Máster en Investigación en Inteligencia Artificial

Grafo convolucional temporal para la predicción de tráfico

Proyecto Fin de Máster

Autor: Ana Arias Botey

Tutor: Dr. José Luis Aznarte

septiembre 2022

Resumen

La predicción del tráfico desempeña un papel importante en la mejora de la eficiencia del sistema de transporte y es de gran importancia para la planificación de rutas, la detección de puntos negros asociados con datos de accidentes o problemas de salud. Los datos de tráfico generan información cada pocos minutos mediante sensores situados en diferentes puntos geográficos, lo que lleva a considerar una relación entre la dimensión del espacio y el tiempo (modelos espacio-temporales). En estudios recientes, se ha producido un aumento de la investigación de modelos de inteligencia artificial aplicados al aprendizaje de la representación de grafos, un tipo de estructura de datos formada por un conjunto de objetos (nodos) y sus relaciones (aristas). Los algoritmos de aprendizaje profundo basados en grafos, son capaces de crear una estructura geométrica más compleja que las redes convolucionales y se ajustan a la topología de la red de carreteras de tráfico. En este trabajo se aplica un modelo novedoso basado en la unión de capas convolucionales aplicadas a grafos para modelar la componente espacial y de capas recurrentes para modelar la parte temporal (T-GCN) obteniendo resultados satisfactorios. El lenguaje de programación empleado es python.

PALABRAS CLAVE: GNN, GCN, LSTM, T-GCN, grafo de tráfico

Abstract

Traffic prediction plays an important role improving the efficiency of the transportation system and it is important for route planning, detection of black spots associated with accident data or health problems. Traffic data generates information every few minutes by sensors located at different geographical points, which leads to consider a relationship between the dimension of space and time (spatio-temporal models). In recent studies, there has been an increase in research on artificial intelligence models applied to learning graph representation, a type of data structure formed by a set of objects (nodes) and their relationships (edges). Deep learning algorithms based on graphs are able to create a more complex geometric structure than convolutional networks and fit the topology of the traffic road network. In this work we propose a novel model based on the union of convolutional layers applied to graphs to model the spatial component and recurrent layers to model the temporal part (T-GCN). The used programming language is Python.

KEYWORDS: GNN, GCN, LSTM, T-GCN, traffic graph

Índice

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

Las ciudades en expansión se enfrentan a problemas relacionados con el transporte. El crecimiento de la economía europea y las necesidades de los ciudadanos en el ámbito de la movilidad, causan una creciente congestión de las infraestructuras viarias y problemas medioambientales. Según estudios realizados [1], la contaminación asociada a la congestión de tráfico, aumenta las posibilidades de desarrollar alergias y agrava los síntomas de las personas sensibles a ellas. La intervención temprana basada en la previsión del tráfico, se considera clave para mejorar la eficiencia de un sistema de transporte, y aliviar los problemas relacionados como el control de los semáforos, la planificación de rutas, la detección de puntos negros asociados con datos de accidentes o problemas de salud.

Para predecir el tráfico, se requiere grandes cantidades de datos y el uso de algoritmos de inteligencia artificial (IA). Los datos de tráfico, generan información cada pocos minutos mediante sensores situados en diferentes puntos geográficos, lo que lleva a considerar una relación entre la dimensión del espacio y el tiempo (modelos espacio-temporales). Los algoritmos que consideran relaciones espacio-temporales surgen en muchos otros contextos, como la predicción de riesgo de enfermedades [2], el consumo eléctrico o el tráfico.

Los modelos de aprendizaje profundo, incluidos las redes neuronales convolucionales y las redes neuronales recurrentes, se han aplicado ampliamente en los problemas de previsión del tráfico para modelar las dependencias espaciales y temporales [3], [4] y [5]. En estudios recientes, se ha producido un aumento de la investigación de modelos de IA aplicados al aprendizaje de la representación de grafos, un tipo de estructura de datos formada por un conjunto de objetos (nodos) y sus relaciones (aristas). Existen muchos problemas donde se puede obtener información de las relaciones del entorno y sus valores. Sin embargo, los algoritmos de aprendizaje automático tradicionales, como los de clasificación o regresión, no suelen contemplar las relaciones entre cada observación

y su entorno. Por ello, en los últimos años, han evolucionado las redes neuronales que son capaces de contemplar la estructura topológica del entorno (mediante grafos o otro tipo de arquitecturas). En concreto, para modelos espacio-temporales, se han aplicado técnicas que generalizan las redes neuronales convolucionales a datos con estructuras no euclídeas, como las redes neuronales aplicadas a grafos (GNN). Algunos estudios destacables aplicados a la previsión de tráfico son [6], [7] o [8] .

Los avances en el aprendizaje de la representación de grafos, han dado lugar a nuevos resultados que han evolucionado el estado del arte en numerosos ambitos, como la visión 3D, los sistemas de recomendación o el análisis de redes sociales. La incorporación de dependencias relacionales en las arquitecturas de aprendizaje profundo contribuye a crear sistemas que puedan aprender, razonar y generalizar a partir de este tipo de datos.

1.1 Objetivo

El objetivo de este trabajo de fin de master es el estudio de modelos de IA aplicados a datos espacio-temporales. En concreto, se estudiarán algoritmos basados en redes neuronales que tienen en cuenta la estructura de un grafo (GNN). Estas técnicas, se aplicarán a la predicción de datos de tráfico en un contexto urbano (ciudad de Madrid). Se emplearán datos de sensores de tráfico que pertenecen al portal web de datos abiertos de la Comunidad de Madrid [9].

Para evaluar los modelos, se utilizarán medidas asociadas a problemas de regresión, debido a que la variable objetivo (intensidad de tráfico) es continua. Algunas de las medidas empleadas son: el error cuadrático medio (RMSE), la media porcentual de errores absolutos (MAPE) o la media de errores absolutos (MAE). Estas medidas se evaluarán en general y en determinadas zonas en particular.

A continuación, se enumeran los objetivos de una manera más detallada:

1. Familiarización con datos geoespaciales y búsqueda de representaciones en python.

2. Investigación de estructuras de grafos que conserven las características topológicas de las calles conservando sus longitudes y direcciones.

3. Evaluación de la complejidad computacional de los algoritmos, analizando cual es el más apropiado en cada aspecto del proyecto.

4. Investigación de modelos de IA en estado del arte para la predicción de datos espacio-temporales.

5. Estudio matemático de los modelos aplicados.

6. Aplicación de modelos de IA basados en grafos en python.

7. Evaluación de los modelos.

1.2 Estructura del proyecto

Este trabajo abarca los siguientes capítulos:

1. Revisión del estado del arte: búsqueda de bibliografía para el estudio de técnicas y modelos para la resolución de datos espacio temporales aplicados al problema del tráfico.

2. Desarrollo teórico T-GCN: descripción de los aspectos matemáticos teóricos de modelos de IA espacio-temporales. Se comienza explicando las redes neuronales convolucionales que constituyen la base de los dos tipos de arquitecturas GNN, grafos convolucionales (GCN) y grafos con capas basadas en técnicas de atención (GAT).

3. Procesamiento de datos y análisis: se explica el procesado de datos espaciales hasta la crecaión del grafo de entrada al modelo y el procesado de los datos de sensores.

4. Modelado y resultados: se discute sobre los parámetros más apropiados para el modelo aplicado y se analizan los resultados.

5. Conclusiones y futuros estudios: dicusión sobre los resultados obtenidos en el proyecto y sus conclusiones. Se establecen futuras lineas de investigación.

Capítulo 2

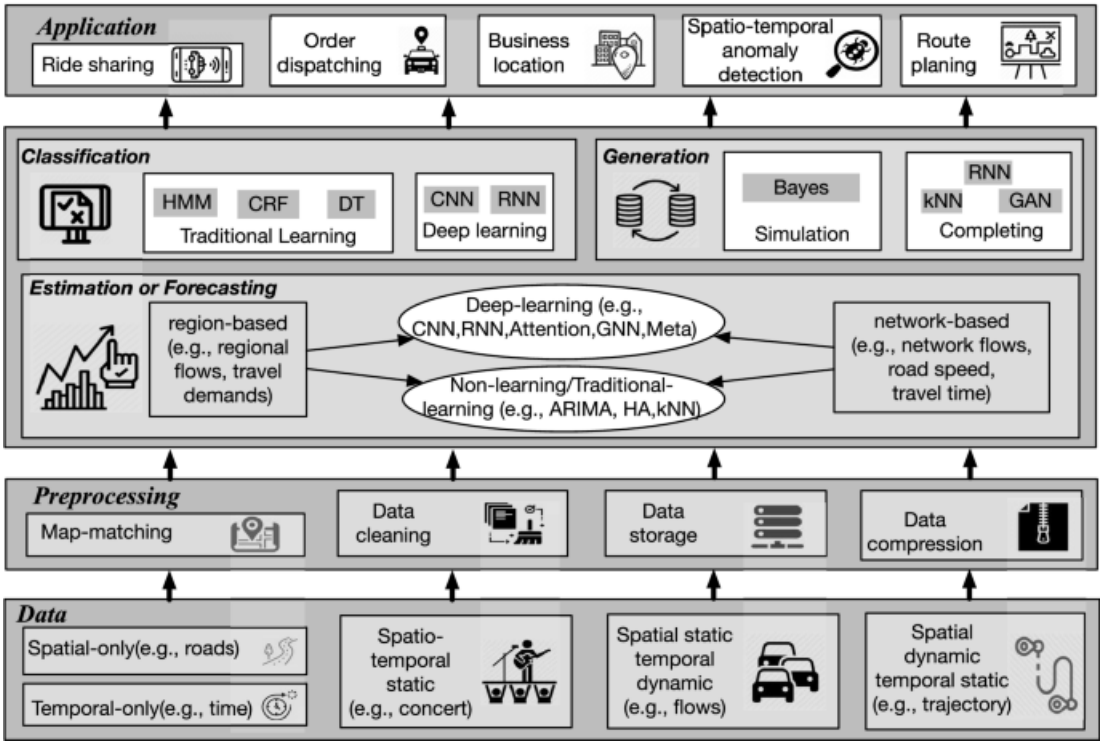
Revisión del estado del arte

La investigación de modelos aplicados a datos espacio-temporales, abarca desde algoritmos clásicos de *machine learning* (ML) hasta la aplicación de redes neuronales. En los algoritmos de ML, resulta complejo modelar simultáneamente este tipo de estructuras y, por ello, a menudo se suelen combinar varios métodos encadenados, como *clustering* junto con algoritmos de clasificación, regresión o de series temporales. Sin embargo, las redes neuronales han demostrado ser capaces de capturar las dependencias espacio-temporales conjuntamente.

En este trabajo se emplean datos de sensores de tráfico geolocoalizados para estudiar modelos de aprendizaje profundo aplicados a datos espacio-temporales. La previsión del tráfico se basa en datos históricos de sensores situados en diferentes puntos geográficos, junto con factores externos que afectan, como el tiempo atmosférico y los días festivos. Por tanto, los datos presentan tanto dependencia espacial, como dependencia temporal, que puede ser estacional. Se pueden distinguir dos problemas asociados a la predicción del tráfico: flujo y velocidad [10]. Este proyecto se centrará en la predicción del flujo (o intensidad), a pesar de que la velocidad del tráfico sigue los mismos pasos en su resolución.

En [11], se recopila literatura sobre la predicción del tráfico con datos espacio-temporales y dividen el problema en cuatro secciones: recopilación de datos geoespaciales, preprocesamiento de datos, predicción del tráfico o clasificación (dependiendo del problema que se quiera resolver) y posibles aplicaciones, como detección de anomalías o planificación de rutas. Se considera que estas cuatro secciones son un buen punto de referencia a seguir en este proyecto. Los pasos descritos en [11], se pueden apreciar en la figura 2.1.

Figura 2.1: Resumen predicciones con tráfico



Fte: [11]

El preprocesamiento de datos cambia en función del modelo empleado. Por ejemplo, en los modelos que utilizan grafos, se necesita crear la estructura y sus conexiones, mientras que en los modelos tradicionales de ML y algunos de aprendizaje profundo no. Por ello, para revisar la literatura asociada a la predicción del tráfico, se crearán tres secciones acordes con los tres tipos de modelos recopilados: enfoques basados en modelos tradicionales de ML, basados en modelos de aprendizaje profundo (en particular, diferentes arquitecturas combinadas de redes neuronales recurrentes y convolucionales) y, por último, enfoques basados en técnicas de aprendizaje profunzo con estructuras de grafos, (GNN).

2.1 Enfoque con Modelos tradicionales

Los modelos tradicionales de series temporales, como los autorregresivos y de media móvil (ARIMA), no pueden manejar conjuntamente las dimensiones del espacio y tiempo. A pesar de ello, hay artículos donde estudian la estacionalidad y realizan predicciones de tráfico con este tipo de modelos como en [12]. En estudios como en [13], tratan de introducir a los modelos ARIMA la componente espacial (STARIMA). En este artículo, observan una correlación entre diferentes series temporales de sensores y, concluyen que,

sensores cercanos (aparentemente conectados por calles) tienen correlación con un determinado decalaje. Esto es debido a que los vehículos que circulan en un instante por el primer sensor, al cabo de un tiempo, también pasan por el siguiente sensor más cercano. Con la velocidad del vehículo y la distancia entre sensores, obtienen el decalaje entre las dos series temporales. Sin embargo, al no tener la estructura de calles ni la dirección, este enfoque pierde rigurosidad.

