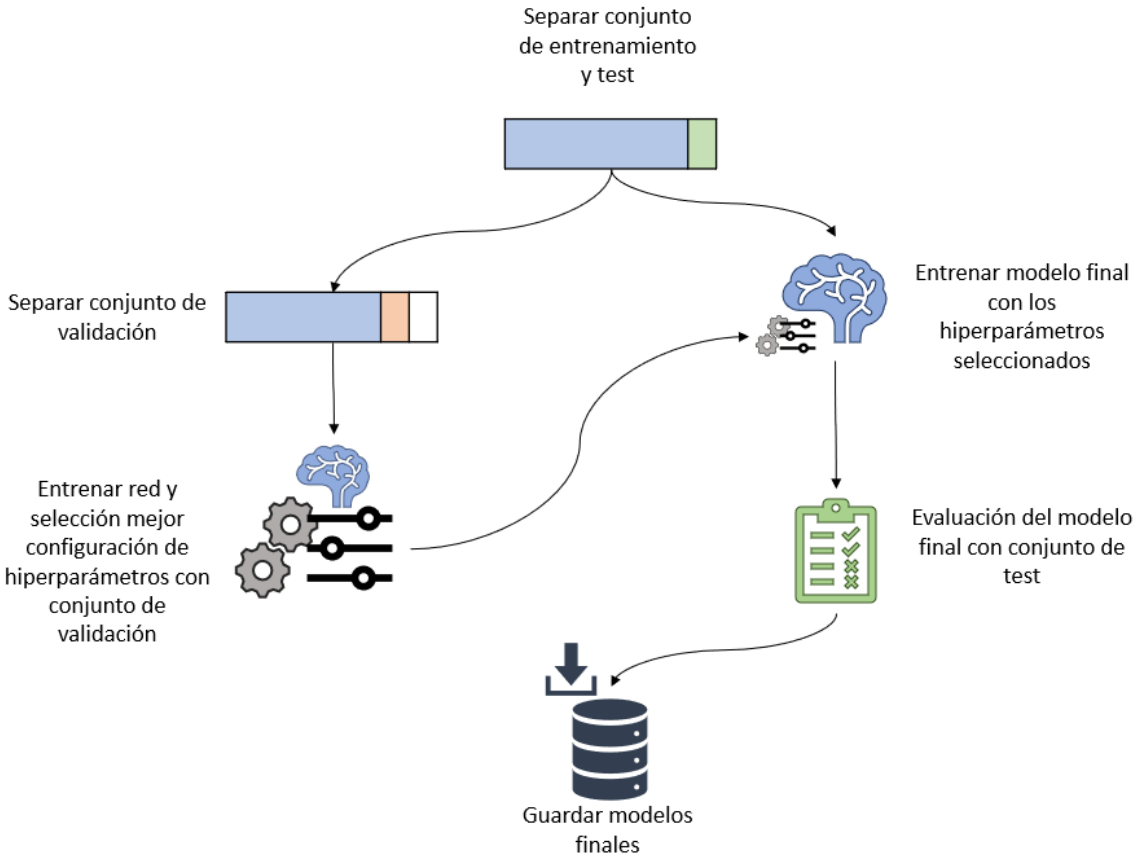


Figura 15: Proceso experimental.

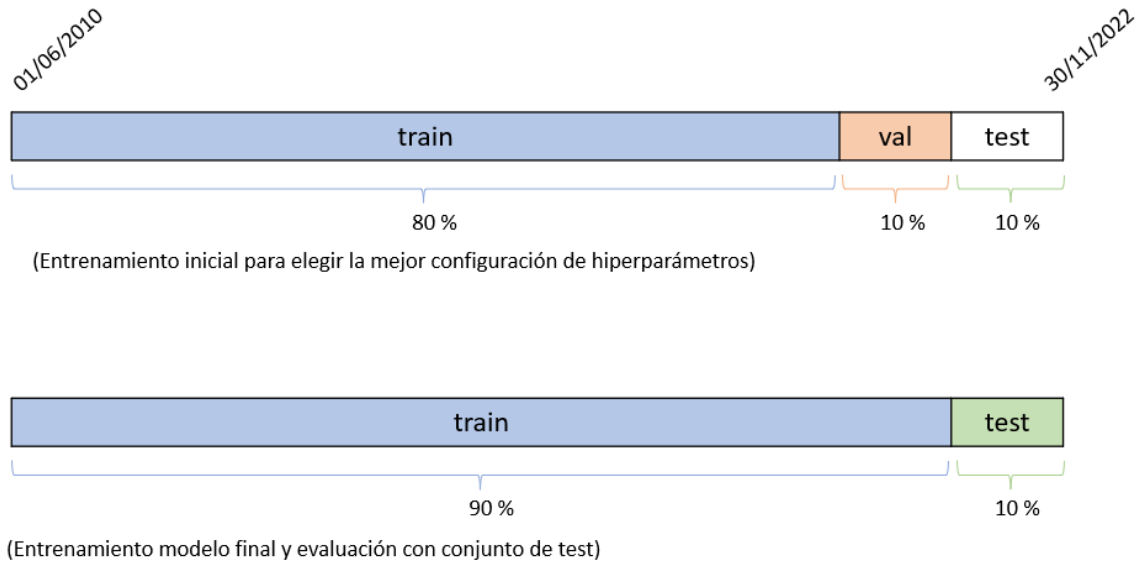


Figura 16: Separación de los datasets.

Tras crear los modelos finales de predicción se pasará a las fases de análisis, diseño e implementación de la aplicación web donde se podrán usar estos para obtener predicciones reales. Dichas fases se explicarán en siguientes capítulos de la memoria.

3.2. Obtención de datos y preprocesado

3.2.1. Datos de la calidad del aire

El ayuntamiento de Madrid pone a la disposición de los ciudadanos un archivo de datos abiertos [37] con información sobre distintos datos que recogen de la ciudad, con la intención de que los ciudadanos tengan la posibilidad de innovar y crear herramientas con dicha información. Este portal cuenta con datos sobre de la calidad del aire que son los usados a lo largo de este estudio.

El ayuntamiento publica datos sobre la calidad del aire desde el 2001 hasta el presente. Estos datos corresponden a la información que recogen las 24 estaciones y puntos de muestreo que hay colocados estratégicamente en la ciudad de Madrid. En la Figura 17 se ven todas las estaciones que existen. Hay diferentes tipos de estaciones dependiendo de la función que desempeñan y se pueden clasificar de la siguiente manera [38],

1. Suburbana (Verde): a las afueras de la ciudad donde hay mayor nivel de ozono.
2. Urbana de fondo (Azul): miden la exposición de los ciudadanos al aire.
3. Urbana de tráfico (Rojo): colocadas estratégicamente en los puntos con mayor influencia de carreteras o calles con emisiones altas.

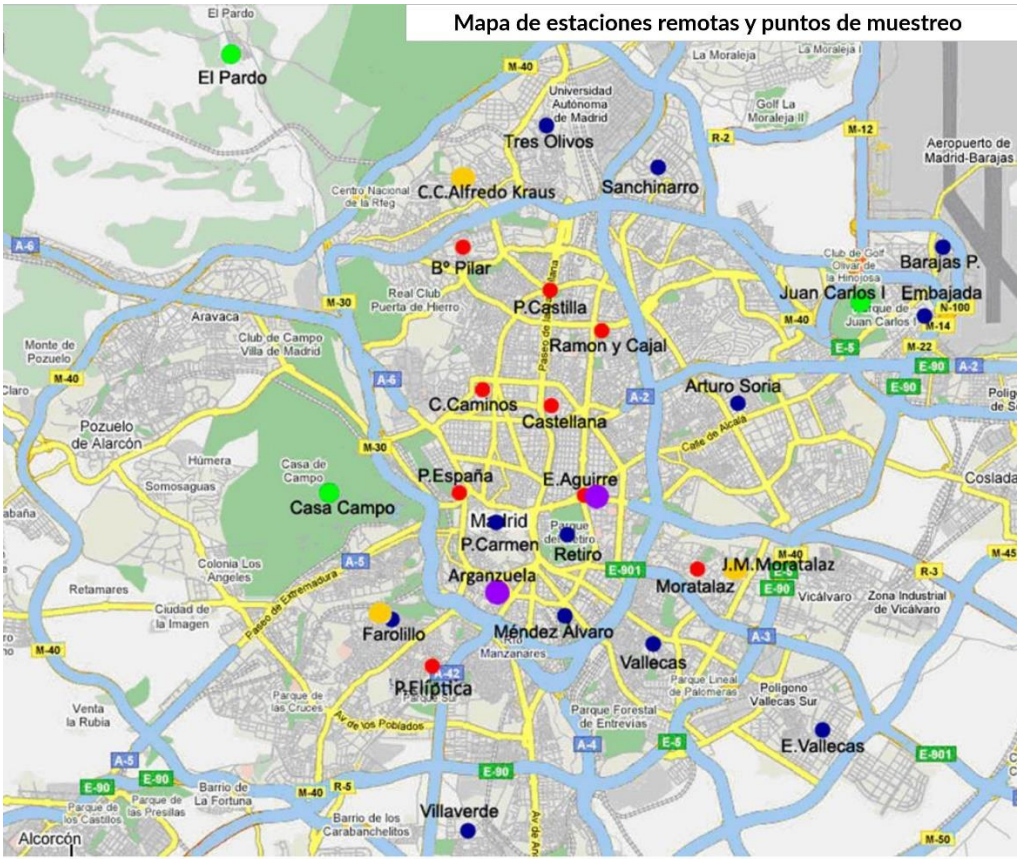


Figura 17: Mapa de estaciones remotas y puntos de control del Ayuntamiento de Madrid [39].

No todas las estaciones tienen los mismos analizadores, aunque todas ellas cuentan con analizador de NO_2 como se puede ver en la Tabla 1.

Estación – Puesto de muestreo	NO_2[1]	SO_2[2]	CO[3]	PM10[4]	PM2,5[5]	O_3[6]
Plaza de España	X	X	X			
Escuelas Aguirre	X	X	X	X	X	X
Ramón y Cajal	X					
Arturo Soria	X					X
Villaverde	X					X
Farolillo	X			X		X
Casa de Campo	X			X	X	X
Barajas Pueblo	X					X
Plaza del Carmen	X	X	X			X
Moratalaz	X	X		X		
Cuatro Caminos	X			X	X	
Barrio del Pilar	X					X
Vallecas	X			X		
Méndez Álvaro	X			X	X	
Castellana	X			X	X	
Parque del Retiro	X					X
Plaza de Castilla	X			X	X	
Ensanche de Vallecas	X					X
Urb. Embajada	X			X		
Plaza Elíptica	X		X	X	X	
Sanchinarro	X			X	X	
El Pardo	X					X
Juan Carlos I	X					X
Tres Olivos	X			X		X

Tabla 1: Distribución de analizadores y equipos de muestreo [39].

Las estaciones que se han escogido para realizar el estudio han sido: Paseo de la Castellana, Plaza del Carmen y Retiro. La primera estación es categorizada como estación urbana de tráfico, debido a la afluencia de coches en la zona, mientras que las otras dos son estaciones urbanas de fondo que se usan para medir la exposición de los ciudadanos al aire.

La primera parte del preprocesado de datos fue obtener todos los archivos relevantes existentes. Las estaciones escogidas para el estudio cuentan con registro desde 2010, así que los datos utilizados para el entrenamiento de las redes neuronales comprenden todos los días desde que hay registro hasta noviembre de 2022.

[1] Dióxido de nitrógeno.
[2] Dióxido de azufre.
[3] Monóxido de carbono.
[4] Partículas en suspensión con un diámetro aerodinámico de hasta 10 µm.
[5] Partículas en suspensión con un diámetro aerodinámico de hasta 2,5 µm.
[6] Ozono troposférico.

Los datos diarios que proporciona el Ayuntamiento de Madrid tienen un formato bastante complejo en archivos ".csv". Cada fila del fichero contiene todos los valores de un mes para cierta estación y cada magnitud medida siendo las columnas del archivo las siguientes:

PROVINCIA;MUNICIPIO;ESTACION;MAGNITUD;PUNTO_MUESTREO;ANO;MES;D01;V01;D02;V02;D03;V03;D04;V04;D05;V05;D06;V06;D07;V07;D08;V08;D09;V09;D10;V10;D11;V11;D12;V12;D13;V13;D14;V14;D15;V15;D16;V16;D17;V17;D18;V18;D19;V19;D20;V20;D21;V21;D22;V22;D23;V23;D24;V24;D25;V25;D26;V26;D27;V27;D28;V28;D29;V29;D30;V30;D31;V31

Esta distribución de los valores no es cómoda para trabajar con la red LSTM por lo que fue necesario un proceso de limpieza de los datos. Dicho proceso se llevó a cabo usando la librería "pandas" de Python que permite manejar y modificar fácilmente ficheros ".csv" y su contenido mediante "dataframes", que permiten estructurar los datos en filas y columnas. Los pasos realizados para la limpieza y modificación de los archivos fueron los siguientes:

1. Filtrar por la magnitud/analizador deseado, es decir, NO_2.
2. Filtrar por la estación deseada de las 24 existentes.
3. Eliminar todas las columnas no necesarias. Solo se guarda la información referente a la estación, la magnitud, el año, el mes y 31 columnas de días.
4. Convertir las columnas a filas.
5. Ordenar los valores por año, mes y día, de más cercano a más lejano.
6. Eliminar los días con valores nulos correspondientes a días no existentes, como lo son el 29, 30, 31 de febrero y aquellos meses que no tienen día 31.

Una vez realizada esta primera limpieza, se apreció que había años con algunos días iguales a 0. Esto significa que la estación estuvo por algún motivo dada de baja o no operativa, por lo tanto, el valor no es válido. Ya que se utilizan series temporales y se considera que el día de la semana influye en el valor de un elemento, se decidió no eliminar los días con valores nulos y reemplazarlos por la media del valor del mismo día de la semana de la semana previa y de la siguiente. Una vez realizada esta limpieza se obtiene un dataset ordenado por fecha con los valores de NO_2 en $\mu g/m^3$ para cada día que puede ser utilizado por la red LSTM, Figura 18.

	DATE	NO2
0	6/1/2010	36
1	6/2/2010	33
2	6/3/2010	27
3	6/4/2010	40
4	6/5/2010	40
...	...	...
4561	11/26/2022	45
4562	11/27/2022	38
4563	11/28/2022	28
4564	11/29/2022	57
4565	11/30/2022	67

4566 rows × 2 columns

Figura 18: Dataset limpio con datos de NO_2 de la estación Paseo de la Castellana.

3.2.2. Datos meteorológicos

Además de los datos de calidad del aire, se han utilizado datos de meteorología procedentes de la Agencia Estatal de Meteorología (AEMET) y su servicio de datos abiertos [40]. Se han usado específicamente los datos de la estación de Cuatro Vientos, ya que es la estación con menos valores nulos, para complementar los datos de calidad de aire y estudiar si su uso mejora o no la predicción de los modelos. A diferencia de los datasets de calidad del aire, estos procedentes de AEMET poseen una estructura mucho más trabajable y por lo tanto el proceso de estructurar los datos fue más sencillo. El fichero cuenta con las siguientes columnas de información:

FECHA;INDICATIVO;NOMBRE;PROVINCIA;ALTITUD;TMEDIA;PRECIPITACION;TMIN;HORATMIN;TMAX;HORATMAX;DIR;VELMEDIA;RACHA;HORARACHA;SOL;PRESMAX;HORAPRESMAX;PRESMIN;HORAPRESMIN

Las variables que se han escogido para entrenar la red neuronal han sido la velocidad media del viento, la temperatura media y la precipitación ya que de acuerdo con varios estudios comentados en el punto 2.1 son las variables más relevantes o las que pueden aportar más información. En el dataset proporcionado por AEMET dichas variables quedan representadas con los nombres *TMEDIA* para temperatura media, *VELMEDIA* para la velocidad media del viento *y PRECIPITACION* para la precipitación media, siendo las unidades de cada una °C, m/s y mm respectivamente.

