UNIVERSITAT POLITÈCNICA DE VALÈNCIA


**Universitat Politècnica de València**

**Departamento de Sistemas Informáticos y Computación**

# Modelo basado en redes neuronales para la predicción de tráfico en la ciudad de Valencia

TRABAJO DE FIN DE MÁSTER

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

*Autor*

Cristian Villarroya Sánchez

*Directores*

Dra. Eva Onaindia de la Rivaherrera
Dr. Carlos Tavares Calafate

*Curso académico*

2020-2021

**Resumen**

Existen multitud de modelos que tratan de predecir la velocidad en vías urbanas e interurbanas, la contaminación acústica provocada por el tráfico en ciudades, o incluso el flujo de tráfico urbano en base a datos históricos de cámaras o del tráfico en los móviles de las personas.

En este estudio se trata de diseñar un modelo que sea capaz de predecir el flujo de tráfico en la ciudad de Valencia, España, en base a los datos recogidos por unas espiras electromagnéticas repartidas por toda la ciudad. Con una buena predicción se podrá ser capaz de prever posibles atascos, y también de evitarlos.

Se ha diseñado un modelo basado en una red neuronal recurrente de tipo Long Short-Term Memory (LSTM) para realizar la predicción del flujo de tráfico en las diferentes calles de Valencia para las diferentes horas del día, así como un estudio de la influencia de las características utilizadas en la precisión del modelo.

**Palabras clave:** Predicción; flujo de tráfico; aprendizaje profundo.

**Abstract**

There are many models that try to predict the speed on urban and interurban roads, the noise pollution caused by traffic in cities, or even the flow of urban traffic based on historical data from cameras or traffic on people's mobile phones.

The aim of this study is to design a model capable of predicting traffic flow in the city of Valencia, Spain, based on data collected by electromagnetic loops distributed throughout the city. With a good prediction it will be possible to foresee possible traffic jams, and also to avoid them.

A model based on a recurrent neural network of the Long Short-Term Memory (LSTM) type has been designed to predict the traffic flow in the different streets of Valencia for the different hours of the day, as well as a study of the influence of the characteristics used in the accuracy of the model.

**Keywords:** Forecasting; Traffic flow; Deep learning.

**Resum**

Existeixen multitud de models que tracten de predir la velocitat en vies urbanes i interurbanes, la contaminació acústica provocada pel trànsit en ciutats, o fins i tot el flux de trànsit urbà sobre la base de dades històriques de cambres o del trànsit en els mòbils de les persones.

En aquest estudi es tracta de dissenyar un model que siga capaç de predir el flux de trànsit a la ciutat de València, Espanya, sobre la base de les dades recollides per unes espires electromagnètiques repartides per tota la ciutat. Amb una bona predicció es podrà ser capaç de preveure possibles embossos, i també d'evitar-los.

S'ha dissenyat un model basat en una xarxa neuronal recurrent de tipus Long Short-Term Memory (LSTM) per a realitzar la predicció del flux de trànsit en els diferents carrers de València per a les diferents hores del dia, així com un estudi de la influència de les característiques utilitzades en la precisió del model.

**Paraules clau:** Predir; flux de trànsit; aprenentatge profund

# Índice general

# Índice de figuras

# Índice de tablas

# Capítulo 1

# Introducción

Con el desarrollo de las ciudades inteligentes, los sistemas inteligentes de transporte (ITS por sus siglas en inglés), se han implantado por todo el mundo para resolver o aliviar los problemas asociados al tráfico. Los sistemas de gestión del tráfico urbano recopilan datos sobre el estado del tráfico en la red de tráfico urbano, incluyendo volumen, ocupación y velocidad. Esta información puede proceder de diversas fuentes, como pueden ser los sistemas GPS, los detectores de bucle, la identificación por radiofrecuencia (RFID por sus siglas en inglés), u otros sistemas.

En el caso de este trabajo, la información del flujo de vehículos proviene de unas espiras electromagnéticas repartidas por toda la ciudad de Valencia, que llevan a cabo el recuento de vehículos en cada una de las calles de la ciudad. Después, en la página web de datos abiertos del Ayuntamiento de Valencia [1], existe una API web de la cual se han extraído los datos utilizados en este trabajo.

La predicción del flujo de tráfico es una cuestión fundamental para investigadores y profesionales del sector del transporte que supone un gran reto, ya que el flujo de tráfico suele presentar una gran falta de linealidad y patrones complejos. Además, pueden ocurrir eventos fuera de lo normal, como es el caso de este año de pandemia, o algo más simple, como un evento multitudinario como por ejemplo un concierto o un partido de fútbol, que alteren drásticamente el comportamiento normal del flujo de tráfico.

Ser capaz de predecir el flujo de tráfico en grandes ciudades otorga un gran beneficio en múltiples áreas, como por ejemplo, el enrutamiento de vehículos o la gestión de la congestión del tráfico. Una buena predicción del flujo de tráfico puede ayudar a evitar estas congestiones, pudiendo ser capaz de cambiar la ruta de un vehículo por otra menos congestionada.

En este proyecto se propone una solución que permite predecir, en cada hora del día, el flujo de tráfico en las calles de la ciudad de Valencia. Para ello se define un modelo de predicción del flujo de tráfico a corto plazo basado en redes

neuronales recurrentes; en concreto, en una red de memoria a largo plazo (LSTM por sus siglas en ingles). La solución propuesta se basa en una predicción del flujo de tráfico a corto plazo, ya que se van a tener en cuenta un pequeño número de horas anteriores para la predicción de las horas posteriores.

Las redes LSTM son un tipo de redes neuronales recurrentes que son apropiadas para modelar datos en serie como series temporales. Una de sus principales ventajas es que son capaces de procesar largas secuencias de información, como es el caso del flujo de tráfico. Esto, unido a que se va a realizar una predicción a corto plazo, las hacen en una buena elección a priori como solución al problema planteado.

El código realizado en este TFM puede encontrarse en mi GitHub [2].

## 1.1 Objetivos

El principal objetivo del TFM es crear un modelo de predicción del flujo de tráfico para la ciudad de Valencia. Con ello se pretende ser capaz de predecir el tráfico en cada calle de la ciudad en las diferentes horas del día.

Este Trabajo de Fin de Máster tiene como objetivos adicionales:

- Desarrollar un modelo de predicción del flujo del tráfico basado en una red neuronal en base a datos históricos obtenidos a partir de espiras electromagnéticas.
- Realizar pruebas con diferentes modelos para conseguir la máxima precisión posible.
- Analizar los resultados obtenidos en las predicciones y compararlos con los datos reales para medir la precisión.
- Definir trabajos futuros.

## 1.2 Estructura

Este TFM está estructurado de la siguiente manera: este primer capítulo incluye una presentación e introducción al TFM, detallando los objetivos y la motivación del mismo.

A continuación, en el capítulo 2, se presenta el estado del arte donde se hará referencia a varios trabajos relacionados.

En el capítulo 3 se explican los procesos de extracción de los datos y el preproceso de los mismos antes de ser importados a una base de datos.

Seguidamente, en el capítulo 4, se detalla el proceso de limpieza de los datos y un proceso de análisis de los mismos, para estudiar las propiedades de estos y ser capaz de tomar decisiones en función de ellas.

Después, en el capítulo 5 se presentan diferentes versiones de los modelos implementados, además de un análisis del error y una comparación de resultados de los mismos.

Finalmente, en el capítulo 6 se presentan las principales conclusiones, así como referencia a trabajos futuros.

# Capítulo 2

# Estado del arte

En este capítulo se describen diferentes investigaciones realizadas en el área de la predicción del flujo de tráfico que guardan relación con este proyecto y se presentan algunas aplicaciones reales de predicción de tráfico. Además, se explica brevemente el concepto de serie temporal y sus componentes, ya que es necesario para entender el desarrollo de este trabajo.

## 2.1 Series temporales

Una serie temporal se define como una colección de observaciones de una variable recogidas secuencialmente en el tiempo y, además, estas observaciones suelen ser recogidas en intervalos de tiempo equidistantes. Con esto se pretende no solo explicar los sucesos del pasado, sino también ser capaz de predecir el futuro observando las muestras pasadas.

Las series temporales se suelen modelar mediante un proceso estocástico, es decir, una secuencia de variables aleatorias. Normalmente, se conoce el valor en un momento $t$, $Y(t)$, y se quiere predecir el valor en un instante posterior $t + 1$, $Y(t + 1)$.

Hay tres aspectos importantes a tener en cuenta a la hora de definir una serie temporal, y estos son la *estacionariedad*, la *estacionalidad* y la *autocorrelación*.

En primer lugar, se dice que una serie temporal es *estacionaria* si sus propiedades estadísticas no cambian con el tiempo, es decir, tiene una media y varianza constante. Además, dentro de la estacionariedad, se estudia la tendencia de los datos, la cual se puede definir como un cambio a largo plazo que se produce en relación a la media de los datos, o el cambio a largo plazo de la media. Es decir, la tendencia muestra si los datos crecen, disminuyen, o se mantienen constantes a lo largo del tiempo.

Por otro lado, la *estacionalidad* hace referencia a las fluctuaciones periódicas

de los datos; por ejemplo, el consumo de electricidad es alto durante el día y bajo por la noche, o el número de parados en España que, en general, aumenta en invierno y disminuye en verano. Estas fluctuaciones pueden darse en diferentes niveles temporales, y pueden ser horarias, diarias, semanales, mensuales, etc.

Finalmente, la *autocorrelación* hace referencia a la similitud entre las observaciones en función del desfase temporal entre ellas, es decir, como de relacionada está una serie temporal con una versión retardada temporalmente de sí misma. La autocorrelación puede ayudar también a identificar la estacionalidad y la tendencia de los datos.

Otro aspecto importante de las series temporales es la descomposición de las mismas. Una serie temporal se compone de tres elementos: la tendencia; la estacionalidad y el error. Antes de explicar los componentes, decir que existen dos tipos de modelos de descomposición, los modelos multiplicativos, dónde la serie temporal es el resultado de multiplicar los tres componentes, y los modelos aditivos, que son el resultado de sumar los tres componentes.

En otras palabras, el modelo aditivo se explica con la fórmula

$$Y_t = T_t + S_t + e_t$$

mientras que el modelo multiplicativo con la fórmula

$$Y_t = T_t * S_t * e_t$$

siendo $Y$ la serie temporal completa, $T$ la tendencia, $S$ la estacionalidad, y $e$ el error. La tendencia y la estacionalidad se han explicado previamente, en cuanto a la parte del error, es el resultado de substraer la estacionalidad y la tendencia a la serie temporal si se utiliza un modelo aditivo, o de dividir la estacionalidad y la tendencia a la serie temporal, si se utiliza un modelo multiplicativo. El escoger un modelo u otro es decisión del diseñador, los modelos multiplicativos se usan si, en los valores de la serie temporal, se observa un crecimiento notable en los datos tanto en amplitud como en longitud. Si los datos no varían en exceso, se usará un modelo aditivo.

Todas estas partes de una serie temporal serán estudiadas en el capítulo 4.2.

## 2.2 Modelos ARIMA

El modelo autorregresivo integrado de promedio móvil (ARIMA por sus siglas en inglés) es un modelo dinámico que utiliza datos de series temporales desarrollado por Box y Jenkins en 1970 [3]. El modelo ARIMA permite describir un valor como una función lineal de datos pasados y errores aleatorios y, además, permite incluir un componente cíclico o estacional (periodicidad o variación de periodo anual, mensual, etc.).

Los modelos ARIMA han tenido una gran aplicación en problemas de predicción de tráfico. Los autores del artículo [4] comentan que uno de los problemas de utilizar los modelos ARIMA es que estos exigen una base de datos sólida para la construcción del modelo. Por ello, proponen una solución a este problema haciendo uso de un modelo ARIMA estacional (SARIMA por sus siglas en inglés) para la predicción a corto plazo del flujo de tráfico con datos de entrada limitados. Los datos de este trabajo proceden de una carretera arterial de la ciudad de Chennai, India, durante 3 días consecutivos. El problema de los modelos ARIMA es que no soportan datos estacionales, por ello, surgen los modelos SARIMA. Estos modelos son una extensión de los modelos ARIMA los cuales captan el comportamiento puramente estacional de una serie, y no tiene que ser eliminado de los datos como sí pasa con los modelos ARIMA. La parte estacional del modelo implica desfases temporales hacia atrás del periodo estacional.

Dong-wei et al. [5] combinan el modelo ARIMA con el filtro de Kalman para construir un algoritmo de predicción del estado del tráfico por carretera. Para ello, utilizan datos recogidos de la ciudad de Beijing, China. Utilizan el filtro de Kalman para construir las ecuaciones de estado, medición y actualización para completar el proceso de entrenamiento. Además, con el filtro de Kalman, resuelven el problema de obtener el mejor filtro lineal basándose en el criterio del mínimo error cuadrático medio. Para los experimentos, utilizaron cuatro tipos diferentes de carreteras de la ciudad de Beijing. Los autores comparan su modelo combinado con un modelo únicamente basado en ARIMA y muestran la mejora obtenida.

Por último, Alghamdi et al. [6] presentan una solución a la congestión del tráfico en la ciudad de California, USA. Los autores proponen un modelo basado en series temporales a corto plazo utilizando datos no gaussianos del flujo de tráfico. A pesar de que la mayoría de la investigación en predicción del flujo de tráfico basada en series temporal considera que los datos son estacionarios, lo que significa que su desviación típica y media no varían en el tiempo, los autores de este artículo consideran que en la predicción a tiempo real de tráfico los datos con una distribución no gaussiana pueden señalar características del tráfico específicas en las que la red de transporte puede experimentar una grave congestión, como puede ser algún tipo de festival, evento deportivo, obras, etc. En base a ello, proponen analizar los datos de series temporales no estacionarios a corto plazo utilizando el modelo ARIMA, convirtiendo las series no estacionarias en series estacionarias determinando el número de diferencias requeridas.

## 2.3 Modelos mixtos de técnicas de aprendizaje automático

Gran parte de la investigación en predicción de flujo de tráfico propone modelos híbridos. Por ejemplo, en el artículo [7], los autores plantean un modelo de predicción del flujo de tráfico en una autopista basado en una combinación del perceptrón multicapa y el método random forest. Para ello, utilizan datos del flujo de tráfico en una autopista en China y, además, añaden información meteorológica, datos sobre festividades locales, datos sobre peajes en la autopista y otras fuentes de datos. El modelo híbrido realiza una predicción del flujo de tráfico usando el método de random forest y otra predicción utilizando un perceptrón multicapa. Finalmente, el resultado de la predicción final será una combinación de ambos resolviendo el peso correspondiente a los modelos utilizando el error mínimo como función objetivo. De esta manera, la predicción final será una combinación ponderada de los dos resultados obtenidos con los dos métodos. Según los resultados obtenidos, se observa una mejora al utilizar su modelo híbrido respecto a utilizar los métodos por separado.

Por otro lado, Li et al. [8] proponen combinar un modelo ARIMA y una red neuronal con funciones de base radial (RBF-ANN por sus siglas en inglés) para predecir el flujo de tráfico a corto plazo. El conjunto de datos utilizado consiste en una serie de datos obtenidos durante las dos primeras semanas de noviembre en intervalos de 5 minutos en la avenida Shinan, en el distrito de Nansha, ciudad de Guangzhou, China. Los autores utilizan el modelo ARIMA para modelar la componente lineal de la serie temporal del flujo de tráfico y, a continuación, utilizan el modelo RBF-ANN para captar la componente no lineal del modelado de los residuos del modelo ARIMA. La predicción final será una combinación de los resultados de predicción de ambos modelos.

En el artículo [9], los autores presentan un método híbrido de aprendizaje profundo multimodal para la previsión del flujo de tráfico a corto plazo haciendo uso de un conjunto de datos sobre el tráfico real en autopistas de Reino Unido. El modelo está formado por una red convolucional, y, por otro lado, unas redes neuronales recurrentes de unidades recurrentes cerradas (GRU por sus siglas en inglés) con un modelo de atención. La capa convolucional es la que se encarga de captar las características no lineales de los datos del tráfico, y la GRU con el modelo de atención se encarga de captar las dependencias temporales a largo plazo.

Por último, los autores de [10] proponen un modelo denominado M-B-LSTM, que es un modelo híbrido donde se construye una red de autoaprendizaje online como capa de mapeo de datos para aprender e igualar la distribución estadística del flujo de tráfico, reduciendo así el efecto del desequilibrio y la distribución y

el problema del sobreajuste en el aprendizaje de la red. Además, se introduce la red profunda bidireccional de memoria a corto plazo (DBLSTM por sus siglas en inglés) para reducir el problema de la incertidumbre mediante el proceso de aproximación de contextos hacia adelante y hacia atrás en la capa de reducción de la estocasticidad; luego, se utiliza la red LSTM para predecir el siguiente estado del flujo de tráfico en la capa de previsión.

## 2.4 Modelos basados en redes neuronales

Uno de los modelos más utilizados hoy en día son las redes neuronales debido a su gran eficacia. En el artículo [11], los autores proponen un modelo de predicción del flujo de tráfico a corto plazo en toda una ciudad basado en una red neuronal convolucional profunda, denominada TFFNET (Traffic Flow Forecasting NETwork). El modelo tiene en cuenta las dependencias espaciales y temporales del flujo de tráfico y, además, es capaz de tener en cuenta factores externos, como pueden ser accidentes, periodos vacacionales u otro tipo de eventos, haciendo uso de otra red neuronal totalmente conectada fusionada con la salida del componente principal de TFFNET. El modelo toma como datos de entrada unas imágenes generadas a partir de unas trazas GPS de taxis.

Otro ejemplo es el propuesto en [12], donde los autores utilizan un modelo basado en una red neuronal profunda para la predicción del flujo de tráfico de un día completo en la ciudad de Seattle. El modelo utiliza un algoritmo de aprendizaje supervisado multicapa para extraer la relación potencial entre los datos de flujo de tráfico, y una combinación de factores clave contextuales. Debido al coste temporal de entrenar una red neuronal profunda, proponen un método de entrenamiento por lotes, reduciendo así el coste temporal en entrenamiento de la red propuesta.

Por otro lado, en el artículo [13], los autores proponen un modelo de predicción del flujo de tráfico a corto plazo que combina el análisis estacional con una red neuronal GRU. El modelo utiliza un algoritmo de detección de características espacio-temporales para definir el intervalo de tiempo de entrada y el volumen de datos espaciales óptimos; después la red GRU procesa la información de dichas características para realizar las predicciones.

Finalmente, en el artículo [14], Kang et al. definen un modelo basado en redes neuronales LSTM para tener en cuenta dependencias temporales a corto y largo plazo. En el artículo, los autores realizan un estudio sobre el efecto de las diferentes configuraciones de entrada en el rendimiento de la predicción del flujo de tráfico.

## 2.5 Herramientas de predicción de tráfico

La investigación en predicción de tráfico urbano es extensa y ha generado algunas plataformas interesantes como C2SMART [15], una iniciativa conjunta de varias universidades americanas que utiliza técnicas de inteligencia artificial para la detección de congestión en redes urbanas. La plataforma C2SMART utiliza los datos de velocidad registrados en los enlaces y cruces de las calles del centro de Seattle. La principal misión de esta plataforma consiste en utilizar los datos generados por las nuevas fuentes emergentes y las tecnologías de detección no tradicionales para desarrollar plataformas de intercambio de datos seguras e interoperables, y soluciones orientadas al sistema. Además de tratar la movilidad urbana, C2SMART realiza análisis urbanos para ciudades inteligentes y proporciona soporte para diseño de infraestructuras de transporte resistentes, seguras e inteligentes.

Otra de las herramientas que sin duda ha tenido un mayor impacto en la predicción de tráfico es la aplicación Google Maps. Esta aplicación, además de ofrecer la mejor ruta a un destino, es capaz de predecir el tráfico que habrá en un determinado tramo a una cierta hora. Como se habla en [16], el modelo que usa Google Maps para estas predicciones es una red neuronal basada en grafos. El modelo combina las condiciones del estado actual de una determinada área con patrones históricos de las carreteras de todo el mundo, a lo que se añade información adicional sobre calidad de la carretera, velocidad permitida, etc. Utilizar una red neuronal basada en grafos permite diseñar conexiones complejas para controlar no solo el tráfico posterior o anterior, sino también a lo largo de las carreteras adyacentes y las intersecciones.

# Capítulo 3

# Extracción y procesado de los datos

En este capítulo se explican los procesos de extracción, procesado y carga de los datos, etapas que son críticas para poder realizar el análisis de los mismos.

## 3.1 Extracción de los datos

Se han realizado dos procesos de extracción de información. El propósito del primero es obtener los datos referentes a la cantidad de vehículos en las diferentes calles, y el objetivo del segundo es obtener las características de las diferentes calles.

### 3.1.1 Número de vehículos en las calles

Los datos acerca de la cantidad de vehículos que circulan por las calles están disponibles en la web de datos abiertos del Ayuntamiento de Valencia [1]. Dicha información es obtenida por medio de espiras electromagnéticas repartidas por toda la ciudad, y cuyas mediciones se comunican a un servidor central. Las espiras se despliegan de manera que hay una espira distinta para cada carril, y existen calles/avenidas con más de un punto realizando el conteo de vehículos, lo cual típicamente ocurre si la calle/avenida es especialmente larga y tiene mucho tráfico. La web del Ayuntamiento ofrece una agregación del resultado de dicho conteo de todas las espiras disponibles en la calle correspondiente al formato vehículos/hora. Esta información es actualizada en tiempo real cada 15 minutos, y para cada calle, habiendo un total de 386 puntos de medición en la ciudad, los cuales representan a las principales vías de la misma.

Cada calle, además de por su nombre, se identifica por un código ATA único; por ejemplo, la calle Blasco Ibáñez tiene el código A47. Es posible que avenidas muy grandes, como es el caso de la Avenida del Cid, estén divididas en diferentes

tramos, cada una con un código ATA distinto, por ejemplo, dicha avenida esta dividida en 4 tramos, cuyos códigos ATA corresponden a A69, A70, A71 y A72.

Existe una API mediante la cual se pueden consultar los datos en diferentes formatos. En este trabajo se ha optado por el formato JSON. Cada ítem del fichero contiene la siguiente estructura:

- DES_TRAMO: Nombre de la calle.
- IDTRAMO: Código identificador de la calle (código ATA).
- MODIFIED: Fecha y hora exacta en la que se ha actualizado la información.
- LECTURA: Cantidad de vehículos/hora.
- COORDINATES: Coordenadas de las diferentes espiras electromagnéticas en la calle correspondiente.
- URI: Enlace al fichero JSON.

Se ha creado un script en Python mediante el cual se realiza periódicamente una consulta a dicha API y se obtiene la información previamente descrita. El script se encarga de generar un nuevo fichero JSON con la información relevante de cada calle. El nuevo fichero JSON posee la siguiente información:

- Descripción: Nombre de la calle.
- ATA: Código identificador de la calle (código ATA).
- Fecha: Fecha en la que se realizó el conteo en formato YYYY-MM-DD.
- Hora: Hora en la que se realizó el conteo en formato HH:MM:SS.
- Coches: Cantidad de vehículos/hora.

Dicho script se ejecuta automáticamente cada 15 minutos para recoger la nueva información, y la añade al fichero JSON.