Otros artículos encontrados en la literatura, utilizan algoritmos basados en k vecinos más cercanos (K-NN) [14] y [15]. Debido a que este algoritmo tiene un tiempo de ejecución muy elevado, en [15] intentan reproducirlo implementándolo de manera distribuida empleando un entorno de *Hadoop*. En este artículo, también consideran la dimensión espacial mediante las correlaciones entre sensores más cercanos, pero presentan las mismas deficiencias expuestas anteriormente sobre la componente espacial. Respecto a [14], es interesante el enfoque del algoritmo K-NN aplicado a un grafo, introduciendo las direcciones de las calles y pudiendo optimizar más la solución. Además, se usan las propiedades típicas de un grafo como agrupamiento o detección de comunidades.

Por último, hay una literatura extensa sobre algoritmos ensambladores de árboles (*XGBoost, LightGBM o Random Forest*). En [16] se usa el algoritmo *XGBoost* después de detectar las estacionalidades de la serie temporal y en [17] aplican el algoritmo *LightGBM* y lo comparan con otros como *Random Forest*.

En estos enfoques se ha demostrado que, a pesar de ser tradicionales y no tener la misma precisión que otros más avanzados, se ha intentado buscar formas de introducir la componente espacial. Algunos artículos, mencionan en las conclusiones la dificultad de procesar los datos de tráfico y el sesgo que pueden introducir al modelo los datos faltantes o anómalos y su procesado. Estos aspectos se tendrán en cuenta en la parte práctica del proyecto escogiendo solo los sensores que sean fiables y tengan pocos periodos sin datos y estableciendo métodos para rellenar los valores faltantes.

Los artículos destacados dentro de esta sección se encuentran en la tabla 2.1:

Tabla 2.1: Literatura recopilada con modelos clásicos

Enfoque con Modelos de *Machine Learning*		
Título y Enlace	Año	Modelo
A Unified STARIMA based Model for Short-term Traffic Flow Prediction [13]	2018	*Space-Time Autoregressive*(STARIMA)
Traffic Congestion Prediction System using K-Nearest Neighbour Algorithm [18]	2020	K-NN (Spark)
Real-time Traffic Congestion Detection using Combined SVM [19]	2013	SVM
Short-Term Traffic Flow Prediction Based on XGBoost [16]	2018	XGBoost
Traffic Prediction Based on Ensemble Machine Learning Strategies with Bagging and LightGBM [17]	2019	LightGBM

Fte: Elaboración propia

2.2 Enfoque con aprendizaje profundo

Los modelos basados en el aprendizaje profundo, presentan propiedades que los hacen más adecuados para la regresión espacio-temporal como su capacidad para aproximar funciones complejas o su facilidad para el aprendizaje de la representación de características, que permite descubrir relaciones más complejas en los datos. En la literatura, se han probado muchas estructuras de redes para la regresión espacio-temporal. Para datos basados en secuencias ordenadas, como series temporales, las redes neuronales recurrentes (RNN), debido a su estructura recursiva, tienen una naturaleza privilegiada. Sin embargo, no es fácil utilizarlas para modelar relaciones espaciales, lo que las hace menos adecuadas para este tipo de problemas. Por esta razón, es frecuente que aparezacan combinadas con otro tipo de arquitecturas como CNN para modelar la información espacial, como en [20]. Las CNN han demostrado ser eficientes con datos espacio-temporales y predicciones a corto plazo, encontrando relaciones espaciales euclideas. En [4], se consideran las relaciones espaciales como una cuadrícula que es posible modelar con CNN. Esta estructura de malla, actúa como una convolución en una imagen, es decir, se realizan operaciones de *max-pooling* con los vecinos más cercanos de la cuadrícula.

Otro tipo de arquitecturas más innovadoras son las de secuencias (Seq2Seq). Esta arquitectura se compone de un codificador y un decodificador. El codificador actúa sobre una secuencia de entrada dada y, el decodificador, produce otra secuencia de salida

para hacer una predicción de varios pasos a la vez. Este enfoque se puede ver en [21].

Por último, otro enfoque aplicado en varios artículos es el uso de redes neuronales residuales [22], donde emplean los residuos de la capa CNN como input de la siguiente capa (tabla 2.2).

Tabla 2.2: Literatura recopilada con redes neuronales

Enfoque con Modelos de *Deep Learning*		
Titulo y Enlace	Año	Modelo
A Spatio-Temporal Spot-Forecasting Framework forUrban Traffic Prediction [4]	2020	CNN
City-Wide Traffic Congestion Prediction Based on CNN, LSTM and Transpose CNN [20]	2020	CNN y LSTM
Spatio-Temporal Traffic Flow Prediction in Madrid: An Application of Residual Convolutional Neural Networks [22]	2021	ResNet
Network Traffic Prediction based on Seq2seq Model [21]	2021	Seq2Seq

Fte: Elaboración propia

Como se ha observado en la literatura recopilada en esta sección, la mayoría se basan en combinaciones o diferentes formas de aplicar CNN. Es por ello que las CNN, han demostrado ser un enfoque satisfactorio para predecir datos con relaciones espaciales.

En la mayoría de los enfoques descritos anteriormente aplicados a la predicción de tráfico, se considera que las relaciones entre sensores de tráfico son euclideas, sin tener en cuenta la dirección de las calles de una ciudad ni sus longitudes. Por ejemplo, dos sensores pueden estar próximos pero no tener calles o carreteras que los comunican, y, por tanto, sus valores no estar relacionados espacial ni temporalmente. El formato de los enfoques que se han presentado en este apartado, funciona bajo el supuesto de que los datos tengan una estructura espacial euclideana, mientras que, los datos de redes de carreteras siguen una estructura más compleja. En el siguiente apartado se verán algunos enfoques donde se aplican estructuras de grafos para modelar de una forma más precisa las calles y carreteras de un mapa.

2.3 Enfoque con grafos y aprendizaje profundo

Los algoritmos de aprendizaje profundo basados en grafos (GNN), son capaces de crear una estructura geométrica más compleja que las CNN y se ajustan a la red de carreteras de tráfico. Varios modelos basados en GNN han demostrado un rendimiento superior a los enfoques anteriores en tareas como el flujo de tráfico por carretera. Basándose en la teoría de grafos, tanto los nodos como las aristas tienen sus propios atributos, que pueden utilizarse posteriormente en las operaciones de convolución. Estos atributos describen diferentes estados del tráfico, por ejemplo, volumen, velocidad, número de carriles, nivel de la carretera, etc. Las redes neuronales basadas en grafos, tienen la misma base que las redes neuronales convolucionales. La diferencia, es la forma de considerar las relaciones espaciales: las CNN consideran los puntos de su entorno de una manera euclidea, mientras que los grafos utilizan relaciones no euclideanas.

En el artículo [23] y en [24], se presenta una revisión de todas las arquitecturas de aprendizaje profundo aplicadas a grafos. En este trabajo, se estudiarán dos tipos de estructuras: grafos formados por capas convolucionales (GCN) y grafos construidos con capas de atención (GAT). En [10] se describen todos los artículos publicados hasta el 2021 sobre GNN y sus distintos enfoques. Algunos de ellos, se han tenido en cuenta en la aplicación práctica de este proyecto (tabla 2.3):

Tabla 2.3: Literatura recopilada con redes neuronales aplicadas a grafos

Enfoque con Modelos de *Deep Learning*		
Titulo y Enlace	Año	Modelo
GMAN: A Graph Multi-Attention Network for Traffic Prediction [8]	2019	GAT
Graph Neural Networks for Traffic Forecasting [6]	2021	GCN
Graph learning-based spatial-temporal graph convolutional neural networks for traffic forecasting [25]	2021	T-GCN
Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting [26]	2019	T-GAT

Fte: Elaboración propia

En los problemas de predicción de tráfico aplicando GNN, uno de los aspectos más importantes es la creación del grafo. En la literatura se han encontrado diferentes enfoques aplicados a la predicción de tráfico. En algunos de ellos, el grafo se construye calculando las distancias de un sensor a los más próximos (grafo no dirigido), sin tener en cuenta

la estructura de calles [26], [25]. En este trabajo se construirá el grafo de una manera novedosa, teniendo en cuenta las rutas entre los sensores y las longitudes de las calles que los comunican.

Capítulo 3

Desarrollo teórico T-GCN

La previsión del tráfico es una tarea difícil debido a sus dependencias espaciales y temporales complejas:

- Dependencia espacial: la estructura topológica de las calles infuye en la variación del volumen de tráfico. El estado del tráfico en las calles próximas al sensor repercute en la predicción de los datos del sensor, por lo que se produce un efecto de retroalimentación.

- Dependencia temporal: el volumen de tráfico cambia dinámicamente a lo largo del tiempo y se refleja principalmente en la periodicidad y la tendencia.

En el capítulo dos, se ha reflejado cómo resolver en la bibliografía la predicción del tráfico mediante enfoques tradicionales y más innovadores. En la tabla 3.1 se recopila un resumen del tratemiento de la dependencia espacial y temporal de los distintos enfoques. En este proyecto se aplica el modelo T-GCN donde se combinan capas GCN con LSTM, para modelar la componente espacial y temporal simultáneamente (T-GCN). Se introduce como entrada al modelo los datos históricos de las series temporales del flujo de tráfico y una matriz de adyacencia que contiene las conexiones de calles y su peso. La red convolucional de grafos se utiliza para capturar la estructura topológica de las calles y las capas recurrentes se utilizan para modelar la componente temporal de las series. El cambio dinámico se obtiene mediante la transmisión de información entre las unidades, para capturar las características temporales. Por último, se obtienen resultados a través de una capa totalmente conectada. Este algoritmo se ha implementado en python mediante la librería StellarGraph [27] y tensorflow [28]. En esta sección, se describen los aspectos matemáticos del modelo T-GCN para realizar la previsión del tráfico basado en la estructura topológica de las calles. Para ello, se dividirá en dos secciones: componente espacial, temporal y combinación de la componente espacial y temporal.

Tabla 3.1: Relaciones espaciales y temporales de los modelos recopilados

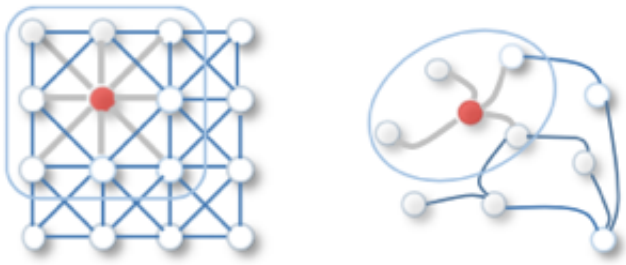
Modelos	Relación temporal	Relación espacial
Modelos tradicionales: ARIMA	Sí. Fácil interpretación	No
Redes recurrentes: LSTM	Sí. Mayor complejidad y precisión que ARIMA en grandes volúmenes de datos	No
Redes convolucionales: CNN	No	Sí (relaciones estrictamente euclídeas)
Redes convolucionales con grafos: GCN	No	Sí (relaciones no euclídeas). Conservan estructuras topológicas
Redes neuronales de grafos con capas de atención: GAT	No	Sí (relaciones no euclídeas). Conservan estructuras topológicas
CNN+LSTM (T-CNN)	Sí. Mediante las capas LSTM	Sí (relaciones euclideas)
GAT+LSTM (T-GCN)	Sí. Mediante las capas LSTM	Sí (relaciones no euclideas)
GCN+LSTM (T-GCN)	Sí. Mediante las capas LSTM	Sí (relaciones no euclideas)

Fte: Elaboración propia

3.1 Componente espacial

Los recientes avances de las redes neuronales profundas, especialmente en las CNN [29], han permitido desarrollar las GNN. Las redes neuronales basadas en grafos, tienen su fundamento en la capacidad de las redes convolucionales de extraer características espaciales localizadas a múltiples escalas y componerlas para construir representaciones. Las claves de las CNN son la conexión local, los pesos compartidos y el uso de múltiples capas [30], siendo estas características importantes para el desarrollo de las GNN. Sin embargo, las CNN solo pueden operar con datos en el espacio euclídeo mientras que las GNN pueden conservar otras propiedades espaciales generalizando el espacio (figura 3.1).

Figura 3.1: Diferencias entre la estructura de una imagen (euclidea) y un grafo (no euclidea)



Fte: [31]

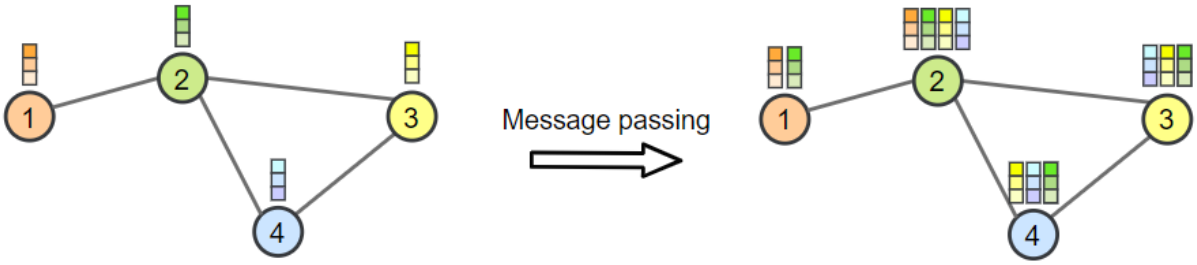
Hay diferentes arquitecturas para las capas de GNN, como grafos formados por capas convolucionales (GCN), grafos formados por capas de atención (GAT) o grafos formados por capas recurrentes (GRN). En esta sección se explicarán la GCN debido a que el algoritmo implementado usa este tipo de capas.