El proceso para su limpieza fue el siguiente:

1. Filtrar por las magnitudes deseadas, en este caso, *TMEDIA*, *VELMEDIA* y *PRECIPITACION*.
2. Eliminar todas las columnas no necesarias.

3. Ordenamos los valores por fecha, de más cercano a más lejano.

Por último, se realiza el mismo proceso de remplazo de valores nulos que para el dataset de calidad del aire. Gracias a esta limpieza se obtiene un dataset ordenado por fecha con los valores de temperatura, viento y precipitación para cada día que puede ser unido al dataset de NO_2. Una vez unido se obtiene el dataset de la Figura 19.

DATE	NO2	TMEDIA	PRECIPITACION	VELMEDIA
2010-06-01	36.0	25.2	0.0	3.1
2010-06-02	33.0	25.2	0.0	1.4
2010-06-03	27.0	24.8	0.0	2.8
2010-06-04	40.0	26.0	0.0	1.9
2010-06-05	40.0	26.3	0.0	3.6
...	...	...	...	...
2022-11-26	45.0	8.7	0.0	1.1
2022-11-27	38.0	8.4	3.0	0.8
2022-11-28	28.0	8.8	0.0	5.3
2022-11-29	57.0	7.8	0.0	0.8
2022-11-30	67.0	8.0	0.0	0.6

4566 rows × 4 columns

Figura 19: Dataset limpio con datos de NO_2, temperatura, precipitación y viento de la estación Paseo de la Castellana.

3.3. Análisis de la serie temporal

A partir de este capítulo se va a explicar el proceso realizado para la estación de control de calidad del aire de Paseo de la Castellana. Para simplificar el proceso se ha decidido estudiar en profundidad solo esta estación y aplicar lo estudiado en las otras dos, Plaza del Carmen y Retiro.

Tras realizar el preprocesado de datos de esta estación, Paseo de la Castellana, se obtiene un dataset compuesto por 4566 valores de NO_2 expresados en $\mu g/m^3$, es decir 4566 días, representados en la gráfica de la Figura 20. Los valores de la serie van desde $1\mu g/m^3$ a 173 $\mu g/m^3$, registrado como valor máximo durante los años con registro, el valor medio es de 36,94 $\mu g/m^3$, y la desviación estándar o "standard deviation" es de 18,28 puntos, Figura 21.

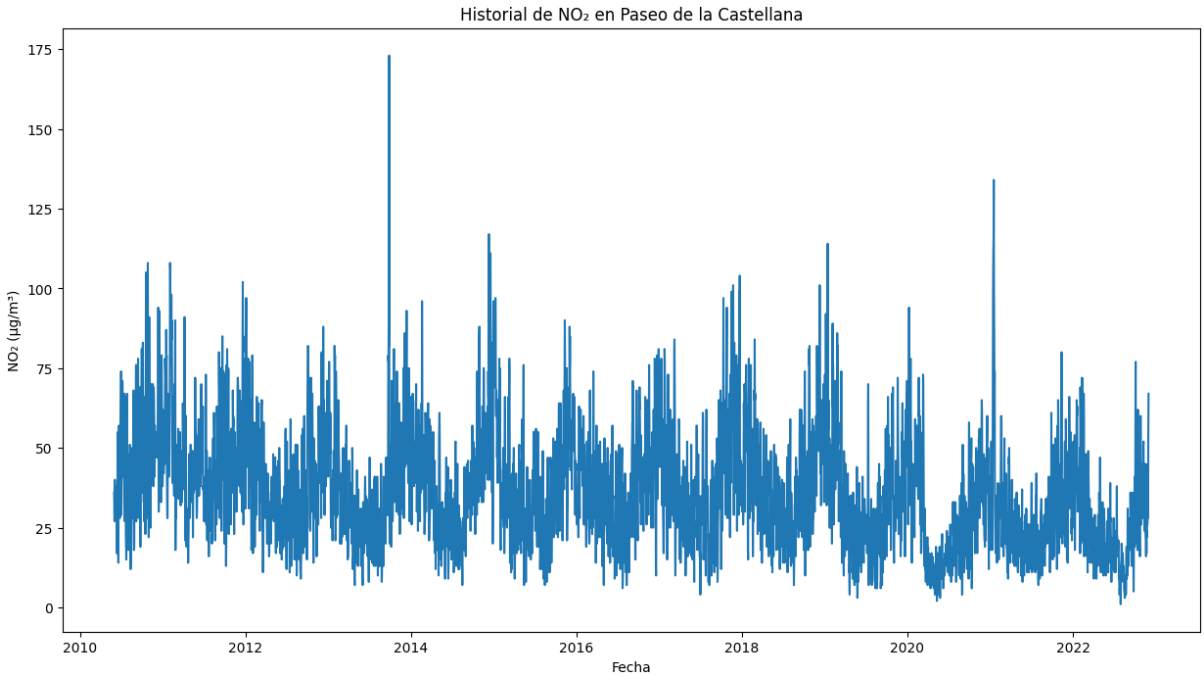


Figura 20: Gráfica del histórico de NO_2 en el Paseo de la Castellana.

	count	mean	std	min	25%	50%	75%	max
NO2	4566.0	36.9424	18.277284	1.0	24.0	34.0	47.0	173.0

Figura 21: Descripción de la serie Paseo de la Castellana.

Como se trabaja con una serie temporal se puede descomponer la gráfica en varias componentes conocidas como estacionalidad, tendencia y ruido. El método "seasonal_decompose" perteneciente a la librería *statsmodels.tsa.seasonal* de Python permite descomponer una serie temporal en estas componentes y ha sido el usado para entender el comportamiento de este serie, Figura 22. La descomposición se ha realizado con un periodo de 365 días para poder apreciar el comportamiento anual a lo largo de los 12 años que hay de registro. En la gráfica resultante se puede apreciar en la tendencia que los valores de NO_2 a lo largo de los años han ido disminuyendo muy poco a poco desde los 50 µg/m^3 a un poco menos de 30 µg/m^3. Se aprecia que existe una estacionalidad anual en la serie ya que todos los años se repite la misma curva, pero podría existir alguna tendencia más referente a periodos más pequeños, como trimestres o semanas. Por último, se puede ver que el ruido de la serie fluctúa entre valores de 30-40 por día, y algunos llegando incluso a más de 50. Es justo este parámetro, el ruido, el que se va a intentar predecir con el uso de la red neuronal LSTM.

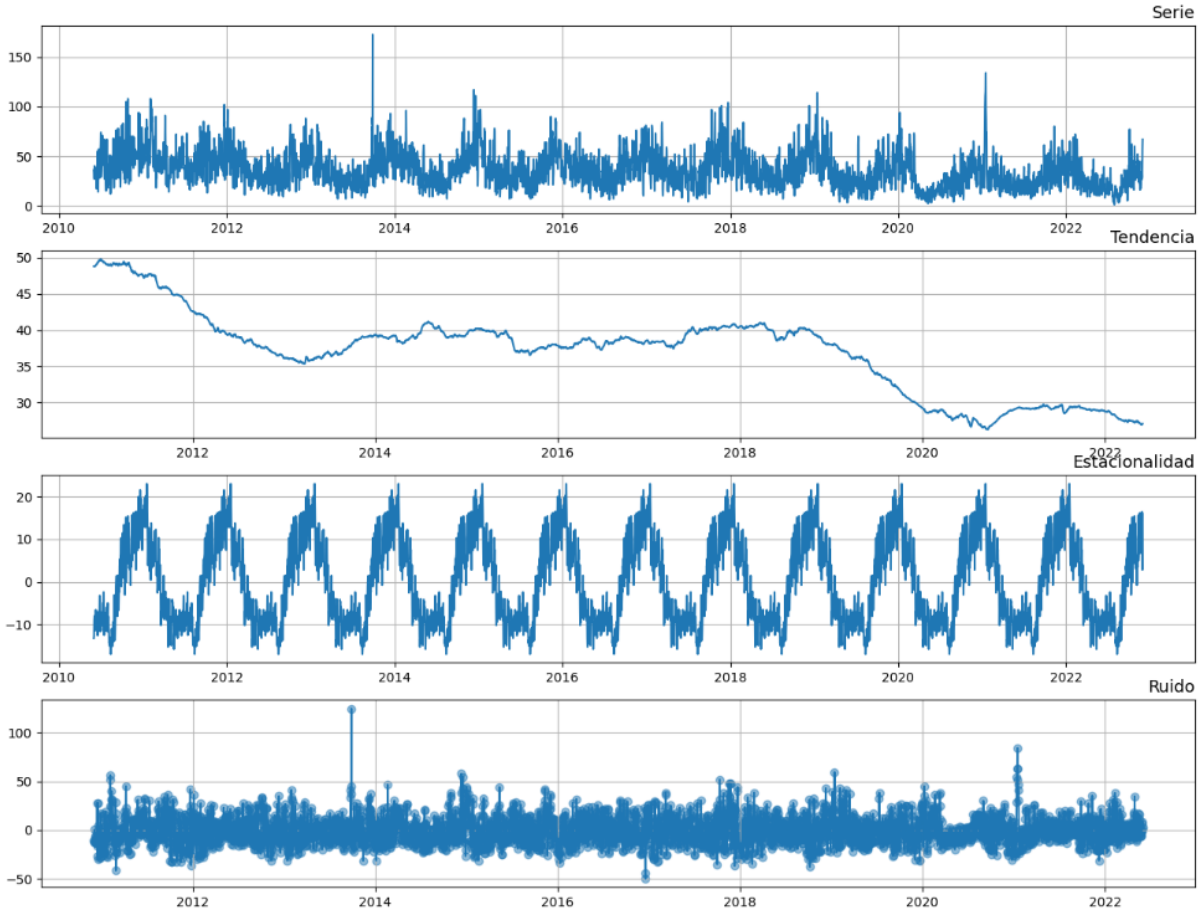


Figura 22: Descomposición de la serie temporal de Paseo de la Castellana.

Además de utilizar la descomposición estacional usando la librería mencionaba, se ha estudiado la estacionalidad de la serie temporal usando la Transformada de Fourier. Como se explica en el punto 2.5 de esta memoria, la Transformada de Fourier es útil para analizar las frecuencias de la serie y obtener aquella más repetida a lo largo de la misma.

El módulo de Python "signal" de la librería *scipy* permite usar la Transformada de Fourier mediante la función *periodogram()*. Esta función genera el espectro de frecuencias de la serie temporal e indica aquellas con más peso en la serie.

Se ha usado la función sobre la serie temporal de Paseo de la Castellana y primero se ha establecido como unidad de ciclo el año para detectar posibles ciclos semanales o trimestrales. En la Figura 23 se aprecia un pico cercano a 0 ciclos/año y otro más pequeño sobre los 50 ciclos año. Ampliando la gráfica en el primer pico, Figura 24, se observa que este corresponde a 1 ciclo/año que hace referencia a la existencia de una estacionalidad anual. El segundo pico, Figura 25, se encuentra entre 52.1 y 52.2 ciclos/año, teniendo en cuenta que un año tiene de media 52.14 semanas contando entre años normales y bisiestos, y la serie temporal usada comprende ambos tipos de años, se puede concluir que la serie también presenta una estacionalidad semanal, aunque más débil que la anual.

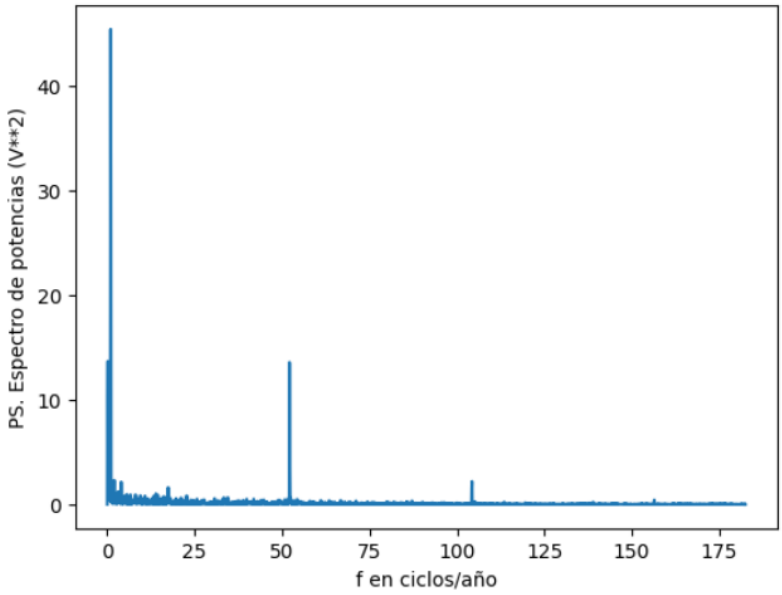


Figura 23: Gráfica de potencia por frecuencia de la serie temporal de Paseo de la Castellana.

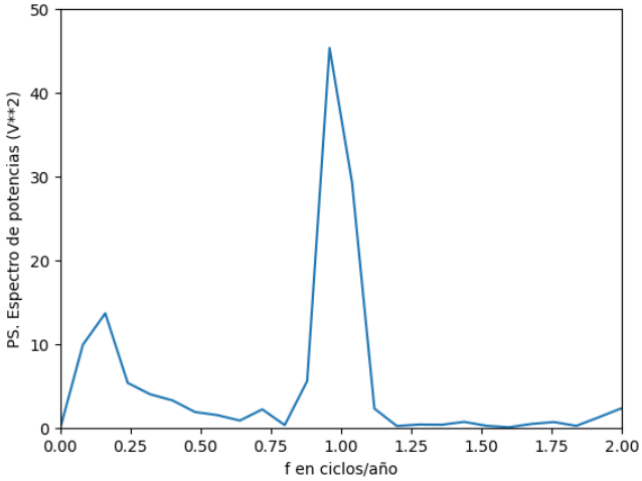


Figura 24: Gráfica primer pico de frecuencia en ciclos/año.

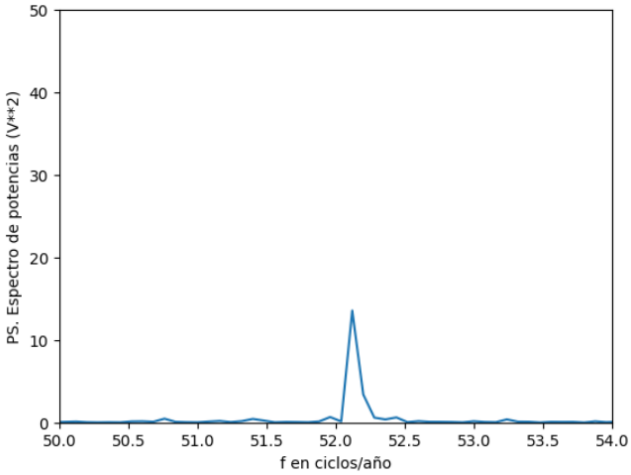


Figura 25: Gráfica segundo pico de frecuencia en ciclos/año.

Otra manera de mostrar los resultados es en periodo en lugar de frecuencia, ya que uno es el inverso del otro, $T = \frac{1}{f}$. De esta manera si en la función *periodogram()* se introduce como parámetro de unidad de ciclo el día, en vez del año como antes, se puede ver en la Figura 26 los números de días que comprende cada ciclo. Nuevamente ampliando la gráfica, Figura 27, se ve que el pico más elevado corresponde aproximadamente 380, prácticamente 1 año, y que el segundo pico corresponde exactamente a 7 días, una semana, Figura 28.