### 3.1.2 Información sobre las calles

Se ha realizado un estudio para las 386 calles disponibles mediante la API. Con dicho estudio, se ha obtenido para cada una de ellas, la siguiente información:

- Tipo de la calle.
- Barrio al que pertenece.

- Código Postal.
- Sentido único o doble sentido.
- Número de carriles.
- Velocidad límite.

Para determinar el tipo de calle ha sido necesario utilizar la herramienta Open Street Map [17]. Para ello, se ha hecho uso de un fichero OSM en el que se tiene un mapa de la ciudad de Valencia. Dicho fichero OSM tiene una estructura XML en la que aparece información de todas las calles existentes en el mapa. De esta manera, se obtienen el tipo de calle (según el listado de Open Street Map) [18], el número de carriles, el sentido, y la velocidad límite.

Por otro lado, para obtener el código postal y el barrio al que pertenece cada calle, se ha hecho uso de la página web del Ayuntamiento de Valencia [19]. Esta web dispone de una función que, en base al nombre de la calle, permite averiguar el barrio y el código postal al que pertenece.

Esta información de las diferentes calles será utilizada posteriormente para realizar diferentes análisis de los datos; por ejemplo, se podrán juntar datos de calles del mismo barrio, datos de calles del mismo tipo u otro tipo de agregaciones, y así estudiar su comportamiento.

## 3.2 Preproceso de los datos

Antes de cargar los datos extraídos de la API en la base de datos, se ha realizado un preproceso de los mismos para dejarlos en un formato específico, así como limpiar el ruido que pudieran contener. Por ejemplo, la API puede devolver valores erróneos, en los que un -1 significa que ha habido un error en la lectura, pero también es posible que devuelva valores muy elevados, debidos también a un fallo de lectura. Estos datos erróneos serán tratados más adelante, con la ayuda de la librería `pandas` [20] de Python.

En este punto, el fichero JSON contiene un campo con la fecha en el que aparecen la fecha y la hora juntas. Se ha optado por separar ambas en dos nuevos valores: un campo fecha que contiene la fecha en formato YYYY-MM-DD, y otro campo hora en el que se tiene la hora en formato HH:MM:SS.

Otra cuestión es que las diferentes lecturas devueltas por la API en un determinado momento no tienen todas el mismo *timestamp*, es decir, a pesar de que la API se actualice cada 15 minutos, la lectura en una calle puede ser a las 13:00:02, y otra lectura de una calle diferente, en la misma actualización de la API, puede ser a las 12:59:57. Por ello, se ha optado por redondear las horas de lectura en

intervalos de 15 minutos. De esta manera, las lecturas quedarían entre los valores HH:00:00, HH:15:00, HH:30:00 o HH:45:00, lo cual facilita la búsqueda y el agrupamiento posterior de los datos.

Finalmente, se han añadido dos nuevos campos al fichero JSON: un campo festivo, el cual permite saber si el día es festivo o no, siguiendo el calendario de la Comunidad Valenciana [21], y otro campo día, que hace referencia al día de la semana al que pertenece la fecha.

Tras estos pasos, el fichero JSON con la información recogida por la API contiene la siguiente información:

- ATA.
- Fecha.
- Día.
- Hora.
- Festivo.
- Coches.

De esta manera, el fichero JSON con la información recogida por la API contiene información sobre el código ATA, la fecha, el día, la hora, si el día es festivo, y el conteo de vehículos en ese momento.

## 3.3 Carga de los datos

El siguiente paso es la carga de los datos en la base de datos. Debido a que se va a estar constantemente añadiendo información, pues se tienen datos nuevos cada 15 minutos, se ha optado por usar una base de datos no SQL, como MongoDB. Además, al tener los datos en formato JSON, se facilita enormemente la carga a esta base de datos.

En este punto se tienen dos ficheros JSON, uno con la información del conteo de vehículos en las calles, y otro con la información de dichas calles. Por simplicidad, se han unido ambos ficheros antes de la carga a la base de datos, con la ayuda de un script en Python. Con ello, se obtiene un nuevo fichero JSON final con la información necesaria, tanto del conteo de vehículos como de la información de las calles. La estructura de dicho fichero, que también es la estructura de la base de datos, tiene la siguiente forma:

- ATA.

- Nombre de la calle.
- Tipo de la calle.
- Barrio al que pertenece.
- Código Postal.
- Sentido único o doble sentido.
- Número de carriles.
- Velocidad límite.
- Fecha.
- Día de la semana.
- Hora.
- Festivo.
- Coches.

Otra de las ventajas de utilizar MongoDB, es que, gracias al gestor de bases de datos Navicat [22], es posible la exportación de la base de datos a diferentes formatos, entre ellos el formato csv. Ya que se va a utilizar Python y la librería `pandas`, se realizará una exportación de la base de datos al formato csv, que facilita el trabajo con dicha librería.

Para concluir este apartado, se muestra un diagrama ilustrando todos los pasos hasta obtener el fichero contenedor de los datos.

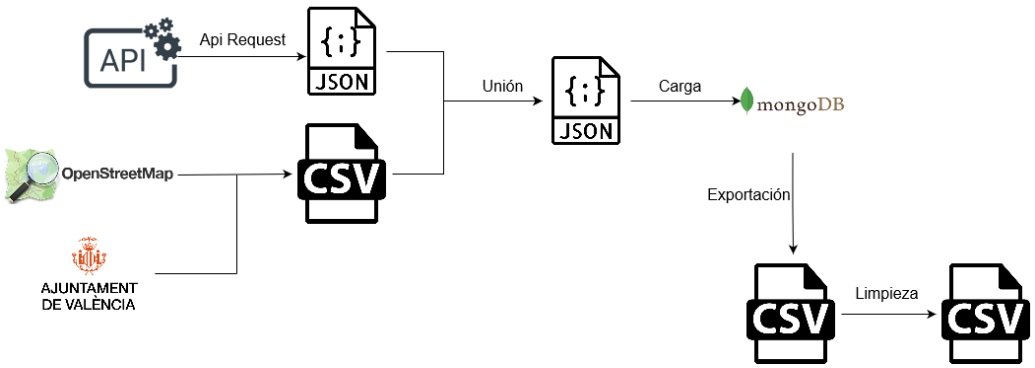


Figura 3.1: Diagrama del proceso ETL.

# Capítulo 4

# Tratamiento de los datos

En este capítulo se efectúa un proceso de análisis de datos para realizar un estudio y detectar patrones en ellos, desarrollando en primer lugar una limpieza de los mismos.

## 4.1 Limpieza de los datos

Inicialmente se dispone, como punto de partida, de una lectura (en vehículos/hora) cada 15 minutos para cada calle. A continuación se muestran los datos para la un tramo de la Avenida del Cid, una de las calles más importantes de Valencia, separadas en dos gráficas, donde la Figura 4.1 muestra datos desde 15 de marzo al 15 de mayo, y la Figura 4.2 muestra los datos desde el 16 de mayo al 20 de junio.

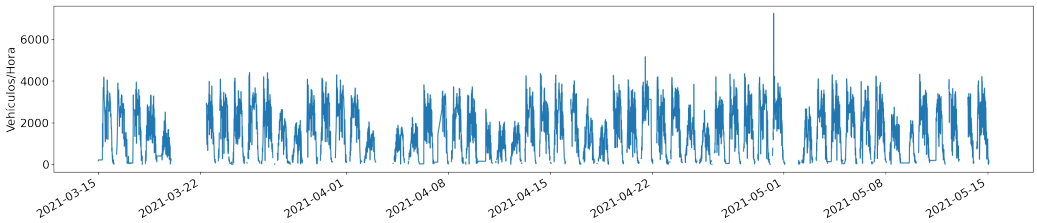


Figura 4.1: Datos del flujo de vehículos en la Avenida del Cid del 15 de marzo al 15 de mayo de 2021.

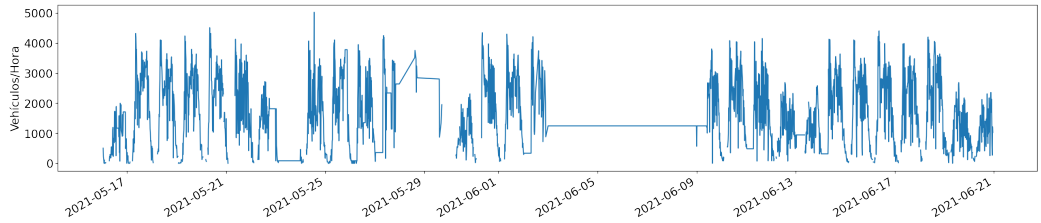


Figura 4.2: Datos del flujo de vehículos en la Avenida del Cid del 16 de mayo al 20 de junio de 2021.

Puede apreciarse que hay errores en las lecturas, siendo posible diferenciarse varios tipos: aquellos en los que no se dispone de ningún dato en ningún momento del día, como se muestra en la Figura 4.3; aquellos en los que faltan datos para alguna hora en particular, pero se disponen de datos para otras horas de ese mismo día, lo cual puede verse en la Figura 4.4; y, por último, cuando por alguna razón se ha obtenido el mismo dato para prácticamente toda un semana de junio, como se observa en la Figura 4.5.

Existen dos tipos de errores a la hora de obtener la lectura de los datos, el primero cuando se obtiene un valor -1 de lectura por parte de la API, y el segundo error que indica que no se ha obtenido ninguna lectura, ya sea por que la API ha fallado, o porque la página web está caída. En el caso de obtener una lectura con valor -1, esta será sustituida por un valor NaN, que facilita su posterior tratamiento.

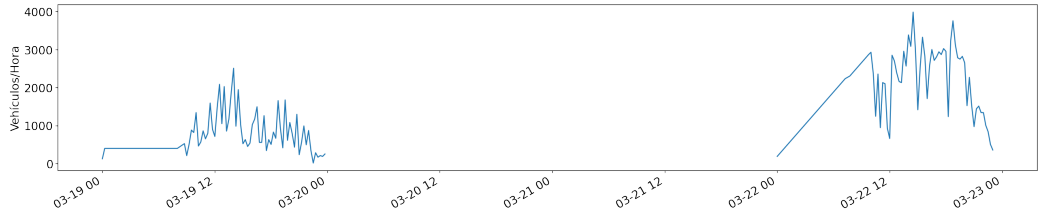


Figura 4.3: Error en los datos del flujo de vehículos en la Avenida del Cid (datos faltantes de día completo).

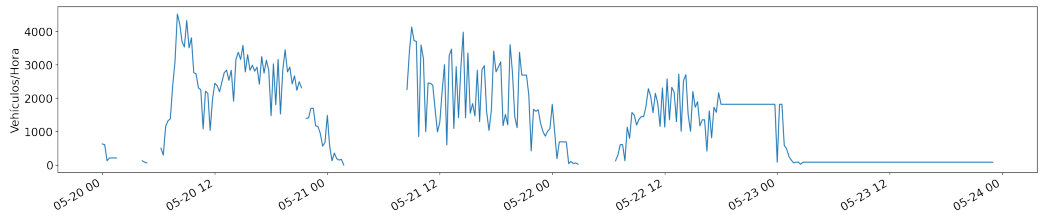


Figura 4.4: Error en los datos del flujo de vehículos en la Avenida del Cid (datos faltantes de horas concretas).

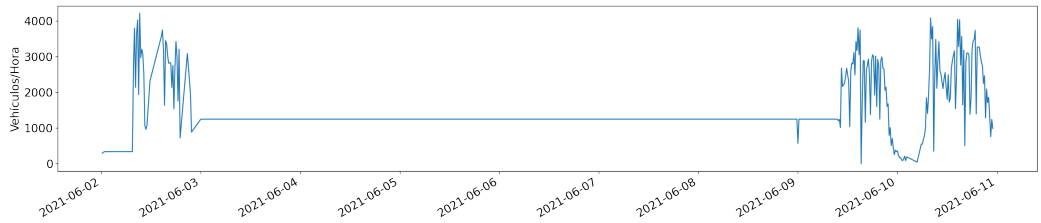


Figura 4.5: Error en los datos del flujo de vehículos en la Avenida del Cid (datos repetidos).

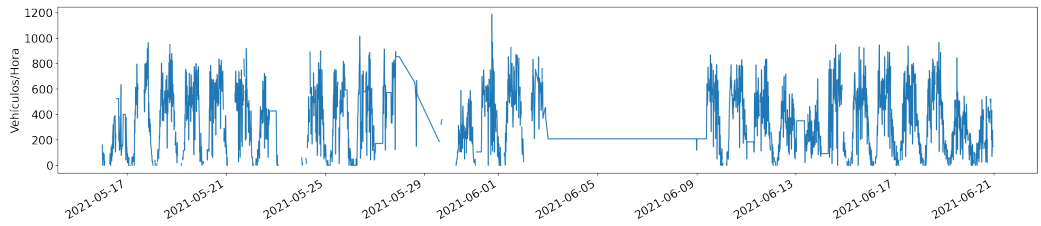


Figura 4.6: Datos del flujo de vehículos en la calle Santos Justo y Pastor del 15 de marzo al 15 de mayo de 2021.

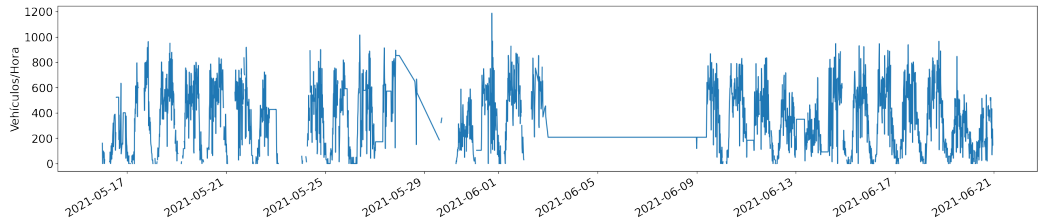


Figura 4.7: Datos del flujo de vehículos en la calle Santos Justo y Pastor del 16 de mayo al 20 de junio de 2021.

Estos errores se repiten para todas las calles en diferentes fechas. Por ejemplo, en las Figuras 4.6 y 4.7 se muestran los datos para la calle Santos Justo y Pastor, una calle no tan transitada como la Avenida del Cid, en la que se puede observar que ocurren los mismos tipos de errores en diferentes fechas.

Para subsanar estos problemas se ha realizado un proceso de limpieza de los datos que puede dividirse en 4 partes:

1. Completar datos vacíos.
2. Agregar los datos existentes (cada 15 minutos) para obtener valores por horas.
3. Completar los días faltantes.
4. Corregir lecturas erróneas (lecturas iguales todo el día).

Previo al proceso de limpieza, se han agrupado los datos por código ATA, ya que el flujo de tráfico no es el mismo en una calle pequeña que en una avenida principal. Después, para cada uno de los grupos, se han separado las lecturas por cada día, para evitar completar datos de un lunes con datos de un martes, por ejemplo.

El primer paso de la limpieza consiste en solucionar la falta de algunos datos en días para los que sí se dispone de datos, es decir, días en los que faltan algunas lecturas. Para ello se han realizado tres pasos:

1. Un paso de interpolación lineal, donde el dato faltante se interpola en función de la lectura en el instante anterior y posterior.
2. Un proceso de relleno hacia delante, ya que puede ser que falten datos para más de una hora seguida, y entonces no podría realizarse una interpolación.
3. Un proceso de relleno hacia atrás.

Con estos tres pasos nos aseguramos que, si hay datos para un día, se dispongan de datos para todas las lecturas realizados a lo largo del día. Tras este paso, la Figura 4.4 cambia a la mostrada en la Figura 4.8.

El segundo paso del proceso de limpieza consiste en agregar los datos de forma que el intervalo de tiempo entre los mismos sea por horas y no cada 15 minutos. Esto se ha realizado por dos motivos, el primero es que hay lecturas consecutivas con una diferencia muy grande entre ellas; por ejemplo, una lectura a las 10:00 puede tener un valor de 5000 vehículos/hora, y a las 10:15 obtenerse una lectura de 500 vehículos/hora, lo cual no parece ser muy realista y podría ser indicativo de un error. Otro motivo es que hay veces que la API no devuelve ningún dato en

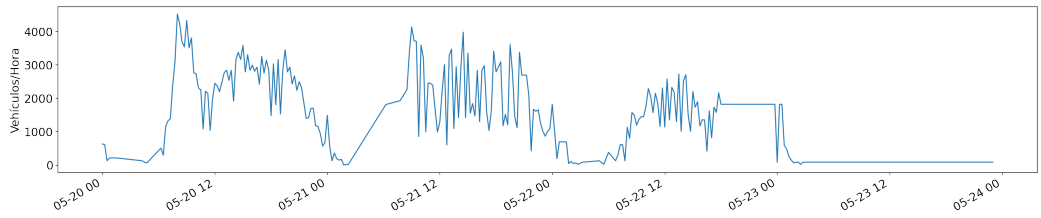


Figura 4.8: Subconjunto de datos en Avenida del Cid tras el primer paso de la limpieza

un determinado momento. Esto no significa que se obtenga un valor de error (es decir, un valor -1) sino que, por algún motivo, la espira no ha realizado la lectura, o la API ha fallado y no se ha obtenido ninguna lectura para ese momento. Por lo tanto, es posible que, para un día, se obtengan las 96 lecturas correspondientes (24 horas $\times$ 4 lecturas por hora), pero también es posible que existan días con 95 lecturas o menos, ya que no se ha recogido ningún dato para un momento de dicho día.

Para la agregación de los datos se ha utilizado la librería `pandas`, la cual permite agregar datos en determinados intervalos de tiempo. Para ello, recoge los datos disponibles para una hora en concreto y hace la media de dichos valores. Tras este proceso la gráfica para la avenida anterior se muestra en las Figuras 4.9 y 4.10.

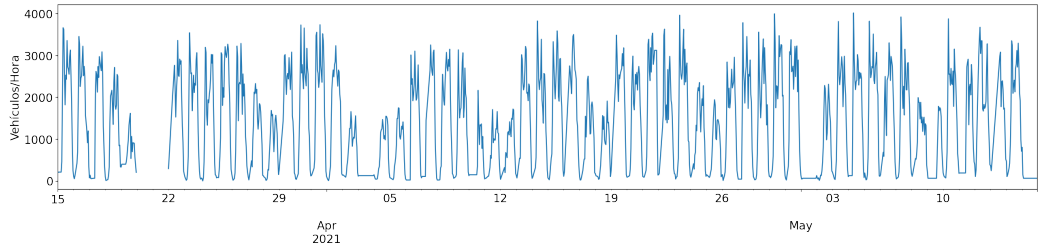


Figura 4.9: Datos de vehículos en la Avenida del Cid tras segunda limpieza del 15 de marzo al 15 de mayo.

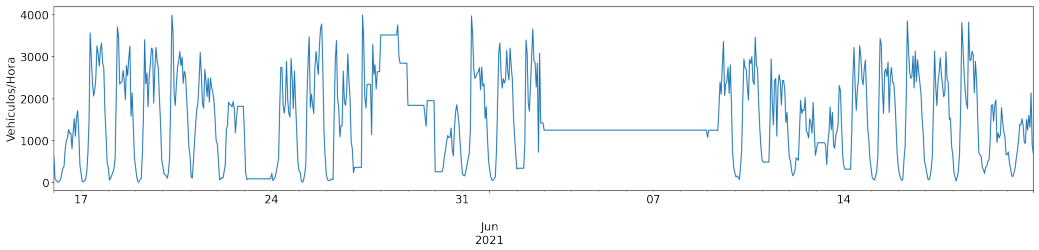


Figura 4.10: Datos de vehículos en la Avenida del Cid tras segunda limpieza del 16 de mayo al 20 de junio

A continuación se aplica el tercer paso de la limpieza, el cual permite completar días enteros faltantes. A pesar de que ahora se tienen 24 lecturas, una para cada hora del día, se siguen observando errores en los datos como consecuencia de los días para los cuales no se dispone de ningún dato. Obviamente, no pueden simplemente eliminarse dichos datos porque los errores pueden ocurrir en diferentes días dependiendo de la calle. En este caso, en la Avenida del Cid, únicamente se observa que falta un día de marzo, pero otras calles pueden tener errores en otros días distintos. Si se optase por simplemente eliminar estos días, obtendríamos series temporales de diferentes tamaños para cada calle, lo cual no permitiría entrenar la red neuronal.

Con ello en mente, se ha optado por detectar los días en los que faltan datos y simplemente copiar los datos para ese mismo día de una semana diferente que sea correcto; por ejemplo, si falta el dato de un lunes, se busca un lunes lo más próximo posible de otra semana y se copian las lecturas de ese lunes a las del lunes faltante. Con esta transformación, los datos obtenidos se representan en las Figuras 4.11 y 4.12.

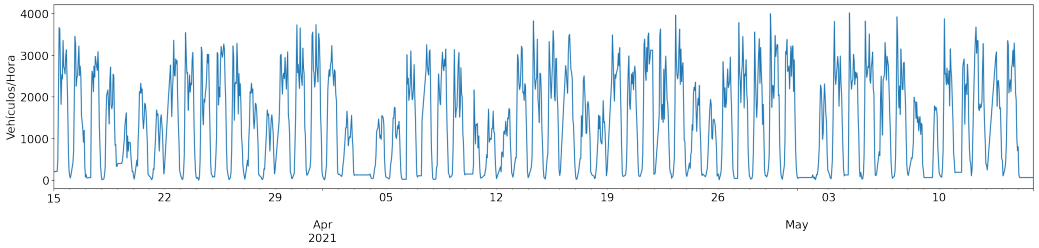


Figura 4.11: Datos del flujo de vehículos en la Avenida del Cid tras la tercera limpieza (del 15 de marzo al 15 de mayo de 2021).

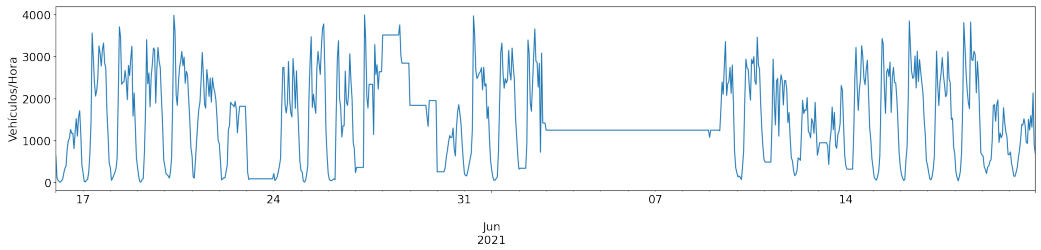


Figura 4.12: Datos del flujo de vehículos en la Avenida del Cid tras tercera limpieza (del 16 de mayo al 20 de junio de 2021).

Finalmente, en el cuarto paso del proceso de limpieza, se han de eliminar las repeticiones en las lecturas. Para ello se ha seguido el mismo proceso que para completar los días faltantes. En primer lugar se detectan las repeticiones (se ha considerado que es un error detectar más de 6 lecturas iguales), y después se sustituyen por el mismo día de la semana con lecturas correctas más próximo. Tras todo este proceso, los datos finales se muestran en las Figuras 4.13 y 4.14.