Las redes convolucionales aplicadas a grafos, fueron introducidas por [32]. Son similares a las redes convoluciones en imágenes, en el sentido de que los parámetros del filtro se comparten en todas las ubicaciones del grafo. Se basan en métodos de intercambio de mensajes, lo que significa que los nodos intercambian información con los vecinos y se envían mensajes entre sí.

Como notación común en todas las secciones de este capítulo, se define grafo, nodo, arista y matriz de adyacencia. Matemáticamente, un grafo G es una tupla formada por un conjunto de nodos V, y un conjunto de aristas E, tal que $G = (V, E)$. Cada arista es un par de dos vértices y representa una conexión entre ellos. La matriz de adyacencia A es cuadrada y sus elementos indican si los pares de vértices son adyacentes, es decir, están conectados, o no. En el caso más simple, A_{ij} es 1 si hay una conexión desde el nodo i a j, y en caso contrario 0. Si hubiera pesos en las aristas, en vez de ser un 1, se introduciría el peso.

El primer paso consiste en que cada nodo crea un vector de características que representa el mensaje que quiere enviar a todos sus vecinos. En el segundo paso, los mensajes se envían a los vecinos, de modo que un nodo recibe un mensaje por cada nodo adyacente. Visualmente, se puede representar como se puede apreciar en la figura 3.2.

Figura 3.2: Estructura GCN



Fte: [32]

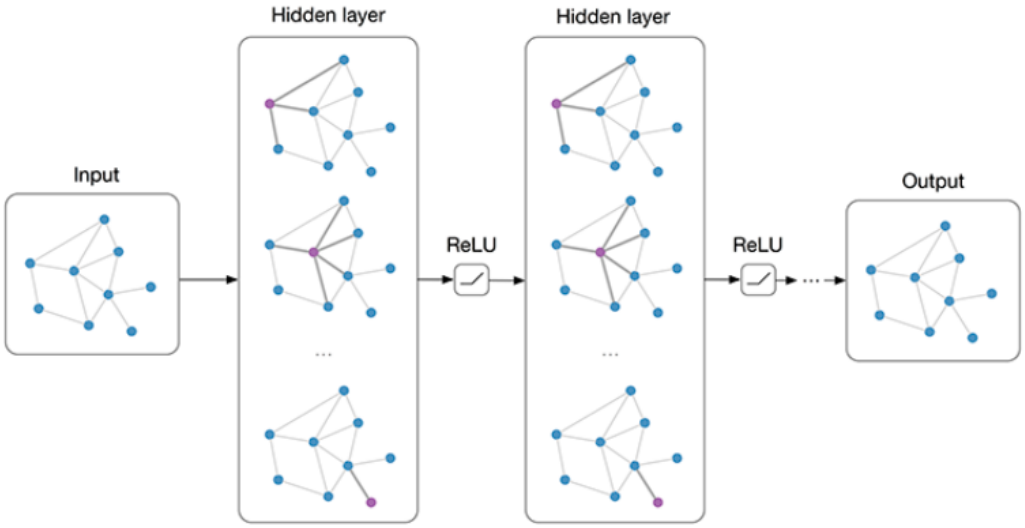
Los valores de salida del primer nodo son la media de sí mismo y del segundo nodo. Del mismo modo ocurre en todos los demás nodos. Formulado en términos matemáticos, es preciso decidir cómo combinar todos los mensajes que recibe un nodo. Como el número de mensajes varía según los nodos, es necesario una operación que funcione para cualquier número (como sumar o sacar la media). Dadas las características anteriores de los nodos $H^{(l)}$, la capa GCN se define como:

$$H^{(l+1)} = \sigma\left(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}H^{(l)}W^{(l)}\right) \tag{3.1}$$

$W^{(l)}$ son los parámetros de peso con los que se tranforman las características de entrada en mensajes ($H^{(l)}W^{(l)}$). A la matriz de adyacencia A, se añade la matriz de identidad para que cada nodo se envíe su propio mensaje también a sí mismo: $\hat{A} = A+I$. Finalmente, se calcula la matriz $\hat{D}$, que es una matriz diagonal con D_{ii}, denotando el número de vecinos que tiene el nodo i. σ representa una función de activación arbitraria (normalmente se utiliza una función de activación basada en ReLU en las GNNs). La ecuación de la convolución enunciada, puede resolverse mediante dos métodos: espectral o espacial. Los métodos espectrales, resuelven una parte de la ecuación mediante el laplaciano del grafo y utilizan la transformada de Fourier, lo que causa que la complejidad computacional sea alta. Los métodos espaciales, utilizan un filtro que consiste en el reparto de parámetros en los vecinos de cada nodo agregando características locales.

Para permitir el intercambio de características entre los nodos más allá de sus vecinos, se aplican múltiples capas GCN con funciones de activación como ReLU. En la siguiente imagen se muestra la concatenación de varias capas (figura 3.3):

Figura 3.3: Combinación de múltiples capas GCN



Fte: [33]

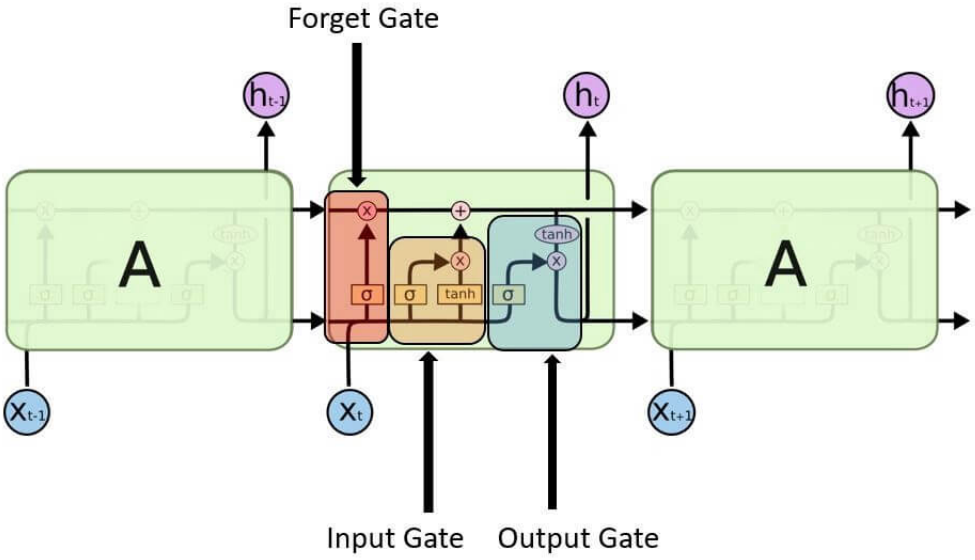
3.2 Componente temporal

Los tipos de redes más utilizadas para modelar la componente temporal son las redes neuronales recurrentes, en concreto las LSTM y las GRU.

Las LSTM tienen conexiones de retroalimentación que las diferencian de las redes neuronales más tradicionales. Esta propiedad permite a las LSTM procesar secuencias enteras de datos sin tratar cada punto de la secuencia de forma independiente, sino conservando información útil sobre los datos anteriores de la secuencia para ayudar a procesar los nuevos puntos de datos. Por ello, las LSTM se utilizan para procesar series temporales.

La aquitectura de una LSTM consta de 3 partes: la puerta donde se decide qué información pasada almacena, la puerta de entrada y la puerta de salida. El esquema de predicción es (figura 3.4):

Figura 3.4: Estructura de una neurona LSTM



Fte: [34]

1. Puerta de memoria (en la figura 3.4 es la parte roja): decide qué información se almacena y descubre los detalles que deben ser descartados del bloque mediante una función sigmoide. Observa el estado anterior h_{t-1} y el contenido de entrada X_t y emite un número entre 0 (omitirlo) y 1 (mantenerlo) para cada número del estado de la celda h_{t-1}.

2. Puerta de entrada (en la figura 3.4 es la parte amarilla): descubre qué valor de la entrada debe utilizarse para modificar la memoria. La función sigmoide decide qué valores se dejan pasar. La función *tanh* dá peso a los valores decidiendo su nivel de importancia.

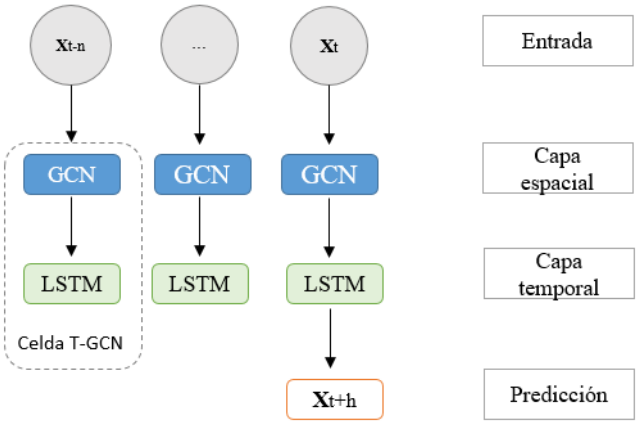
3. Puerta de salida (en la figura 3.4 es la parte azul): la entrada y la memoria del bloque se utilizan para decidir la salida. La función sigmoide decide qué valores dejar pasar por 0 o 1. Y la función *tanh* decide qué valores se dejan pasar por 0, 1. Y la función *tanh* da ponderación a los valores que se pasan, decidiendo su nivel de importancia que va de -1 a 1 y se multiplica con una salida sigmoide.

Estos tres pasos se repiten iterativamente tantas veces como instantes de tiempo haya.

3.3 Combinación de la componente espacial y temporal

Para capturar las dependencias espaciales y temporales de los datos de tráfico al mismo tiempo se propone una red T-GCN con una capa convolucional de grafo GCN y otra recurrente LSTM. En resumen, el modelo T-GCN puede tratar la dependencia espacial y temporal. Por un lado, la red convolucional de grafos se utiliza para capturar la estructura topológica de la red vial urbana para obtener la dependencia espacial. Por otro lado, la capa recurrente se utiliza para captar la variación dinámica de la información del tráfico en las carreteras para obtener la dependencia temporal y para realizar tareas de predicción del tráfico (figura 3.5):

Figura 3.5: Estructura T-GCN



Fte: elaboración propia

Capítulo 4

Procesamiento de datos y análisis

4.1 Formulación del problema

En esta sección, se expone matemáticamente los elementos que participan en la resolución del problema de predicción de tráfico mediante GNN:

Definición 1. Grafo de red de calles. Se define como un grafo $G(V, E, A)$ donde V es el conjunto de nodos que representan la intersección de calles, E son las aristas que representan los segmentos de calles entre cada intersección y A es la matriz de adyacencia.

Definición 2. Grafo dual de red de calles. Se define el grafo dual $G^*(V^*, E^*, A^*)$ a partir del grafo $G(V, E, A)$ como:
- V^* es el conjunto de nodos de G^*, que corresponden con las aristas E del grafo G. Es decir, cada nodo está asociado a una arista del grafo G y representa un segmento de una calle entre dos intersecciones.
- E^* es el conjunto de aristas de G^*, donde cada arista $e^* \in E^*$ está asociada a un nodo V de G. Pueden existir varias aristas e^* asociadas a un mismo nodo $v \in V$ dependiendo del número de aristas e adyacentes a G.

Definición 3. intensidad de Tráfico. Para cada calle V^* del grafo G^*, el valor $x_i^t \in X$ representa la intensidad del tráfico dentro del nodo v_i en el momento t.

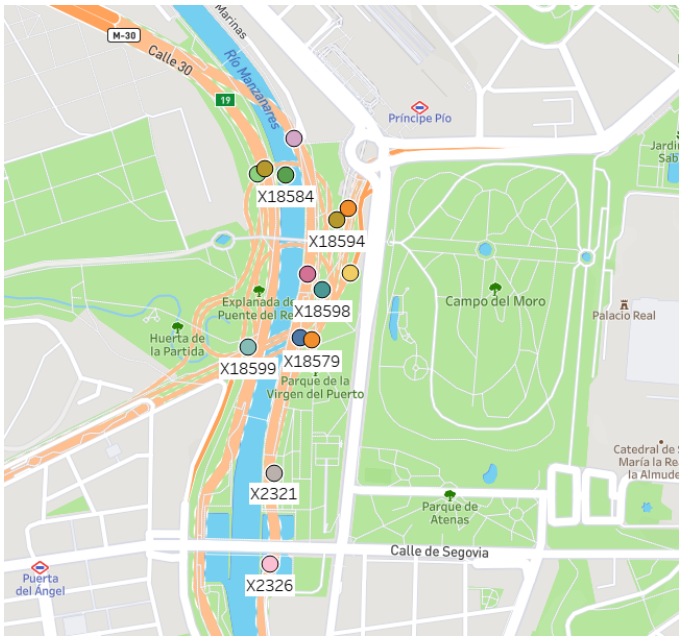
Definición 4. Predicción de la intensidad de tráfico. la predicción de la intensidad de tráfico en el grafo G se define como la función f que genera la predicción $y = f(X, \varepsilon; G^*)$ donde y representa el estado de tráfico del nodo n predicho a partir de la ventana $x_n = (x_n^1, ..., x_n^t)$, siendo t el número de espacios temporales en la ventana de histórico y ε los factores externos como el tiempo atmosférico o los eventos de calendario.