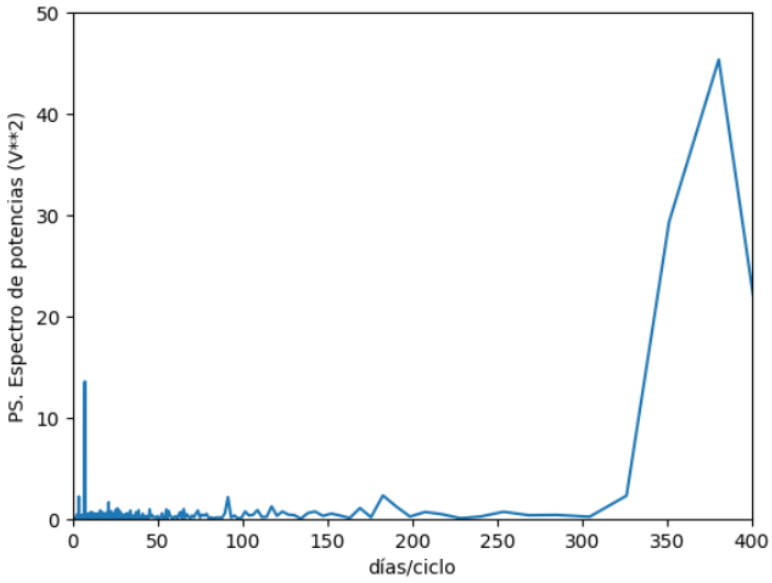


Figura 26: Gráfica de potencia por periodo de la serie temporal de Paseo de la Castellana.

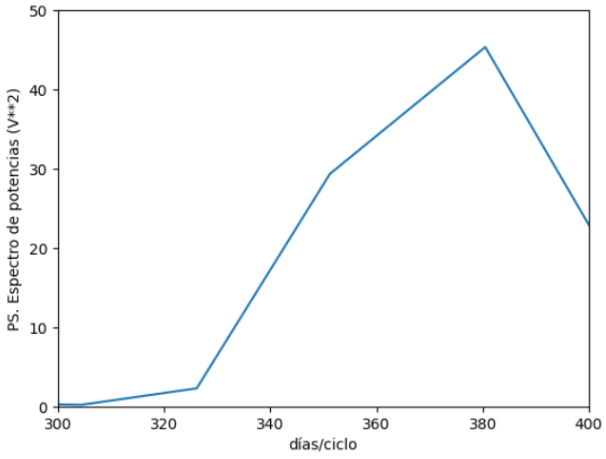


Figura 27: Gráfica pico más elevado de periodo días/ciclo.

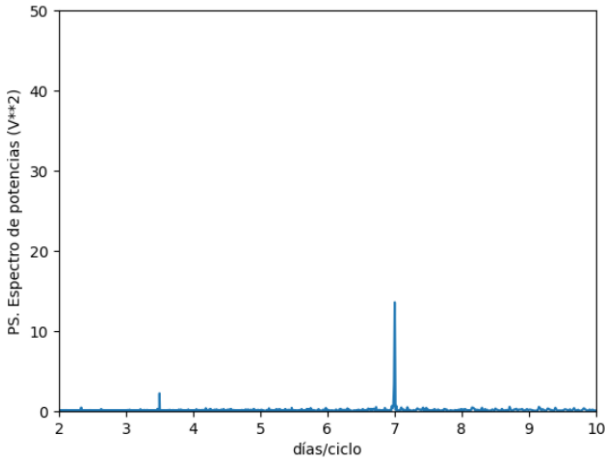


Figura 28: Gráfica segundo pico de periodo días/ciclo.

Tras realizar este análisis de la estacionalidad de la serie se decidió añadir como variables auxiliares al dataset la Transformada de Fourier en función de senos y cosenos representando ciclos anuales, ya que fue la frecuencia que obtuvo mayor potencia. Esto es posible realizarlo sencillamente usando la clase *Fourier* perteneciente a la librería *statsmodels.tsa.deterministic* de Python. Esta función tiene dos posibles parámetros, el periodo, que en este caso corresponde con 365 tiempos de periodo para completar un ciclo, y el orden, que indica el número de parejas de senos y cosenos que se generan, en este caso 2, ya que el periodograma dio unos 380 días [35]. Una vez obtenidas las parejas para el periodo seleccionado se extrapolan al dataset limpio de NO_2 usando la función *in_sample*, y se obtienen las curvas adaptadas a las fechas necesarias. En la Figura 29 se muestran dichas curvas adaptadas a los 4566 valores de NO_2 del dataset, y en la Figura 30 una ampliación de la imagen para las 500 primeras fechas. Por último, se añadió las curvas de Fourier al dataset que ya contaba con los atributos de meteorología para crear un nuevo dataset completo, Figura 31.

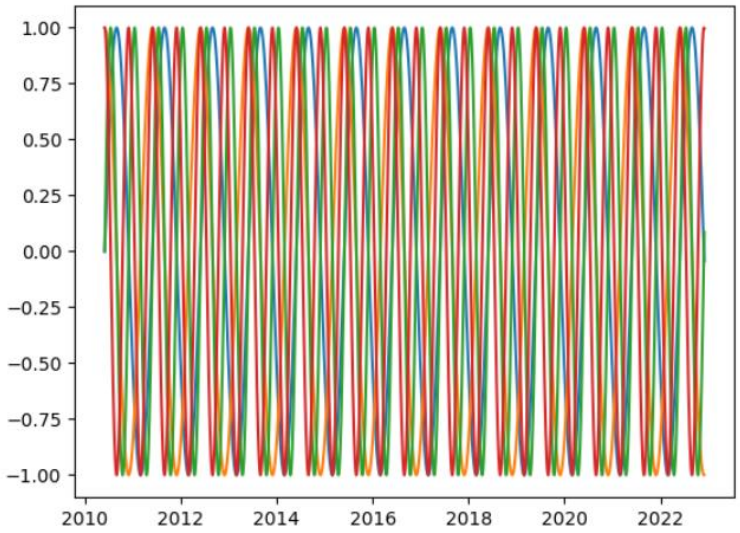


Figura 29: Gráfica de las curvas de sen y cos obtenidas con Fourier.

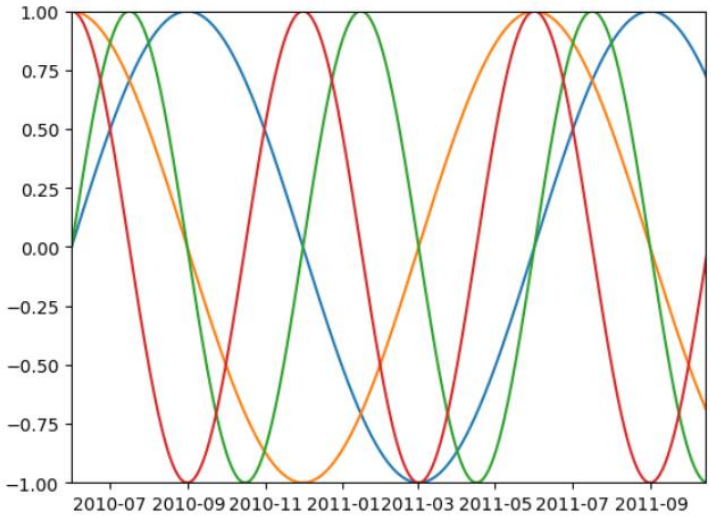


Figura 30: Ampliación de la gráfica de las curvas de sen y cos a las 500 primeras fechas.

DATE	NO2	TMEDIA	PRECIPITACION	VELMEDIA	sin(1,365)	cos(1,365)	sin(2,365)	cos(2,365)
2010-06-01	36.0	25.2	0.0	3.1	0.000000	1.000000	0.000000	1.000000
2010-06-02	33.0	25.2	0.0	1.4	0.017213	0.999852	0.034422	0.999407
2010-06-03	27.0	24.8	0.0	2.8	0.034422	0.999407	0.068802	0.997630
2010-06-04	40.0	26.0	0.0	1.9	0.051620	0.998667	0.103102	0.994671
2010-06-05	40.0	26.3	0.0	3.6	0.068802	0.997630	0.137279	0.990532
...	...	...	...	...	...	...	...	...
2022-11-26	45.0	8.7	0.0	1.1	0.025818	-0.999667	-0.051620	0.998667
2022-11-27	38.0	8.4	3.0	0.8	0.008607	-0.999963	-0.017213	0.999852
2022-11-28	28.0	8.8	0.0	5.3	-0.008607	-0.999963	0.017213	0.999852
2022-11-29	57.0	7.8	0.0	0.8	-0.025818	-0.999667	0.051620	0.998667
2022-11-30	67.0	8.0	0.0	0.6	-0.043022	-0.999074	0.085965	0.996298

4566 rows × 8 columns

Figura 31: Dataset de NO_2 y meteorología con las curvas de Fourier como atributos.

3.4. Modelos y experimentación

Para intentar obtener un modelo de red LSTM capaz de realizar la mejor predicción posible se decidió establecer tres datasets diferentes. El primero cuenta únicamente con una variable, el nivel de NO_2 medio diario medido en $\mu g/m^3$. El segundo posee como variables adicionales la temperatura, el viento y la precipitación media para cada día. Y el tercero añade al segundo dataset valores de seno y coseno que provienen de la transformada de Fourier para periodos de un año. De esta manera se cuenta con estos 3 datasets para la estación a estudiar:

1. NO_2. Figura 18.
2. NO_2, temperatura, viento, precipitación. Figura 19.
3. NO_2, temperatura, viento, precipitación, Transformada de Fourier. Figura 31.

Para cada uno de estos datasets se ha estudiado cuál es la arquitectura que consigue minimizar el MSE (Mean Squared Error), es decir, el error cuadrático medio de la red para cada uno de los horizontes propuestos en el sistema (1, 2, 3, 4 y 5 días vista). Este error se calcula tomando la diferencia entre cada valor predicho por el modelo y el valor real correspondiente, elevándola al cuadrado y luego tomando el promedio de todos ellos [42]. Al ser cuadrático el error siempre es positivo y cuanto más cercano a 0 sea más precisa será la predicción de la red. Este tipo de medida penaliza más los errores grandes debido a la operación de elevar al cuadrado, lo que significa que los errores más grandes tendrán un impacto mayor en el valor final del MSE. La fórmula es la siguiente:

$$MSE = \frac{1}{n}\sum(y_i - \hat{y}_i)^2$$

donde n es el número de observaciones, y_i es el valor real e $\hat{y}_i$ es el valor predicho por la red.

Para el entrenamiento de las redes se dividió el dataset de 4566 filas en 80% entrenamiento, 10% validación y 10% test. Por lo tanto, para cada uno de los entrenamientos se cuenta con los siguientes datasets:

1. Entrenamiento, 3652 filas
2. Validación, 457filas
3. Test, 457 filas

Para entender el funcionamiento de la red LSTM y alguno de los parámetros que se han modificado se presenta a continuación una explicación acompañada de imágenes del input de la red para el modelo con la configuración 1, es decir, con una solo variable de entrada, el valor del NO_2. La entrada de una red LSTM es una matriz en vez de un simple array, debido a que cada fila es una ventana de x días previos a la predicción del valor y no solo el valor anterior como sería si estuviéramos hablando de un array. Por lo tanto, para los datasets de entrenamiento, validación y test de este proyecto, y utilizando una ventana de 7 días como ejemplo para entrenar la red tendríamos una matriz de 3645x7, 449x7, 449x7 dimensiones para X y 3645x1, 449x1, 449x1 para Y respectivamente, Tabla 2. Siendo X los valores introducidos a la red e Y los valores a predecir, necesarios para el aprendizaje supervisado. De manera más gráfica se puede ver en la Figura 32 cuál sería la distribución real de la matrix *x_train* e *y_train* que se usa para entrenar la red, teniendo en cuenta que el valor x_{t-1} corresponde al último día de registro de NO_2 que hay en el dataset de *train* (entrenamiento).

Dataset	Dimensión
x_train	3645x7
y_train	3645x1
x_val	449x7
y_val	449x1
x_test	449x7
y_test	449x1

Tabla 2: Distribución de datasets y sus dimensiones para entrenar una red con una ventana de 7 días y 1 variable de entrada.

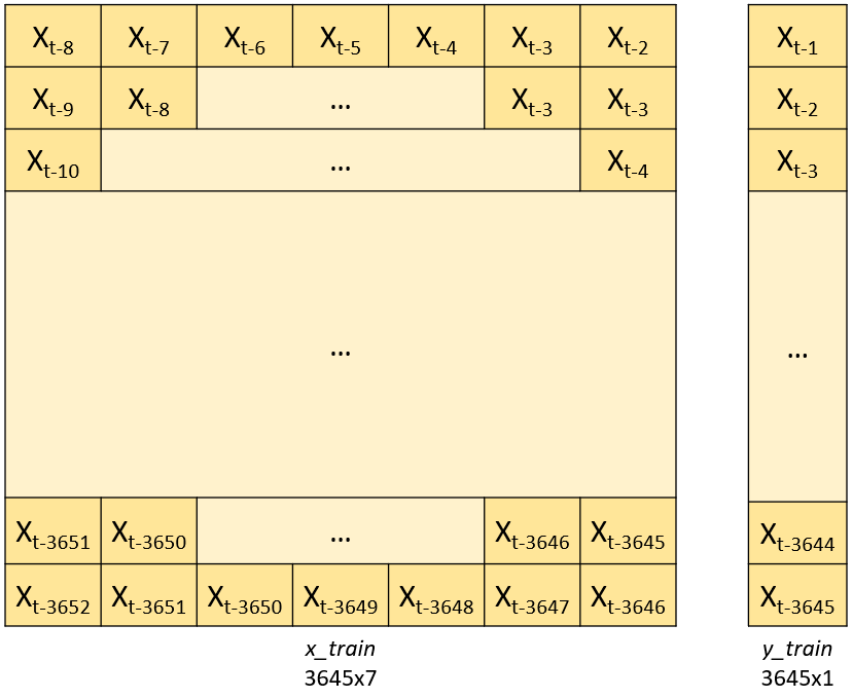


Figura 32: Matrices de los datasets *x_train* e *y_train* para ventana de 7 días y 1 variable de entrada.

En el caso de que hubiera más de una variable con la que entrenar la red, es decir más de una variable como input como es el caso de la configuración 2 que incluye los datos meteorológicos, la matriz de entrada varía un poco con respecto a la explicada anteriormente. Ahora las matrices poseen las dimensiones de la Tabla 3. Se añade una tercera dimensión que indica el número de matrices para cada entreno, haciendo referencia al número de variables de entrada que haya, en el caso de la configuración 2 son 4: NO_2, temperatura, viento y precipitación. En la Figura 33 se puede apreciar cómo es la configuración de las matrices. Esta misma estructura se aplica a la configuración 3, siendo el número de matrices 7.

Dataset	Dimensión
x_train	3645x7x4
y_train	3645x1x1
x_val	449x7x4
y_val	449x1x1
x_test	449x7x4
y_test	449x1x1

Tabla 3: Distribución de datasets y sus dimensiones para entrenar una red con una ventana de 7 días y 4 variables de entrada.