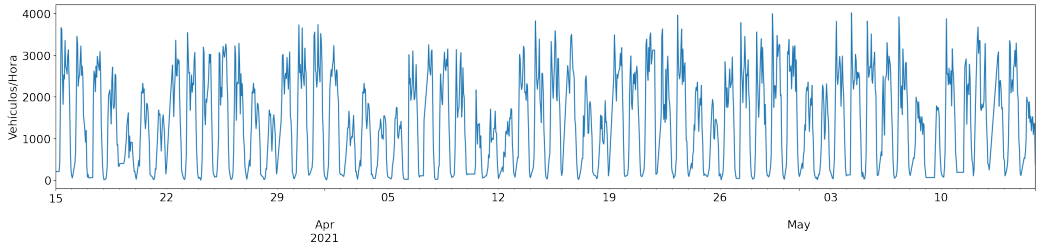


Figura 4.13: Datos del flujo de vehículos en la Avenida del Cid tras proceso de limpieza (del 15 de marzo al 15 de mayo de 2021).

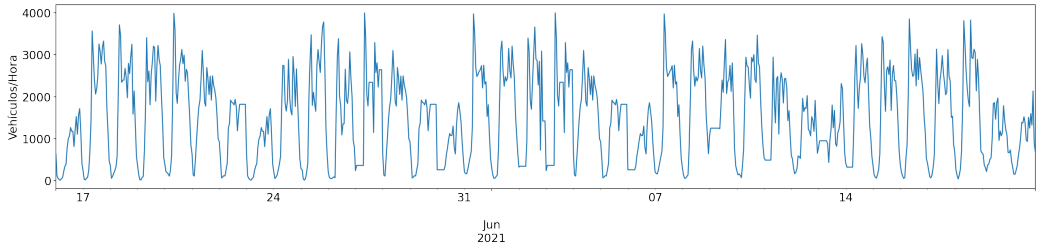


Figura 4.14: Datos del flujo de vehículos en la Avenida del Cid tras proceso de limpieza (del 16 de mayo al 20 de junio de 2021).

Este proceso se ha repetido para las 386 calles disponibles.

## 4.2 Análisis de los datos

Una serie temporal consiste en una serie de datos ordenados en el tiempo en la que el tiempo es la variable independiente, y el objetivo es realizar una predicción para el futuro. Como se ha comentado anteriormente, hay tres aspectos importantes que caracterizan una serie temporal: la *estacionariedad*, la *estacionalidad* y la *correlación* de los datos. El objetivo de este apartado es estudiar dichos aspectos.

### 4.2.1 Tendencias

Antes de entrar a analizar los componentes, se va a estudiar primeramente las tendencias de los datos en tres niveles temporales: mensualmente, diariamente y por horas. Para ello, se han utilizado los datos resultantes de la agregación de todas las calles, y se han agrupado según el mes, el día o la hora.

En primer lugar, en la figura 4.15 se observa una tendencia ligeramente ascendente a lo largo de los meses. Esto puede ser debido principalmente a que en abril, y en la primera semana de mayo, había restricciones de movilidad debido al estado de alarma, lo que se traduce en que no se podía circular por la calle entre las 00 y las 06 horas, ni se permitía cambiar de comunidad autónoma. El estado de alarma acabó el 9 de mayo y, como es lógico, el flujo de tráfico a partir de este momento se incrementa. No se ha tenido en cuenta el mes de marzo para el estudio de estas gráficas debido a que la API web de la que se han extraído los datos estuvo fuera de servicio 1 semana, por lo que no sería lógico utilizar los datos de marzo debido a que falta una semana completa del mes. No es el caso del mes de junio que, a pesar de que anteriormente se han mostrado los datos hasta el 20 de junio, sí que se dispone de los datos hasta el 30 junio.

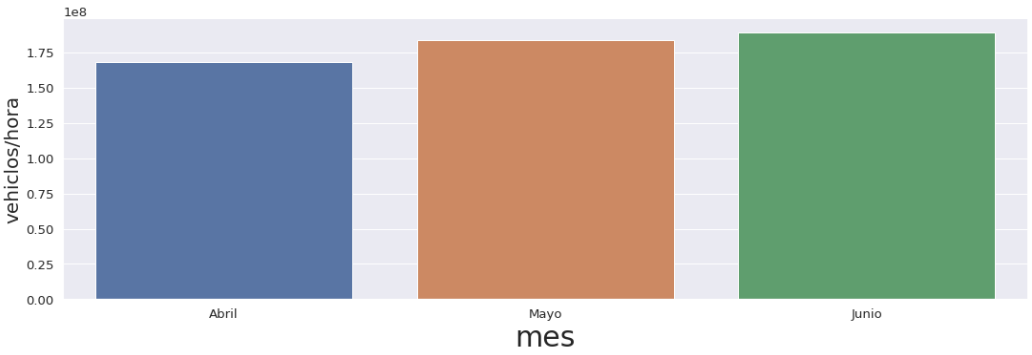


Figura 4.15: Suma de vehículos/hora de todas las calles de Valencia cada mes.

Analizando los datos en función del día, en la figura 4.16 se observa que el flujo de tráfico es mayor entre semana que los fines de semana. Esto puede ser

debido a que la gente utiliza más el coche entre semana para ir a trabajar. Durante el fin de semana hay personas que se trasladan a segundas residencias, o bien las que residen en la ciudad prefieren utilizar otro medio de transporte, o no utilizar ningún tipo de vehículo e ir andando.

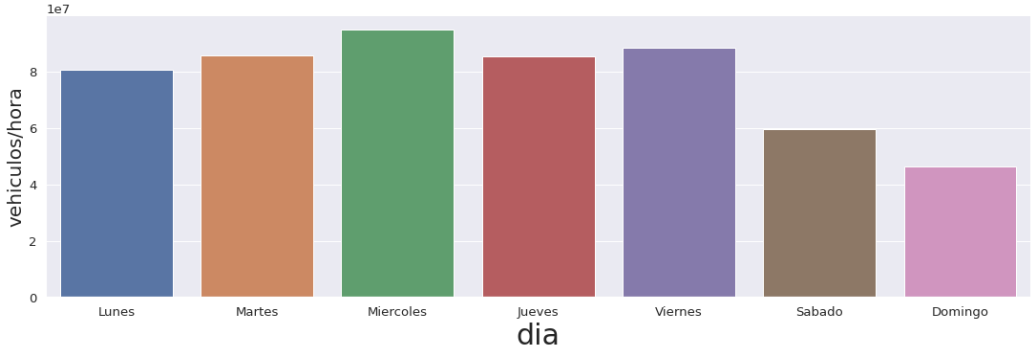


Figura 4.16: Suma de vehículos/hora de todas las calles de Valencia cada día.

Finalmente, en la figura 4.17 se muestran los datos por horas en los que se aprecia, como es lógico, que hay un flujo de tráfico mucho menor durante la noche que durante el día y, además, hay que considerar que estuvieron presentes las restricciones por el estado de alarma previamente comentadas. Por otro lado, se observa que las horas más concurridas son aquellas en las que la gente suele entrar o salir de trabajar, a las 08:00, 09:00, 14:00, 18:00 o 19:00 horas

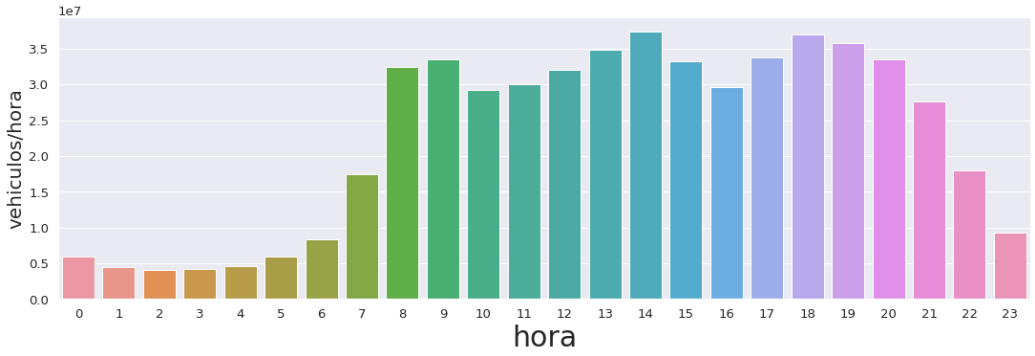


Figura 4.17: Suma de vehículos/hora de todas las calles de Valencia cada hora.

### 4.2.2 Estacionalidad

Para estudiar la estacionalidad, se han agregado los datos de todas calles para cada una de las horas de la serie temporal. A continuación, se ha hecho uso del método `seasonal_decompose` de la librería `statsmodels`. Este método

descompone la serie temporal en 4 gráficas: (1) datos observados; (2) tendencia; (3) estacionalidad y (4) el error, que se obtiene restando la estacionalidad a la serie temporal original. Los modelos utilizados para realizar este análisis son aditivos. No se han podido utilizar modelos multiplicativos ya que, en la serie temporal, existen valores iguales a 0, lo cual impide crear estos modelos y además, como se ha dicho previamente, no se aprecia un crecimiento notable ni en amplitud ni en longitud en los datos, por lo que es mejor elección utilizar un modelo aditivo.

Al igual que en el apartado anterior, se ha estudiado la frecuencia mensual, semanal y diaria. En la figura 4.18 se muestra la serie temporal completa que se va a descomponer.

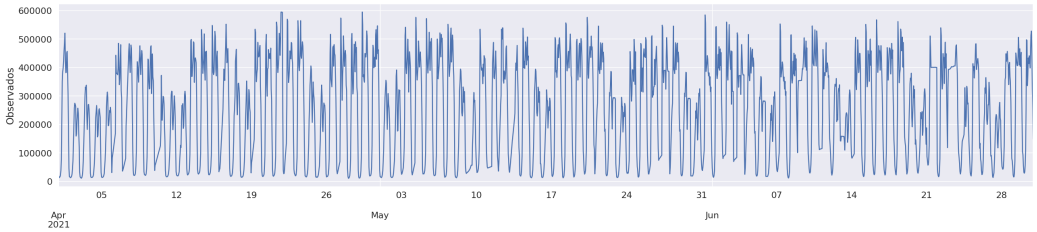


Figura 4.18: Serie temporal completa con datos agregados.

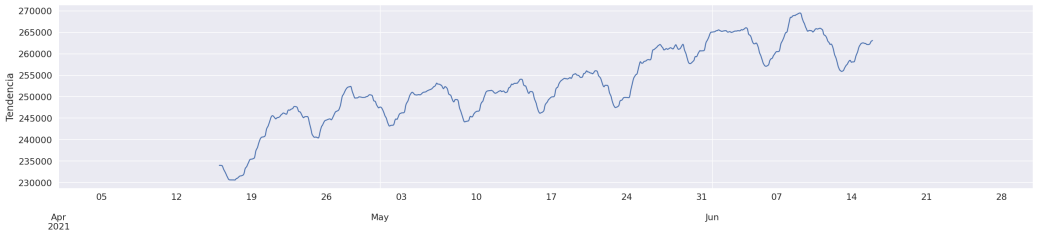


Figura 4.19: Tendencia de los datos a nivel mensual.

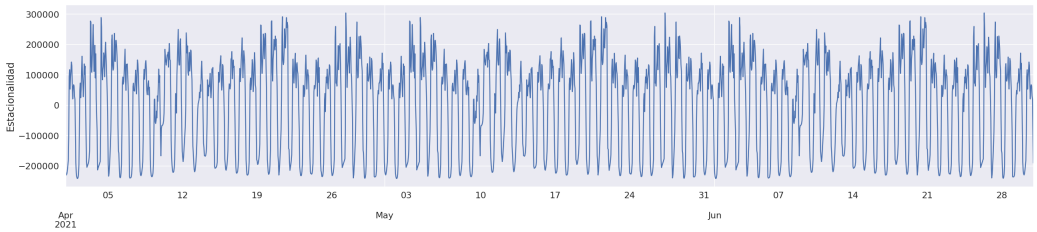


Figura 4.20: Estacionalidad de los datos a nivel mensual.


Figura 4.21: Error de los datos a nivel mensual.

En primer lugar, estudiamos la serie con frecuencia mensual. La figura 4.19 muestra la tendencia de los datos a nivel mensual donde, al igual que se apreciaba en la figura 4.15, se ve que sigue una tendencia ascendente, aunque esta no es muy pronunciada, teniendo en cuenta los valores el eje Y.

En la figura 4.20 se muestra la estacionalidad con frecuencia mensual. La estacionalidad puede verse como una fluctuación sobre la tendencia. Una estacionalidad positiva indica valores por encima de la tendencia, la estacionalidad cero indica valores iguales a la tendencia y la estacionalidad negativa indica valores por debajo de la tendencia. La estacionalidad muestra, por lo tanto, las subidas y bajadas de los datos. En general, puede verse que la gráfica es muy fluctuante con ondas definidas tanto en el margen superior como el inferior.

Para acabar con la frecuencia mensual, en la gráfica 4.21 se muestra el error. En esta gráfica puede verse que, al principio, en el mes de abril, el patrón del error es más uniforme; sin embargo, en los meses siguientes se hace más variable, y las fluctuaciones son más amplias.


Figura 4.22: Tendencia de los datos con frecuencia semanal.

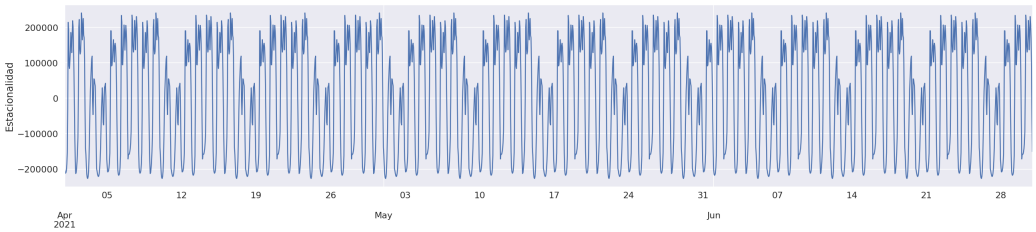


Figura 4.23: Estacionalidad de los datos con frecuencia semanal

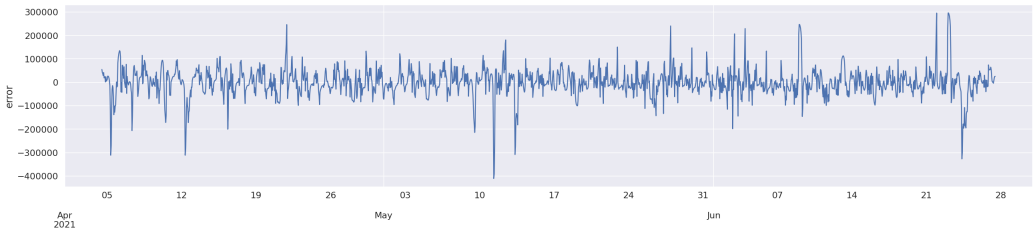


Figura 4.24: Error de los datos a con frecuencia semanal.

Si se descompone la serie con frecuencia semanal (ver figura 4.22) se observa una tendencia ascendente, aunque al final de la serie temporal las últimas semanas del mes de junio muestran un flujo de tráfico menor que las anteriores. No obstante, aunque esta tendencia ascendente es algo superior a la apreciada en la frecuencia mensual, tampoco parece ser muy notable.

Observando la estacionalidad semanal, se aprecia que, al inicio, el flujo es mayor que al final. La gráfica 4.23 comienza el día 1 de abril, que es jueves, y acaba el día 30 de junio, que es miércoles. Puede apreciarse que hay días con una estacionalidad más baja, que corresponderían a los fines de semana (el primer caso de esto serían los días 3 y 4 de abril), e inclusive el lunes es algo más bajo que el resto. Esto concuerda con lo visto anteriormente en la figura 4.16, dónde se aprecia que en los días entre semana hay un mayor flujo que en los días del fin de semana.

Para acabar con el estudio con frecuencia semanal, en la figura 4.24 se muestra la gráfica con el error para los datos con dicha frecuencia. En este caso, las fluctuaciones no son tan pronunciadas como con la frecuencia mensual. De hecho, las fluctuaciones se mantienen alrededor del 0, a pesar de que sí hay ciertas oscilaciones muy pronunciadas, que son minoritarias.

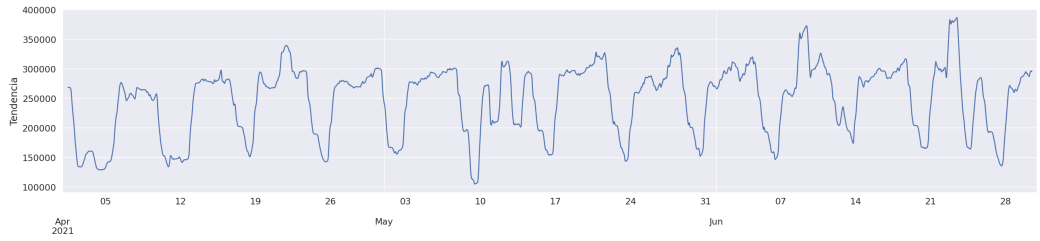


Figura 4.25: Tendencia de los datos con frecuencia diaria

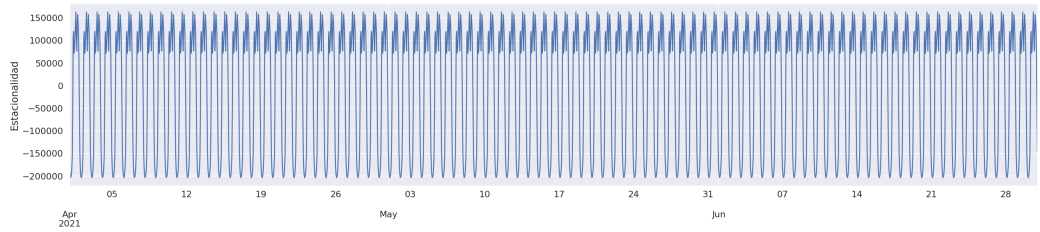


Figura 4.26: Estacionalidad de los datos con frecuencia diaria

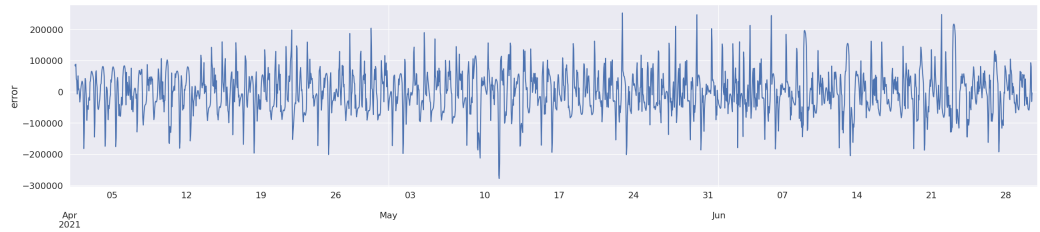


Figura 4.27: Error de los datos con frecuencia diaria

Para acabar el estudio de la estacionalidad, en la figura 4.25 se muestra la tendencia de los datos con frecuencia diaria. En este caso, la tendencia ascendente no es tan clara; sin embargo, sí puede apreciarse que los datos van creciendo, aunque en menor medida que en los casos anteriores. Además, pueden apreciarse más fluctuaciones que en los anteriores casos, pues al ser menor la frecuencia, aquí se aprecia lo comentado previamente de que los fines de semana hay menor flujo de tráfico que los días entre semana.

En cuanto a la estacionalidad, se aprecian las subidas y bajadas en los datos que reflejan lo que ya se observó en la gráfica 4.17, dónde se ve que en las horas de la noche hay un menor flujo de tráfico que en las horas del día, así como los picos en las horas dónde la gente entra o sale de trabajar.

Finalmente, la figura 4.27 muestra el error correspondiente a la frecuencia diaria de los datos. Puede observarse un comportamiento similar al que ocurría con

la frecuencia semanal: el error se mantiene centrado alrededor de 0 y no fluctúa en exceso. A simple vista, puede parecer que el error en este caso fluctúa más que con la frecuencia semanal; sin embargo, observando los índices de ambas gráficas, los valores del eje Y para frecuencia diaria oscilan entre -300.000 y +200.000, mientras que, con frecuencia semanal, oscilan entre -400.000 y +300.000, por lo que las fluctuaciones son menores en el caso de la frecuencia diaria.

Como conclusión a este apartado, analizando el error, unas fluctuaciones menores en el mismo implican una mejor descomposición de la serie temporal. Esto significa que el modelo se ajusta mejor a los datos y, por lo tanto, es posible obtener predicciones más exactas. Con esto en mente, se puede concluir que la peor de las tres descomposiciones es la frecuencia mensual, mientras que, la frecuencia semanal y diaria son mejores opciones, por lo que a partir de este apartado, no se va a tener más en cuenta la frecuencia mensual para el análisis. Además, entre la frecuencia semanal y la frecuencia diaria, parece que la frecuencia diaria es incluso mejor opción.

### 4.2.3 Estacionariedad

El siguiente paso es estudiar la estacionariedad de la serie temporal. Una serie es estacionaria si su media y varianza son constantes a lo largo del tiempo.

Para realizar este estudio, se ha utilizado el método del test de Dickey-Fuller. Este método es una prueba estadística denominada de raíz unitaria. La intuición subyacente a esta prueba es que permite determinar la intensidad con la que una serie temporal está definida por una tendencia.

La hipótesis nula de la prueba es que la serie temporal puede ser representada por una raíz unitaria, que no es estacionaria, es decir, que tiene alguna estructura dependiente del tiempo. La hipótesis alternativa, que rechaza la hipótesis nula, implica que la serie temporal es estacionaria.

Se interpreta el resultado utilizando el p-valor de la prueba. Un p-valor por debajo de un umbral, como el 5 % o el 1 %, indica que se rechaza la hipótesis nula, y, por lo tanto, la serie es estacionaria; por lo contrario, un p-valor por encima del umbral indica que no se rechaza la hipótesis nula, y, por lo tanto, que la serie no es estacionaria. Este umbral puede variar en función de los intereses; en este caso se ha definido que un valor p mayor a 0,05 confirma la hipótesis nula, y los datos serían no estacionarios, y un valor menor a 0,05 rechaza la hipótesis nula, y por lo tanto los datos serían estacionarios.

Para realizar esta prueba se ha hecho uso de la librería `statsmodel` [23] de Python. Se ha realizado este experimento para las mismas frecuencias temporales que en apartados anteriores: diaria, semanal y mensual.

Este estudio se va a realizar sobre los 2 datos de los errores obtenidos en el apartado anterior, con frecuencia diaria y semanal.

En primer lugar, para el error con frecuencia diaria, se obtienen los siguientes valores:

- ADF Stadistic: -16,4220
- P-Value: 0,00
- Valores críticos:
  - 1 %: -3,433
  - 5 %: -2,863
  - 10 %: -2,567

Cuanto más positivo es el resultado del test, el valor estadístico Dickey-Fuller aumentado (ADF Stadistic por sus siglas en inglés), más posibilidades de aceptar la hipótesis nula y, por lo tanto, de que los datos no sean estacionarios. Por el contrario, cuanto más negativo sea, mayor probabilidad de que los datos sean estacionarios. Se puede ver que el valor ADF es menor que los valores críticos, lo cual significa que se puede rechazar la hipótesis nula y, por lo tanto, que la serie temporal es estacionaria. Los valores críticos hacen referencia a los intervalos de confianza mediante los cuales la hipótesis será rechazada o no. Por ejemplo, el valor crítico de 1 %, permitiría rechazar o aceptar la hipótesis nula con una confianza del 99 %, el 5 % con una confianza del 95 % y el 10 % con una confianza del 90 %. Estos valores críticos suelen estar precalculados y dependen del número de datos de los que se dispone, y de si se ha eliminado la tendencia de los datos o no. Si el valor estadístico es más negativo que el valor crítico, entonces se rechaza la hipótesis nula, lo que indica que los datos son estacionarios; en caso contrario, la hipótesis se aceptaría, y los datos no serían estacionarios.