4.2 Datos iniciales

El código de la parte práctica de este proyecto se encuentra en un repositorio público de *github* con la dirección [35].

Se han empleado diferentes fuentes de datos abiertos para la predicción del tráfico en Madrid. La información de sensores, se ha recopilado de la página web de la Comunidad Madrid [9] y se estructura en dos tablas con la información temporal de los sensores y su geolocalización (latitud y longitud). De esta forma, es posible visualizar cómo se encuentran distribuidos los sensores y su información temporal. Por ejemplo, la visualización de 14 sensores de la región de Madrid-Río es (figura 4.1):

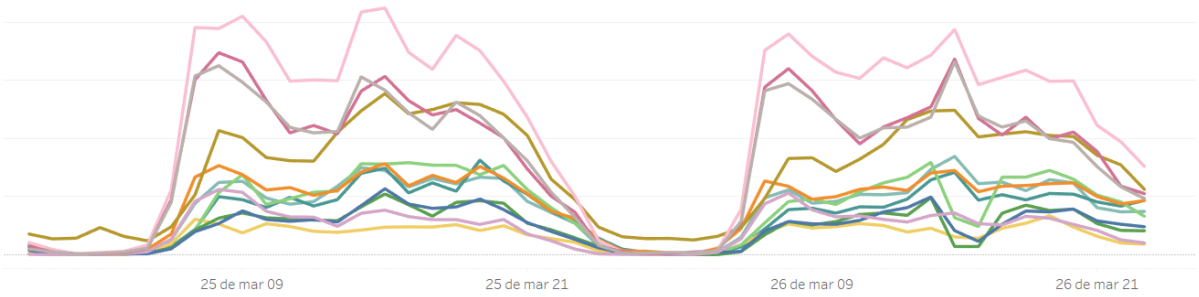
Figura 4.1: Zona Madrid-Río



Fte: elaboración propia

Hay más de 2500 sensores encontrados en la página oficial de la comunidad de Madrid. Sin embargo, algunos de ellos no tienen datos durante un periodo de tiempo considerable (esto se detalla en el apartado 4.4). El número de sensores seleccionado es 1244. Respecto a la tabla de la información temporal de los sensores, contiene los datos de intensidad y valocidad del tráfico cada 10 minutos. En este trabajo, se ha agregado el dato a una granularidad horaria (en el apartado 4.2.2 se detallará el procesado de las series temporales) y se predice la intensidad de tráfico. Para los sensores de la figura 4.1, un ejemplo del aspecto de su información de la intensidad de tráfico para 2 días se muestra en la figura 4.2, donde se puede apreciar las series temporales de intensidad de tráfico para un ejemplo del 25 y 26 de marzo de 2021 y para los sensores representados en la figura 4.1.

Figura 4.2: Zona Madrid-Río: ejemplo serie temporal intensidad de tráfico



Fte: elaboración propia

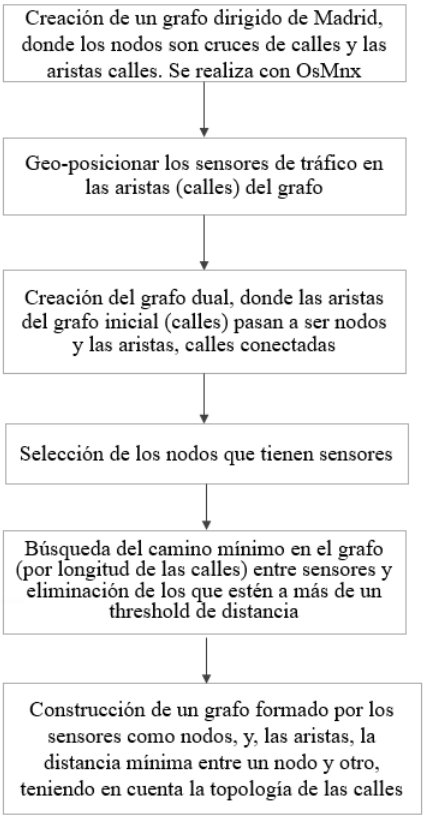
Se observan diferentes estacionalidades donde la más destacada es la diaria. A partir de las 5 de la mañana comienza a incrementar la intensidad del tráfico. Existen subestacionalidades como en las horas centricas de la mañana o en horas del medio día.

4.3 Creación del grafo geoespacial

La tarea de crear un grafo geoespacial representando la topología de las calles de Madrid y el posicionamiento de los sensores, ha sido la etapa que ha abarcado más tiempo en este proyecto. La dificultad de este proceso reside principalmente en la complejidad computacional de eliminación de nodos (apartado 4 de esta sección). En los enfoques de la literatura donde aplican algoritmos basados en grafos, se preserva la estructura topológica de las calles con sus direcciones. Sin embargo, la consideración de las longitudes de las calles y la introducción de ellas como pesos en el grafo, es un enfoque novedoso con escasa literatura.

En figura 4.3 se exponen los pasos que se han realizado para la creación del grafo de entrada al modelo predictivo. El objetivo del procesado de datos del grafo es conseguir simplificarlo preservando la estructura topológica de las calles (conexiones y sentidos de calles) y se considere sus longitudes como pesos del grafo. A continuación, se explican detalladamente cada uno de los pasos de la figura:

Figura 4.3: Pasos para la construcción del grafo de tráfico



Fte: elaboración propia

1. Construcción de un grafo con las calles de Madrid: para construir un grafo que represente la topología de las calles de Madrid se ha empleado la librería de python *osMnx* [36] que permite descargar datos geoespaciales de *OpenStreetMap* y modelar, proyectar, visualizar y analizar las redes de calles del mundo real mediante grafos. Con las funciones propias de esta librería, es posible obtener un grafo de las calles de Madrid, donde los nodos sean cruces de calles y las aristas calles. El resultado de la función es una tabla con la información de los nodos (cruces de calles) y otra con la información de las aristas (segmentos de calles). Por ejemplo, en la zona de Madrid-Río representada en la 4.2, se obtiene el grafo de la figura 4.4 donde los nodos son puntos verdes y representan intersecciones de calles y los segmentos de calles se representan en azul. La información se presenta en dos tablas con información geoespacial, donde una contiene la información de los nodos y la otra la información de las aristas.

Figura 4.4: Grafo con nodos intersecciones de calles y aristas segmentos de calles

Grafo zona Madrid-Río

Contenido tabla de nodos

y	x	geometry
40.421264	-3.684452	POINT (-3.68445 40.42126)
40.420234	-3.687794	POINT (-3.68779 40.42023)
40.417360	-3.688989	POINT (-3.68899 40.41736)
40.414991	-3.688980	POINT (-3.68898 40.41499)
40.412775	-3.688901	POINT (-3.68890 40.41277)

Contenido tabla de aristas

u	v	oneway	lanes	length	highway	maxspeed	tunnel	geometry
171946	26513145	True	4	41.093	tertiary	50	NaN	LINESTRING (-3.68445 40.42126, -3.68442 40.42163)
171948	1209331142	True	3	14.084	primary	50	NaN	LINESTRING (-3.68779 40.42023, -3.68796 40.42020)
171951	1209331009	True	NaN	18.733	residential	NaN	NaN	LINESTRING (-3.68899 40.41736, -3.68885 40.41749)
171951	991794380	True	3	54.211	tertiary	50	NaN	LINESTRING (-3.68899 40.41736, -3.68903 40.41687)
171952	26486617	True	3	26.251	tertiary	NaN	NaN	LINESTRING (-3.68898 40.41499, -3.68897 40.41475)

Fte: elaboración propia

Este grafo tiene un total de nodos 143072 y 200042 aristas. Como se puede apreciar en la información de las aristas (tabla 4.1), contienen dirección y longitud. La descripción de cada una de las variables de las dos tablas es:

Tabla 4.1: variables de las aristas para el grafo de red de calles

grafo de red de calles: variables de las aristas	
variable	descripción
u	nodo de partida
v	nodo de llegada
oneway	si es posible circular por la vía en las dos direcciones
lanes	número de carriles de la vía
length	longitud del segmento entre el nodo u y v
highway	etiqueta utilizada para clasificar cualquier tipo de vía
maxspeed	máxima intensidad en el segmento de la vía
tunnel	si contiene un tunel el segmento entre u y v
geometry	geometría de la geolocalización del segmento de vía entre u y v

Fte: elaboración propia

Las calles tienen carriles con sentido diferente, se reflejan en la tabla como aristas distintas. La variable *oneway* indica si la calle admite vehículos en ambos sentidos.

Tabla 4.2: variables de los nodos para el grafo de red de calles

grafo de red de calles: variables de los nodos	
variable	descripción
y	coordenada geográfica de la latitud del nodo
x	coordenada geográfica de la longitud del nodo
street count	número de vías que intersecan en el nodo
geometry	punto geométrico formado por las coordenadas (y, x)

Fte: elaboración propia

2. Geoposicionamiento de los sensores de tráfico sobre las aristas del grafo: sobre la tabla de aristas explicada en el paso anterior, se selecciona la variable *geometry*, que contempla el polígono formado por todos los límites del segmento de la calle al que pertenece. Mediante una función de la librería de python *osMnx*, se asocian las latitudes y longitudes de los sensores de tráfico a las geometrías de las aristas. Esta función consiste en contrastar si el punto del sensor está dentro de la geometría definida de la calle. En caso de que esté dentro, el sensor se asocia a la arista. Los sensores de tráfico tienen una precisión de 6 decimales en su latitud y longitud, que equivale a un área de 11 centímetros. Se observa como en Madrid-Río la asignación de las aristas del grafo es precisa (figura 4.5):

Figura 4.5: Asociación de los sensores a aristas del grafo inicial

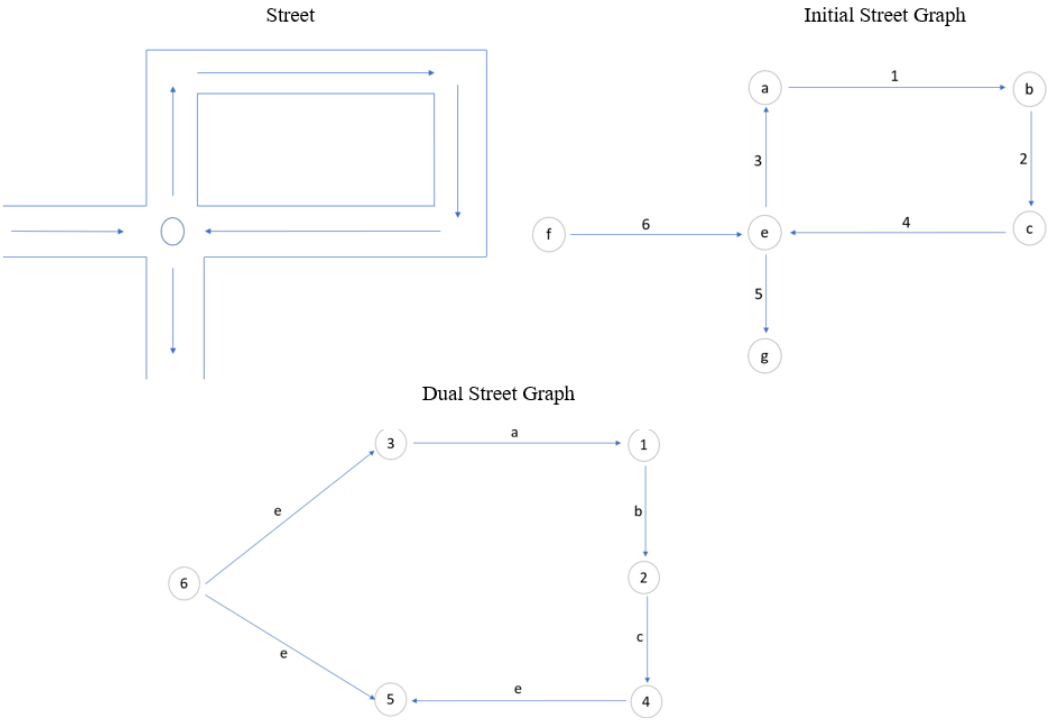
id	nombre	latitud	longitud	from	to
1047	03FL08PM01	40.419552	-3.720851	26080693	32636493
6827	18NC20PM01	40.418307	-3.721659	21723292	1881246852
6839	17NC80PM01	40.415183	-3.722345	21723290	21723292
6840	17NC65PM01	40.413740	-3.722429	21723296	21723290
1045	18RU04PM01	40.419163	-3.721048	32636490	315266750
1012	18RA66PM01	40.419861	-3.722097	32636475	3402272923
1016	18NC52PM01	40.420443	-3.721929	21723313	21723233
1014	18RL09PM01	40.419889	-3.722677	32636476	3402272922
1044	03FT08PM01	40.419124	-3.720757	26080693	32636493
1043	18RT11PM01	40.418333	-3.720762	32636490	315266287
1013	18XC46PM01	40.419971	-3.722532	21724021	3402292034

Fte: elaboración propia

3. Construcción del grafo dual: debido a que los algoritmos de GNN aplicados a problemas de regresión deben contener la información a predecir en los nodos y no la de

las aristas, es necesario pasar la información de los sensores a los nodos. Para realizar este proceso, se ha calculado el grafo dual, siguiendo la definición matemática de grafo dual de calles realizada al inicio de este capítulo. El grafo definido en el paso 1 está formado por los nodos como intersecciones de calles y las aristas como los segmentos de calles que hay entre cada intersección. El grafo dual definido contiene, en cada nodo, la información de las aristas del grafo anterior, y, en las aristas, los segmentos conectados. En el ejemplo de la figura 4.6 se observa cómo se ha construido el nuevo grafo:

Figura 4.6: Construcción del grafo dual



Fte: elaboración propia

Este proceso se ha realizado transformando la notación de las dos tablas generadas de la información de los nodos y aristas. Dentro de la tabla de información de aristas, la conexión de u a v, se denota con el nombre X_i, siendo i un número entero generado para cada arista. Esto se puede observar en la siguiente figura 4.7 con los nodos de la zona de Madrid-Río como ejemplo:

Figura 4.7: Datos de construcción del grafo dual

id sensor → arista grafo inicial → nodo grafo dual

id	u	v	name_node
1012	32636475	3402272923	X18584
1013	21724021	3402292034	X2390
1014	32636476	3402272922	X18585
1016	21723313	21723233	X2333
1043	32636490	315266287	X18595
1044	26080693	32636493	X9457
1045	32636490	315266750	X18594
1047	26080693	32636493	X9457
6827	21723292	1881246852	X2323
6833	32636471	315266288	X18579
6835	32636471	315262454	X18580
6836	32636494	674492230	X18598
6837	32636495	315264896	X18599
6839	21723290	21723292	X2321
6840	21723296	21723290	X2326

Fte: elaboración propia

En el recuadro rojo aparece el identificador del sensor asociado a la arista del grafo inicial (cuadro azul). En gris aparece el nombre del nodo del grafo dual.