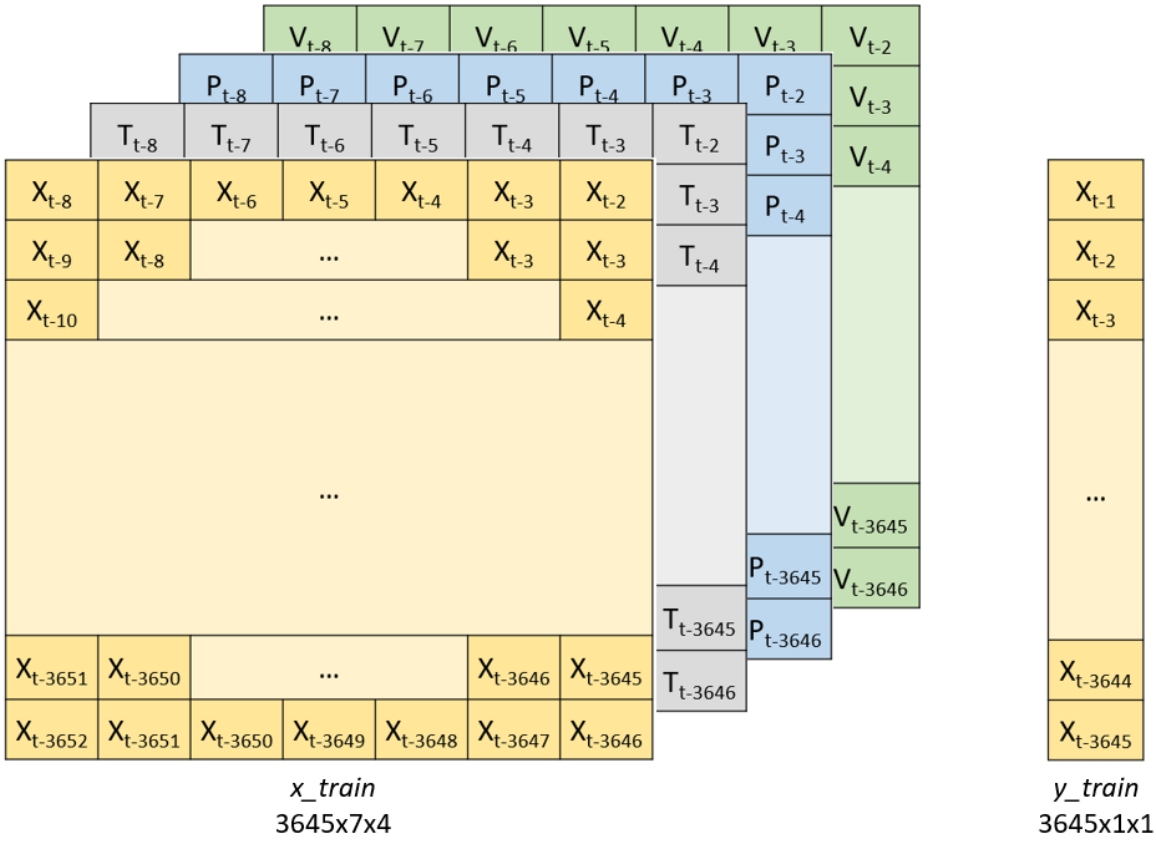


Figura 33: Matrices de los datasets *x_train* e *y_train* para ventana de 7 días y 4 variables de entrada.

Para realizar el entrenamiento y experimentar qué modelo da mejores resultados se han modificado diferentes hiperparámetros de la red LSTM mediante el método de *grid_search*. Este método consiste en probar todas las configuraciones posibles y guardar sus resultados en un fichero [43]. Los hiperparámetros modificados junto a los valores probados y la explicación de cada uno de ellos se muestran a continuación:

- **Ventana del historial** [7, 14, 30]: número de días en la ventana que usa la red para el entrenamiento de cada input.

- **Horizonte** [1, 2, 3, 4, 5]: número de días vista al cual se hace la predicción.
- **Patience** [3, 5, 7]: valor usado por la función *EarlyStopping* como límite. Dicha función hace que la red deje de entrenar cuando cierta métrica deja de mejorar [44]. En este caso la métrica a medir es el MSE.
- **Neuronas** [20, 60, 100, 140]: de la capa oculta de la red LSTM.
- **Batch_size** [4, 8, 16]: número de patrones que se propagan antes de que la red realice backpropagation para entrenar.

Con la evaluación de cada uno de estos hiperparámetros se han juntado los resultados de las tres configuraciones posibles (1, 2 y 3) y se van a mostrar a continuación los tres mejores resultados para predecir de 1 a 5 días vista. Se ha estudiado por separado el mejor modelo para predecir cada día vista, Tablas 4, 5, 6, 7 y 8, y por otro lado se ha estudiado el mejor modelo capaz de generar la predicción para los 5 días siguientes a la fecha, Tabla 9.

Conf.	Parámetros	Train_loss	Val_loss	Train_mae	Val_mae	epochs
2	ventana: 7 horizonte: 1 patience: 5 neuronas: 60 batch_size: 16	0.00554	**0.00342**	0.05713	0.04393	46
2	ventana: 7 horizonte: 1 patience: 3 neuronas: 20 batch_size: 16	0.00573	**0.00344**	0.05787	0.04378	44
2	ventana: 14 horizonte: 1 patience: 5 neuronas: 140 batch_size: 16	0.00624	**0.00344**	0.06061	0.04435	24

Tabla 4: Mejores configuraciones de red a 1 día vista.

Conf.	Parámetros	Train_loss	Val_loss	Train_mae	Val_mae	epochs
2	ventana: 7 horizonte: 2 patience: 3 neuronas: 60 batch_size: 16	0.00727	**0.00552**	0.06625	0.05416	75
2	ventana: 14 horizonte: 2 patience: 5 neuronas: 20 batch_size: 16	0.00728	**0.00552**	0.06633	0.05404	77
2	ventana: 7 horizonte: 2	0.00885	**0.00553**	0.07336	0.05289	8

	patience: 3 neuronas: 140 batch_size: 8					

Tabla 5: Mejores configuraciones de red a 2 días vista.

Conf.	Parámetros	Train_loss	Val_loss	Train_mae	Val_mae	epochs
2	ventana: 14 horizonte: 3 patience: 3 neuronas: 60 batch_size: 16	0.00786	**0.00649**	0.06888	0.05776	59
2	ventana: 7 horizonte: 3 patience: 3 neuronas: 60 batch_size: 8	0.00952	**0.00649**	0.07624	0.05573	6
2	ventana: 7 horizonte: 3 patience: 5 neuronas: 60 batch_size: 8	0.00927	**0.00652**	0.07527	0.05578	8

Tabla 6: Mejores configuraciones de red a 3 días vista.

Conf.	Parámetros	Train_loss	Val_loss	Train_mae	Val_mae	epochs
2	ventana: 7 horizonte: 4 patience: 3 neuronas: 60 batch_size: 8	0.00974	**0.00679**	0.07709	0.05656	6
2	ventana: 7 horizonte: 4 patience: 3 neuronas: 100 batch_size: 8	0.00988	**0.00687**	0.07762	0.05642	6
3	ventana: 7 horizonte: 4 patience: 3 neuronas: 20 batch_size: 8	0.00919	**0.00695**	0.07469	0.05754	18

Tabla 7: Mejores configuraciones de red a 4 días vista.

Conf.	Parámetros	Train_loss	Val_loss	Train_mae	Val_mae	epochs
2	ventana: 7 horizonte: 5 patience: 3 neuronas: 140 batch_size: 8	0.01020	**0.00709**	0.07894	0.05733	6
2	ventana: 7 horizonte: 5 patience: 3 neuronas: 100 batch_size: 8	0.00986	**0.00711**	0.07745	0.05713	6
2	ventana: 7 horizonte: 5 patience: 5 neuronas: 140 batch_size: 8	0.00979	**0.00713**	0.07717	0.05702	8

Tabla 8: Mejores configuraciones de red a 5 días vista.

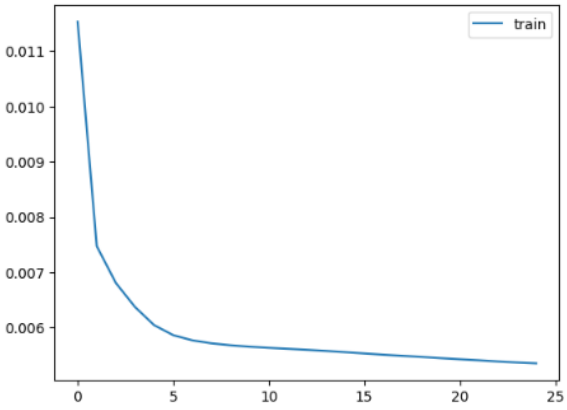
Conf.	Parámetros	Train_loss	Val_loss	Train_mae	Val_mae	epochs
2	ventana: 14 horizonte: 5 patience: 3 neuronas: 60 batch_size: 16	0.007708	**0.006093**	0.068079	0.055723	68
2	ventana: 14 horizonte: 5 patience: 5 neuronas: 60 batch_size: 16	0.007680	**0.006192**	0.067955	0.056156	73
2	ventana: 7 horizonte: 5 patience: 5 neuronas: 140 batch_size: 8	0.009235	**0.006215**	0.074801	0.054668	7

Tabla 9: Mejores configuraciones de red 1 a 5 días vista.

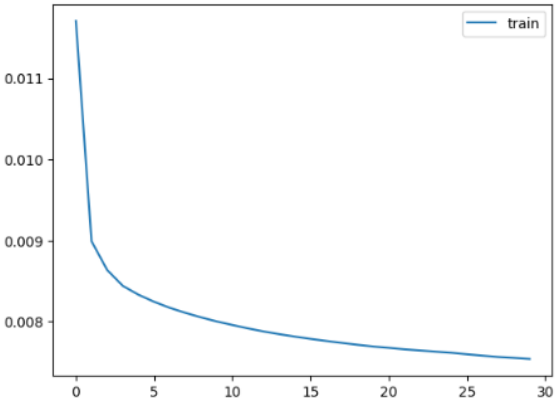
Tras obtener las mejores tres configuraciones de parámetros para cada día vista se decidió usar la primera, la que obtuvo el valor más bajo de la medida *validation_loss*, para generar los modelos definitivos. De estos resultados se observa que la configuración que mejor resultados ofrece es aquella que añade los valores meteorológicos, pero no las curvas de la Transformada de Fourier.

A continuación, se muestran los resultados de los modelos finales realizados para la estación de control de Paseo de la Castellana, Tablas 10 y 11. Estos modelos finales se realizaron usando un 90% del dataset inicial para entrenamiento y un 10% para test. Por lo tanto, de las 4200 filas iniciales se reparte de la siguiente manera el entramiento y test de los modelos finales:

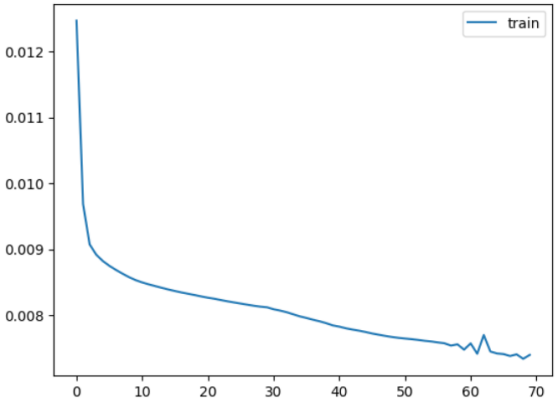
1. Entrenamiento, 4102 filas
2. Test, 457 filas



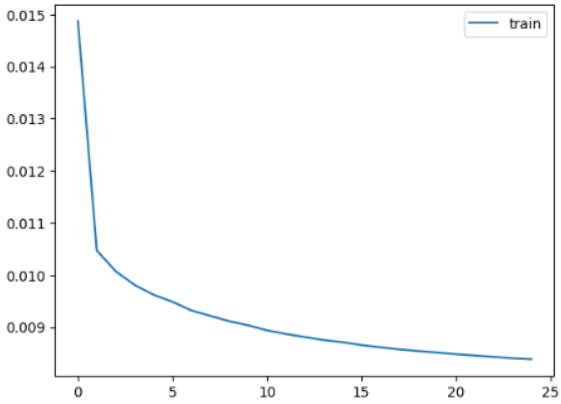
modelo 1 día vista



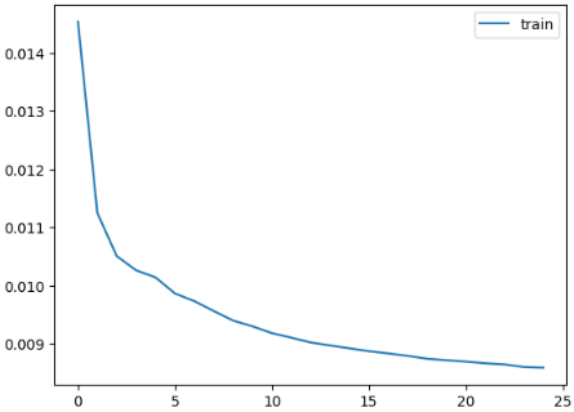
modelo 2 días vista



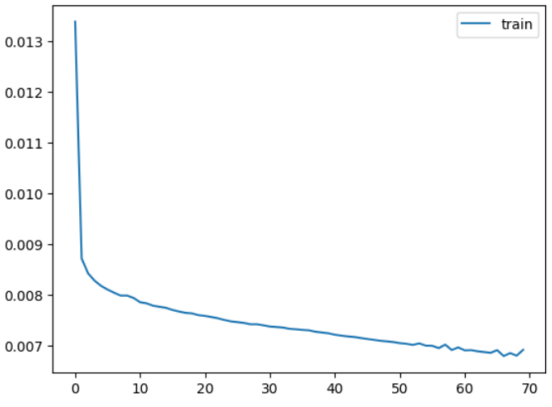
modelo 3 días vista



modelo 4 días vista



modelo 5 días vista



modelo 1-5 días vista

Tabla 10:Gráficas de evolución del MSE de los modelos finales para Paseo de la Castellana.

	Paseo de la Castellana		
Día vista	**Test_loss**	**RMSE**	**MAE**
1	0.003467	10.07	7.35
2	0.004535	11.516	9.003
3	0.004763	11.802	9.448
4	0.005013	12.107	9.355
5	0.005031	12.13	9.288
1-5	0.004558	11.546	8.918

Tabla 11: Resultados para el fichero de test de los modelos finales para Paseo de la Castellana.

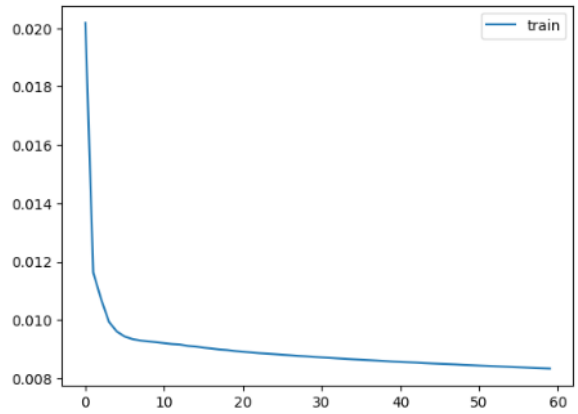
Para valorar el resultado de los modelos finales se usa la medida de precisión Root Mean Squared Error (RMSE), esta medida nos permite conocer el error medio que comete la red en las mismas unidades que las que tiene la serie temporal [45]. Por lo tanto, para el modelo de 1 día vista con valor de 10.07 indica que la red se confunde de media 10 puntos por predicción. Para poder comprobar si es más preciso usar los modelos entrenados específicamente para predecir cada día vista o si por lo contrario es más preciso el modelo que predice directamente los próximos cinco días, es necesario realizar la media de los valores de RMSE de los modelos específicos.

Media RMSE modelos específicos: 11.525

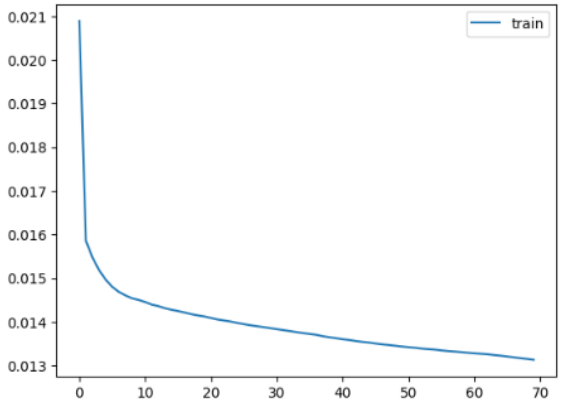
RMSE modelo 1-5 días vista: 11.546

La diferencia de error entre ambos es del orden del segundo decimal, por lo tanto, se puede concluir que no es relevante la diferencia. Aun así, como la media del error de los modelos específicos es menor, se han utilizado dichos modelos para realizar las predicciones.