Otro indicativo de que la serie es estacionaria es el p-valor que, en este caso, tiene un valor igual a 0. Esto es un indicativo de que se rechaza la hipótesis nula. Con estos 3 indicativos, se puede concluir que los datos son estacionarios para la frecuencia diaria.

En cuanto a la frecuencia semanal, se obtienen los siguientes valores:

- ADF Stadistic: -9,8807
- P-Value: 0,00
- Valores críticos
  - 1 %: -3,434
  - 5 %: -2,863
  - 10 %: -2,568

Como se puede ver, ocurre lo mismo que con la frecuencia diaria; sin embargo, el valor estadístico es mayor.

De los resultados obtenidos se puede deducir que, a medida que se aumenta la frecuencia, los datos se vuelven menos estacionarios, aunque en estas 2 frecuencias, todos son estacionarios. Esto se puede justificar ya que, a pesar de que la tendencia de los datos es ascendente, esta no es muy pronunciada. Uno de los motivos de este crecimiento es el fin del estado de alarma comentado previamente.

Si bien no se dispone de datos del mes de julio, se esperaría que, en caso de disponer de ellos, la serie se volviese más estacionaria, ya que los meses de marzo y abril pueden considerarse como meses excepcionales, debido al estado de alarma.

### 4.2.4 Auto-correlación

En este apartado se va a estudiar la auto-correlación, ya que es una herramienta potente de análisis para modelar los datos de las series temporales. La auto-correlación es una representación matemática del grado de similitud entre una serie temporal y una versión desfasada temporalmente de ella misma en intervalos de tiempo sucesivos. La diferencia con la correlación es que, en lugar de calcularse entre dos características diferentes, se calcula con una versión atrasada de si misma. En primer lugar, en la figura 4.28 se muestra la auto-correlación de los datos hasta un valor de atraso de 168, que correspondería a 1 semana (24 horas $\times$ 7 días). Como se puede ver en la gráfica, los atrasos pequeños tienden a tener correlaciones más altas. Además, se ve que hay atrasos que obtienen una correlación negativa. Una auto-correlación de +1 indica una auto-correlación positiva perfecta, lo que implica que un aumento en un valor de una serie supone un aumento en un valor de la otra serie retardada. El valor que tiene exactamente +1 es el primero de la gráfica, y corresponde a la auto-correlación de la serie temporal consigo misma, sin ningún desfase temporal. Por otro lado, un valor -1 indica una auto-correlación negativa perfecta, lo que implica que un aumento en un valor de una serie supone un decremento en un valor de la otra serie retardada.

Finalmente, en la figura 4.29, se muestra la correlación hasta un valor de atraso de 24, equivalente a un día. Al igual que en el caso anterior, existen valores de auto-correlación positivos y negativos, y los atrasos pequeños tienen una auto-correlación más alta. Parece que la gráfica es igual a la anterior, pero con menor amplitud, habiendo 24 valores en lugar de 168 valores.

La capacidad de comparar la relación entre puntos de datos pasados y presentes presenta una ventaja única. Si se puede asociar el valor actual a $k$ periodos anteriores, esto significa que se puede encontrar también una relación con los valores que vienen después de $k$ periodos de tiempo.

Para concluir este apartado, se puede deducir de las gráficas 4.29 y 4.28 que

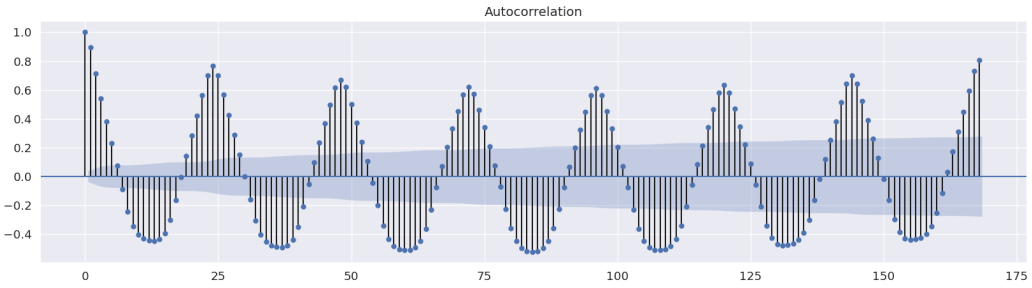


Figura 4.28: Auto-correlación de los datos con retraso semanal.

los atrasos pequeños son una buena elección a tener en cuenta en el futuro modelo de predicción. El primer punto, como se ha comentado previamente, corresponde a la auto-correlación de la serie consigo misma, el segundo punto a un desfase de 1 hora, etc. El siguiente punto, que supera el valor de 0.95, es el segundo valor, que corresponde a un desfase temporal de 1 hora. Con un desfase de 2 horas se obtiene un valor de alrededor de 0.75, los cuales son unos valores aceptables. A partir de estos desfases, la auto-correlación ya no es tan alta, por lo que no se presentan como buenas opciones para el modelo de predicción.

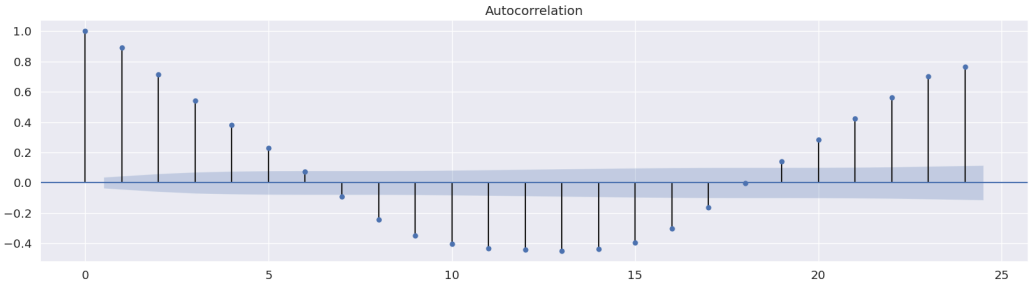


Figura 4.29: Auto-correlación de los datos con retraso diario.

# Capítulo 5

# Implementación del modelo

En este capítulo se realizará una implementación del modelo neuronal basado en una red LSTM. Además, se ofrece una explicación de este tipo de redes de cara a ayudar a la comprensión del modelo implementado.

## 5.1 Redes neuronales LSTM

Las redes de memoria a largo plazo, o simplemente LSTMs, son un tipo especial de redes neuronales recurrentes capaces de aprender dependencias a largo plazo. Fueron introducidas por Hochreiter y Schmindhuber [24] en 1997, y posteriormente han sido popularizadas y mejoradas por la comunidad científica. Este tipo de redes fueron diseñadas explícitamente para paliar la limitación de las redes neuronales recurrentes que solo son capaces de aprender dependencias temporales a corto plazo.

Esta capacidad de recordar dependencias temporales a largo plazo es la que hace a este tipo de redes una buena solución a problemas en los que los datos forman una serie temporal, como es el caso del flujo de tráfico.

La arquitectura de las redes LSTM puede verse en la figura 5.1.

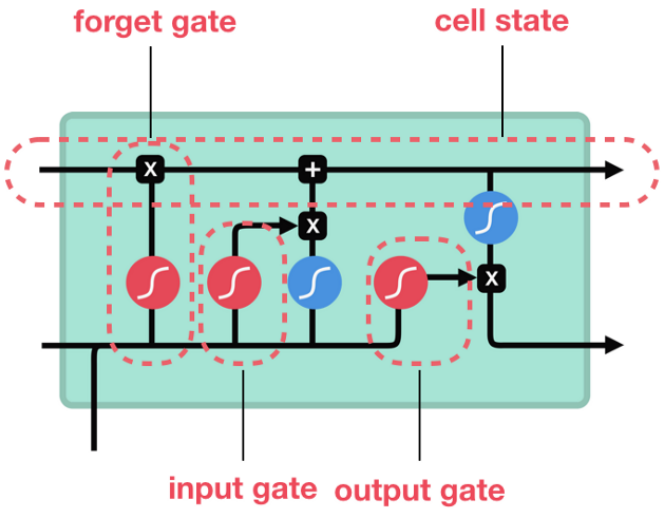


Figura 5.1: Arquitectura de una red LSTM.

Una capa LSTM consiste en un conjunto de bloques conectados de forma recurrente que reciben el nombre de bloques de memoria. Cada uno de ellos contiene unos mecanismos llamados puertas que regulan la cantidad de información que es recordada, y cual es olvidada.

La celda de estado actúa como una autopista que transporta la información a lo largo de los bloques, por lo que puede considerarse como la memoria de la red. De esta manera, permite que la información dé los primeros pasos temporales sea capaz de llegar a los más posteriores, reduciendo los efectos de la memoria a corto plazo.

A medida que la celda de estado avanza, la información se añade o elimina a través de las puertas. Las puertas pueden ser consideradas como diferentes redes neuronales que deciden la información que entra en la celda de estado. Esto es importante ya que les permite ser capaces de aprender qué información es relevante para mantener u olvidar durante el entrenamiento.

En primer lugar, la puerta de olvido es la encargada de decidir qué información se va a olvidar o recordar. Por esta puerta entra la información del estado oculto anterior, así como la información de la entrada actual, y después pasan por la función sigmoide, que toma valores entre 0 y 1. Cuanto más cerca esté de 0, más olvida la información, y cuanto más cerca de 1, más información recuerda.

Para actualizar la celda de estado, se dispone de la puerta de entrada. Primero, pasa el estado oculto anterior y la entrada actual a una función sigmoide, para decidir qué valores se actualizarán y cuales no. Además, se pasa el estado oculto y

la entrada actual a una función tangente hiperbólica, para transformar los valores al rango de $-1$ a $+1$. Finalmente, la salida de esta tangente hiperbólica se multiplica con la salida de la función de activación sigmoide, siendo la sigmoide la que decide cuál es la información a recordar, y cuál olvidar.

Con estas dos partes, ya se tiene información suficiente para actualizar la celda de estado. Primeramente, la celda de estado y la puerta de salida son multiplicadas elemento a elemento (*element-wise multiplication*). Con esto se permite olvidar información si se multiplica por valores cercanos a 0. Después se realiza una suma elemento a elemento de los valores de la salida de la multiplicación previa con los elementos de la puerta de entrada. Finalmente, la salida de esta suma es el nuevo estado de la celda de estado.

Por último, se tiene la puerta de salida, que es la encargada de decidir qué información debe ser transferida al siguiente estado oculto. Es importante recordar que el estado oculto contiene información de entradas previas. Primero se pasa el estado oculto y la entrada actual a una función sigmoide. Después pasa el estado de la celda recientemente modificada a la función tangente hiperbólica. A continuación, se multiplica la salida de la función de activación tangente hiperbólica por la salida de la función de activación sigmoide para decidir qué información se transmite al estado oculto. Finalmente, la salida de esta última operación será el estado oculto. La nueva celda de estado y el nuevo estado oculto son transferidos al siguiente bloque LSTM.

Para concluir este primer apartado de introducción, a modo de resumen, la puerta de olvido decide la información que es relevante mantener a lo largo del tiempo y la puerta de entrada decide qué información es relevante para añadir del estado actual; por último, la puerta de salida determina cuál debe ser el siguiente estado oculto.

## 5.2 Implementación del perceptrón multicapa

En primer lugar se va a implementar un modelo inicial, que sirva como referencia para comparaciones con implementaciones futuras. Este modelo inicial está basado en la red neuronal más simple, un perceptrón multicapa (MLP por sus siglas en inglés).

Un MLP es una red neuronal feedforward, sin ningún tipo de recurrencia, que está formada por una capa de entrada, al menos una capa oculta, y una capa de salida en la que, exceptuando los nodos de entrada, cada neurona utiliza una función de activación. En el proceso de aprendizaje se utiliza la técnica de retropropagación del error para aprender los pesos de la red.

Para el desarrollo de las implementaciones se ha hecho uso de Google Colab [25], que permite ejecutar código Python en la nube y proporciona GPUs al

usuario. En cuanto a la implementación de las redes neuronales, se ha elegido la librería Keras [26].

### 5.2.1 Descripción del experimento

Antes de comentar la implementación del MLP, es conveniente explicar el experimento que se va a realizar.

La idea de este primer experimento es, a partir de los datos de todas las calles de Valencia, predecir el flujo de tráfico que habrá en todas y cada una de ellas a la vez. Para ello, lo primero que es necesario realizar es dividir el conjunto de datos en tres conjuntos: entrenamiento, validación y test.

Los datos del conjunto de validación serán utilizados para ajustar los parámetros de la red, entrenando la misma con el conjunto de entrenamiento. En este ajuste, se incluye el estudio del número de capas ocultas, el tamaño de las capas, el tamaño del batch, las épocas de entrenamiento, el valor del factor de aprendizaje, etc. Una vez se tengan los parámetros óptimos de la red, el conjunto de validación será añadido al conjunto de entrenamiento, siendo el resultado de esta adición el nuevo conjunto de entrenamiento. De esta manera, se dispone de más datos para entrenar la red, y los resultados obtenidos sobre el conjunto de test serán los que se utilicen para comparar los diferentes modelos.

Se dispone de datos del flujo de tráfico desde el 15 de marzo de 2021 hasta el 20 de junio de 2021. El conjunto de datos de entrenamiento lo forman los datos comprendidos entre las fechas del 15 de marzo al 17 de mayo, siendo un 65 % del total. Por otro lado, el conjunto de validación lo forman los datos comprendidos entre el 18 de mayo y el 6 de junio, representando un 20 % del total. Finalmente, el conjunto de test está formado por los datos entre el 7 junio y el 20 de junio, representando un 15 % del total.

### 5.2.2 Preparación de los datos

Además de dividir el conjunto de datos, es necesario transformar la serie temporal de manera que los datos puedan ser introducidos en la red neuronal. Esta transformación consiste en convertir la serie temporal en un problema de aprendizaje supervisado, donde se tengan unos vectores de entrada $X$ y un vector de salida $y$.

La manera de realizar esta transformación es utilizando instantes de tiempo anteriores como variables de entrada, y utilizar el siguiente instante de tiempo como variable de salida. En otras palabras, se toman los valores en el instante de tiempo $t$ como valores de entrada, y los valores en el instante $t + 1$ como valores de salida.

Este método se conoce como el método de la ventana deslizante, o método de retardo. El número de instantes de tiempo anteriores que se utilicen se denomina

como el tamaño de la ventana o el valor de atraso. En este experimento se va a utilizar un valor de atraso igual a 2, pues, como se vio en el apartado anterior, es el valor con una mayor auto-correlación. Esto significa que se van a utilizar los valores en el instante $t-1$ y $t$ para predecir los valores en el instante $t+1$.

Existen dos tipos de análisis de series temporales en función del número de muestras observadas: univariante, donde se trata un conjunto de datos en los que sólo se observa una única variable en cada momento o, multivariante, en el que en el conjunto de datos se observan dos o más variables en cada momento.

Como se ha comentado previamente, en este primer experimento se va a predecir el tráfico de todas las calles a la vez, por lo que sería un análisis multivariante, pues se disponen de los datos de todas las calles de Valencia.

En este punto, el vector de entrada tiene una estructura tridimensional, donde la primera dimensión es el número de muestras, es decir, la longitud de la serie temporal, la segunda es el valor de atraso utilizado, y la tercera es el número de características de la serie temporal, que en este caso son las 386 calles disponibles. El formato del vector es [muestras, valor de atraso, características].

El MLP necesita como entrada un vector unidimensional, por lo que es necesario transformar este vector tridimensional en una única dimensión. No obstante, al ser un análisis multivariante, se tienen múltiples vectores, uno para cada paso de tiempo, por lo que la forma final del vector tiene que ser de la forma [muestras, longitud].

La longitud del vector de entrada se puede calcular como el valor de atraso multiplicado por el número de características; después se utiliza este valor para aplanar el vector de entrada. Por ejemplo, si la longitud de la serie temporal es de 1510, el valor de atraso es 2 y se tienen 386 características, la longitud del vector de entrada sería 772 (valor de atraso * características); aplanando el vector, la forma del vector de entrada quedaría de (1510, 772), o lo que es lo mismo, 1510 muestras con 772 valores cada una. De esta manera, el vector de entrada ya está listo para ser introducido al MLP.

### 5.2.3 Descripción del modelo y sus parámetros

La estructura del MLP se muestra en la figura 5.2. En primer lugar, una capa densa de entrada de tamaño 100 y con una dimensión de entrada de 772, que sería el resultado de multiplicar las características, 386 calles, por el valor de atraso, que es igual a 2. A continuación, se tiene una capa oculta densa de tamaño 40, y finalmente una capa de salida densa de tamaño 386, para ser capaz de predecir todas las calles. Tanto la capa de entrada como la capa oculta tienen una función de activación ReLU, mientras que la capa de salida tiene una función de activación lineal.

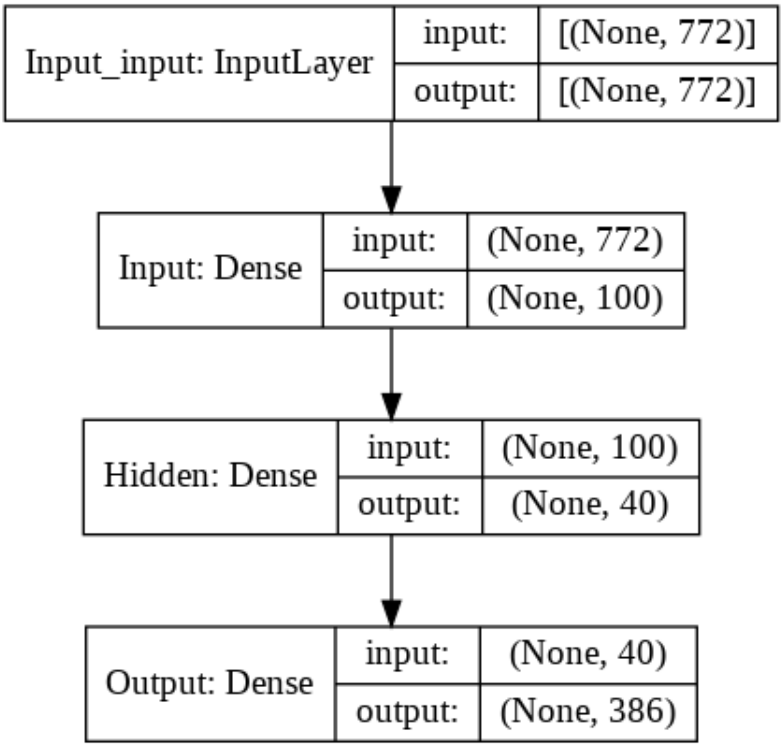


Figura 5.2: Arquitectura utilizada para el MLP.

En cuanto a los parámetros de la red, se ha definido un tamaño de batch de 24, haciendo que cada batch coincida con el tamaño de 1 día, un factor de aprendizaje igual a $10^{-4}$ y, además, se ha utilizado Adam como optimizador del gradiente. Se han definido 200 épocas de entrenamiento como límite; sin embargo, se ha utilizado la técnica del *early stopping* [27] que consiste en, si el resultado en una época de entrenamiento no mejora al resultado obtenido en la época anterior, el entrenamiento acaba. No obstante, se ha utilizado un *patience value*, o valor de paciencia, igual a 3, esto significa que el entrenamiento acaba si el resultado no mejora en 3 épocas de entrenamiento, no solo en una única época. Esta técnica se ha aplicado sobre el conjunto de validación, es decir, si el resultado en el conjunto de validación no mejora tras 3 épocas de entrenamiento, se para el entrenamiento.

Como se ha comentado previamente, estos parámetros se han definido según las pruebas realizadas sobre el conjunto de validación, evitando así un sobre entrenamiento de los datos de test, lo cual no es para nada conveniente para predicciones futuras. Además, debido a la componente aleatoria del proceso de entrenamiento, cada prueba se ejecutó 10 veces y, después, se realizó la media de todas las ejecuciones, siendo dicha media el objeto de comparación entre las diferentes configuraciones de la red.

Se ha visto que, al utilizar una función de activación lineal en la capa de salida, en lugar de una ReLU, el modelo es capaz de predecir valores negativos en algunos casos extremos. No obstante, se probó a utilizar una función de activación ReLU en lugar de una lineal y el resultado fue bastante peor, por lo que finalmente se optó por una función de activación lineal. Ya que es imposible tener un flujo de tráfico negativo, los valores de las predicciones necesitarían algún tipo de post-proceso que transforme estos valores negativos a otros valores, por ejemplo, a 0. Sin embargo esto no es crítico ya que, como se ha dicho, esto ha ocurrido en casos muy extremos donde el flujo de tráfico suele ser 0, lo cual no es normal y solo ocurre en ciertas calles muy pequeñas. Esto ha ocurrido en los modelos expuestos más adelante en este documento.

### 5.2.4 Entrenamiento del modelo

Para poder comparar resultados en diferentes calles, se ha decidido realizar un escalado de los datos del flujo de vehículos en el rango 0 y 1, pues en un avenida grande pueden pasar un número elevado de vehículos por hora y, en una calle pequeña, simplemente pasar unos pocos vehículos. Además, las redes neuronales trabajan mejor si se les otorgan valores en dicho rango como entrada, que introduciéndoles valores muy elevados. Para esto, se ha hecho uso del escalador *MinMaxScaler* disponible en la librería Scikit-Learn [28].

Como métricas del error, para evaluar el resultado del modelo, se han utilizado tres medidas: el error cuadrático medio (MSE por sus siglas en inglés), el error absoluto medio (MAE por sus siglas en inglés), y la raíz del error cuadrático medio (RMSE por sus siglas en inglés).

Con todo ello, se ha obtenido el siguiente resultado tras 37 épocas de entrenamiento:

- MAE conjunto entrenamiento: 0,0683
- MSE conjunto entrenamiento: 0,0118
- RMSE conjunto entrenamiento: 0,1085
- MAE conjunto validación: 0,0776
- MSE conjunto validación: 0,0145
- RMSE conjunto validación: 0,1204

La gráfica de la evolución del MAE puede verse en la figura 5.3, la evolución del MSE en la figura 5.4, y la evolución del RMSE en la figura 5.5

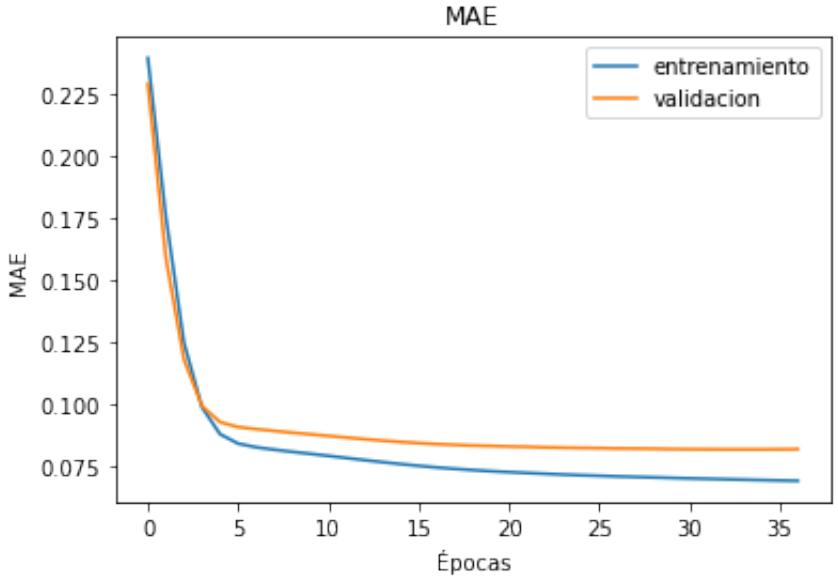


Figura 5.3: Evolución del MAE en el MLP.