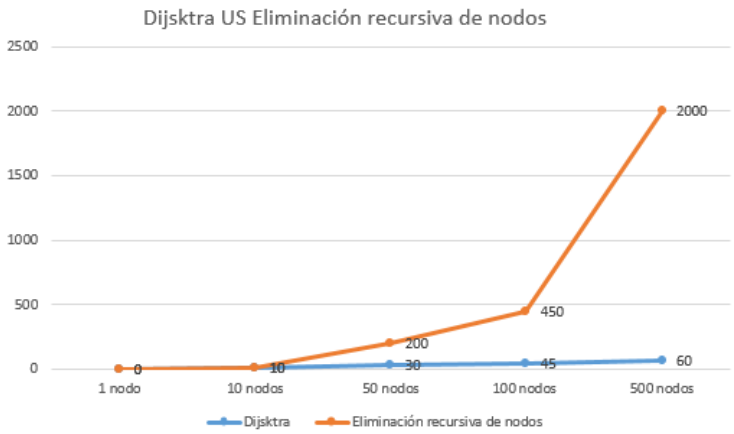
Para calcular la tabla de aristas, se conectan todos los nodos X_i que estén conectados mediante u y v y se cambia su notación. Para ello, se selecciona la variable u sobre la tabla de la figura 4.7 y se cambia el nombre al establecido del grafo dual, como siguiente paso se encuentran todos los nodos v conectados con u y se cambia la notación.

4. Eliminación de los nodos sin sensores: el grafo inicial del que se parte en el paso 1, contiene todas las calles donde algunas de ellas tienen un sensor asociado (paso 2) y otras no. Una vez está creado el grafo dual (paso 3) con la información de los sensores en los nodos correspondientes, se eliminan aquellos que no contengan un sensor. Para ello, se han realizado varios enfoques distintos. Es importante realizar la eliminación de nodos conservando las conexiones y longitudes de calles, en caso contrario, se pierden las propiedades más importantes del grafo. Esta tarea es la que ha abarcado más tiempo en su realización debido a la complejidad computacional de los algoritmos implementados. En la figura 4.8 se comparan tiempos de ejecución de los diferentes enfoques probados.

El primer enfoque realizado, se ha descartado debido a su elevada complejidad computacional. Dado un grafo G y un nodo v_i del conjunto de nodos V, iterativamente se elimina el nodo v_i y se conectan sus nodos padre v_{i-1} con sus hijos v_{i+1}, sumando el peso de las aristas desde v_{i-1} a v_{i+1}. Este proceso se repite hasta eliminar los nodos que

no tengan sensores del grafo. Al ser el número de nodos sin un sensor mucho mayor al número de nodos con sensor asociado, se requiere eliminar muchos más nodos de los que se conservan. Al ir eliminando nodos intermedios en cada iteración, el número de conexiones incrementa y, en consecuencia, aumenta el tiempo de ejecución. Este algoritmo se ha elaborado mediante una implementación propia y el código se encuentra en *github* ([35]). Como se observa en la figura 4.8, a medida que se elimina mayor cantidad de nodos, el algoritmo va creciendo en tiempo de ejecución (el eje de abcisas se encuentra en segundos). Para futuros trabajos, una forma de mejorar el tiempo de ejecución de este algoritmo puede ser seleccionando los nodos a eliminar en cada iteración en función del número de hijos y padres con los que estén conectados.

Figura 4.8: Dijkstra y eliminación recursiva de nodos



Fte: elaboración propia

El segundo enfoque realizado, tiene una complejidad computacional elevada pero más reducible que el anterior y es factible emplearlo. Por ello, se ha seleccionado este procedimiento para el cálculo. Este algoritmo sigue los siguientes pasos:

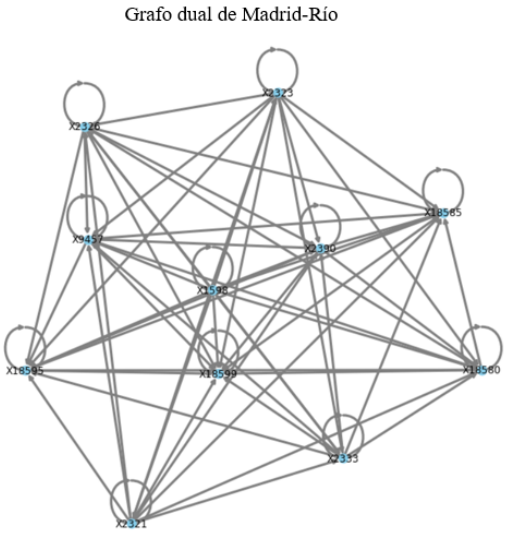
- Selección de los nodos que tienen sensores del grafo dual: se parte del supuesto de que siempre existe uno o varios caminos conectados por calles entre un sensor y otro cualquiera. Se necesita calcular cuál es el camino mínimo que los conecta en base a las longitudes de las calles. Por ello, se seleccionan todos los nodos que tienen sensores del grafo dual.

- Cálculo del producto vectorial entre sensores: se realizan todas las combinaciones de conexiones entre nodos con sensores.

- Aplicación del algoritmo de Dijkstra: se ha implementado en la librería de python *networkx* para cada conexión entre sensores. El resultado final es una matriz de adyacencia donde cada par de sensores tiene como peso la distancia mínima entre ellos. De esta forma, se introducie como peso de las aristas la suma de las longitudes de los segmentos

de calle del camino mínimo de un sensor a otro. Dependiendo del algoritmo de Dijkstra implentado el tiempo de ejecución varía. En la librería de python empleada se usa el algoritmo *treeset* ([37]).

Figura 4.9: Matriz de distancias topológicas del grafo dual



Matriz de distancias entre sensores siguiendo la topología de calles

to \ from	X1598	X2321	X2323	X2326	X2333	X2390	X9457	X18580	X18585	X18595	X18599
X1598	0	7.405	7.534	7.758	6.607	5.334	7.589	3.464	3.015	3.399	3.687
X2321	7.405	0	259	353	1.186	2.755	2.694	238	687	494	1.634
X2323	7.534	259	0	613	927	3.015	2.954	368	817	624	1.605
X2326	7.758	353	613	0	1.539	3.108	3.047	415	864	671	1.811
X2333	6.607	1.186	927	1.539	0	3.264	2.857	831	1.280	1.087	1.141
X2390	5.334	2.755	3.015	3.108	3.264	0	3.051	1.139	690	1.521	1.239
X9457	7.589	2.694	2.954	3.047	2.857	3.051	0	1.234	1.297	954	287
X18580	6.929	476	736	830	1.663	2.279	2.469	0	449	381	1.459
X18585	6.031	1.374	1.634	1.728	2.561	1.381	2.593	449	0	830	1.010
X18595	6.798	988	1.248	1.341	2.174	3.041	1.907	381	830	0	1.241
X18599	7.373	3.268	3.210	3.621	2.283	2.477	574	1.459	1.010	1.241	0

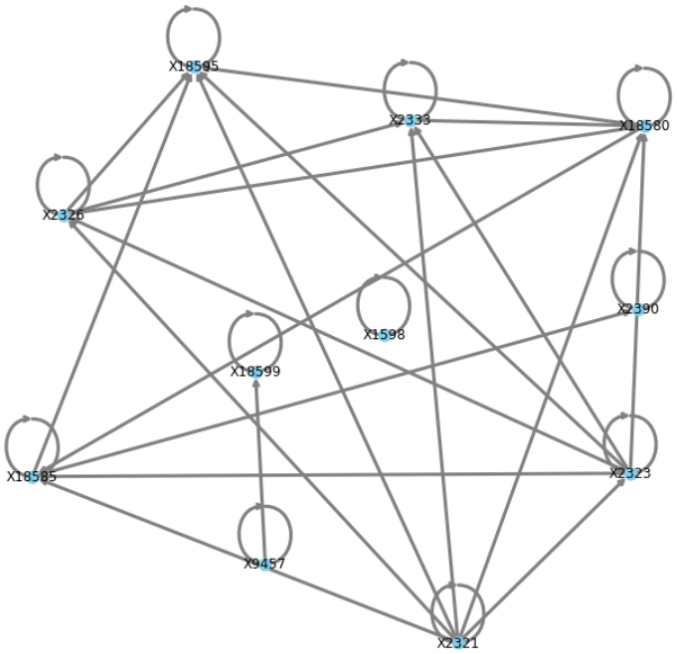
Fte: elaboración propia

En la imagen 4.2 se puede observar como están distribuídos los sensores en el mapa tomando el ejemplo de los sensores de Madrid-Río mostrados en las figuras 4.1 y 4.2. Se puede apreciar que la matriz de distancias de sensores de 4.9 no es simétrica, por lo que se están considerando las direcciones de las calles (en caso contrario, la matriz sería simétrica). Destacan los sensores $X2321, X2323$ y $X2326$ por encontrarse a poca distancia topológica entre ellos. De la misma manera, estos tres sensores están relacionados con $X18580, X18585$ y $X18595$. Al situar en el mapa $X2321, X2323$ y $X2326$, se observa que se encuentran en la M-30 situados en la misma dirección por lo que sus datos es posible que estén relacionados. Por otro lado, se aprecia que los sensores $X18580, X18585$ y 18595 están fuera de la M-30 pero se conectan con ella. Además, entre ellos también tienen una relación por estar en una distancia topológica cercanos. Por último, el sensor $X9457$ se encuentra dentro de la misma calle que el $X18599$ y el sentido de la calle es desde el primero al segundo.

5. Eliminación de conexiones poco significativas entre nodos: como se ha explicado en el paso anterior, se han calculado las distancias mínimas entre todos los nodos. En la matriz de adyacencia que contiene el ejemplo de Madrid-Río de 4.9 se aprecia como algunos nodos se encuentran a distancias muy altas, por lo que los datos de los sensores no estarán relacionados. Para simplificar el grafo y eliminar complejidad al algoritmo, se han seleccionado únicamente las distancias que sean menores a un umbral. Realizando un análisis descriptivo, se ha seleccionado un umbral de 850 metros.

Sobre el ejemplo anterior, escogiendo el umbral de 850 metros, se aprecia el siguiente grafo de conexiones simplificado (4.10). La matriz de adyacencia se simplifica tomando la figura 4.9 y filtrando por el umbral seleccionado:

Figura 4.10: Grafo final simplificado

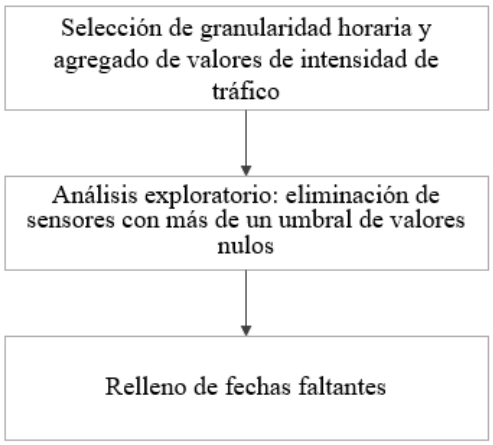


Fte: elaboración propia

4.4 Descripción de datos temporales

Los datos de series temporales que se han utilizado son de intensidad de tráfico, expresada como el número de vehículos que transitan. La granularidad inicial de las series temporales es cada 10 minutos pero el dato se ha agregado a cada hora. Este proceso se ha realizado sumando los valores de todas las intensidades por hora y por sensor. Una vez agregado el dato, se ha estudiado por cada sensor el número de valores faltantes. Los sensores que tengan más de 72 horas faltantes desde 2021-01-01 a 2021-01-12, son eliminados por falta de datos. En total el número de sensores seleccionados es de 1244. Los datos faltantes se han rellenado por el valor anterior no nulo. El proceso seguido se puede ver en la figura 4.11.