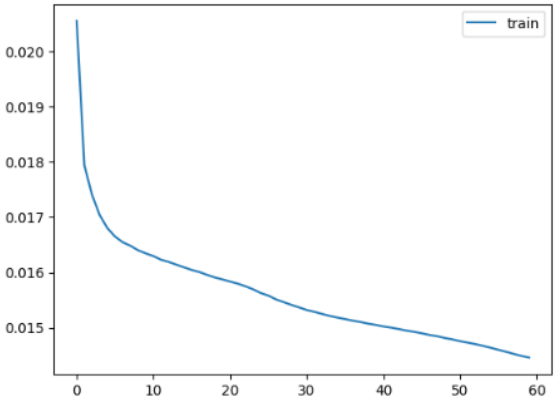
Teniendo en cuenta el estudio completo realizado para la estación de Paseo de la Castellana se ha decidido usar los mismos valores de configuración óptimos para obtener los modelos finales de las estaciones de Plaza del Carmen y Retiro obteniendo los siguientes valores de RMSE para los modelos creados, Tablas 12, 13, 14 y 15:



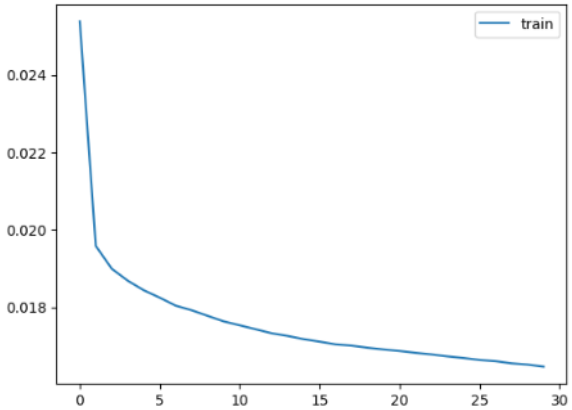
modelo 1 día vista



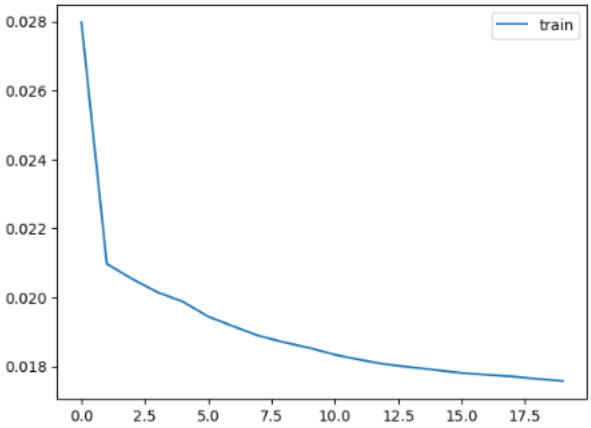
modelo 2 días vista



modelo 3 días vista



modelo 4 días vista

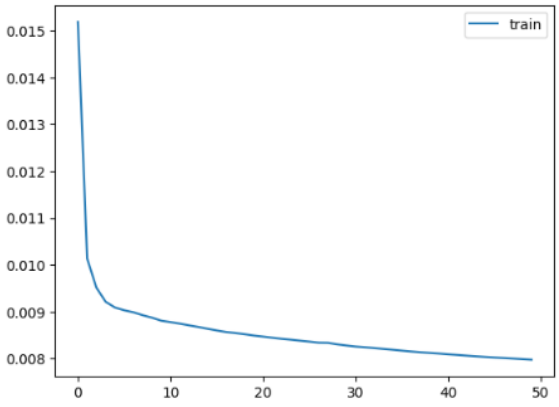


modelo 5 días vista

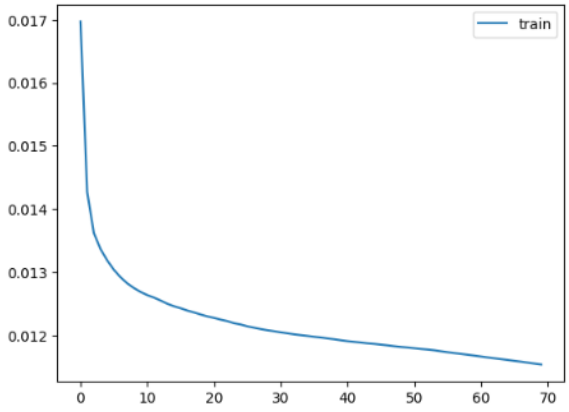
Tabla 12: Gráficas de evolución del MSE de los modelos finales para Plaza del Carmen.

	Plaza del Carmen		
Día vista	**Test_loss**	**RMSE**	**MAE**
1	0.005983	9.051	6.991
2	0.008667	10.893	8.717
3	0.009633	11.484	9.200
4	0.010633	12.065	9.532
5	0.010992	12.267	9.567

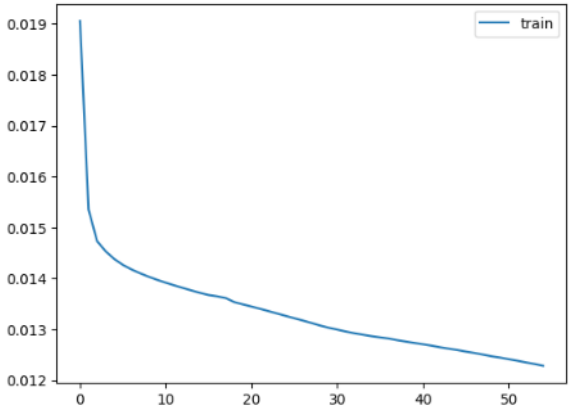
Tabla 13: Resultados para el fichero de los test de los modelos finales para Plaza del Carmen.



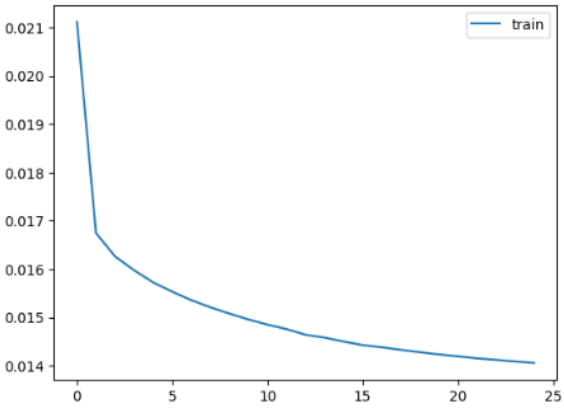
modelo 1 día vista



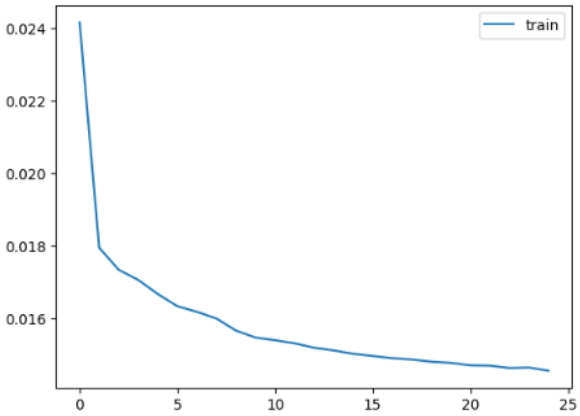
modelo 2 días vista



modelo 3 días vista



modelo 4 días vista



modelo 5 días vista

Tabla 14: Gráficas de evolución del MSE de los modelos finales para Retiro.

	Retiro		
Día vista	**Test_loss**	**RMSE**	**MAE**
1	0.005844	8.333	6.034
2	0.007681	9.553	7.284
3	0.008543	10.075	7.795
4	0.008814	10.234	7.746
5	0.008883	10.274	7.732

Tabla 15: Resultados para el fichero de test de los modelos finales para estación Retiro.

3.5. Evaluación de resultados

Para comprobar la precisión de los modelos finales se ha tomado como medida de evaluación el RMSE como se mencionó en el apartado anterior. No se ha usado el MAE ya que no contempla el error absoluto en las predicciones, como si lo hace el RMSE. Esta medida permite apreciar cuantas unidades posee de error el modelo en la misma unidad de medida que la que posee la variable a predecir, en este caso el NO_2. Para poder comparar dicho error primero es necesario entender los datos de NO_2, cómo se comportan a lo largo de la serie temporal y los rangos en los que sus niveles pueden ser perjudiciales para los seres humanos.

Para la estación de Paseo de la Castellana el valor mínimo que se detectó fue 1µg/m^3 y el máximo 173µg/m^3, siendo la media 37µg/m^3. En el caso de Plaza del Carmen el valor mínimo fue 1µg/m^3, el máximo 118µg/m^3 y la media 42µg/m^3. Por último, en la estación de Retiro el valor mínimo corresponde a 2µg/m^3, el máximo a 111µg/m^3 y la media a 29µg/m^3.

El RMSE de las estaciones ha sido el siguiente en función de los días vista a los que se ha realizado la predicción, Tabla 16:

Días vista	Paseo de la Castellana	Plaza del Carmen	Retiro
1	10.07	9.051	8.333
2	11.516	10.893	9.553
3	11.802	11.484	10.075
4	12.107	12.065	10.234
5	12.13	12.267	10.274
Media	11.525	11.152	9.694

Tabla 16: RMSE de los modelos finales.

Para todas las estaciones el error aumenta a medida que se realiza la predicción a más días vista debido a que es más complejo para la red realizar dicha predicción con cada vez menos valores próximos en el tiempo. El único día que esto no ocurre es en el quinto día vista de la estación de Retiro, y se puede deber a que los modelos óptimos han sido obtenidos a partir de la estación de Paseo de la Castellana y no de esta. Por otro lado, la estación que obtiene un mayor error de media es la estación Paseo de la Castellana a pesar de haber realizado el estudio sobre ella. Una posible causa de esto es que esta estación es la única de las tres que está colocada en una zona de tráfico, y teniendo en cuenta que la cantidad de tráfico no es una variable auxiliar en la red ha podido obtener peores resultados por este motivo.

De acuerdo con el CAQI (Índice Común de Calidad del Aire) el Ayuntamiento de Madrid ofrece como referencia de calidad de ciertos parámetros medioambientales el Índice de Calidad del Aire [46] que se realiza teniendo en cuenta los valores límites marcados por la legislación. Dentro de este índice los valores horarios para el NO_2 son los indicados es la Figura 34. El índice diario se realiza con el valor máximo alcanzado en las 24 horas del día usando el mismo índice.

Muy bueno	Bueno	Regular	Malo	Muy malo
0-50	51-100	101-200	201-400	>400

Figura 34: Índice de Calidad del Aire para el Dióxido de Nitrógeno (NO_2).

Teniendo en cuenta que los valores usados en este estudio han sido diarios, que corresponden a la media todos los valores horarios registrados en un día, la media de las tres estaciones usadas entra dentro del rango "Muy bueno" de calidad del aire.

El error medio de los modelos es de 11.525µg/m^3, 11.152µg/m^3 y 9.694µg/m^3 para las estaciones de Paseo de la Castellana, Plaza del Carmen y Retiro respectivamente. Si se tiene en cuenta que el valor máximo puede ser de 400µg/m^3 y el mínimo 0µg/m^3, los errores obtenidos son muy pequeños. Aun así, la precisión de los modelos depende de cómo se quieran utilizar los valores obtenidos. Si el usuario desea comprobar si los próximos días el nivel de calidad del aire va a ser Muy bueno, Bueno o Regular podrá obtener muy buena precisión, pero si por el contrario su intención es obtener una precisión de menos de los valores de error de la

Tabla 16, entonces haría falta realizar un análisis nuevo con nuevos atributos que intenten mejorar más los modelos.

En meteorología se suelen comparar los resultados de los modelos de predicción con lo que se llama “persistencia”, que consiste en tomar como predicción para el día siguiente el valor del día actual. Aunque parezca trivial, a veces es difícil ganar a la persistencia porque en ciertos periodos de tiempo se repite la misma situación meteorológica. Utilizando esta técnica, se pueden comparar los resultados obtenidos por los modelos con los resultados que se obtienen de desplazar los valores x días vista sobre la serie temporal, es decir, si la intención es predecir 1 día vista, basta con desplazar la serie temporal 1 día más a cada fecha y comprobar el error que se obtiene. A esta deviación se le denomina “lag” o “persistencia” y realizándolo en las tres estaciones para los 5 días vista se obtienen los siguientes resultados:

Días vista	Paseo de la Castellana	Plaza del Carmen	Retiro
1	11.559	10.136	9.723
2	14.427	13.095	12.050
3	15.840	14.475	12.940
4	16.223	15.354	13.525
5	16.453	15.892	13.864
Media	14.900	13.790	12.420

Tabla 17: RMSE usando lag.

Como se muestra en la Tabla 17, todos los valores de error son mayores que los conseguidos con los modelos de red LSMT, por lo tanto, aunque los resultados no sean tan bajos como lo deseado se ha conseguido mejorar la persistencia.

4. ANÁLISIS DEL SISTEMA

4.1. Especificación de requisitos

Para definir lo que debe hacer el sistema y cómo debe hacerlo se ha dividido la especificación en requisitos funcionales y no funcionales. Cada requisito estará definido usando una tabla con el siguiente formato, Tabla 18:

ID	
Nombre	
Descripción	

Tabla 18: Ejemplo de estructura de requisito.

A continuación, la definición de cada atributo de la tabla:

- **ID**: es el identificador único de cada requisito y su nomenclatura debe ser la siguiente:
  - R: indica que el elemento es un requisito.
  - F/NF: indica el tipo de requisito pudiendo este ser Funcional o No Funcional.
  - Número: indica la posición del requisito dentro de la lista de requisitos de cada tipo.
- **Nombre**: identifica de manera resumida al requisito.
- **Descripción**: explica el requisito de manera detallada.

4.1.1. Requisitos funcionales

ID	**RF-01**
Nombre	Visualización de estaciones de control
Descripción	El usuario podrá ver en un mapa las distintas estaciones de control existentes en el sistema con sus localizaciones en él y sus respectivos nombres.
ID	**RF-02**
Nombre	Selección de estación de control
Descripción	El usuario deberá poder elegir sobre que estación de control desea realizar la predicción de NO_2.
ID	**RF-03**
Nombre	Selección de fecha actual
Descripción	El usuario podrá elegir que fecha desea utilizar como fecha actual desde la cual realizar la predicción. Esta fecha podrá ser cualquier día entre el 14 de enero de 2023 y el 25 de junio de 2023.
ID	**RF-04**

Nombre	Gráfica de predicción
Descripción	La aplicación web devolverá al usuario una gráfica que contenga los cinco valores predichos y los catorce días previos de historial.
ID	**RF-05**
Nombre	Uso de modelos
Descripción	La aplicación podrá acceder a los modelos neuronales guardados para cada configuración existente.
ID	**RF-06**
Nombre	Uso de histórico de datos
Descripción	La aplicación podrá acceder a los datos de histórico de NO_2 y meteorológicos para introducirlos como input a los modelos de predicción.

Tabla 19: Requisitos funcionales del sistema.

4.1.2. Requisitos no funcionales

ID	**RNF-01**
Nombre	Disponibilidad
Descripción	La aplicación web podrá ser accedida mediante los navegadores Google Chrome, Firefox, Microsoft Edge y Safari, todos en sus dos versiones más recientes.
ID	**RNF-02**
Nombre	Diseño del gráfico de resultados
Descripción	La gráfica de resultados deberá mostrar en un color los valores reales y en otro los valores predichos, además deberá incorporar una separación visual que indique dónde empiezan los cinco días predichos.
ID	**RNF-03**
Nombre	Formato de los datos de entrada
Descripción	Los datos que se utilizan como input para los modelos deberán ser guardados en formato ".csv" para poder ser leídos correctamente.

Tabla 20: requisitos no funcionales del sistema.

4.2. Casos de uso

Para los casos de uso del sistema se ha planteado un diagrama en forma de grafo de estos, así como una descripción detallada de cada uno de ellos. También se ha creado una matriz de trazabilidad entre los casos de uso del sistema y los requisitos funcionales definidos anteriormente.