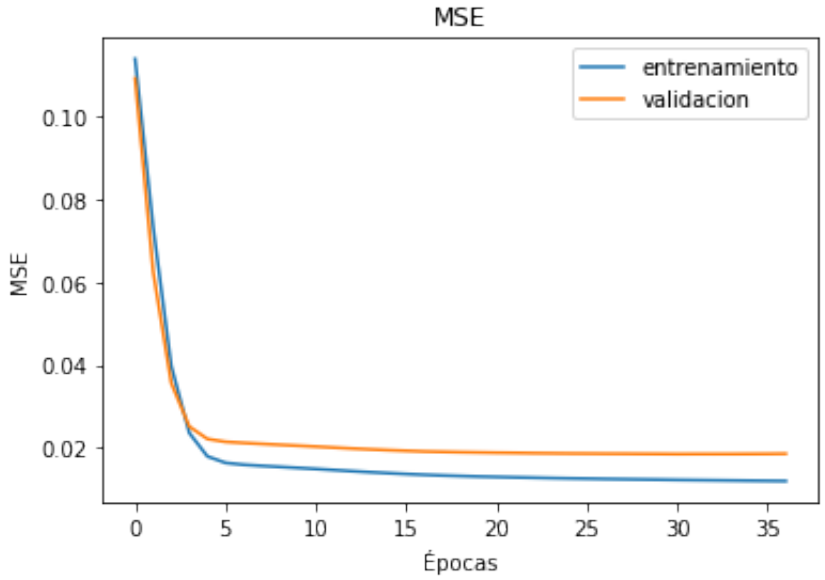


Figura 5.4: Evolución del MSE en el MLP.

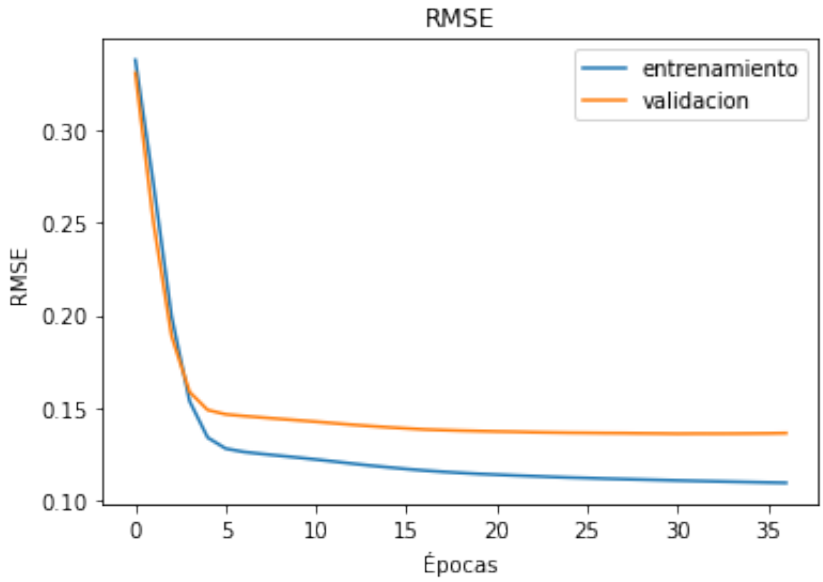


Figura 5.5: Evolución del RMSE en el MLP.

De las gráficas se puede observar que, a partir de aproximadamente 5 épocas de entrenamiento, el resultado mejora muy lentamente. Esto ocurre para las 3 métricas - MAE, MSE y RMSE - y tanto para el conjunto de entrenamiento como para el de validación. Sin embargo, se ha optado por dejar el entrenamiento durante más épocas, y que sea el *early stopping* el encargado de parar el entrenamiento, tras 3 épocas sin mejora, evitando así un sobre entrenamiento de la red.

### 5.2.5 Evaluación del modelo

A continuación, con la red entrenada, se ha realizado la predicción para los datos de test, obteniendo los siguientes resultados:

- MAE conjunto de test: 0,0772
- MSE conjunto de test: 0,0110
- RMSE conjunto de test: 0,1049

Los resultados obtenidos con este conjunto, servirán como punto de partida para realizar mejoras al modelo.

Finalmente, se muestran unas gráficas con algunas predicciones, comparándose con los resultados reales. En la figura 5.6 se muestra la predicción del día 8 de junio de 2021. En la gráfica puede verse que el modelo no es capaz de predecir los

valores más extremos de los valores reales, llegando a estar por debajo en aproximadamente 2000 vehículos en el pico más alto, a las 16 horas, aunque el primer pico de la mañana, a las 9 horas, el modelo ha sobreestimado el valor. Durante el resto del día ambas gráficas siguen más o menos el mismo patrón.

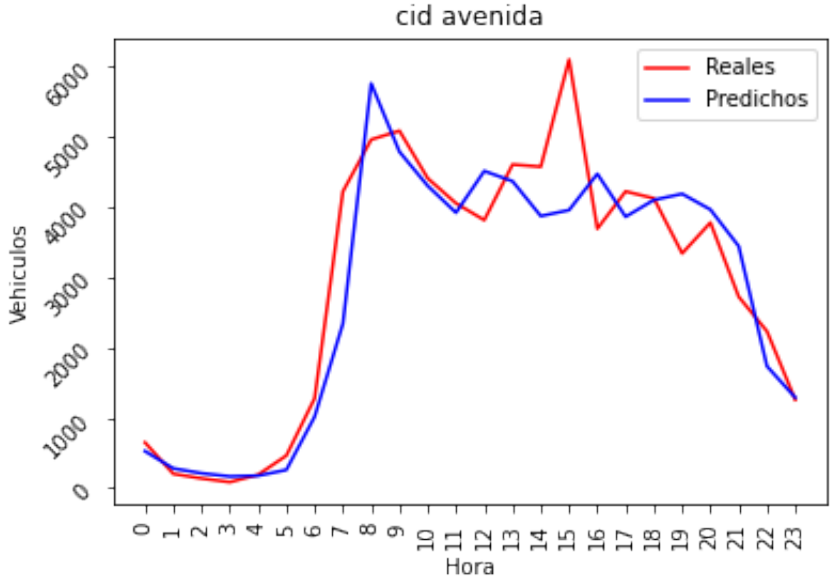


Figura 5.6: Predicción del día 8 de junio en la Avenida del Cid con MLP.

En la figura 5.7 se muestra la comparación del mismo día pero en la calle Santos Justo y Pastor, que es una calle no tan transitada como la Avenida del Cid. En este caso, como el flujo de tráfico no es tan elevado, los valores predichos no se distancian tanto en las horas con más tráfico, aunque parece observarse el mismo suceso, el modelo no es capaz de predecir los valores más altos; por ejemplo, a las 8 de la mañana o a las 18 horas, ambos valores se distancian bastante en comparación al resto de horas del día. No obstante, en general, la predicción parece buena.

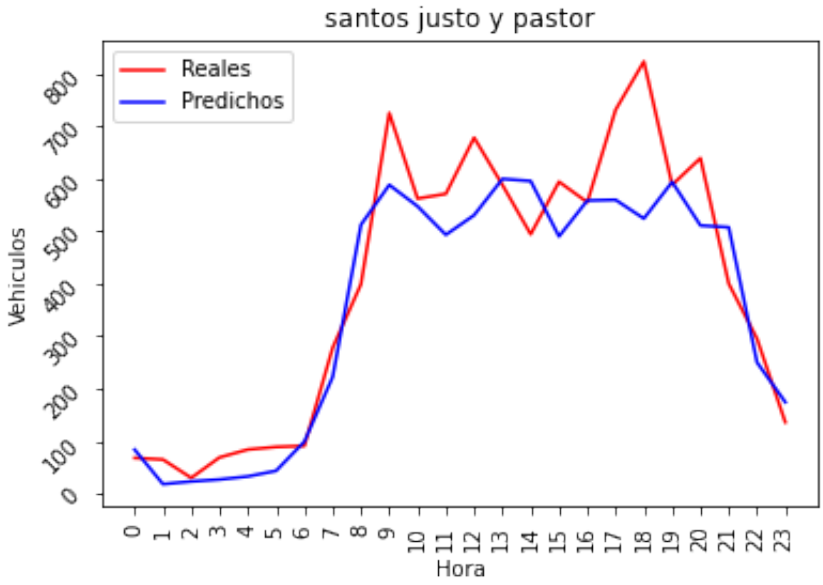


Figura 5.7: Predicción día 8 de junio en la calle Santos Justo y Pastor con MLP

Finalmente, en la gráfica 5.8, aparece la predicción del mismo día que en las gráficas anteriores, ahora para la calle Barón de San Petrillo, una calle pequeña no muy transitada del barrio de Benimaclet. En este caso, se aprecia que la predicción no se ajusta tanto como se ajustaba en las figuras anteriores; de hecho, la gráfica tiene muchos más picos que los anteriores casos, y el modelo no es capaz de predecirlos tan acertadamente.

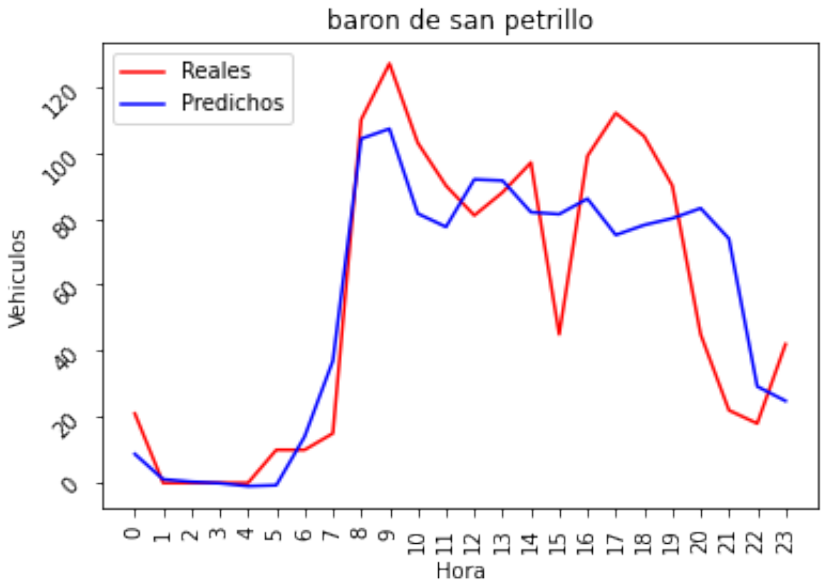


Figura 5.8: Predicción día 8 de junio en la calle Barón de San Petrillo con MLP.

Para concluir este apartado, de estas 3 gráficas se puede intuir que es posible que, para predecir el flujo de tráfico en unas calles muy transitadas como la Avenida del Cid, no es necesario el flujo de tráfico en otras calles menos transitadas, ya que el modelo no es capaz de llegar a predecir valores de tráfico tan elevados. Esto será objeto de estudio más adelante.

## 5.3 Implementación LSTM

En este apartado se va a desarrollar una implementación de una red LSTM, con el objetivo de mejorar los resultados obtenidos previamente con la implementación MLP.

El primer experimento es el mismo que se ha realizado en el apartado anterior: predecir el flujo de tráfico en las 386 calles de Valencia en función del tráfico de todas y cada una de ellas, siendo el valor de atraso igual a 2.

### 5.3.1 Preparación de los datos

Como en el apartado anterior, es necesario un proceso de transformación de los datos de la serie temporal convirtiéndolos en un problema de aprendizaje supervisado, creando los vectores de entrada $X$ y un vector de salida $y$, aunque, en este caso concreto, el proceso es ligeramente diferente. Para la creación de los vectores se ha utilizado el mismo método que en el apartado anterior (método de la ventana

deslizante), por lo que el vector de entrada adopta el siguiente formato [muestras, valor de atraso, características]. Sin embargo, las redes LSTM, al contrario que las redes MLP, no necesitan como entrada un vector unidimensional; concretamente, necesitan de un vector tridimensional, con la forma que ya tiene el vector de entrada, por lo que, en este caso, no es necesario aplanar el vector como sí ocurrió en el MLP.

### 5.3.2 Descripción del modelo y sus parámetros

La arquitectura elegida para la red LSTM ha sido la mostrada en la figura 5.9. En primer lugar, una capa de entrada de tipo LSTM con 150 unidades y con la forma (1510, 2, 386), para que pueda ser introducido el vector de entrada que, como se ha dicho, adopta el formato (muestras, valor de atraso, características). Conectada a la capa de entrada, se ha añadido una capa de dropout con un valor de 0,2. Las capas dropout se encargan de desactivar un número de neuronas de forma aleatoria, en función del valor introducido. En este caso, cada neurona tendría un 20 % de probabilidades de desactivarse, de manera que no tendrían influencia para el proceso de entrenamiento de la red, ayudando así a evitar el sobre entrenamiento de la red. A continuación, aparece una capa oculta densa de 24 unidades y, finalmente, una capa densa de salida de tamaño 386. Las funciones de activación, exceptuando la capa de salida que es una función lineal, son de tipo ReLU.

Los parámetros utilizados han sido un tamaño de batch de 24, un factor de aprendizaje igual a $10^{-4}$, Adam como optimizador del gradiente y un límite de 200 épocas de entrenamiento, aunque al utilizar la técnica del *early stopping*, con un valor de paciencia igual a 3, es posible que el entrenamiento acabe antes.

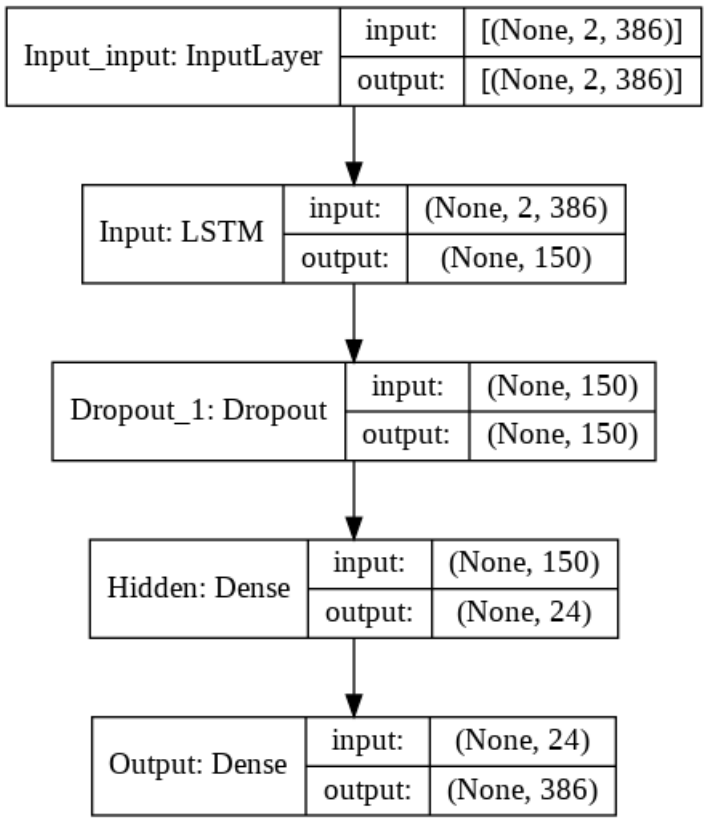


Figura 5.9: Arquitectura utilizada para la red LSTM.

Al igual que con el estudio de los parámetros del MLP, esta configuración se ha elegido en base a los resultados obtenidos con el conjunto de validación en 10 ejecuciones, debido a la componente aleatoria del proceso de entrenamiento, y después, comparando la media de los mismos, y guardando la configuración del mejor resultado obtenido. Al acabar este paso, el conjunto de validación también se ha añadido al conjunto de entrenamiento.

Se probaron diferentes configuraciones en busca de la mejor configuración. Por ejemplo, se probó a añadir y quitar más capas LSTM, añadir más capas densas, capas de dropout, el tamaño de las capas, etc. Otra de las pruebas fue sustituir la red LSTM normal por una red LSTM bidireccional, que son una ampliación de las LSTM donde se entrenan dos LSTM, en lugar de una, con la secuencia de entrada. La primera utiliza la secuencia de entrada tal cual, y la segunda una copia invertida de la secuencia de entrada, lo cual puede proporcionar un contexto adicional a la red. Sin embargo, la arquitectura que mejores resultados obtuvo fue la descrita en este apartado.

### 5.3.3 Entrenamiento del modelo

El entrenamiento de la red se ha realizado con los parámetros comentados previamente y, además, se ha realizado un escalado de los datos en el rango [0, 1]. Al igual que en el MLP, se se han utilizado las mismas 3 métricas para el error, el MSE, MAE y RMSE. Tras 40 épocas de entrenamiento, se han obtenido los siguientes resultados:

- MAE conjunto entrenamiento: 0,0703
- MSE conjunto entrenamiento: 0,0123
- RMSE conjunto entrenamiento: 0,1111
- MAE conjunto validación: 0,0801
- MSE conjunto validación: 0,0154
- RMSE conjunto validación: 0,1241

Comparando estos resultados con los obtenidos en el MLP, se puede ver que no son tan diferentes. En cuanto al conjunto de entrenamiento, las 3 métricas son más bajas, lo cual puede verse en la tabla 5.1 y 5.2, y por lo tanto alcanzan mejores resultados, en el MLP que en la red LSTM; sin embargo estas diferencias no son muy elevadas. En el conjunto de validación ocurre lo mismo: las 3 métricas son menores en la red MLP que en la LSTM. Ambos entrenamientos han necesitado de un número de épocas similar: 37 la red MLP, y 40 la red LSTM.

| Modelo | MAE | MSE | RMSE |
|---|---|---|---|
| MLP | 0,0683 | 0,0118 | 0,1085 |
| LSTM | 0,0703 | 0,0123 | 0,1111 |

Tabla 5.1: Comparativa resultados MLP y LSTM sobre el conjunto de entrenamiento.

| Modelo | MAE | MSE | RMSE |
|---|---|---|---|
| MLP | 0,077 | 0,0145 | 0,1361 |
| LSTM | 0,0801 | 0,0145 | 0,1204 |

Tabla 5.2: Comparativa resultados MLP y LSTM sobre el conjunto de validación.

En las figuras 5.10, 5.11 y 5.12 se muestra la evolución del MAE, MSE y RMSE, respectivamente, para los conjuntos de validación y entrenamiento. Al igual que ocurría con el MLP, más allá de aproximadamente 5 épocas, el aprendizaje se vuelve más lento; de hecho, el error en el conjunto de validación llega a ser menor que en el de entrenamiento en alguna de las primeras épocas. La convergencia es más o menos igual de rápida con la red LSTM que con la red MLP. No obstante, para poder decir que un modelo es mejor que otro, se van a estudiar los resultados sobre el conjunto de test más adelante.

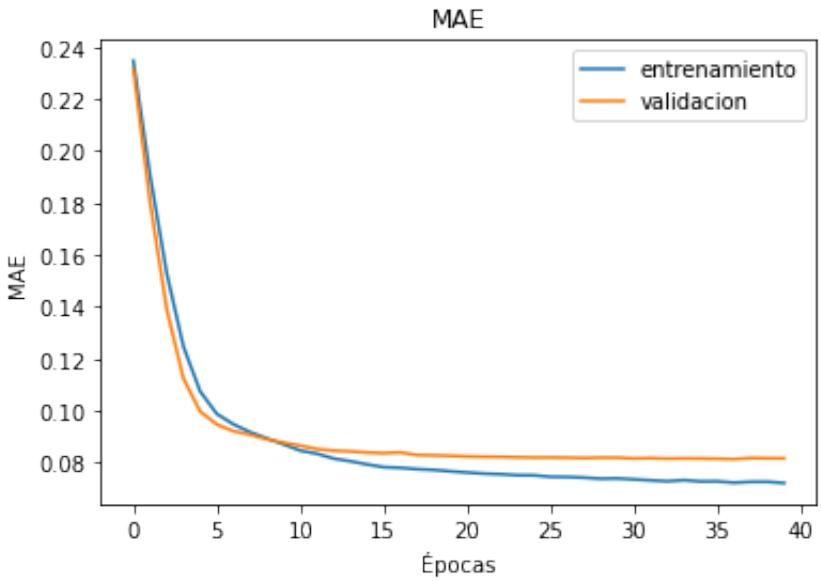


Figura 5.10: Evolución del MAE en la red LSTM.

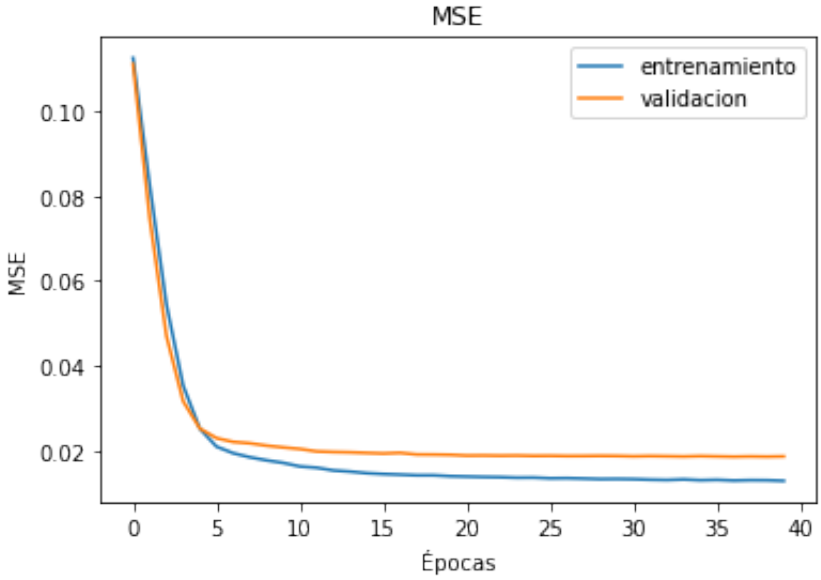


Figura 5.11: Evolución del MSE en la red LSTM.