Figura 4.11: Preprocesamiento de series temporales

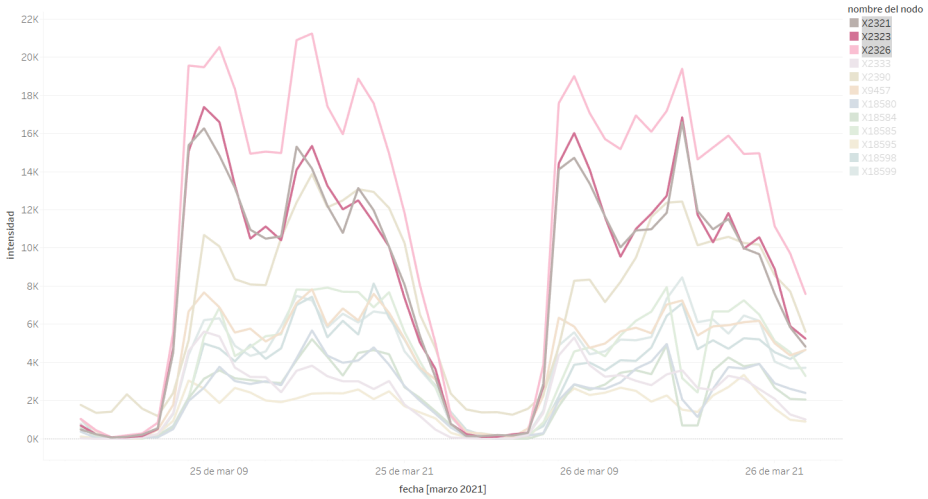


Fte: elaboración propia

Como se ha indicado en los apartados anteriores, las series temporales tienen una dependencia espacial y temporal.

Dentro de la dependencia temporal se han encontrado diferentes patrones de estacionalidad. Tomando como ejemplo las series de la zona de Madrid-Río $X2321, X2323$ y $X2326$, se puede apreciar (figura 4.12) que las tres tienen una estacionalidad diaria con subestacionales dentro de las horas del día (entre las 11 y las 13 de la mañana hay menos intensidad de vehículos). A pesar de que en el gráfico no se puede apreciar debido a que solo se muestran dos días, existe una estacionalidad semanal y anual, donde los mismos patrones de vacaciones o navidades se repiten cada año.

Figura 4.12: Series temporales de la zona Madrid-Río

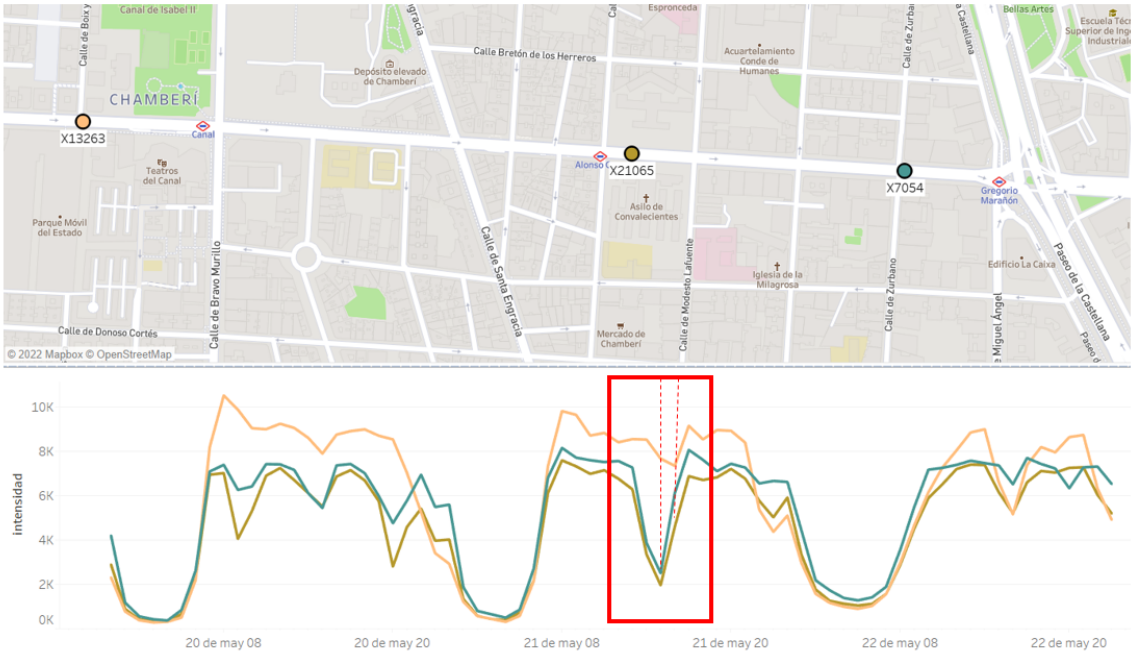


Fte: elaboración propia

La dependencia espacial en las series temporales se puede apreciar en atascos, donde

el flujo de vehículos se propaga a lo largo de las calles cercanas conectadas. Cuando ocurre esta situación, la estacionalidad diaria se desvirtúa y la intensidad de tráfico sigue un patrón diferente. Esto ocurre, por ejemplo, en la serie temporal de datos de la calle José Abascal de Madrid, que contiene 3 sensores. Se puede observar en la figura 4.13 que para el viernes 21 de mayo toma valores diferentes al patrón que sigue diariamente. Buscando el motivo, en los reportes de la página de datos de Madrid, se ha encontrado en los informes diarios que ese día había retenciones en esta calle. En los datos de las series temporales se puede observar que a las tres de la tarde desciende el número de vehículos (intensidad) en los sensores más cercanos a la Castellana ($X7054$ y $X21065$). A las cuatro de la tarde se puede apreciar que desciende ligeramente la serie temporal amarilla (sensor $X13263$).

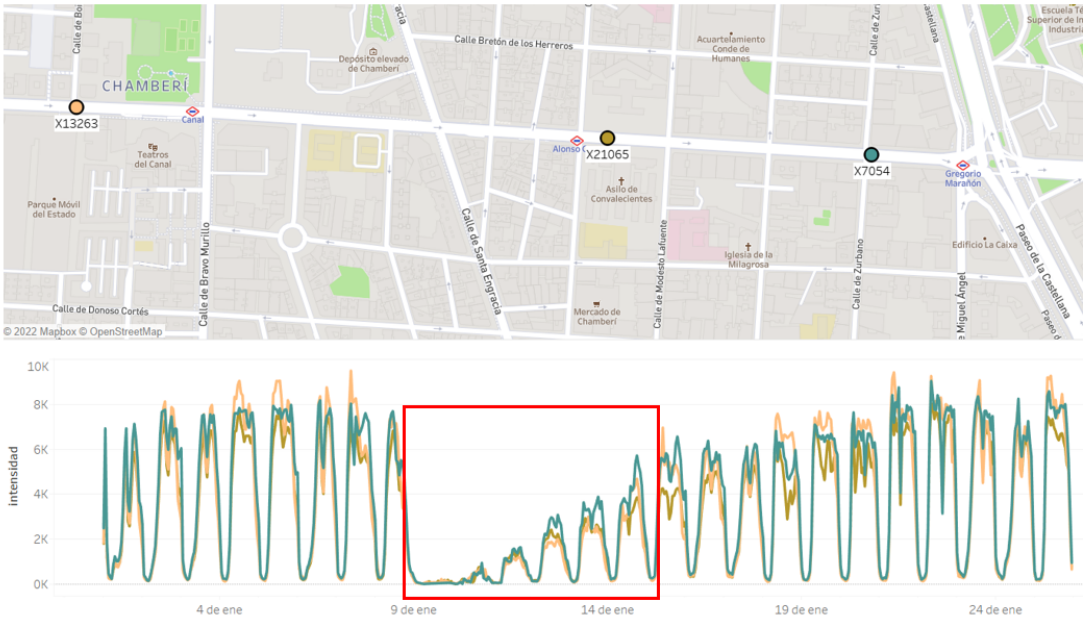
Figura 4.13: Series temporales de la calle José Abascal



Fte: elaboración propia

Por último, existen otros factores que pueden alterar las estacionalidades como días festivos o temporales. El temporal filomena afectó en gran medida al tráfico, esto se puede ver en la siguiente figura 4.14:

Figura 4.14: Efecto filomena



Fte: elaboración propia

4.5 Introducción de datos al modelo

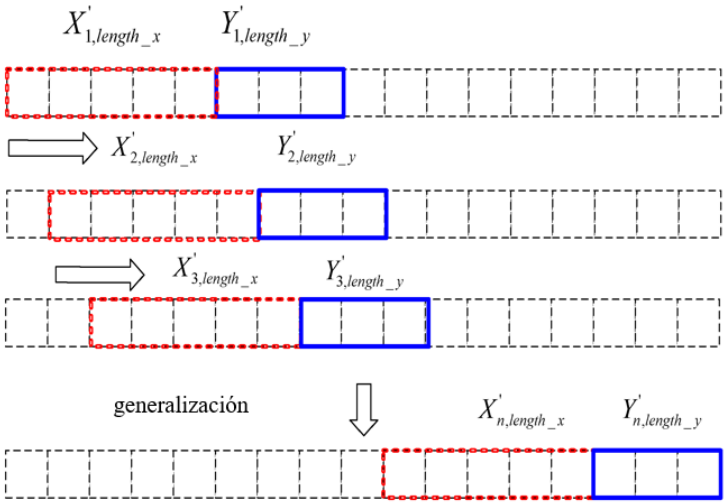
Se tienen dos conjuntos de datos: la matriz de adyacencia con las relaciones espaciales de los sensores y los datos de las series temporales de la intensidad de tráfico. Los datos de relaciones espaciales son constantes a lo largo del tiempo, es decir, la matriz de adyacencia no cambia.

Para introducir los datos temporales de intensidad de tráfico al modelo T-GCN, se separan en entrenamiento, validación y test. El conjunto de validación se introduce junto con el de entrenamiento al aprendizaje de la red neuronal para evitar el sobreajuste. Con el conjunto de test se utiliza el modelo ajustado para realizar las predicciones. Se ha dejado un 15 por ciento del dataset para validación. El número de periodos que se reservan para el test es de 24 y 42 periodos. Como siguiente paso, se normalizan los datos entre 0 y 1 (figura 4.16) tanto de las series temporales de intensidad de tráfico como de la matriz de adyacencia. Los datos de la matriz de adyacencia contienen los pesos de las longitudes de las calles donde longitudes elevadas indica que los sensores están más alejados. Para normalizarlos se calcula $\frac{1}{x_i}$ donde i es cada una de las componentes de la matriz. Para normaliar los datos de series temporales se realiza la transformación $\frac{x_i - max(x_i)}{max(x_i) - min(x_i)}$.

Debido a que el modelo T-GCN contiene capas LSTM, los datos que recibe la red

neuronal siguen el mecanismo de entrada a una capa LSTM basado en ventanas, donde se crea una secuencia observaciones pasadas ($X_1, ..., X_n$ siendo n es el número de valores que se utilizan para predecir en el primer paso) para la predicción del dato actual ($Y_1, ..., Y_j$ siendo j es el número de periodos que se quieren estimar). En la predicción del valor posterior al actual, la secuencia se mueve un instante temporal hacia delante, introduciendo el valor de la iteración anterior predicho (figura 4.15).

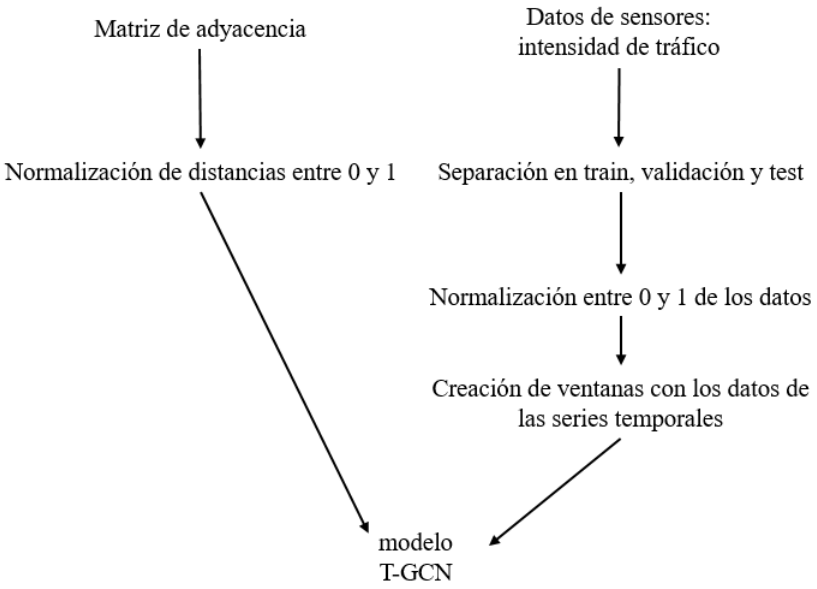
Figura 4.15: Estructura de entrada de los datos temporales al modelo T-GCN



Fte: elaboración propia

El proceso que se ha seguido para la preparación de los datos al formato de entrada del modelo T-GCN han sido:

Figura 4.16: Separación en entrenamiento y test



Fte: elaboración propia

Capítulo 5

Modelado y resultados

En esta sección, se evalua el rendimiento de predicción del modelo T-GCN en el conjunto de sensores seleccionados con los valores de intensidad de tráfico. Se realiza el experimento de predecir la siguiente hora y las siguientes dos horas. Se usan 4 medidas para evaluar los resultados:

1. Raíz cuadrada de la media de los errores al cuadrado (RMSE):

$$RMSE = \sqrt{\frac{1}{MN}\sum_{j=1}^{M}\sum_{j=1}^{N}(y_i^j - \hat{y}_i^j)^2} \tag{5.1}$$

2. Media de los errores al cuadrado (RMSE):

$$MSE = \frac{1}{MN}\sum_{j=1}^{M}\sum_{j=1}^{N}(y_i^j - \hat{y}_i^j)^2 \tag{5.2}$$

3. Media de errores absolutos (MAE):

$$MAE = \frac{1}{MN}\sum_{j=1}^{M}\sum_{j=1}^{N}|y_i^j - \hat{y}_i^j| \tag{5.3}$$

4. Media porcentual de errores al cuadrado (MAPE):

$$MAPE = \frac{100}{MN}\sum_{j=1}^{M}\sum_{j=1}^{N}|\frac{(y_i^j - \hat{y}_i^j)^2}{y_i^j}| \tag{5.4}$$

donde y_i^j representa el tráfico real y $\hat{y}_i^j$ representa el valor predicho. M es el número de periodos de una serie temporal y N es el número de sensores que se han predicho.