4.2.1. Diagrama de casos de uso

El siguiente diagrama, Figura 35, muestra las relaciones entre los casos de uso existentes en el sistema.

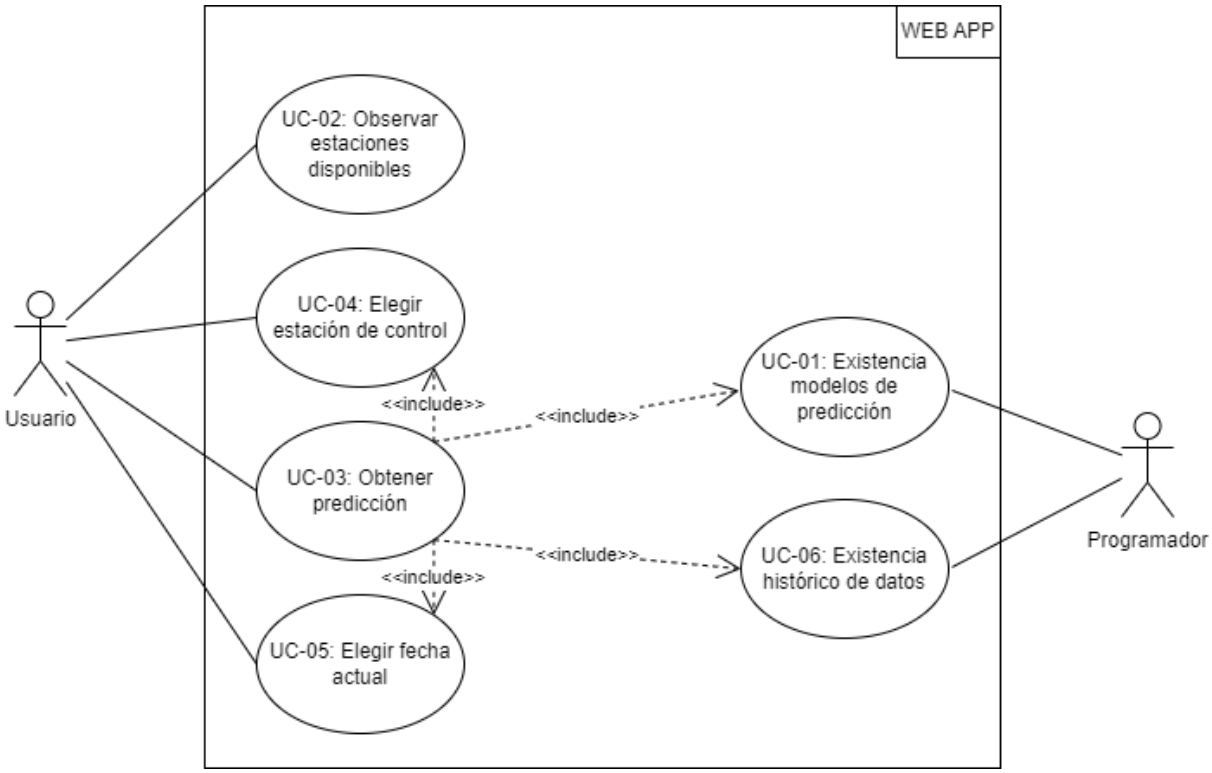


Figura 35: Diagrama de casos de uso del sistema.

4.2.2. Descripción de casos de uso de alto nivel

Para cada caso de uno se ha utilizado la siguiente tabla como estructura:

ID	
Nombre	
Actores	
Resumen	
Tipo	

Tabla 21: Ejemplo de estructura para la descripción de un caso de uso de alto nivel.

A continuación, la definición de cada atributo de la tabla:

- **ID**: identificador único del caso de uso y su nomenclatura de ser la siguiente:
  - UC: define que el elemento es un caso de uso.
  - Número: indica la posición del caso de uso dentro de la lista de todos los existentes.
- **Nombre**: identifica de manera resumida al caso de uso.
- **Actores**: define que actores participan.
- **Resumen**: descripción textual de la funcionalidad asociada al caso de uso mostrando a su vez los pasos que sigue el actor para realizar la operación.
- **Tipo**: pudiendo ser primario, secundario u optativo dependiendo la frecuencia de uso o nivel de importancia.

ID	**UC-01**
Nombre	Existencia modelos de predicción
Actores	Programador
Resumen	El programador tendrá que crear un modelo de red LSTM para cada estación y día vista, guardar el modelo en la carpeta *modelos* del repositorio siguiendo la nomenclatura "<nº estación>_modelo_<día>_dia.h5". También tendrá que añadirlo al diccionario que hay en el código de la aplicación web, con nombre de fichero *application.py*.
Tipo	Primario
ID	**UC-02**
Nombre	Observar estaciones disponibles
Actores	Usuario
Resumen	El usuario deberá entrar en la url de la aplicación web y observar la imagen del mapa de Madrid que muestra la localización de las estaciones y sus nombres.
Tipo	Secundario
ID	**UC-03**
Nombre	Obtener predicción
Actores	Usuario
Resumen	El usuario tendrá que acceder a la aplicación web, elegir una estación de control sobre la que realizar la predicción y la fecha actual. Por último, tendrá que pulsar el botón "Obtener predicción".

	Este caso de uso incluye los casos de uso UC-01, UC-04, UC-05 y UC-06 ya que los necesita para realizar la predicción.
Tipo	Primario
ID	**UC-04**
Nombre	Elegir estación de control
Actores	Usuario
Resumen	El usuario tendrá que entrar en la url de la aplicación web y en el desplegable de "Elija una estación de control" tendrá que elegir una de las estaciones disponibles sobre la cual realizar la predicción.
Tipo	Primario
ID	**UC-05**
Nombre	Elegir fecha actual
Actores	Usuario
Resumen	El usuario tendrá que entrar en la url de la aplicación web y en el desplegable de "Elija la fecha actual" tendrá que elegir una de las fechas disponibles desde la que realizar la predicción.
Tipo	Primario
ID	**UC-06**
Nombre	Existencia histórico de datos
Actores	Programador
Resumen	El programador tendrá que descargar y limpiar el histórico de NO_2 para las estaciones disponibles y el histórico meteorológico de la estación de Cuatro Vientos, guardarlos en la carpeta *data* del repositorio siguiendo la nomenclatura "aire_<nº estación>_data.csv" para las estaciones y "meteo_data.csv" para los datos meteorológicos. También tendrá que añadirlo al diccionario que hay en el código de la aplicación web, con nombre de fichero *application.py*.
Tipo	Primario

Tabla 22: Descripción de los casos de uso de alto nivel.

4.2.3. Matriz de trazabilidad

Se incluye a continuación la matriz de trazabilidad, la cual vincula los casos de uso del sistema con los requisitos funcionales definidos previamente en el apartado 4.1.1. Los pares de requisitos-casos de uso que poseen una relación directa entre sí están señalados con una "X".

	UC-01	**UC-02**	**UC-03**	**UC-04**	**UC-05**	**UC-06**
RF-01		X				
RF-02				X		
RF-03					X	
RF-04			X			
RF-05	X		X			
RF-06			X			X

Tabla 23: Matriz de trazabilidad.

5. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

Para que los modelos creados puedan ser usados se ha decidido crear una aplicación web para que los usuarios puedan interactuar con ella y obtener predicciones de la calidad del aire.

5.1. Modelos finales y su uso

Para poder usar los modelos finales y obtener una predicción es necesario introducir como input de la red los datos de NO_2, lluvia, viento y precipitación de 7 o 14 días previos dependiendo del modelo. El ayuntamiento de Madrid no ofrece datos en tiempo real de los últimos 14 días por lo tanto se descartó poder hacer predicciones en tiempo real. Se optó por utilizar los datos del año 2023 ya que no se usaron ni para entrenar ni como test de los modelos finales. Por lo tanto, el usuario puede elegir una fecha desde el día 14 de enero hasta el 25 de junio de 2023 y realizar la predicción sobre cualquiera de los días que se comprenden.

Una vez que el usuario elige la estación de control de aire y la fecha deseada, el programa de Python calcula los datos necesarios para los modelos y realiza la predicción de 5 días vista.

5.2. Aplicación web

La aplicación web ha sido creada usando Streamlit. Este framework es open source y permite crear aplicaciones web interactivas basadas en machine learning y datos usando como lenguaje de programación Python [30]. Como el proyecto desarrollado son modelos de machine learning y necesitan que el usuario seleccione ciertos parámetros, Streamlit se convierte en un framework perfecto para realizar un desarrollo rápido y sencillo de la aplicación web.

El diagrama de sistema cliente-servidor corresponde al de la Figura 36 y comprende los siguientes bloques o elementos:

- **Navegación usuario**: Este es el navegador web que el usuario utiliza para acceder a la aplicación. El usuario interactúa con la interfaz de usuario generada por Streamlit y envía solicitudes al servidor.
- **Servidor (Streamlit App)**: En este servidor se ejecuta la aplicación web desarrollada con Streamlit. La aplicación procesa las solicitudes del usuario, realiza cálculos y renderiza la interfaz de usuario dinámica. El código de la aplicación se encuentra en el repositorio de GitHub.
- **Repositorio GitHub**: El repositorio de GitHub contiene tanto el código fuente de la aplicación web como los datos y los modelos de machine learning necesarios para su funcionamiento. Esto incluye archivos como conjuntos de datos, configuraciones y cualquier otro recurso necesario para la aplicación y los modelos.

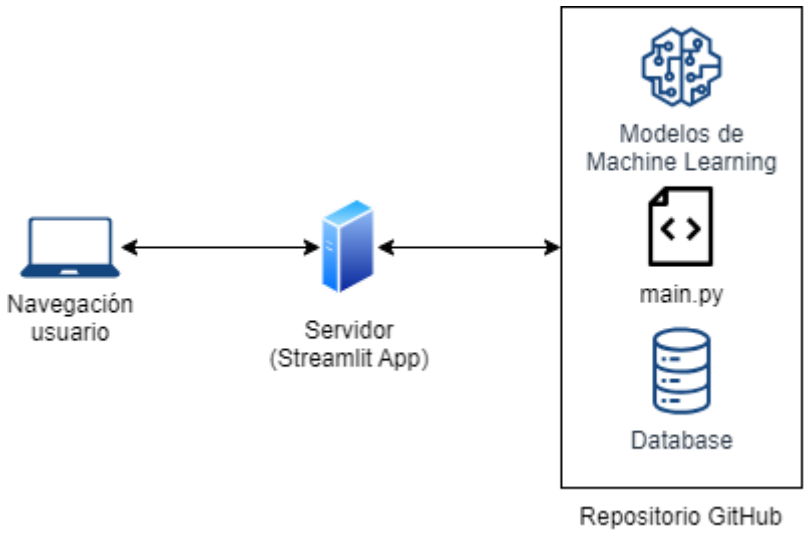


Figura 36: Diagrama de sistema cliente-servidor

La interfaz de la aplicación se ha diseñado para que comprenda todos los elementos necesarios en una sola página haciendo de esta manera que la aplicación sea muy sencilla y fácil de usar por el usuario. Comprende los siguientes elementos que se pueden ver en la Figura 37:

1. Mapa orientativo de las estaciones de control de calidad del aire disponibles con sus respectivos nombres.
2. Desplegable para elegir la estación de control, Figura 38. El valor por defecto es “Paseo de la Castellana”.
3. Desplegable mensual para elegir la fecha que simula el día actual desde el cual realizar la predicción, Figura 39. El valor por defecto es 2023/01/14.
4. Botón de realizar la predicción de los próximos 5 días.

Figura 37: Interfaz principal de la aplicación web.

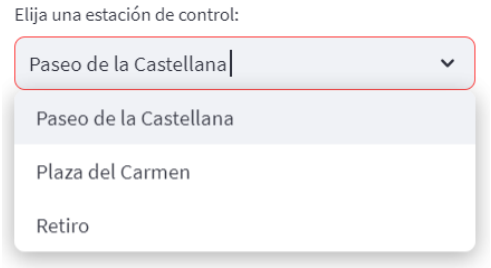


Figura 38: Desplegable estación de control.

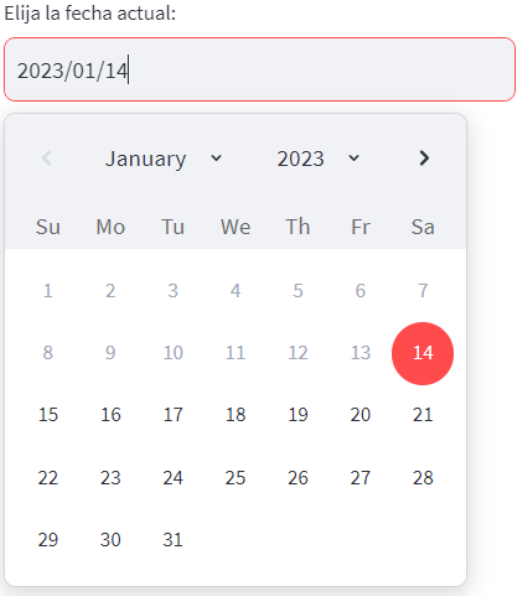


Figura 39: Desplegable fecha actual.

Cuando el usuario presiona el botón de “Obtener predicción” se realizan los cálculos internos necesarios y se usan los modelos finales para proporcionar la predicción en forma de gráfico. El gráfico que se muestra en pantalla muestra en color azul el nivel de NO_2 de los últimos 14 días registrados y el valor real de los 5 días vista. En naranja se muestran los valores que se han predicho por los modelos finales con un relleno también naranja entre el valor real y el predicho. Además, la gráfica cuenta con una línea discontinua gris que separa los días previos de los días vista predichos, y también con una indicación de en qué índice de calidad del aire se encuentran los valores de NO_2, siendo “Muy bueno” y “Bueno” la franja verde, “Regular” la franja naranja y “Malo” la roja. Dicho índice se base en la Figura 34 explicada en el punto 3.5, y se muestra para los usuarios justo debajo del botón de “Obtener predicción”. En resumen, los elementos que aparecen tras presionar el botón de “Obtener predicción” son los siguientes que se pueden ver también en la Figura 40:

5. Índice de Calidad del Aire para el NO_2.
6. Gráfica

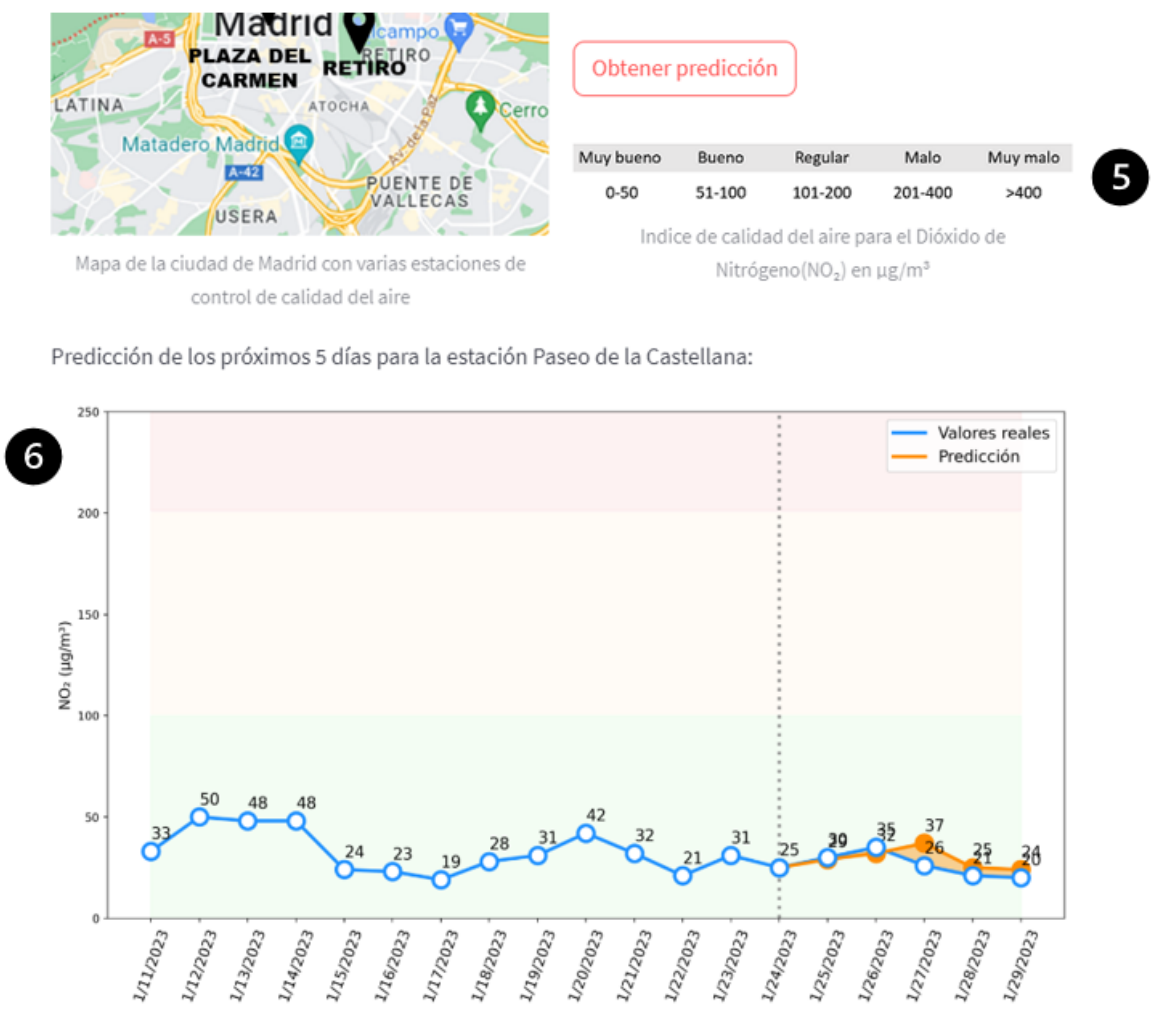


Figura 40: Interfaz de predicción de la aplicación web.