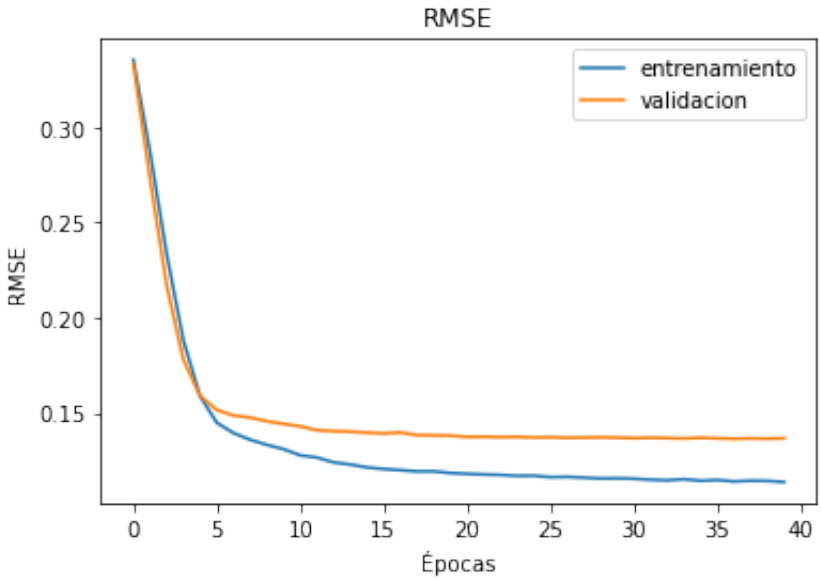


Figura 5.12: Evolución del RMSE en la red LSTM.

### 5.3.4 Evaluación del modelo

Se ha realizado la predicción sobre el conjunto de test con la red LSTM entrenada, obteniendo los siguientes resultados:

- MAE conjunto de test: 0,0760.
- MSE conjunto de test: 0,0108.
- RMSE conjunto de test: 0,1039.

Comparando los resultados con los obtenidos en el MLP, se ve que, para el conjunto de test, el modelo basado en la red LSTM ha obtenido mejores resultados, aunque la mejora no es muy elevada. Esto puede verse en la tabla 5.3. La mejora obtenida con la red LSTM ha sido de un 1,6 % tomando el MAE como referencia, de un 1,8 % con el MSE y de un 0,9 % para el RMSE.

Esto hace pensar que es posible que la red MLP se haya ajustado más a los datos del conjunto de entrenamiento, quizás llegando al sobre entrenamiento, mientras que la LSTM, gracias al dropout entre otras cosas, no ha sufrido ese sobre entrenamiento y obtiene mejores resultados para el conjunto de test.

| Modelo | MAE | MSE | RMSE |
|---|---|---|---|
| MLP | 0,0772 | 0,0110 | 0,1049 |
| LSTM | 0,0760 | 0,0108 | 0,1039 |

Tabla 5.3: Comparativa resultados MLP y LSTM sobre el conjunto de test.

A continuación, se presentan las predicciones para el día 8 de junio de 2021 en las mismas 3 calles que se mostraron en el apartado anterior. En la figura 5.13 se muestra la comparación de los datos reales, y la predicción el 8 de junio. Ocurre algo similar a lo que ocurrió con el MLP: el modelo no es capaz de llegar a predecir valores tan elevados. Aun así, las gráficas parecen seguir un mismo patrón. Si se compara con la obtenida con el MLP, esta predicción se antoja algo mejor.

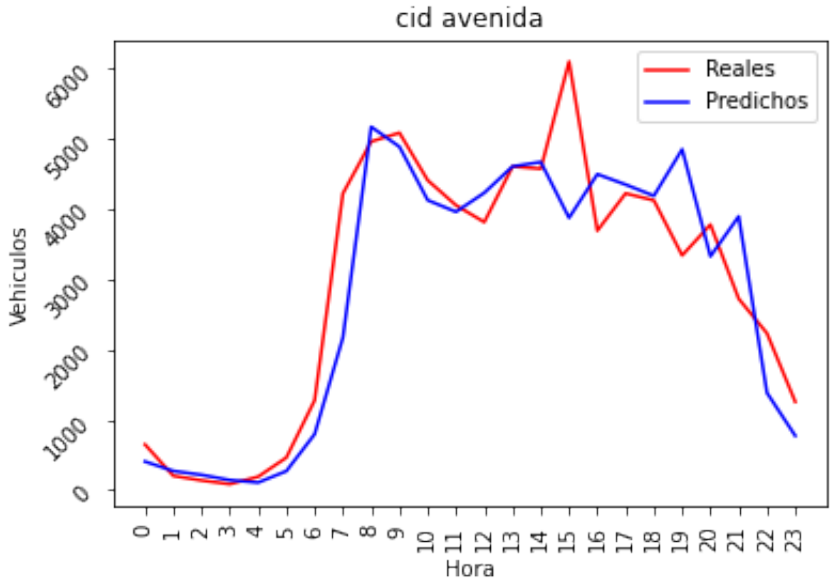


Figura 5.13: Predicción para el día 8 de junio en la Avenida del Cid usando LSTM.

En la figura 5.14 se muestra la predicción para el mismo día en la calle Santos Justo y Pastor. Al igual que en la figura anterior, si se compara con la obtenida en la red MLP, ocurre algo similar: los picos más elevados están subestimados, y para el resto del día ambas gráficas siguen un patrón similar. A grandes rasgos, la predicción parece buena.

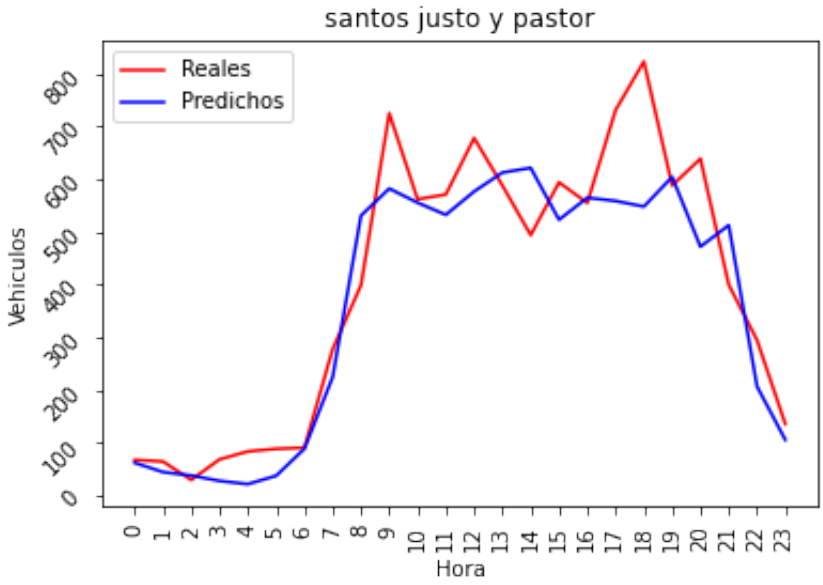


Figura 5.14: Predicción para el día 8 de junio en la calle Santos Justo y Pastor usando LSTM.

Finalmente, la figura 5.15 muestra la predicción en la calle Barón de Petrillo. Como ha ocurrido en las calles anteriores, los resultados obtenidos son similares a los resultados obtenidos con el MLP: el modelo no es capaz de predecir bien tantas fluctuaciones en el flujo del tráfico, y parece la peor predicción de las tres.

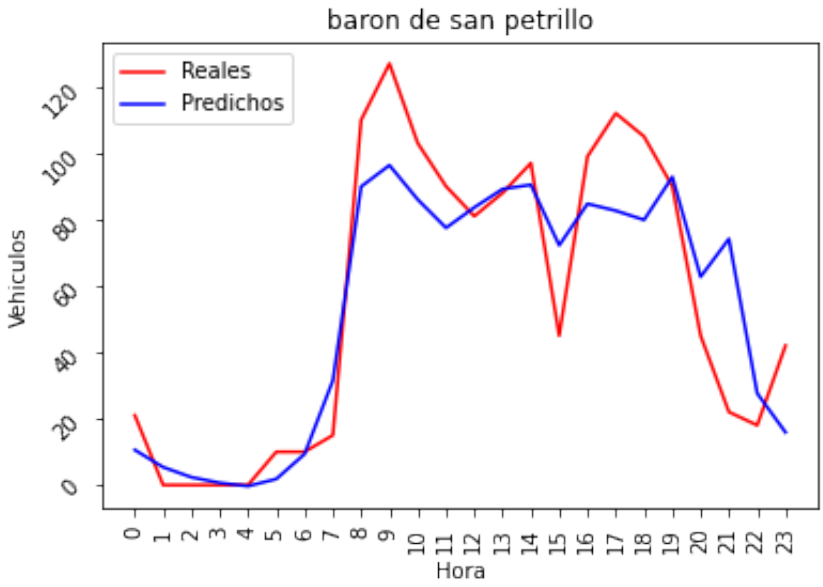


Figura 5.15: Predicción para el día 8 de junio en la calle Barón de San Petrillo usando LSTM.

### 5.3.5 Estudio del error

Como se comentó en el apartado anterior, se va a caracterizar el error, para ver cuales son las mayores fuentes del mismo. Para ello, se va a hacer uso del MAE. Los resultados mostrados anteriormente son la media del total de la métrica en cuestión, por ejemplo, para el conjunto de test se obtuvo un MAE de 0,0760, y dicho resultado viene de dividir la suma del MAE de cada una de las 386 calles, que sería un total de 29,317, y después dividir dicho resultado por 386, obteniendo así el MAE medio de 0,0760.

En la figura 5.16 se muestra el porcentaje de MAE que representa cada tipo de calle sobre el MAE total. Como se puede ver, las que más aportan al MAE son las calles residenciales y las primarias con un 32 % y un 31,4 %, respectivamente, entre otras cosas por que son las mayores en número, siendo 120 calles principales y 120 residenciales. Seguido a estas dos, aparecen las calles secundarias y las terciarias, con un 17,7 % y un 16,4 %, siendo un total de 69 calles secundarias y 63 terciarias. Finalmente, con un 1,7 % y un 0,8 % están las entradas y salidas, con un total de 3 entradas y 7 salidas. Se puede ver, como es lógico, que cuanto mayor sea el número de calles de cada tipo, más contribuyen al error total.

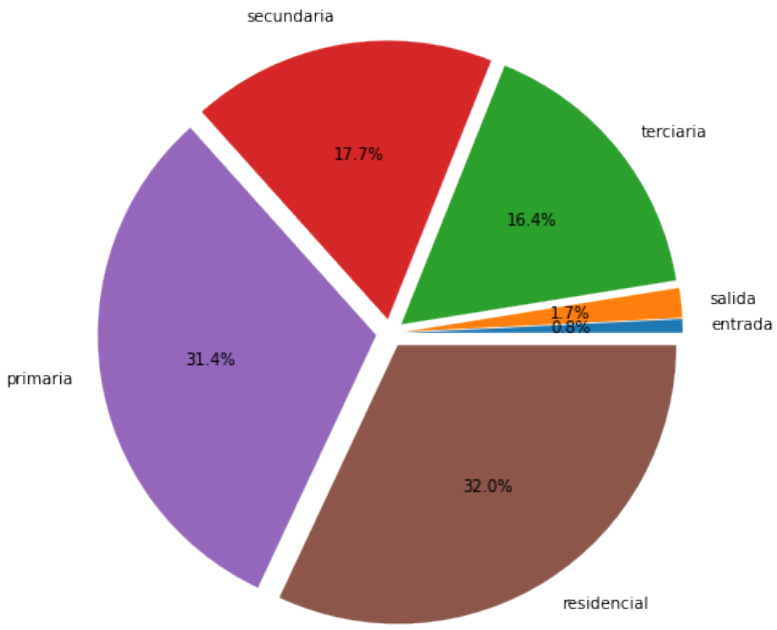


Figura 5.16: Porcentaje de error según tipo de calle.

Para desglosar más el error, a continuación se van a mostrar las gráficas con las 10 calles de cada tipo con mas MAE, pudiendo observar de esta manera las mayores fuentes de error, según el tipo de calle. En estas gráficas, además, se muestra una linea roja que representa el MAE medio de todas las calles de dicho tipo.

En primer lugar, la figura 5.17 muestra las 10 calles residenciales con mayor MAE. Puede verse que hay una calle, la calle Pintor Genaro, donde se obtiene un MAE que dobla a los valores del resto de calles, esta calle es una calle muy pequeña y poco transitada del barrio La Saidia, ocurre algo similar a lo visto en las figuras 5.15 y 5.8, existen demasiados picos y el modelo falla al predecirlos. El resto de calles, a excepción de la 2 calle con mas MAE (Barcas-Pintor Sorolla) que tiene un 0,15 de MAE, se ve que no están tan distantes de la media, siendo esta un valor de aproximadamente 0,09.

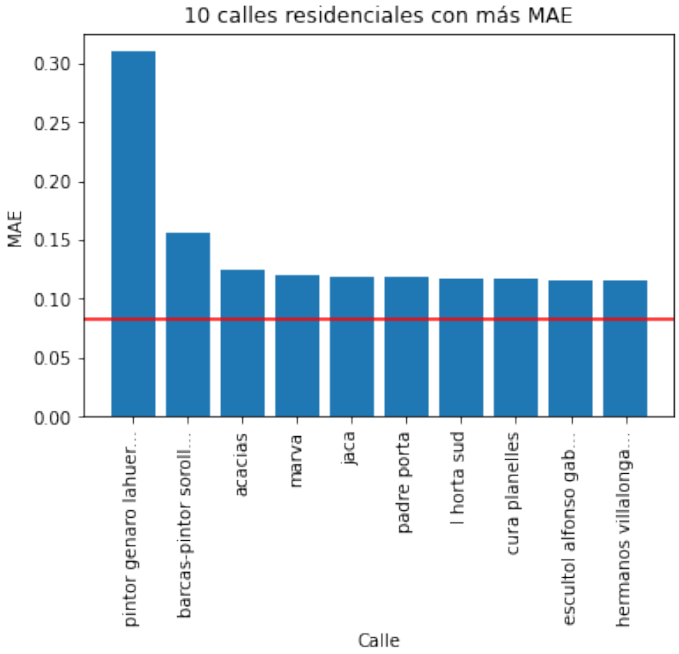


Figura 5.17: Gráfica de las 10 calles residenciales con mayor MAE.

En la figura 5.20 se muestran las 10 calles primarias con más MAE. En este caso, son dos calles las que tienen un valor que dobla al valor del resto de calles, que son el Paseo Ciudadela y un trozo de la Avenida del Cid. La Avenida del Cid es una avenida muy grande y en la página web del ayuntamiento de Valencia, está dividida en 4 trozos, el trozo que aparece en esta gráfica es un trozo diferente al estudiado en los apartados anteriores. Están considerados como si fueran 4 calles diferentes, pero que realmente son la misma divida en tramos. Para ilustrar esto, en la figura 5.18 se muestra un imagen de OpenStreetMaps con el tramo de la Avenida del Cid seleccionado, el cual corresponde al tramo que aparece en la figura 5.20 y cuyo código ATA es A69. Por otro lado, el tramo de la Avenida del Cid estudiado en capítulos anteriores y mostrado en las figuras 5.13 y 5.6 aparece en la figura 5.19, cuyo código ATA es A72.


Figura 5.18: Tramo de la avenida del cid con código ATA A69

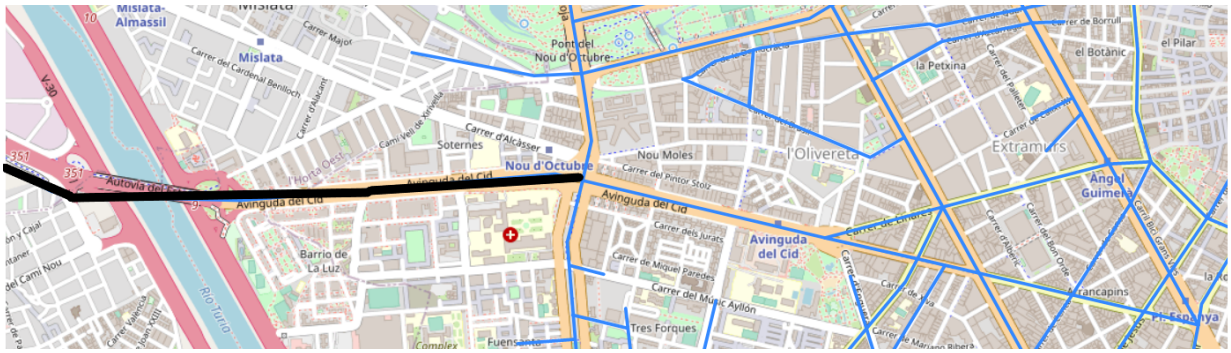


Figura 5.19: Tramo de la avenida del cid con código ATA A72

En este caso, no son calles poco transitadas, pero aun así, el modelo ha sido incapaz de predecir el flujo de tráfico en ellas, subestimando en gran medida el valor de la predicción. La media en este caso parece rondar el valor de 0,09 y exceptuando las calles Prolongación Juan y Pérez Galdós, que se alejan algo más de la media, el resto poseen valores cercanos a esta.

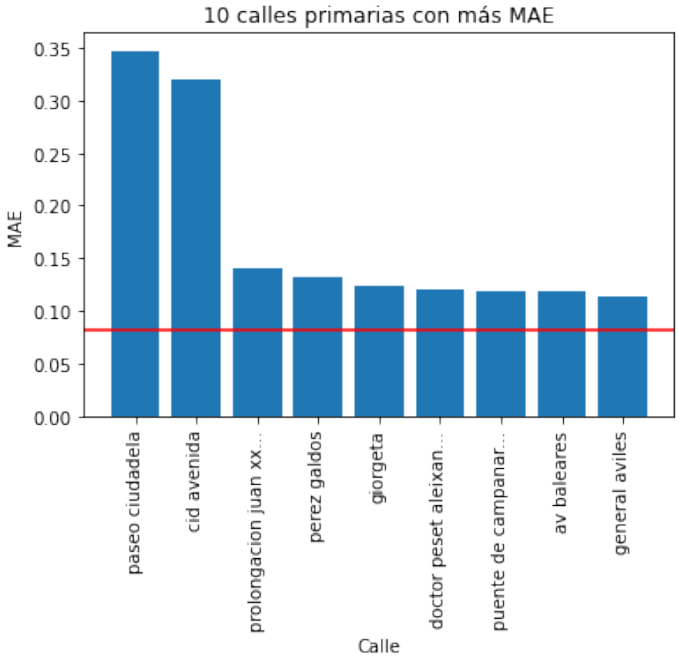


Figura 5.20: Gráfica de las 10 calles primarias con mayor MAE.

Las calles secundarias con mayor MAE aparecen en la figura 5.21 en la que se ve que vuelve a aparecer un extremo, la calle San José de Calasanz. En este caso, ha ocurrido algo diferente a los casos anteriores, esta calle a partir del día 24 de mayo hasta el 20 de junio, la calle tuvo que ser cortada por obras o alguna otra anomalía, entonces, los datos que devuelve la página web en esas fechas no son datos normales. La razón por la que no se han eliminado estos días en el proceso de limpieza es por que sí tiene datos, por ejemplo, hay lecturas con valor 1, con valor 10, con valor 3, etc. Es decir, posee valores pero estos valores no se asemejan a los valores en condiciones normales. La media vuelve a rondar el valor de 0,09 y en este caso, parece que el resto de calles se acercan mas a dicho valor que en las gráficas anteriores.

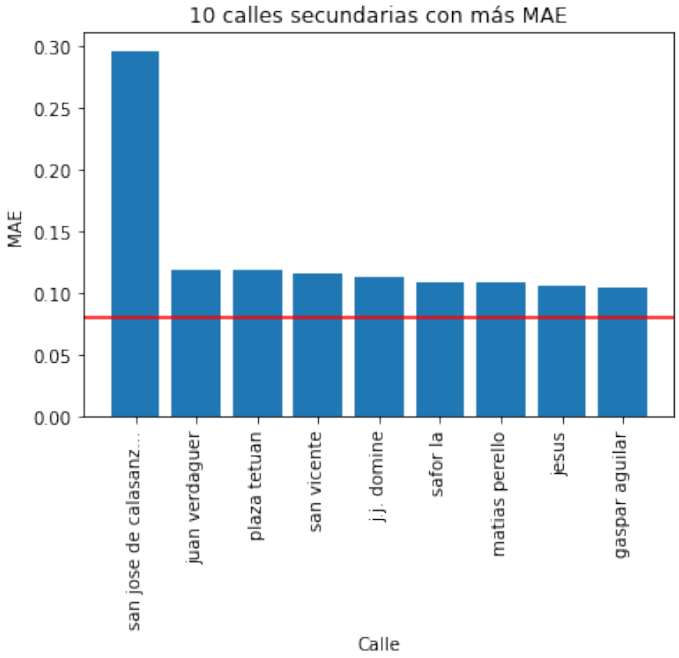


Figura 5.21: Gráfica de las 10 calles secundarias con mayor MAE.

En el caso de las calles terciarias, no parece haber un caso tan extremo como en los casos anteriores, como se puede ver en la figura 5.22. Además, el hecho de no tener un extremo, hace que la media baje hasta un 0.08. En esta ocasión, las calles se alejan más de la media, por ejemplo, la calle con mayor MAE, la Avenida del Mediterráneo, dobla la media, aunque esto no se debe a ninguna anomalía como en el caso anterior, simplemente el modelo subestima o sobrestima los valores reales, lo que puede aplicarse al resto de calles de este tipo. En este tipo de calles hay casos en los que sobrestima en exceso los valores reales, esto se debe a que, son calles tan poco transitadas que en ciertos momentos no circule ningún vehículo y a cierta hora pasen un numero más alto de vehículos, dando lugar a los picos vistos previamente en las figuras 5.8 y 5.15, haciendo fallar al modelo.

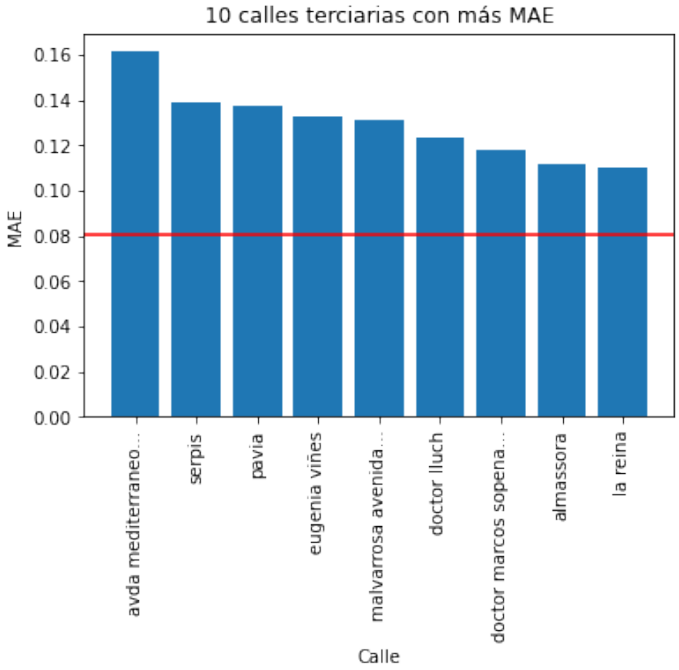


Figura 5.22: Gráfica de las 10 calles terciarias con mayor MAE.