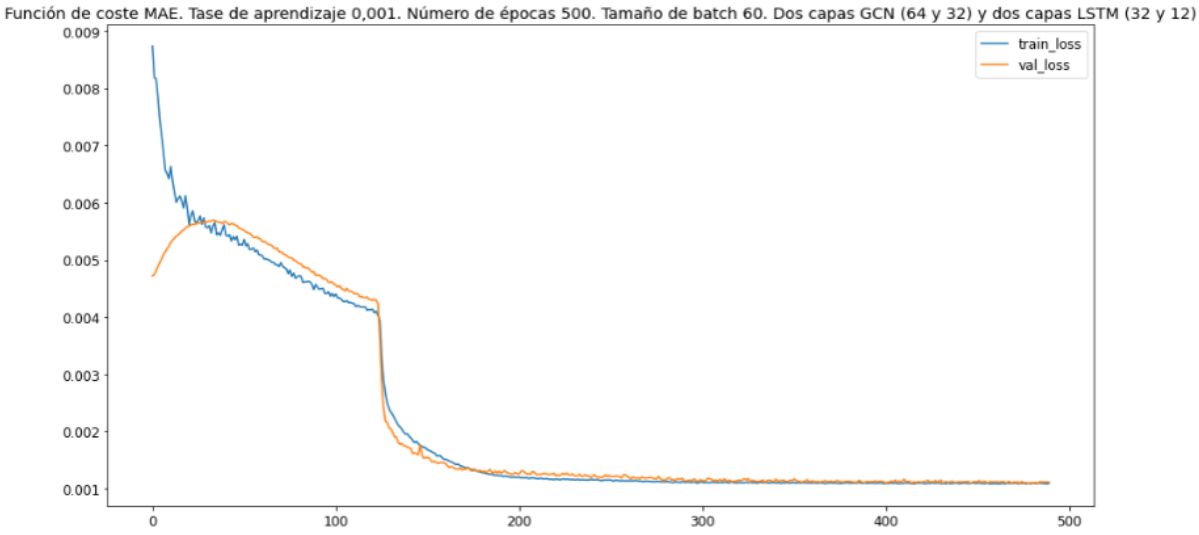
5.1 Hiperparámetros

Como la muestra de test solo se evalua en el modelo final seleccionado, los hiperparámetros del modelo T-GCN se han seleccionado tomando las muestras de entrenamiento y validación. Algunos de los parámetros estudiados son la tasa de aprendizaje, el tamaño de los lotes (número de agrupaciones que se hacen con los datos en el aprendizaje de la red por cada época), número de épocas de entrenamiento, número de capas ocultas y dimensiones de neuronas ocultas. La tasa de aprendizaje seleccionada es 0,001. Se ha probado con el tamaño de los lotes entre 30, 60 y 100, escogiendo 60. El número de épocas del modelo final es 500 debido a que a partir de este valor la función de pérdidas permanece constante.

Las capas que se utilizan es un parámetro importante para el modelo T-GCN debido a que afectan en gran medida a la precisión de la predicción, de la misma forma que el número de neuronas dentro de cada capa. Cuanto mayor sea número de capas y neuronas más compleja es la red, por lo que se tenderá al sobreajuste. Se ha probado con una, dos y cuatro capas GCN combinadas con una y dos capas LSTM. Finalmente, se han seleccionado dos capas GCN y dos capas LSTM debido a que un número más elevado hace que la función de coste se estabilice a las pocas iteraciones e incrementa la complejidad computacional.

Respecto al número de neuronas ocultas, se han seguido tres estrategias diferentes. En la primera, se ha decidido ir disminuyendo el número de dimensiones en cada capa. Este tipo de estructura se suele usar en redes neuronales que también combinan múltiples capas como *autoencoder* ([38]). Se ha probado con varias combinaciones siguiendo este método, siendo una de ellas la reflejada en la figura 5.1 (donde el eje de abcisas es el valor de la función de coste evaluada con la métrica MAE y el eje de ordenadas es el número de épocas), formada por dos capas GCN que contienen 64 y 32 neuronas y la dos capa LSTM que contienen 32 y 16. La figura 5.1 se ha realizado para la predicción de 24 horas. Al visualizar la función de coste obtenida en entrenamiento y validación se observa que a partir de la época 120 la función permanece constante. Esta situación ocurre con otras combinaciones que se han realizado siguiendo esta estrategia, por lo que se han explorado otro tipo de composiciones.

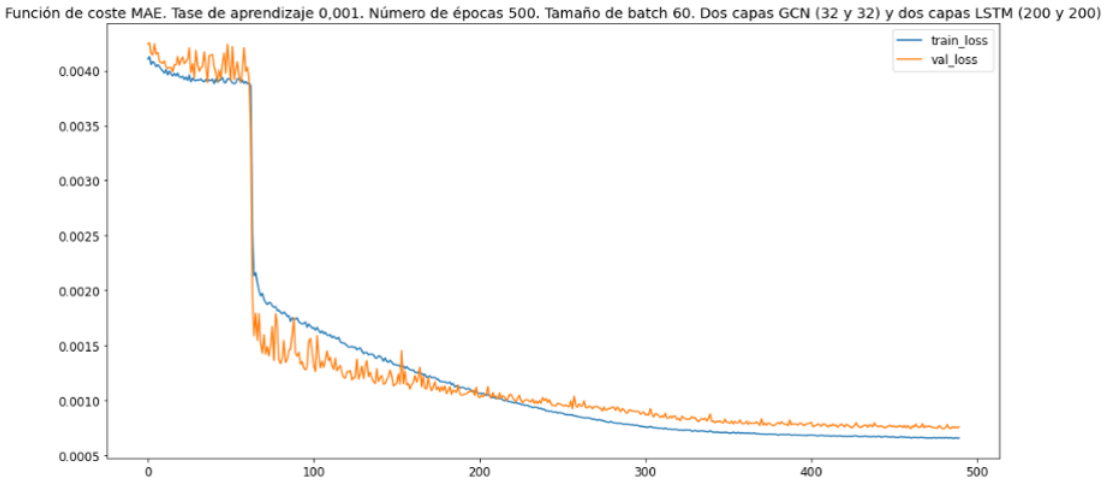
Figura 5.1: T-GCN reducción de la dimensión de neuronas



Fte: elaboración propia

La segunda estrategia consiste en fijar el mismo número de neuronas en las capas GCN y las capas LSTM. Siguiendo los análisis realizados en [39], se ha probado en las capas GCN con 16, 32 y 64 neuronas ocultas y en las capas LSTM con 100 y 200 neuranas ocultas. Uno de los modelos obtenidos ha sido el de la figura 5.2 con las capas GCN con 32 neuronas ocultas y las capas LSTM con 200 neuronas ocultas.

Figura 5.2: T-GCN mismo número de neuronas en los tipos de capas

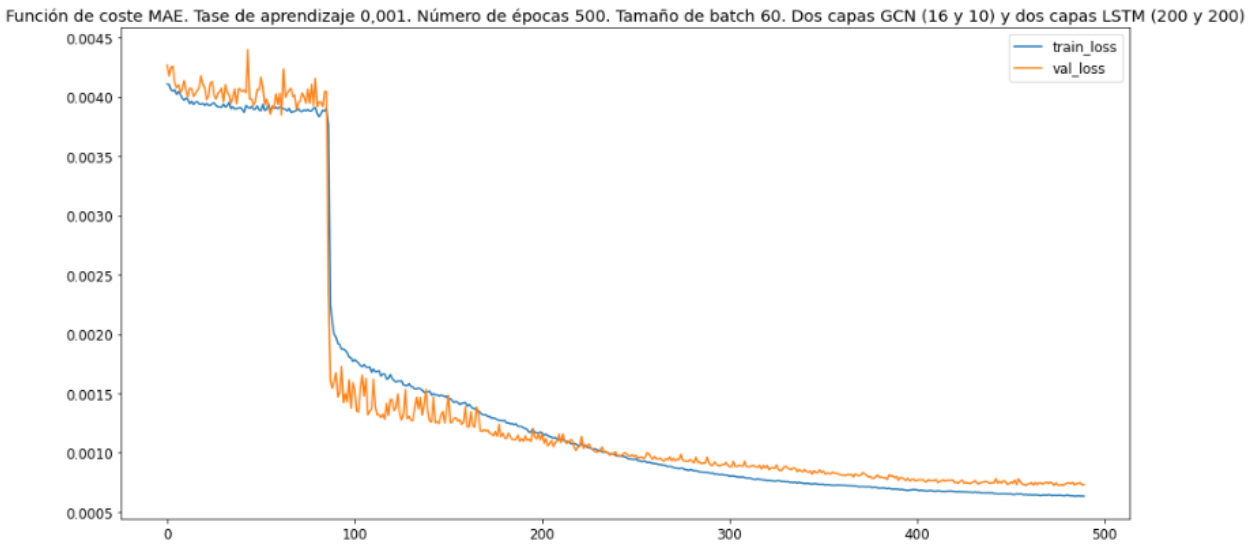


Fte: elaboración propia

La última estrategia es disminuir el número de neuronas ocultas en las capas GCN y fijar las mismas en las capas LSTM y viceversa. En este caso, el modelo que mejores resultados ha obtenido está formado por 16 y 10 neuronas en las capas GCN y 200 y 200 neuronas en las capas LSTM. Se observa que la gráfica de la figura 5.2 es muy similar a

la de 5.3, por lo que se escoge este último modelo debido a que tiene menor complejidad.

Figura 5.3: T-GCN reducción de dimensión GCN y mismo número de neuronas en LSTM

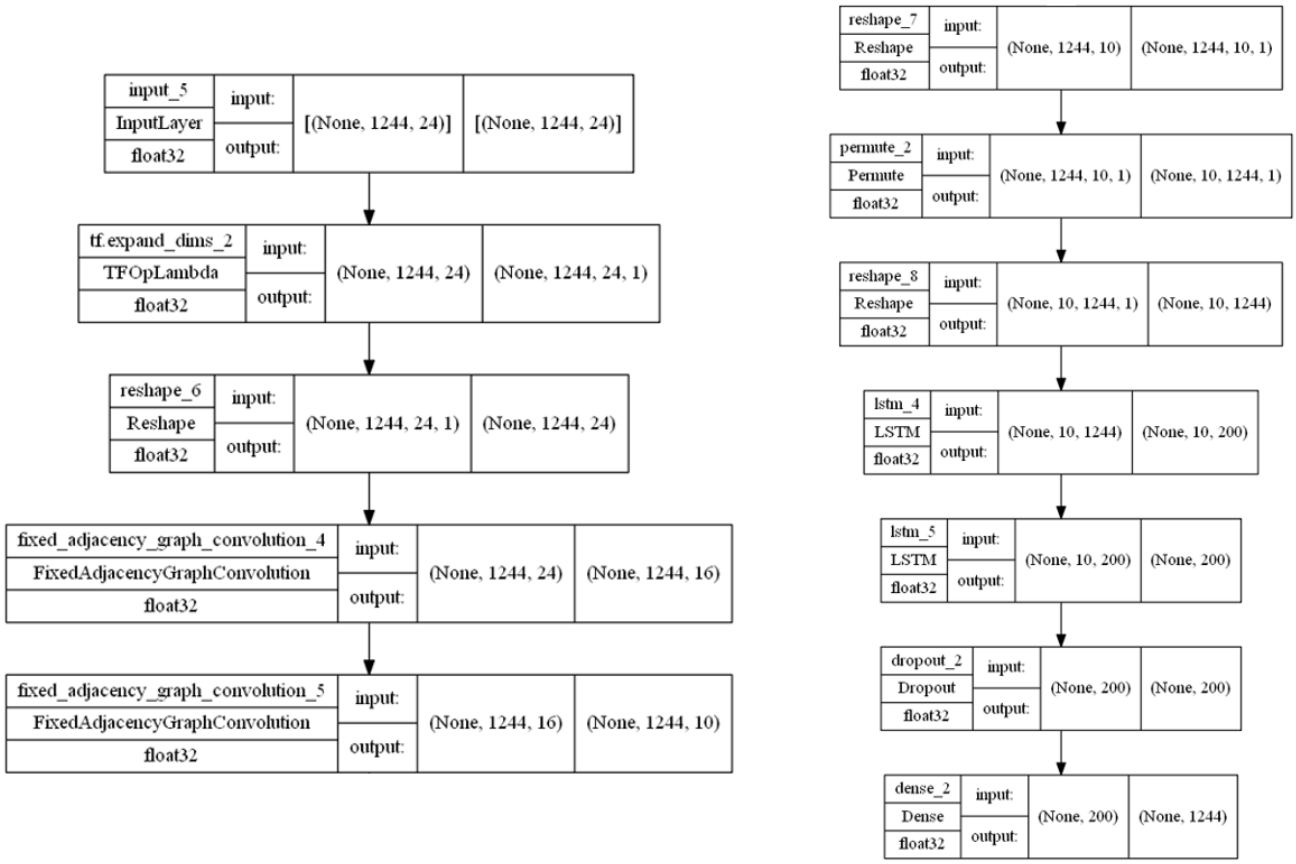


Fte: elaboración propia

Por último, la función de activación escogida para las capas GCN es *relu* y para las capas LSTM *tanh*.