5.3. Deployment y URL

Para que la aplicación pueda usarla todo el mundo hay que desplegarla y crear una URL de acceso. Todo esto se puede realizar también de manera muy sencilla y gratis gracias a Streamlit Community Cloud [47] que forma parte de Streamlit.

Para ello se ha depositado todo el código de la aplicación en este repositorio público de GitHub. Dicho repositorio cuenta con el archivo principal de la aplicación llamado *application.py*, una carpeta llamada *modelos* que almacena los modelos finales necesarios para realizar las predicciones de las tres estaciones, otra carpeta llamada *data* que guarda las imágenes usadas y los archivos de datos de NO_2 (*aire_35_data.csv*, *aire_48_data.csv* y *aire_49_data.csv*) y meteorológicos (*meteo_data.csv*), y por último, el archivo *requirements.txt* que guarda el nombre y la versión de los módulos usados en el código de la aplicación web, necesario para realizar el despliegue.

Una vez creado el repositorio se realizó el despliegue con Streamlit Community Cloud y se creó la URL prediccion-aire-madrid-victoria.streamlit.app desde la cual se puede acceder y utilizar la aplicación web.

6. PLANIFICACIÓN Y ENTORNO SOCIOECONÓMICO

En este capítulo se va a explicar la planificación que ha seguido el proyecto, se enseñará el diagrama de Gantt correspondiente además del presupuesto y el coste total, y por último una explicación del entorno socioeconómico del mismo.

6.1. Planificación

Con respecto a la planificación del proyecto, se identificaron las tareas a realizar y se dividieron en distintas fases que abordar para la elaboración del proyecto. El proyecto ha comprendido un total de 114 días laborables, se ha trabajado de lunes a viernes, excluyendo fines de semana, por lo tanto, ha tenido un total de 5 meses de duración. Las distintas fases juntos con sus correspondientes tareas, la fecha de inicio y de fin de estas, más el número de días trabajados se contempla en la Tabla 24.

	Tarea	Inicio	Fin	Días laborales
Fase 1	Investigación sobre el tema de estudio	3/04/2023	7/04/2023	7
	Estudio de proyectos similar	10/11/2023	11/04/2023	
Fase 2	Estudio del estado del arte	12/04/2023	25/04/2023	10
Fase 3	Búsqueda de datos necesarios	26/04/2023	27/04/2023	20
	Limpieza de datos	28/04/2023	11/05/2023	
	Análisis de la serie temporal	12/05/2023	16/05/2023	
Fase 4	Experimentación de modelos de redes LSTM	17/05/2023	13/06/2023	30
	Evaluación de los resultados	14/06/2023	20/06/2023	
	Generación de modelos finales	21/06/2023	27/06/2023	
Fase 5	Creación de la aplicación web	28/06/2023	11/07/2023	12
	Implementación y despliegue de la aplicación web	12/07/2023	13/07/2023	
Fase 6	Redacción de la memoria	14/07/2023	25/08/2023	31
Fase 7	Revisión de la memoria	28/08/2023	31/08/2023	4

Tabla 24: Planificación del proyecto dividido en fases y tareas.

Como soporte visual de la planificación del proyecto de la Tabla 24 se agrega el diagrama de Gantt en la Figura 41.

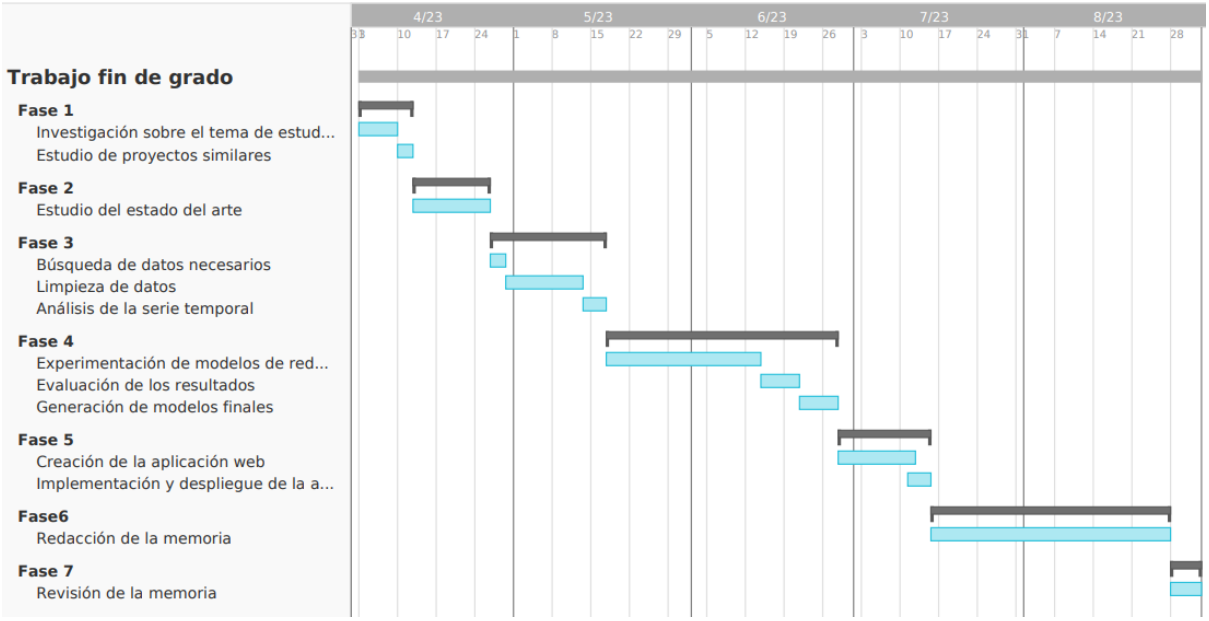


Figura 41: Diagrama de Gantt.

6.2. Presupuesto

Este apartado recoge la estimación del coste total de proyecto. Para calcular el presupuesto se han tenido en cuenta costes de personal, de hardware, software y costes indirectos que se explican en profundidad a continuación. El Impuesto de Valor Añadido (IVA) del 21% se añadirá en el apartado de coste total.

6.2.1. Costes de personal

La planificación del proyecto ha estipulado que va a tener una duración total de 114 días laborales. El personal que trabaja en este proyecto es un ingeniero informático, en este caso la alumna, y un jefe de proyecto, el tutor. El horario del ingeniero informático es una jornada parcial de 4 horas diarias para poder compaginarlo con su horario académico y laboral. En la Tabla 25 se observa las horas trabajadas, los días y horas totales, y el coste total incluyendo la Seguridad Social (SS) del 30%.

Empleado	Coste/hora	Días totales	Horas totales	Coste total	Coste total (con SS)
Jefe de proyecto	32 €	7	28	896 €	1.164,80 €
Ingeniero informático	20 €	114	456	9.120 €	11.856 €
Total					13.020,80 €

Tabla 25: Costes de personal.

6.2.2. Costes de hardware

Para la posible realización de este proyecto ha sido necesario hacer uso de un equipo informático completo. En específico de un portátil, un monitor y un ratón para poder realizar cada una de las tareas del proyecto. Como ya se contaba con todo el equipo necesario el valor viene dado de aplicar el periodo de amortización sobre el tiempo de uso en el proyecto, que fue de 5 meses. Usando la siguiente fórmula se obtiene el coste total de cada equipo:

$$Total\ equipo = Precio\ equipo * \frac{Tiempo\ proyecto\ (meses)}{Periodo\ de\ vida\ (año) * 12\ (\frac{meses}{año})}$$

Equipo	Precio/unidad (con IVA)	Periodo de vida	Coste total (5 meses)
Xiaomi Mi Laptop Aire 13.3”	646,19 €	4 años	67,32 €
Monitor LG M2232D-PZ	120 €	5 años	10 €
Ratón Zelotes T-60	13,99 €	4 años	1,46 €
Total			78,78 €

Tabla 26: Costes de hardware.

6.2.3. Costes de software

Para la realización del proyecto también han hecho falta ciertas herramientas de software, como un editor de código, programas relacionados con aprendizaje automático, un framework de aplicaciones web y también software de redacción de documentos para la realización de la presente memoria. Los costes totales de esta sección han sido de 0 € debido a la gratuidad de los programas usados. Aun así, se pueden ver en la Tabla 27.

Herramienta	Coste total
Visual Studio Code	0 €
Google Colab	0 €
Microsoft Office 365 Education	0 €
Streamlit	0 €
Total	0 €

Tabla 27: Costes de software.

6.2.4. Costes indirectos

El proyecto se ha realizado online por lo tanto hay que tener en cuenta los costes indirectos correspondientes a un servicio mensual de fibra óptica y la factura de la luz.

Descripción	Coste/mes (con IVA)	Coste total (5 meses)
Internet Fibra óptica 1Gb	60 €	300 €
Factura de la luz/1 persona	32 €	160 €
Total		460 €

Tabla 28: Costes indirectos.

6.2.5. Coste total estimado

Una vez realizado todos los costes tanto directos como indirectos se calcula el coste total estimado del proyecto en la Tabla 29. En dicha tabla se incluye el IVA del 21%, un margen de beneficio de 10% para la venta del proyecto y un margen de riesgo del 4%.

Descripción	Coste
Costes de personal	13.020,80 €
Costes de Hardware	78,78 €
Costes de Software	0 €
Costes indirectos	460 €
Total	**13.559,58 €**
% margen beneficio	10%
% margen error	5%
Precio de venta	**15.593,52 €**
Precio de venta con IVA (21%)	**18.868,16 €**

Tabla 29: Coste total estimado.

El coste de venta del proyecto es de 15.593,52 euros, y si se le añade el IVA es de 18.868,16 euros.

6.3. Entorno socioeconómico

La predicción de series temporales es un campo que se lleva estudiando y desarrollando varios años dentro del mundo de la estadística y más recientemente en el del machine learning gracias a las redes neuronales recurrentes. El poder de las redes LSTM y su capacidad de aprender en secuencia han hecho posible que se usen para muchas aplicaciones, siendo algunas de ellas las siguientes:

- Procesamiento del lenguaje natural (NLP): traducción automática y generación de texto.
- Reconocimiento de voz: transcripción de audios.
- Análisis de sentimientos: clasificación de opiniones.
- Pronóstico y series temporales: predicción del clima, de la contaminación del aire y de ventas.
- Salud y medicina: Análisis de señales y diagnósticos médicos.
- Video y procesamiento de imágenes: Etiquetado de objetos y generación de subtítulos en vídeos.

Para el correcto funcionamiento de estas redes hace falta una infraestructura tecnológica adecuada, incluyendo servidores y una gran capacidad de procesamiento y almacenaje de datos. Es muy importante contar con acceso a datos precisos, actualizados y completos si se quiere realizar un estudio con estas redes. En este caso el ayuntamiento de Madrid cuenta con una base de datos con acceso gratuito a la población donde se pueden encontrar datos de las diferentes estaciones de control de calidad del aire en la ciudad los cuales han sido la base de estudio de este proyecto.

Este proyecto podría tener un impacto en la salud pública ya que la contaminación del aire está asociada con una gran variedad de problemas de salud incluyendo enfermedades respiratorias y cardiovasculares. Por lo tanto, una predicción de estos valores podría contribuir a la toma de decisiones para proteger la salud pública y reducir los riesgos para los ciudadanos.

También cabe destacar que un proyecto como este podría aumentar la conciencia pública sobre el problema mundial que supone la contaminación del aire y a su vez motivar a los ciudadanos a tomar acciones para reducir la misma gracias a proporcionarles información transparente de fácil acceso.

Reducir los niveles de contaminación podría suponer impactos económicos positivos a largo plazo. Una mejor calidad del aire podría disminuir a los costes asociados con la atención médica y las bajas laborales por enfermedades relacionadas directamente con la contaminación del aire. Además, que una reducción de la contaminación puede aumentar el atractivo para la inversión en la ciudad y para su turismo.

Como parte negativa del proyecto cabe comentar que los resultados obtenidos son mejorables, sería necesario un plan de financiamiento sólido y una fuente sostenible de ingresos para garantizar la mejora y la continuidad de este a largo plazo.

7. CONCLUSIONES Y TRABAJOS FUTUROS

En este apartado de la memoria se van a presentar las conclusiones del proyecto y los trabajos futuros que se podrían realizar tomando como referencia este proyecto.

7.1. Conclusiones

Tras la realización del proyecto se puede concluir que se han cumplido los objetivos principales explicados en el apartado 1.2 de esta memoria. Se ha realizado un aprendizaje sobre la situación del aire en la ciudad de Madrid, así como se ha aprendido sobre los efectos que tiene sobre las personas y el planeta unos niveles de contaminación altos. Se ha conseguido obtener y limpiar satisfactoriamente datos históricos de dominio público para poder usarlos en la realización del trabajo. Se ha estudiado en profundidad la serie temporal del problema y se han implementado atributos a las redes LSTM en función de los resultados de dicho análisis. El trabajo principal de este proyecto ha sido el aprendizaje sobre las redes LSTM y la creación de modelos de predicción basados en ellas, que ha resultado ser satisfactorio también. Además, se han analizado los modelos de predicción y se han optimizado mediante la variación de sus hiperparámetros. Por último, en torno a la creación de una interfaz usable por los usuarios cabe destacar que también se ha conseguido mediante el desarrollo de una aplicación web funcional.

Los resultados obtenidos en los modelos de predicción son bastante buenos si los comparamos con el margen de ruido que presentaba la serie temporal en el apartado 3.3, siendo estos valores de entre 30-40 unidades por día y el error generado ronda las unidades. Aun así los resultados son mejorables, los modelos generados no son capaces de predecir correctamente los picos de contaminación, aunque el error medio no es alto.