En el caso de las entradas, únicamente se disponen datos de 3 estas, que pueden verse en la figura 5.23. Estas entradas son 3 entradas del norte de la ciudad. En la ciudad hay más entradas, sin embargo, pueden estar consideradas como otro tipo. Por ejemplo, una salida sur es la calle que tiene por nombre Carrera Malilla y está considerada como una calle terciaria. No se aprecia ningún caso extremo, aunque cada una de ellas tiene una diferencia de valor de en torno a 0,2. Es posible que en ciertos momentos exista una retención en alguna de ellas y esto contribuya a aumentar el MAE, aunque no parece reflejarse en un aumento drástico del error, como ocurría en figuras anteriores.

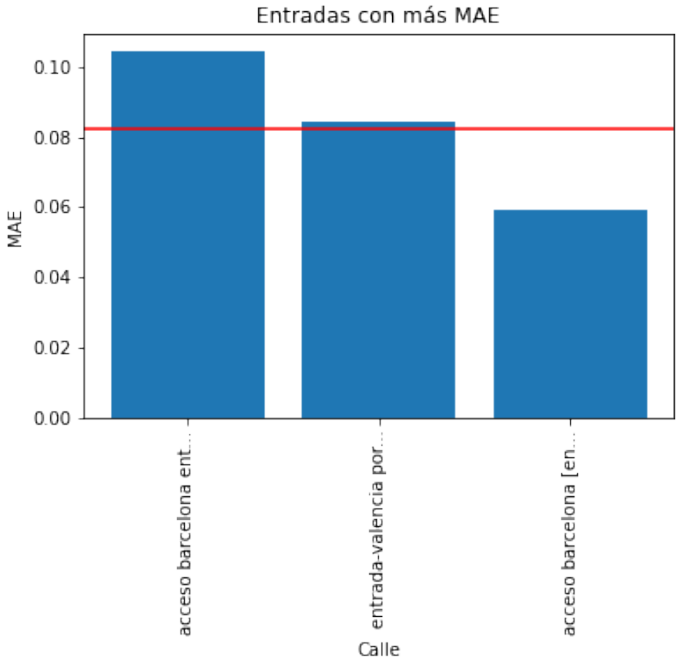


Figura 5.23: Gráfica del MAE de las 3 entradas.

Finalmente, en la figura 5.24 se muestran las salidas. Al igual que en las entradas, existen más salidas pero tienen definido otro tipo de calle. Ocurre algo similar a las entradas, no existen valores extremos y exceptuando la última de ellas que tiene un MAE pequeño en comparación a las demás, todas poseen un valor similar.

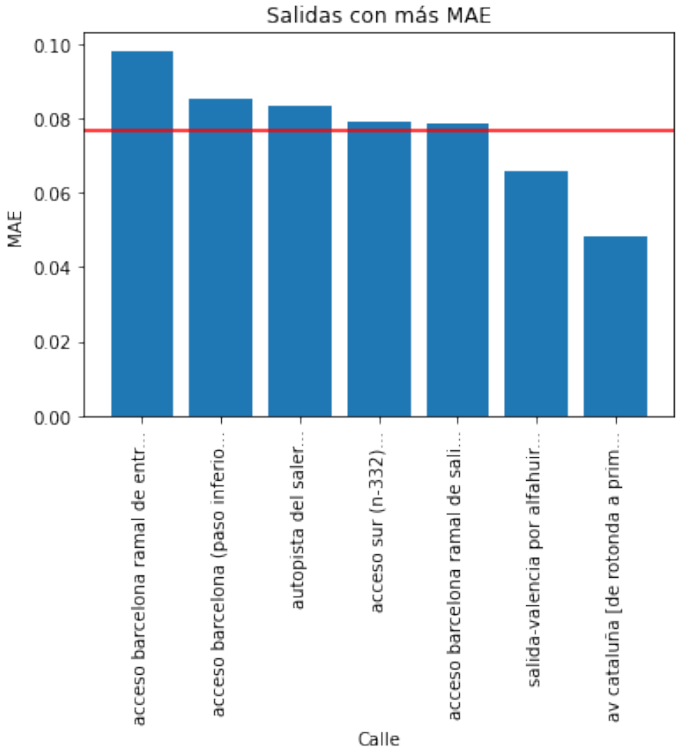


Figura 5.24: Gráfica del MAE de las 7 salidas.

Para el caso de la Avenida del Cid, en el tramo cuya predicción se ha mostrado en la figura 5.19, se ha obtenido un MAE de 0,0845. Esta calle está considerada como una calle primaria, por lo que si se compara este resultado con la media de dicho tipo de calle, se observa que está ligeramente por debajo de la misma.

En la calle Santos Justo y Pastor, que está considerada un calle terciaria, se ha obtenido un MAE de 0,0494. La media de las calles terciarias se sitúa en un 0,08, por lo que la predicción en esta calle parece ser bastante buena en comparación a la media de este tipo de calles.

Por último, la calle Barón de San Petrillo, que se considera una calle residencial, ha obtenido un MAE de 0,0739. La media de las calles residenciales es de aproximadamente 0,09, con lo cual, esta predicción es algo mejor que la media de este tipo de calles, aunque no tan buena como la obtenida en la calle Santos Justo y Pastor y quizá, algo mejor que la obtenida en la Avenida del Cid.

De este análisis del error, se puede deducir que es necesario hacer un modelo más selectivo en lugar de utilizar todas las calles juntas, lo cual tiene lógica ya que para predecir el tráfico en la Avenida del Cid, no aporta nada el tráfico en la calle Barón de San Petrillo que, además de estar en otro barrio, no tiene un flujo de tráfico ni si quiera parecido.

## 5.4 Mejora de la implementación LSTM

En este apartado se va realizar una implementación de un modelo más selectivo, que sea capaz de mejorar el resultado obtenido en aquellas fuentes de mayor error vistas en el apartado anterior.

### 5.4.1 Descripción del experimento

En el análisis del error puede verse que hay calles con demasiado error en comparación a la media, como puede verse en las figuras 5.20, 5.17 y 5.21. Uno de los motivos por los que esto puede ser debido es a que el flujo de tráfico en una calle pequeña no tiene nada que ver con una calle mayor, y que además está en un barrio diferente.

Se ha optado por realizar un modelo donde, en lugar de estudiar el tráfico de las 386 calles a la vez, se estudie el flujo de tráfico de una calle junto a sus calles cercanas. Parece lógico pensar que el flujo de tráfico de una calle depende de aquellas calles que están más cerca a ella; en concreto, las calles que entran o salen de dicha calle. Sin embargo, tras realizar varios experimentos, y estudiando el flujo de tráfico en una calle y sus entradas y salidas, y después realizando el mismo experimento pero únicamente con la calle en cuestión en sus entradas, y otro con la calle con sus salidas, se descubrió que los mejores resultados se obtenían estudiando el flujo de tráfico en una calle y en sus calles de entrada.

En este apartado se va a tratar de mejorar el resultado obtenido en aquellas calles con un mayor error haciendo uso de este planteamiento, estudiando el flujo en dicha calle y en sus calles de entrada. El conjunto de datos está dividido de la misma manera que en apartados anteriores, y la diferencia ahora está en el número de calles utilizadas.

### 5.4.2 Arquitectura del modelo

Como en este caso no se van a tener tantas calles de entrada como en el caso anterior, se ha optado por cambiar la arquitectura del modelo anterior; no obstante, también está basado en una red LSTM.

Para la gran mayoría de calles, aparecen 3 calles que entran a dicha calle, haciendo un total de 4 calles a estudiar, contando la propia calle. Por lo que se ha optado por realizar un proceso de ajuste sobre este grupo, las de 4 calles de entrada, utilizando el conjunto de validación. Tras un proceso de ajuste, tomando como referencia los resultados obtenidos sobre el conjunto de validación, la arquitectura que mejores resultados ha obtenido se muestra en la figura 5.25. Los resultados se han evaluado tras realizar 10 ejecuciones con cada una de las configuraciones probadas, y después una media de ellas.

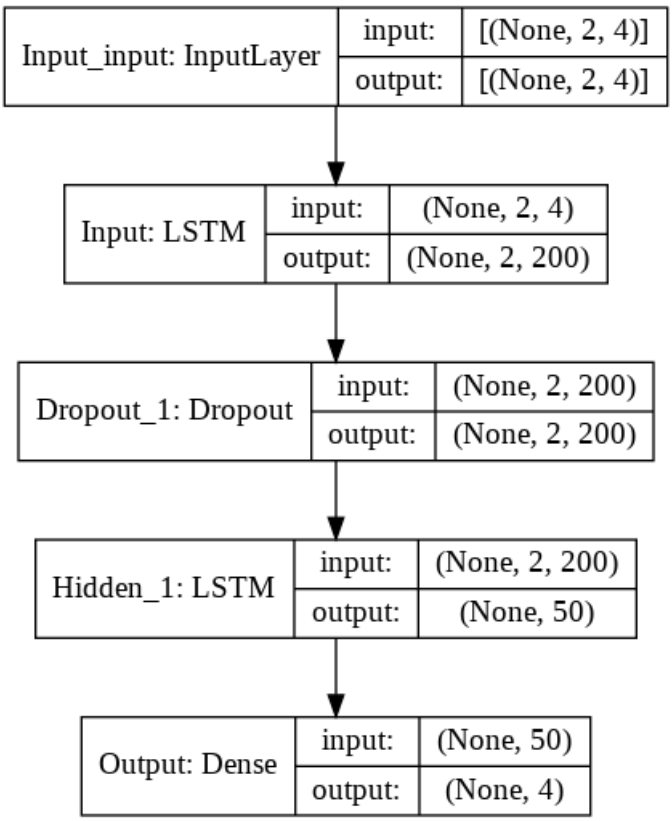


Figura 5.25: Arquitectura utilizada para la mejora de la red LSTM.

En esta ocasión se tiene una capa de entrada LSTM con 200 unidades, que requiere como entrada un vector tridimensional de manera que las dimensiones sean muestras, valor de atraso y características; como ya se explicó en el apartado anterior, el valor de atraso utilizado sigue siendo igual a 2. Después de esta capa de entrada, se añade una capa de dropout con valor de 0.2, y una capa oculta de tipo LSTM con 50 unidades. Finalmente, se tiene una capa de salida de tamaño igual a las características. Al igual que en apartados anteriores, todas las funciones de activación son de tipo ReLU, excluyendo la de la capa de salida, cuya función de activación es lineal. Se ha realizado también un escalado de los datos hacia el rango [0, 1].

Los parámetros de la red son similares a los de apartados anteriores: factor de aprendizaje igual a $10^{-4}$, tamaño de batch igual a 24, límite de 200 épocas de entrenamiento con *early stopping* con un valor de paciencia igual a 3, y Adam como optimizador del gradiente.

### 5.4.3 Entrenamiento del modelo

En este apartado, se muestra el resultado del entrenamiento del modelo para el tramo de la Avenida del Cid con código ATA A69, mostrado en la figura 5.18, y sus calles de entrada que corresponden a otro tramo de dicha avenida, la calle Lorca y la calle Ayora. No obstante, este proceso de entrenamiento debe realizarse para todas y cada una de las calles y sus entradas solo que, por no extender mucho el presente documento, únicamente se muestra el resultado de dicha calle. Tampoco se muestran los resultados del ajuste de los parámetros por la misma razón, simplemente se muestra el resultado sobre el conjunto de test.

Tras 131 épocas de entrenamiento sobre el conjunto que contiene los datos de entrenamiento y validación, se han obtenido los siguientes resultados:

- MAE conjunto entrenamiento: 0,0722
- MSE conjunto entrenamiento: 0,0126
- RMSE conjunto entrenamiento: 0,1121
- MAE conjunto test: 0,0881
- MSE conjunto test: 0,0153
- RMSE conjunto test: 0,1237

Comparando los resultados obtenidos sobre el conjunto de test, parece que los resultados obtenidos con la red LSTM anterior son más bajos, pero el experimento es diferente y, por lo tanto, para ver si realmente se obtiene una mejora, se van a comparar los resultados sobre el conjunto de test no en grupo, si no en calles individuales. En la tabla 5.4 se muestra la comparativa de los resultados obtenidos sobre el conjunto de entrenamiento entre la red LSTM original y esta LSTM mejorada y en la figura 5.5 la misma comparativa sobre el conjunto de test, donde puede verse lo comentado.

| Modelo | MAE | MSE | RMSE |
|---|---|---|---|
| LSTM original | 0,0703 | 0,0123 | 0,1111 |
| LSTM mejorada | 0,0722 | 0,0126 | 0,1121 |

Tabla 5.4: Comparativa resultados LSTM original y LSTM mejorada sobre el conjunto de entrenamiento.

| Modelo | MAE | MSE | RMSE |
|---|---|---|---|
| LSTM | 0,0760 | 0,0108 | 0,1039 |
| LSTM mejorada | 0,0881 | 0,0153 | 0,1237 |

Tabla 5.5: Comparativa resultados LSTM original y LSTM mejorada sobre el conjunto de test.

En la figura 5.26 se muestra la evolución del MAE para el conjunto de entrenamiento y el de test. En esta ocasión necesita de más épocas de entrenamiento ya que la arquitectura es más profunda. Se aprecia que la curva no es tan suave como en las gráficas 5.10 o 5.3. Lo mismo pasa con el MSE, que aparece en la figura 5.27, y el RMSE que se muestra en la figura 5.28.

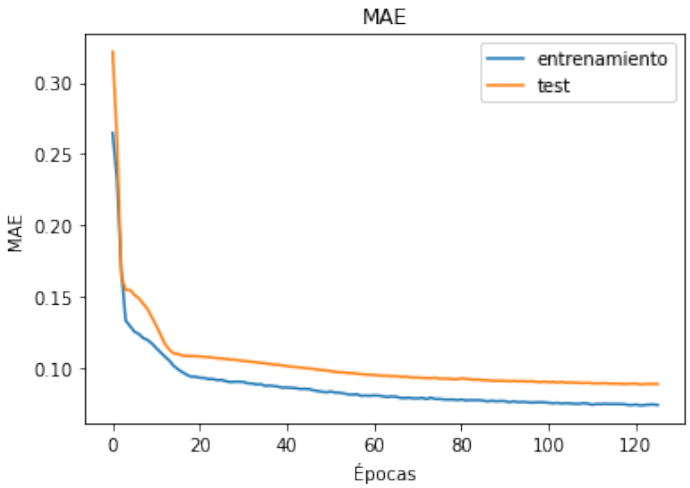


Figura 5.26: Evolución del MAE en la red LSTM mejorada para la Avenida del Cid.

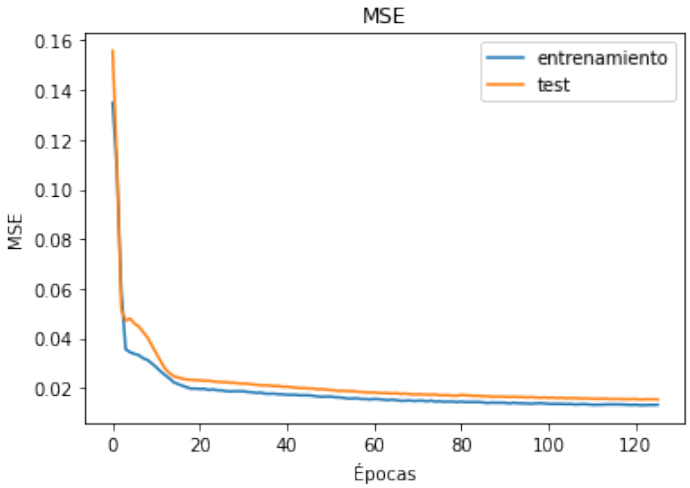


Figura 5.27: Evolución del MSE en la red LSTM mejorada para la Avenida del Cid.

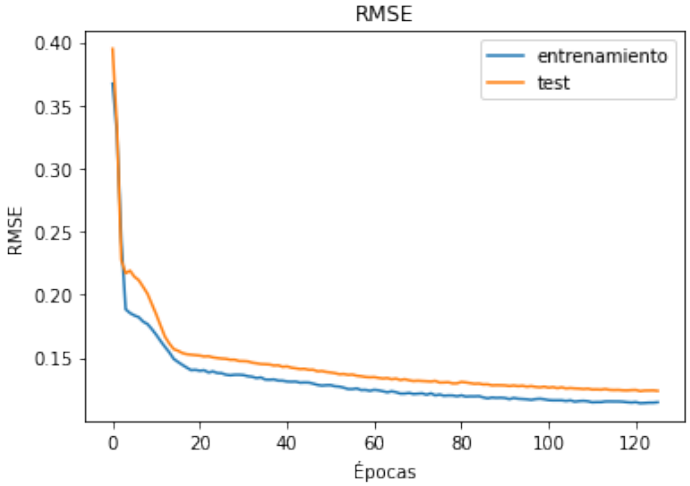


Figura 5.28: Evolución del RMSE en la red LSTM mejorada para la Avenida del Cid.

### 5.4.4 Evaluación del modelo

Como se ha comentado anteriormente, más que evaluar el modelo en términos globales, se va a estudiar el impacto en aquellas calles que tenían un mayor MAE. Antes de empezar, decir que para cada calle se ha entrenado un modelo con dicha calle y sus respectivas calles de entrada. En este apartado únicamente se muestran

los resultados sobre los conjuntos de test de dichas calles para ver la comparativa entre los resultados de cada una de ellas.

El tramo de la Avenida del Cid mostrado en la figura 5.18 tiene un MAE de 0,3276, y ese mismo tramo con nuestro nuevo modelo mejorado ha obtenido un MAE de 0,1314. Esto supone una mejora del 59,88 % con respecto a la red LSTM original. En la figura 5.29 se muestra el resultado de la predicción con el modelo mejorado del día 8 de junio de 2021, y en la figura 5.30 se muestra la predicción del mismo día con el modelo original, donde se aprecia que claramente el modelo original falló, subestimando mucho los valores predichos.

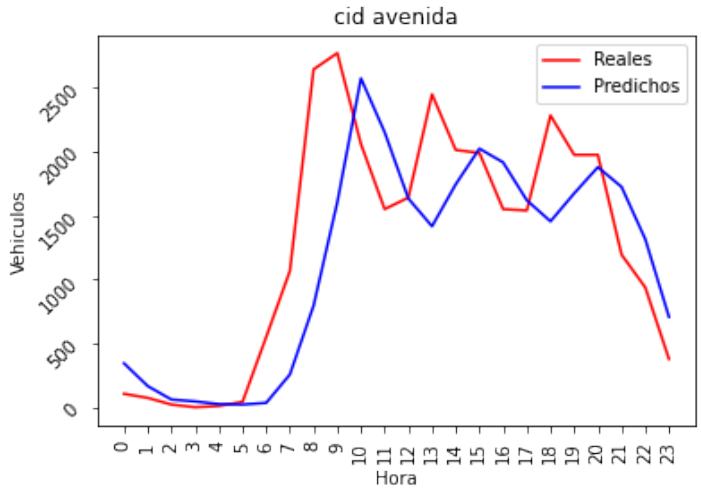


Figura 5.29: Predicción para el día 8 de junio en la Avenida del Cid usando la red LSTM mejorada.

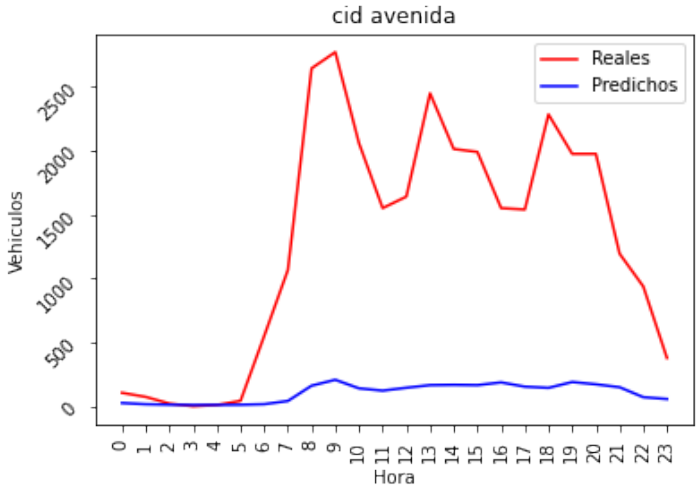


Figura 5.30: Predicción para el día 8 de junio en la Avenida del Cid usando la red LSTM original.

Por otro lado, el tramo de la Avenida del Cid, mostrado en la figura 5.19, cuya predicción se muestra en la figura 5.13, obtuvo un MAE de 0,0845. Con este nuevo modelo mejorado, ese mismo tramo ha obtenido un MAE de 0,0937, lo cual supone un empeoramiento del 9,78 %.

El resultado de la predicción de este tramo el día 8 de junio de 2021 se muestra en la figura 5.31, que es el mismo día que en la figura 5.13. Si se comparan ambas gráficas, se ve que evidentemente la predicción era mejor con el modelo original que con este modelo mejorado.

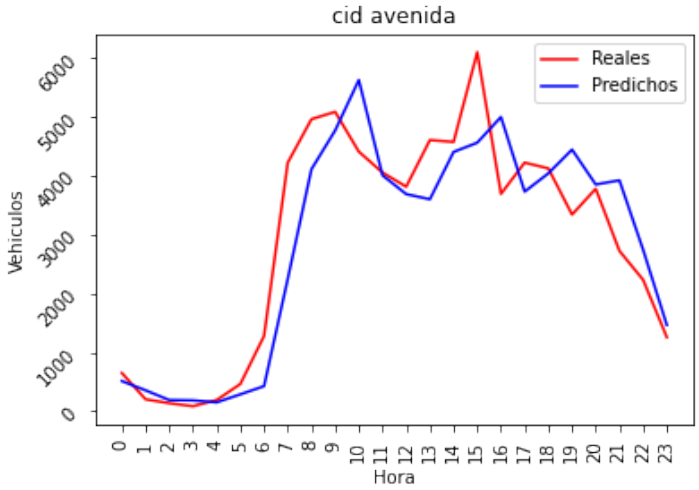


Figura 5.31: Predicción para el día 8 de junio en la Avenida del Cid usando la red LSTM mejorada.

Otra de las mayores fuentes de error fue el Paseo Ciudadela, que obtuvo un MAE de 0,3477. El resultado obtenido con el modelo mejorado se ha obtenido un MAE de 0.0993. La mejora en esta ocasión es de un 71,45 %. Para poder comparar los resultados de la predicción y poder ver dicha mejora, en la figura 5.32 se observa la predicción con el modelo LSTM mejorado, que se ve que claramente es mejor que la predicción con el modelo LSTM original, mostrado en la figura 5.33, donde, como pasaba anteriormente, se subestima en exceso los valores.

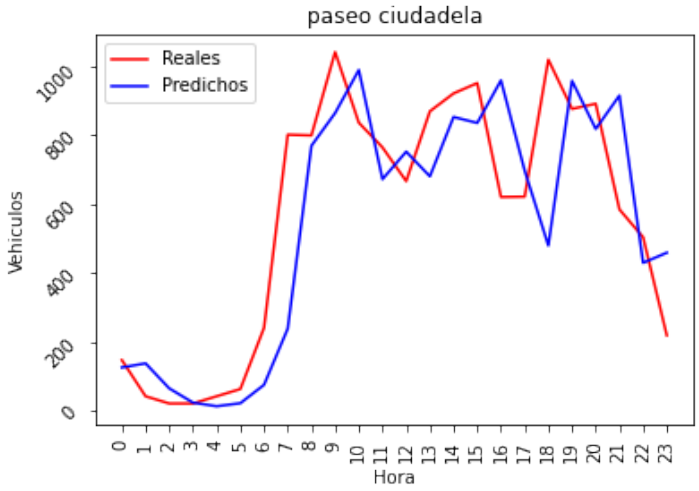


Figura 5.32: Predicción para el día 8 de junio en el Paseo Ciudadela usando la red LSTM mejorada.

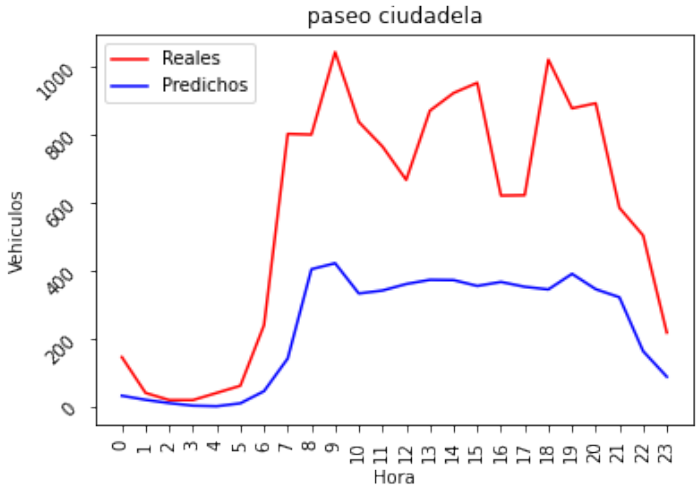


Figura 5.33: Predicción para el día 8 de junio en el Paseo Ciudadela usando la red LSTM original.

La calle Pintor Genaro Lahuerta también obtuvo un MAE muy elevado, de 0,3081. Con este modelo mejorado, esta misma calle ha obtenido un MAE de 0,3324, lo que supone un empeoramiento del 7,33 %. Esta calle es una calle muy pequeña con muy poco flujo de tráfico y muy variable, además de que únicamente dispone de 2 calles de entrada, por lo que este modelo no dispone de suficientes datos para predecir esta calle, y hace que empeore el resultado comparado con el modelo LSTM original. La predicción de esta calle con el modelo mejorado puede verse en la figura 5.34, que se ve peor que la predicción con el modelo original, mostrado en la figura 5.35.

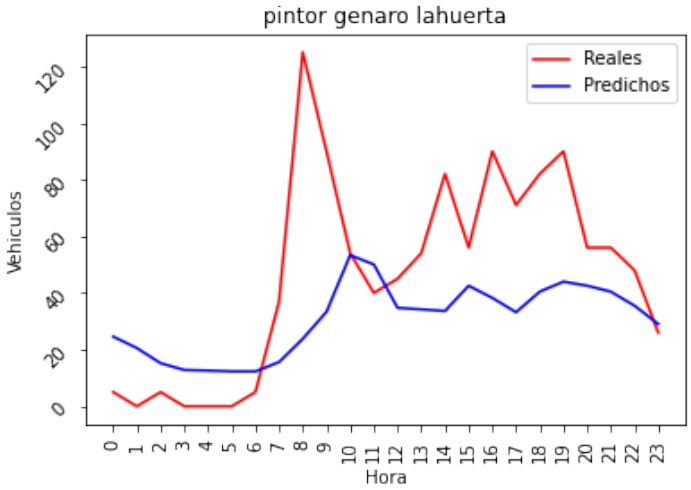


Figura 5.34: Predicción para el día 8 de junio en la calle Pintor Genaro Lahuerta usando la red LSTM mejorada.

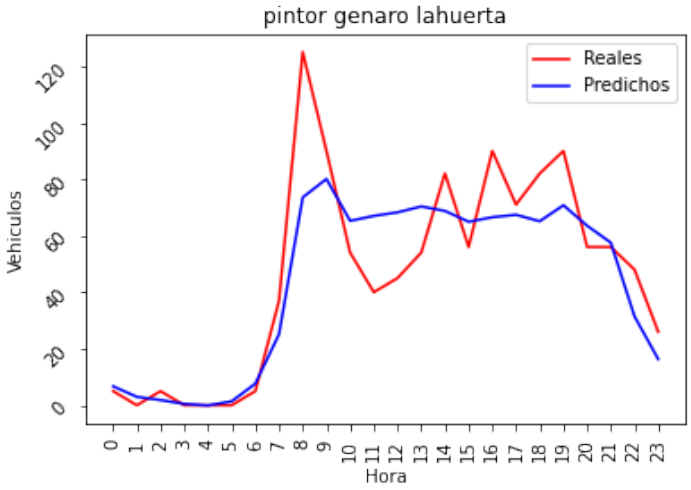


Figura 5.35: Predicción para el día 8 de junio en la calle Pintor Genaro Lahuerta usando la red LSTM original.

Una calle cuyo MAE también fue alto, pero no tan elevado como los anteriores, fue la Avenida del Mediterráneo. El MAE obtenido fue de 0,1630, mientras que con este modelo mejorado se ha obtenido un MAE igual a 0,1235, lo que supone una mejora del 24,24 %. Se puede ver que la predicción con el modelo mejorado, en la figura 5.36, es mejor que la obtenida con el modelo LSTM original, en la figura 5.37.

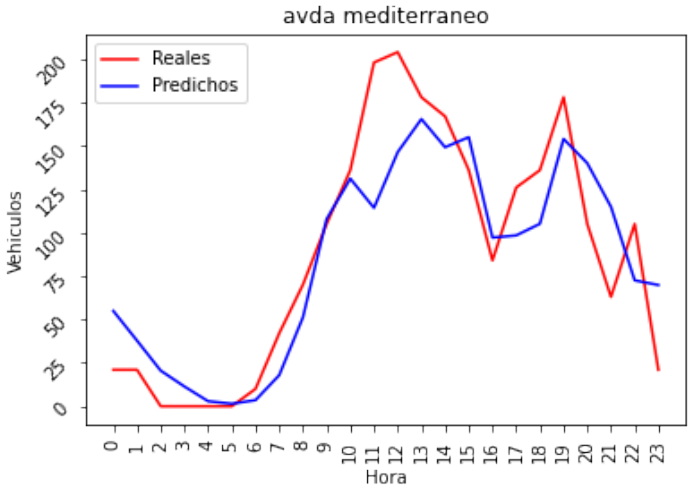


Figura 5.36: Predicción para el día 8 de junio en la Avenida del Mediterráneo usando la red LSTM mejorada.

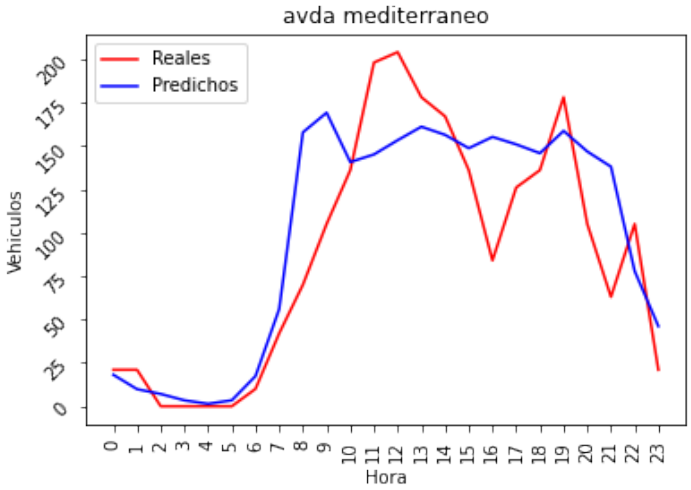


Figura 5.37: Predicción para el día 8 de junio en la Avenida del Mediterráneo la red LSTM original.

La calle Santos Justo y Pastor, con el modelo LSTM original, obtuvo un MAE de 0,0494, mientras que con el modelo LSTM mejorado se ha obtenido un MAE de 0,0524. Este resultado ha empeorado un 5,63 % y, si se comparan la gráfica obtenida con el modelo LSTM original, que puede verse en la figura 5.14, y la obtenida con este modelo mejorado, mostrado en la figura 5.38, también se aprecia que es algo mejor la gráfica con el modelo original, aunque la diferencia no es tan grande como pasaba en la Avenida del Cid.

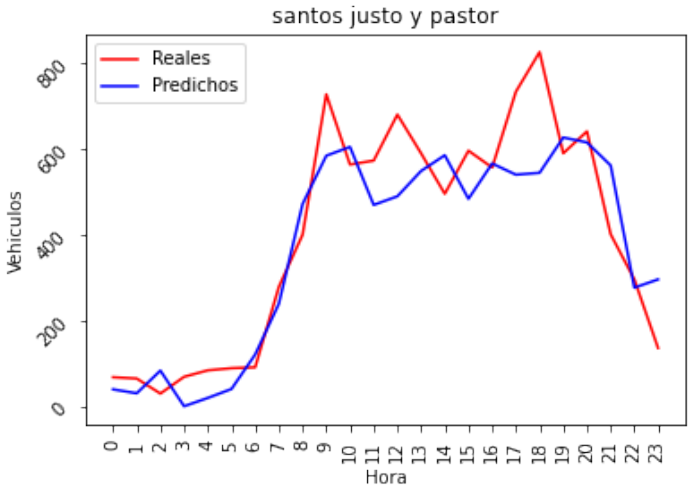


Figura 5.38: Predicción para el día 8 de junio en la calle Santos Justo y Pastor usando la red LSTM mejorada.

Para acabar este apartado de comparación, la calle Barón de San Petrillo, con el modelo LSTM original, obtuvo un MAE de 0,0739. Con el nuevo modelo LSTM mejorado, se ha obtenido un MAE igual a 0,0783, suponiendo un empeoramiento del 5,64 %. La figura 5.39 muestra la predicción con el modelo mejorado y en la que se ve, al igual que en los casos anteriores, que el modelo original obtuvo una mejor predicción.

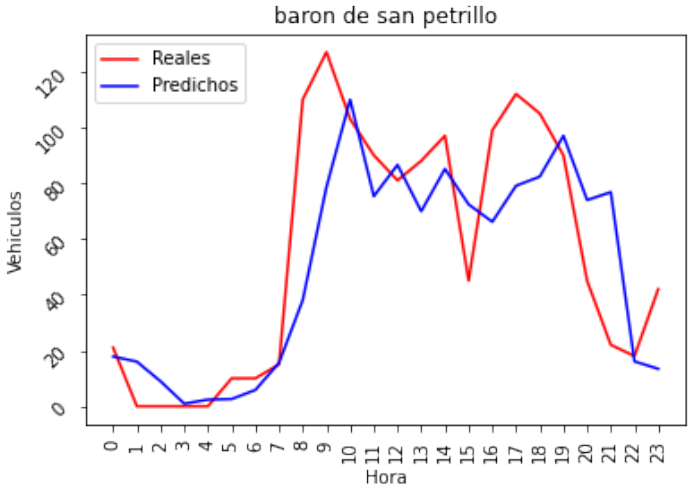


Figura 5.39: Predicción para el día 8 de junio en la calle Barón de San Petrillo usando la red LSTM mejorada.

Para concluir este apartado, se ha visto que realizando este modelo más selectivo, en lugar de entrenar y predecir todas las calles a la vez, se han obtenido mejoras muy significativas en aquellas fuentes de mayor error. Sin embargo, aquellas predicciones que no obtuvieron tan malos resultados, o aquellas calles que son pequeñas y con poco flujo de tráfico, no obtienen mejora y, además, empeoran el resultado. No obstante, si se comparan las mejoras con respecto a los empeoramientos, se ve que las mejoras son mucho más significativas. Por ejemplo, para el tramo de la Avenida del Cid con código ATA A72, que no obtuvo un MAE muy elevado, se ha empeorado un 9,78 %; en cambio, en el tramo que obtuvo un MAE muy elevado, cuyo ATA es A69, se ha obtenido una mejora de un 71,45 %.

# Capítulo 6

# Conclusiones

Para finalizar este trabajo de fin de máster, se van a exponer las principales conclusiones obtenidas del desarrollo del mismo, y plantear trabajos futuros a partir del presente documento.

## 6.1 Conclusiones

En este trabajo de fin de máster se ha presentado el problema de la predicción del flujo del tráfico, muy presente en la comunidad científica. Además, se ha realizado en primer lugar una explicación de las series temporales y sus componentes. Tras ello, se ha realizado un estudio de los diferentes modelos y técnicas utilizadas en el estado del arte, así como aplicaciones reales existentes que abordan este problema.

A continuación, se ha realizado una explicación en detalle del proceso de extracción y pre-proceso de los datos, necesario para dejar los datos en un formato común. Seguidamente, se ha llevado a cabo una descripción de todo el proceso de limpieza de los datos, clave para el correcto funcionamiento del algoritmo.

Posteriormente, se ha llevado a cabo un análisis de la serie temporal formada por los datos del flujo de tráfico. Este análisis es fundamental, ya que con él se ha conseguido entender el comportamiento de los mismos, y se ha sido capaz de justificar los parámetros posteriores utilizados en las redes neuronales, sobre todo el valor de atraso utilizado en la serie temporal.

Tras dicho análisis, se ha realizado una primera implementación de un modelo basado en la red neuronal más simple, el perceptrón multicapa, para tener un punto de partida y, a partir de ahí, intentar mejorar los resultados. Con dicho fin, se ha realizado una implementación de un modelo basado en una red LSTM que, debido a su naturaleza, funciona muy bien en problemas de series temporales, como es el caso de la predicción del flujo de tráfico.

Sobre los resultados de la red LSTM, se ha llevado a cabo un proceso de análisis para ver cuales son las mayores fuentes de error, y ser capaz de mejorar el resultado de estas y, con ello, del modelo final.

Finalmente, en base al análisis del error, se ha planteado una solución más selectiva a la basada en el primer modelo LSTM, que es capaz de reducir drásticamente el error en las calles con mayor error obtenidas en el modelo inicial. Sin embargo, en otras calles cuya predicción era más aceptable, verificamos que el resultado empeora ligeramente.

De los resultados obtenidos, se ha visto que, para el primer experimento, en el que se quiere predecir el flujo de tráfico en las 386 calles de Valencia a la vez, haciendo uso de los datos de todas y cada una de ellas, el modelo basado en la red LSTM superó al modelo basado en el MLP en un 1,6 % si se tiene en cuenta el MAE, y un 1,8 % si se tiene en cuenta el MSE. Tiene sentido que, a pesar de que la diferencia no sea excesiva, se obtengan mejores resultados con la red LSTM que con la red MLP, debido a la simpleza de la segunda red que, a pesar de ello, funciona bastante bien.

Sin embargo, este primer experimento de predecir todas las calles a la vez implica que todas calles dependen unas otras, lo cual no parece tener mucho sentido, pues una avenida grande en el barrio de Algirós no parece tener relación alguna con una calle pequeña de los Poblados Marítimos. Es por esto que se realizó un estudio de las fuentes de error de este modelo para compararlos con un modelo mucho más selectivo.

En primer lugar, se intentó predecir el tráfico en una calle teniendo en cuenta el tráfico de esa misma calle, y el tráfico de sus calles de entrada y de salida; posteriormente, el mismo experimento, pero con solo las calles de salida y, finalmente, con la calle y sus calles de entrada, siendo este último el que mejores resultados ha obtenido. Por ello, se ha realizado un modelo para varias calles de Valencia, las que más error producían en el modelo anterior, y las 3 calles de ejemplo vistas en ambas redes, para poder comparar resultados y comprobar que, efectivamente, se obtienen mejores predicciones de esta manera.

De los resultados obtenidos, se ha visto que, realizando esta selección, se ha mejorado las fuentes de error dónde el modelo subestimaba de manera muy excesiva las predicciones. Sin embargo, en aquellas calles dónde el flujo de tráfico es pequeño, el resultado mejora, pero no tanto, e incluso en aquellas calles menos transitadas, como ha sido el caso de la calle Pintor Genaro Lahuerta, el modelo ha obtenido peores resultados.

Finalmente, decir que, por falta de tiempo, no se ha podido implementar este modelo selectivo para las 386 calles, ya que requiere una selección manual de vías de entrada y salida. Sin embargo, viendo los resultados y siguiendo la lógica, parece que es una mejor idea utilizar el tráfico de una calle y el de sus entradas como datos para realizar las predicciones, y no utilizar el tráfico de todas las calles a la

vez, por lo que la solución final propuesta sería implementar un modelo selectivo dónde, para predecir el tráfico de una calle, únicamente se utilicen datos de dicha calle y de sus calles de entrada.

## 6.2 Trabajos futuros

Una clara línea futura de este trabajo es poder realizar la implementación del modelo selectivo con todas de las calles de Valencia, y no solo con unas pocas. A pesar de que se ha tratado de escoger calles de diferentes barrios, y con diferentes comportamientos, ya que hay avenidas grandes y calles pequeñas, sería interesante ver el resultado de predecir todas las calles, para así poder comparar los resultados de ambos modelos en términos globales.

Otra posible línea podría ser implementar otro tipo de modelos, no basados en redes neuronales, como un modelo ARIMA o SARIMA, que han sido introducidos en este trabajo de fin de máster.

Finalmente, podrían realizarse modelos que tuvieran en cuenta no solo el flujo de tráfico, si no también que tengan en cuenta otros factores como datos climatológicos, que tengan en cuenta si un día es festivo o no, etc. En definitiva, que se utilicen otro tipo de datos que puedan afectar al flujo de tráfico normal.

# Bibliografía


[1] Ajuntament de València. Intensidad de los puntos de medida de tráfico (espiras electromagnéticas). [Online; accessed 10-septiembre-2021]. URL: https://www.valencia.es/dadesobertes/es/dataset/?id=intensidad-de-los-puntos-de-medida-de-trafico-espiras-electromagn

[2] Cristian Villarroya Sánchez. Tfm. https://github.com/CristianViSa/TFM, 2021.

[3] Granville Tunnicliffe Wilson. Time series analysis: Forecasting and control,5th edition, by george e. p. box, gwilym m. jenkins, gregory c. reinsel and greta m. ljung, 2015. published by john wiley and sons inc., hoboken, new jersey, pp. 712. isbn: 978-1-118-67502-1. *Journal of Time Series Analysis*, 37:n/a–n/a, 03 2016. doi:10.1111/jtsa.12194.

[4] S. Kumar and L. Vanajakshi. Short-term traffic flow prediction using seasonal arima model with limited input data. *European Transport Research Review*, 7, 09 2015. doi:10.1007/s12544-015-0170-8.

[5] Dong-wei Xu, Yong-dong Wang, Limin Jia, Yong Qin, and Hong-hui Dong. Real-time road traffic state prediction based on arima and kalman filter. *Frontiers of Information Technology & Electronic Engineering*, 18:287–302, 02 2017. doi:10.1631/FITEE.1500381.

[6] Taghreed Alghamdi, Khalid Elgazzar, Magdi Bayoumi, Taysseer Sharaf, and Sumit Shah. "forecasting traffic congestion using arima modeling". In *15th International Wireless Communications Mobile Computing Conference (IWCMC)*, pages 1227–1232, 06 2019. doi:10.1109/IWCMC.2019.8766698.

[7] D. Xu and Y. Shi. A combined model of random forest and multilayer perceptron to forecast expressway traffic flow. In *2017 7th IEEE International Conference on Electronics Information and Emergency Communica-*

*tion (ICEIEC)*, pages 448–451, 2017. doi:10.1109/ICEIEC.2017.8076602.

[8] K. Li, C. Zhai, and J. Xu. Short-term traffic flow prediction using a methodology based on arima and rbf-ann. In *2017 Chinese Automation Congress (CAC)*, pages 2804–2807, 2017. doi:10.1109/CAC.2017.8243253.

[9] Shengdong Du, Tianrui Li, Xun Gong, and Shi-Jinn Horng. A hybrid method for traffic flow forecasting using multimodal deep learning, 2019. arXiv:1803.02099.

[10] Q. Zhaowei, L. Haitao, L. Zhihui, and Z. Tao. Short-term traffic flow forecasting method with m-b-lstm hybrid network. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–11, 2020. doi:10.1109/TITS.2020.3009725.

[11] Shangyu Sun, Huayi Wu, and Longgang Xiang. City-wide traffic flow forecasting using a deep convolutional neural network. *Sensors*, 20:421, 01 2020. doi:10.3390/s20020421.

[12] Licheng Qu, Wei Li, Wenjing Li, Dongfang Ma, and Yinhai Wang. Daily long-term traffic flow forecasting based on a deep neural network. *Expert Systems with Applications*, 121:304–312, 2019. URL: https://www.sciencedirect.com/science/article/pii/S0957417418308017, doi:https://doi.org/10.1016/j.eswa.2018.12.031.

[13] G. Dai, C. Ma, and X. Xu. Short-term traffic flow prediction method for urban road sections based on space–time analysis and gru. *IEEE Access*, 7:143025–143035, 2019. doi:10.1109/ACCESS.2019.2941280.

[14] D. Kang, Y. Lv, and Y. Chen. Short-term traffic flow prediction with lstm recurrent neural network. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6, 2017. doi:10.1109/ITSC.2017.8317872.

[15] University Transportation Center (funded by USDOT) at the New York University via a grant to the University of Washington. An artificial intelligence platform for network-wide congestion detectionand prediction usingmultisourcedata, 2019. [Online; accessed 10-septiembre-2021]. URL: https://c2smart.engineering.nyu.edu/wp-content/uploads/2019/07/Transportation_AI%20Platform_FinalReport_C2SMART_Wang.pdf.

[16] DeepMind. Traffic prediction with advanced graph neural networks, 2020. [Online; accessed 10-septiembre-2021]. URL: https://deepmind.com/blog/article/traffic-prediction-with-advanced-graph-neural-networks.

[17] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017. [Online; accessed 10-septiembre-2021].

[18] OpenStreetMap Wiki. Main page — openstreetmap wiki,, 2020. [Online; accessed 21-mayo-2021]. URL: https://wiki.openstreetmap.org/wiki/Key:highway#Roads.

[19] Ajuntament de València. Mi barrio. [Online; accessed 10-septiembre-2021]. URL: https://www.valencia.es/web/guest/cas/la-ciudad/mi-barrio.

[20] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010. [Online; accessed 10-septiembre-2021]. doi:10.25080/Majora-92bf1922-00a.

[21] Calendarios Laborales. Calendario laboral valencia 2021, 2021. [Online; accessed 10-septiembre-2021]. URL: https://www.calendarioslaborales.com/calendario-laboral-valencia-2021.html.

[22] PremiumSoft™ CyberTech Ltd. Navicat. https://www.navicat.com/es/, 2002. [Online; accessed 10-septiembre-2021].

[23] Skipper Seabold and Josef Perktold. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010. [Online; accessed 10-septiembre-2021].

[24] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[25] Ekaba Bisong. *Google Colaboratory*, pages 59–64. Apress, Berkeley, CA, 2019. [Online; accessed 10-septiembre-2021]. doi:10.1007/978-1-4842-4470-8_7.

[26] Francois Chollet et al. Keras, 2015. [Online; accessed 10-septiembre-2021]. URL: https://github.com/fchollet/keras.

[27] François Chollet et al. Keras early stopping. https://keras.io/api/callbacks/early_stopping/, 2015. [Online; accessed 10-septiembre-2021].

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. [Online; accessed 10-septiembre-2021].