Con respecto a la utilidad de la aplicación web, no se ha podido usar para el propósito inicial, que era que se pudiese realizar la predicción en tiempo real, es decir que los usuarios pudiesen generar la predicción desde la fecha actual del día que se conectan a la web. Esto se ha debido a que el Ayuntamiento de Madrid no publica los datos de calidad del aire en tiempo real de los 14 días previos, sino solo del día actual en formato horario. Por lo tanto, la solución del problema queda fuera del alcance del programador. Aun así se ha planteado una solución útil, la aplicación web permite a los usuarios realizar la predicción sobre cual día comprendido entre las fechas 14/01/2023 y 25/06/2023.

7.2. Conclusiones personales

A título personal me gustaría comentar que estoy muy contenta con el resultado del proyecto. Elegí este tema porque siempre he estado muy concienciada con el cambio climático y con la calidad del aire que respiramos los habitantes de la ciudad de Madrid y pensé que era la mejor manera de poder aprender de manera simultánea el uso de redes LSTM en problemas de series temporales. Entré en el proyecto no sabiendo nada sobre estas redes y lo he acabado

aprendiendo sobre una nueva área del machine learning y aportando mi granito de arena para combatir la contaminación del aire.

7.3. Trabajos futuros

Todas las decisiones tomadas a lo largo de este proyecto han afectado a los resultados de este, por lo que hay aspectos que se pueden mejorar o implementar para conseguir mejores resultados. A continuación, se presentan ciertas ideas para realizar trabajos futuros que se basan en lo aprendido a lo largo de este proyecto.

- **Elección de variables auxiliares**: para este proyecto se han usado como variables auxiliares datos meteorológicos de viento, precipitación y temperatura, y senos y cosenos provenientes de la Transformada de Fourier. Aun así, hay muchas más variables que podrían ayudar a realizar una mejor predicción como pueden ser los datos de tráfico, el valor del día de la semana, el calendario laboral o el número de personas usando transporte público.

- **Realización del estudio sobre un dataset inicial con más datos**: el dataset usado cuenta con datos solo desde 2010. Si el estudio se hiciese sobre estaciones de calidad de aire de otra ciudad que posea un histórico mucho más antiguo se podría entrenar los modelos con más datos, lo cual siempre favorece al resultado final.

- **Probar con redes convolucionales**: aunque estas redes usan principalmente para tratar imágenes, también pueden extraer tendencias y estacionalidad por lo que también se podría comparar el uso de estas redes con las usadas en este proyecto, las redes LSTM.

- **Utilización de validación cruzada en el entrenamiento**: esta técnica que consiste en dividir el set de entrenamiento y validación en diferentes subsets puede ayudar a que la red no sufra "overfitting" o sobre aprendizaje en el entrenamiento.

- **Probar con más combinaciones de hiperparámetros**: en este proyecto se probaron ciertas unidades para los hiperparámetros ajustables, pero se puede probar un rango mayor de unidades por cada uno de ellos.

- **Realizar un estudio individual de las estaciones**: en este proyecto para simplificar y por falta de tiempo, se realizó el estudio sobre una sola estación, Paseo de la Castellana y la configuración óptima generada se extendió a dos estaciones más sin estudiarlas individualmente. Para obtener resultados óptimos para cada estación habría que realizar el análisis a cada una de ellas.

- **Extender la aplicación web a más estaciones**: el proyecto solo cuenta con 3 estaciones, pero la red de estaciones de la ciudad de Madrid es mucho más grande, así que se podría expandir a todas ellas en un futuro.

BIBLIOGRAFÍA

[1] "Las nuevas Directrices mundiales de la OMS sobre la calidad del aire tienen como objetivo evitar millones de muertes debidas a la contaminación del aire". World Health Organization. https://www.who.int/es/news/item/22-09-2021-new-who-global-air-quality-guidelines-aim-to-save-millions-of-lives-from-air-pollution (acceso: 2 de agosto de 2023).

[2] "Contaminación atmosférica". World Health Organization. https://www.who.int/es/health-topics/air-pollution (acceso: 2 de agosto de 2023).

[3] "Normativa del Ayuntamiento de Madrid sobre reutilización". Portal de datos abiertos del Ayuntamiento de Madrid. https://datos.madrid.es/portal/site/egob/menuitem.3efdb29b813ad8241e830cc2a8a409a0/?vgnextoid=9023234be7924710VgnVCM2000001f4a900aRCRD&vgnextchannel=830512b9ace9f310VgnVCM100000171f5a0aRCRD&vgnextfmt=default (acceso: 5 de agosto de 2023).

[4] "Acuerdo de 10 de diciembre de 2018 de la Junta de Gobierno de la Ciudad de Madrid, por el que se aprueba definitivamente el Protocolo de Actuación para Episodios de Contaminación por Dióxido de Nitrógeno en la Ciudad de Madrid - SEDE ELECTRÓNICA". https://sede.madrid.es/portal/site/tramites/menuitem.5dd4485239c96e10f7a72106a8a409a0/?vgnextoid=fae7e00548f26610VgnVCM1000001d4a900aRCRD&vgnextchannel=6b3d814231ede410VgnVCM1000000b205a0aRCRD&vgnextfmt=default (acceso: 5 de agosto de 2023).

[5] "Áreas de Prioridad Residencial APR (Información histórica)". Portal de datos abiertos del Ayuntamiento de Madrid. https://datos.madrid.es/portal/site/egob/menuitem.c05c1f754a33a9fbe4b2e4b284f1a5a0/?vgnextoid=fd6dc7a31bbc3510VgnVCM1000001d4a900aRCRD&vgnextchannel=374512b9ace9f310VgnVCM100000171f5a0aRCRD&vgnextfmt=default (acceso: 5 de agosto de 2023).

[6] Resolución de la creación de los Distintivos Ambientales: *Ministerio del Interior*. 21 de abril de 2016.

[7] "Plan de calidad del aire y cambio climático. Plan A". Portal de transparencia del Ayuntamiento de Madrid. https://transparencia.madrid.es/portales/transparencia/es/Transparencia-por-sectores/Medio-ambiente/Aire/Plan-de-calidad-del-aire-y-cambio-climatico-Plan-A-2017-2020/?vgnextoid=fab664457127f510VgnVCM1000001d4a900aRCRD&vgnextchannel=33d9508929a56510VgnVCM1000008a4a900aRCRD (acceso: 5 de agosto de 2023).

[8] S. de M. A. y R. Naturales. "La atmósfera, esencial para el mantenimiento de la vida", *gob.mx*. http://www.gob.mx/semarnat/articulos/atmosfera (acceso: 17 de abril de 2023).

[9] E. Martínez Ataz, Contaminación atmosférica. Cuenca: Ediciones de la Universidad de Castilla-La Mancha, 2004. [En Línea] Disponible en: https://elibro.net/es/ereader/uc3m/54954?page=33

[10] “Calidad del aire y salud”, Comunidad de Madrid, 11 de abril de 2017. https://www.comunidad.madrid/servicios/salud/calidad-aire-salud (acceso: 17 de abril de 2023).

[11] R. Ruiz-Páez *et al.*, “Short–term effects of air pollution and noise on emergency hospital admissions in Madrid and economic assessment”, *Environmental Research*, vol. 219, p. 115147, feb. 2023, doi: 10.1016/j.envres.2022.115147. Acceso: abril 2023.

[12] A. H. Barrú, “METEOROLOGÍA Y CONTAMINACIÓN ATMOSFÉRICA. PECULIARIDADES DE LA ZONA URBANA DE JAÉN”, Facultad de Ciencias Experimentales, Dpto. de Física Aplicada, España, Informe Técnico. [En línea]. Disponible en: https://dialnet.unirioja.es/descarga/articulo/1202715.pdf

[13] M. P. Viñals, *Series temporales*, 2ª ed. Universitat Politècnica de Catalunya, 2001. [En línea]. Disponible en: https://books.google.es/books?id=zEmCOBABiQ4C&lpg=SL16-PA7&hl=es&pg=SL16-PA5#v=onepage&q&f=false

[14] “Precio, cotización en tiempo real y noticias de IBEX 35”. Google Finance. https://www.google.com/finance/quote/INDI:INDEXBME (acceso: 18 de abril de 2023).

[15] C. Esparza Catalán, "Series Temporales", CSIC, España, Informe Técnico. [En línea]. Disponible en: http://humanidades.cchs.csic.es/cchs/web_UAE/tutoriales/PDF/SeriesTemporales.pdf.

[16] “Qué es la Inteligencia Artificial | Solver”, IA SOLVER, 30 de marzo de 2021. https://iasolver.es/que-es-la-inteligencia-artificial/ (acceso: 23 de agosto de 2023).

[17] P. Isasi Viñuela and I. M. Galván León, “Introducción a las Redes de Neuronas Artificiales” en *Redes de neuronas artificiales: un enfoque práctico*, Madrid: Pearson Prentice Hall, 2004, 1-22.

[18] P. Isasi Viñuela and I. M. Galván León, “Perceptron multicapa” en *Redes de neuronas artificiales: un enfoque práctico*, Madrid: Pearson Prentice Hall, 2004, 45-74.

[19] I. Neutelings. “Neural networks”, TikZ.net. https://tikz.net/neural_networks/ (acceso: 6 de junio de 2023).

[20] I. G. R. Gavilán. “Catálogo de componentes de redes neuronales (II): funciones de activación”, Ignacio G.R. Gavilán. http://ignaciogavilan.com/catalogo-de-componentes-de-redes-neuronales-ii-funciones-de-activacion/ (acceso: 6 de junio de 2023).

[21] “What is Gradient Descent?”. IBM. https://www.ibm.com/mx-es/topics/gradient-descent (acceso: 28 de agosto de 2023).

[22] P. Isasi Viñuela and I. M. Galván León, “Redes de neuronas recurrentes” en *Redes de neuronas artificiales: un enfoque práctico*, Madrid: Pearson Prentice Hall, 2004, 103-122.

[23] “Understanding LSTM Networks”. colah’s blog. https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (acceso: 10 de junio de 2023).

[24] S. Hochreiter y J. Schmidhuber, " Long Short-Term Memory ", *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997. [En línea]. Disponible en: http://www.bioinf.jku.at/publications/older/2604.pdf. Acceso: junio 2023.

[25] J. Korstanje, “Fourier Transform for Time Series”, Medium, 31 de octubre de 2021. https://towardsdatascience.com/fourier-transform-for-time-series-292eb887b101 (acceso: 25 de junio de 2023).

[26] “¿Qué es Python? - Explicación del lenguaje Python – AWS”. *Amazon Web Services, Inc.* https://aws.amazon.com/es/what-is/python/ (acceso: 20 de agosto de 2023).

[27] “TensorFlow Core | Aprendizaje automático para principiantes y expertos”. https://www.tensorflow.org/overview?hl=es-419 (acceso: 20 de agosto de 2023).

[28] “Package overview — pandas 2.0.3 documentation”. https://pandas.pydata.org/docs/getting_started/overview.html (acceso: 20 de agosto de 2023).

[29] “What is NumPy? — NumPy v1.25 Manual”. https://numpy.org/doc/stable/user/whatisnumpy.html (acceso: 20 de agosto de 2023).

[30] “Streamlit • A faster way to build and share data apps”. Streamlit. https://streamlit.io/ (acceso: 1 de agosto de 2023).

[31] María Medina, “mariamedp/pycones-contaminacion-madrid”, GitHub. https://github.com/mariamedp/pycones-contaminacion-madrid/tree/3e0ddfa9f81429b98afed0ecc5e77043fd58ad1e (acceso: 25 de junio de 2023).

[32] S. Kumari y S. K. Singh, “Machine learning-based time series models for effective CO2 emission prediction in India”, *Environ Sci Pollut Res*, jul. 2022, doi: 10.1007/s11356-022-21723-8.

[33] C. Zhan, S. Li, J. Li, Y. Guo, Q. Wen and W. Wen, "Prediction of Air Quality in Major Cities of China by Deep Learning", 2020 16th International Conference on Computational Intelligence and Security (CIS), Guangxi, China, 2020, pp. 68-72, doi: 10.1109/CIS52066.2020.00023.

[34] L. Jovova and K. Trivodaliev, "Air Pollution Forecasting Using CNN-LSTM Deep Learning Model," 2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 2021, pp. 1091-1096, doi: 10.23919/MIPRO52101.2021.9596860.

[35] J. Luo, Y. Gong, “Air pollutant prediction based on ARIMA-WOA-LSTM model”, Atmospheric Pollution Research, Volume 14, Issue 6, 2023, 101761, ISSN 1309-1042, doi: 10.1016/j.apr.2023.101761.

[36] V. Pintado, "Predicción de la calidad del aire de Madrid mediante modelos supervisados", Trabajo fin de máster, Dpto. de Ciencia de Datos, Universitat Oberta de Catalunya, España, 2019. [En línea]. Disponible en: https://openaccess.uoc.edu/bitstream/10609/99446/7/gabvilpiTFM0619memoria.pdf

[37] "¿Qué son?". Portal de datos abiertos del Ayuntamiento de Madrid. https://datos.madrid.es/portal/site/egob/menuitem.400a817358ce98c34e937436a8a409a0/?vgnextoid=eba412b9ace9f310VgnVCM100000171f5a0aRCRD&vgnextchannel=eba412b9ace9f310VgnVCM100000171f5a0aRCRD&vgnextfmt=default (acceso: 24 de marzo de 2023).

[38] "Calidad del aire. Estaciones de control". Portal de datos abiertos del Ayuntamiento de Madrid. https://datos.madrid.es/portal/site/egob/menuitem.c05c1f754a33a9fbe4b2e4b284f1a5a0/?vgnextoid=9e42c176313eb410VgnVCM1000000b205a0aRCRD&vgnextchannel=374512b9ace9f310VgnVCM100000171f5a0aRCRD&vgnextfmt=default (acceso: 24 de marzo de 2023).

[39] "Red de estaciones fijas de control de calidad del aire". Portal Web del Ayuntamiento de Madrid. https://www.madrid.es/portales/munimadrid/es/Inicio/Medio-ambiente/Direcciones-y-telefonos/Red-de-estaciones-fijas-de-control-de-calidad-del-aire/?vgnextfmt=default&vgnextoid=aeca16d591bbf710VgnVCM2000001f4a900aRCRD&vgnextchannel=864f79ed268fe410VgnVCM1000000b205a0aRCRD (acceso: 24 de marzo de 2023).

[40] AEMET. "AEMET OpenData". https://opendata.aemet.es/centrodedescargas/inicio (acceso: 29 de julio de 2023).

[41] "Seasonality". Kaggle. https://kaggle.com/code/ryanholbrook/seasonality (acceso: 31 de julio de 2023).

[42] "How to interpret MSE (simply explained)". Stephen Allwright. https://stephenallwright.com/interpret-mse/ (acceso: 25 de julio de 2023).

[43] J. Brownlee. "How to Grid Search Deep Learning Models for Time Series Forecasting". MachineLearningMastery. https://machinelearningmastery.com/how-to-grid-search-deep-learning-models-for-time-series-forecasting/ (acceso: 24 de junio de 2023).

[44] K. Team. "Keras documentation: EarlyStopping". Keras. https://keras.io/api/callbacks/early_stopping/ (acceso: 25 de julio de 2023).

[45] "What is a good RMSE value? Simply explained". Stephen Allwright. https://stephenallwright.com/good-rmse-value/ (acceso: 25 de julio de 2023).

[46] "Índice de Calidad del Aire". Portal de Calidad del aire. https://airedemadrid.madrid.es/portales/calidadaire/es/Bases-de-datos-y-publicaciones/Bases-de-datos-de-calidad-del-aire/Indices-y-zonas/Indice-de-Calidad-del-Aire/?vgnextfmt=default&vgnextoid=303d635a41187710VgnVCM1000001d4a900aRCRD

&vgnextchannel=480285a1259d7710VgnVCM2000001f4a900aRCRD (acceso: 28 de julio de 2023).

[47] “Streamlit Community Cloud • Streamlit”. Streamlit. https://streamlit.io/cloud (acceso: 1 de agosto de 2023).