El modelo final tiene el esquema de la figura 5.4. El flujo comienza en la parte izquierda de la imagen y continúa con la parte derecha. Dentro de cada paso se indica la operación que se aplica a los datos, el tipo, la dimensión al llegar al paso y la dimensión de salida. Los tres primeros pasos realizan operaciones de transformación de las dimensiones para poder introducir los datos a las dos capas GCN con 16 y 10 neuronas. Cuando los datos salen de la segunda GCN, se vuelve a transformar las dimensiones para conseguir el formato de entrada de las capas LSTM con 200 dimensiones. Finalmente, hay una capa de *dropout* para evitar el sobreajuste y la última capa densa con una neurona (en el caso de predicción de 1 hora) y una función de activación *relu* que permite devolver los valores predichos en el rango de cero a infinito.

Figura 5.4: Composición del modelo ajustado T-GCN



Fte: elaboración propia

5.2 Evaluación del modelo general y comparación con el modelo base

Se evaluan los resultados de todas las series temporales de los sensores a corto y largo plazo (1, 2 y 24 horas del día 26 de octudre de 2021) mediante las medidas definidas al inicio de este capítulo: MAE, RMSE, MSE, MAPE. Como se puede apreciar en los ejemplos de series temporales de intensidad de tráfico que se han mostrado a lo largo del proyecto (figuras 4.13, 4.12 y 4.2), las series temporales tienen una estacionalidad muy marcada diaria, semanal y anual. En cada sensor, si no existe ninguna irregularidad, los valores de intensidad de tráfico del día anterior serán muy similares a los del actual si son días lectivos. Por tanto, seleccionar los valores del día anterior como predicción del día siguiente, podría ser un buen modelo aproximado. Se comparan los resultados obtenidos con el modeo T-GCN (con los hiperparámetros seleccionados en el apartado 5.1) con el modelo base. A continuación, se muestran los resultados a 1, 2 y 24 horas.

- Predicción de la siguiente hora (tabla 5.1): tomando la muestra de test que no se ha utilizado para la selección de hiperparámetros ni para el entrenamiento del modelo, se predice una hora. Los resultados de los dos modelos en la tabla 5.1 muestran que el modelo T-GCN predice mejores resultados a una hora.

Tabla 5.1: Evaluación general a una hora

Métricas	**T-GCN**	**Modelo base**
MAE	219,78	338,76
MSE	172606,82	521120,19
MSE	415,45	721,88
MAPE	14,35	9,52e15

Fte: Elaboración propia

- Predicción de las siguientes 2 horas (tabla 5.2): tomando dos horas de predicción en vez de una, se aprecia que la métrica de MAE, MSE y RMSE son más bajas las del modelo T-GCN. Se aprecia que el valor del MAPE es mayor a 100, esto es debido a que el valor predicho es mayor que el valor real. Tomando la ecuación del valor del MAPE, definida al inicio del capítulo, se observa que es posible que el valor sea mayor que 100 si ocurre esta situación.

Tabla 5.2: Evaluación general a 2 horas

Métricas	**T-GCN**	**Modelo base**
MAE	179,83	286,47
MSE	134195,99	376611,97
RMSE	366,32	613,68
MAPE	74977,09	9e15

Fte: Elaboración propia

- Predicción de las siguientes 24 horas (tabla 5.3): realizando la predicción de los siguientes 24 periodos, el modelo base obtiene métricas más bajas que el modelo T-GCN ajustado. Esto se puede apreciar principalmente en las medidas MAE, MSE, RMSE. Sin embargo, en la siguiente sección, se observará que el modelo T-GCN en casos particulares de atascos o accidentes se ajusta mejor.

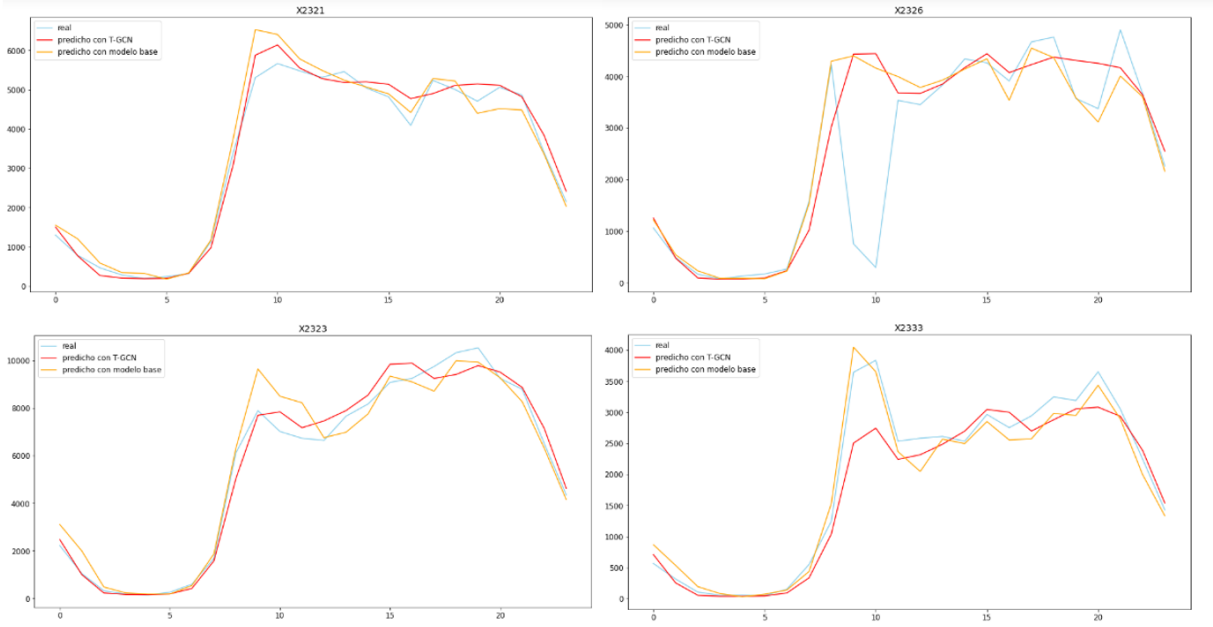
Tabla 5.3: Evaluación general a 24 horas

Métricas	**T-GCN**	**Modelo base**
MAE	407,08	298,79
MSE	1115917,312	775782,04
RMSE	1056,36	880,78
MAPE	6492,25	7,07e15

Fte: Elaboración propia

Se ha demostrado que el modelo T-GCN tiene métricas más bajas en las prediccines de datos a corto plazo (1 y 2 horas). En la predicción a 24 horas, el modelo base presenta mejores predicciones. Esto es debido a que en condiciones normales donde no hay demasiados atascos y la intensidad de vehículos es la habitual, repetir el patrón del día anterior es un buen ajuste. Al ser las métricas empleadas una media de los resultados de cada una de las series temporales, es posible que el modelo base actúe mejor que T-GCN. Por otro lado, realizar la predicción a 24 horas, no permite predecir los atascos puntuales que ocurran a una determinada hora del día. Esto hace que el modelo T-GCN tienda a predecir valores que sean como el patrón que más a ocurrido en periodos anteriores y que el modelo base obtenga buenos resultados. Un ejemplo de las predicciones del modelo a 24 horas en la zona de Madrid-Río (presentadas en la figura 4.2) es el de la figura 5.5, donde en azul se muestra el valor real, en rojo el predicho por el modelo T-GCN y en naranja el modelo base. El eje de abcisas es la intensidad del tráfico y el de ordenadas las horas

Figura 5.5: Ejemplo predicciones a 24 horas en Madrid-Río



Fte: elaboración propia

Se observa que en la serie temporal X2326 de la figura 5.5, tanto el modelo T-GCN como el modelo base, no llegan a predecir la irregularidad en la intensidad del tráfico que se produce a las 10 de la mañana. El modelo base no lo ajusta debido a que repite el patrón del día anterior y no se produjo esa irregularidad. Respecto al modelo T-GCN, no predice este valor debido a que se han realizado predicciones de 24 periodos seguidos. Para poder valorar qué realiza el modelo cuando ocurre una situación inesperada como un atasco, es necesario hacer predicciones a corto plazo, recibiendo el valor real de la hora anterior y prediciendo la siguiente.

5.3 Evaluación del impacto de un atasco

En esta sección se evaluaa qué predice el modelo en el grupo de series temporales afectadas por un atasco. Se realizan predicciones de manera horaria, es decir, recibiendo el valor real de la hora anterior y prediciendo el siguiente. Buscando en la web puntos atascados para el día 26 de octubre de 2021, se ha encontrado un choque entre dos camiones en el puente de ventas que provocó retenciones de tráfico durante varias horas. Esto se puede apreciar en las noticias de la figura 5.6.

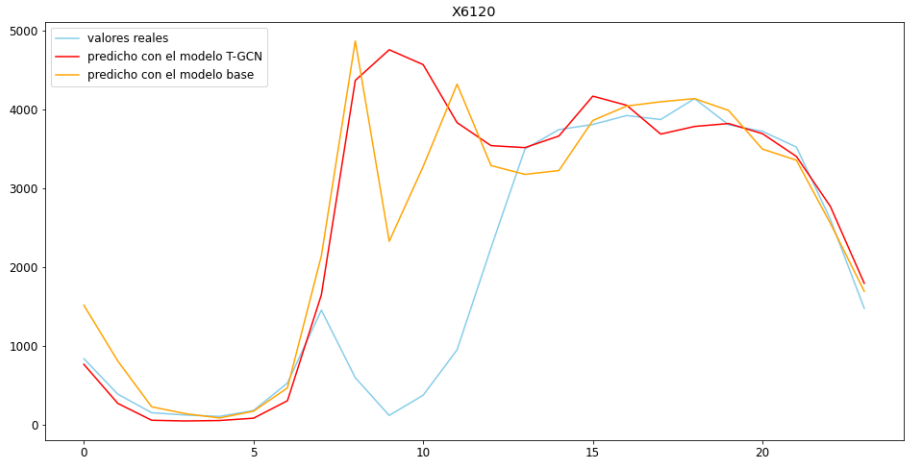
Figura 5.6: Retenciones en puente de ventas

Fte: elaboración propia

Hay un sensor situado en el puente de ventas donde disminuyó de manera considerable la intensidad de vehículos durante el periodo de tiempo que se produjo el atasco. Observando los resultados de las predicciones a 24h del modelo T-GCN y del modelo base, se

obtiene (figura 5.7):

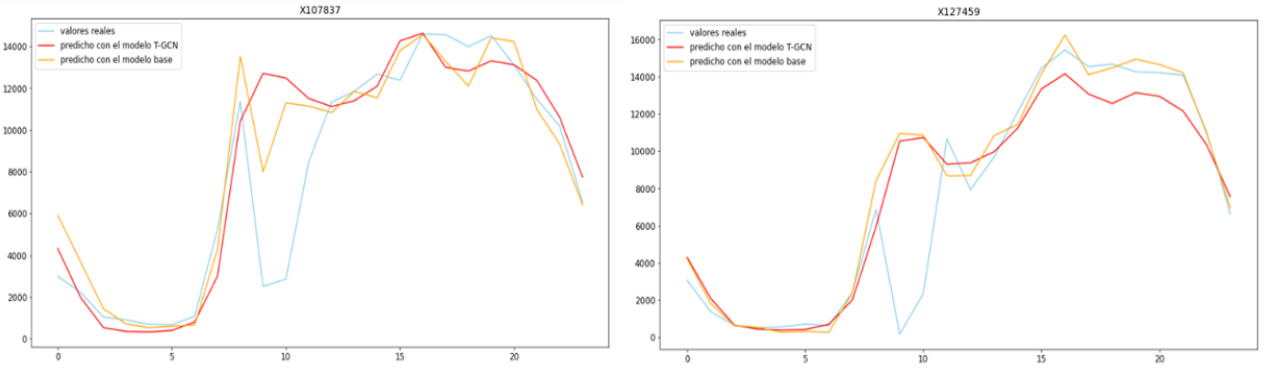
Figura 5.7: Retenciones en puente de ventas



Fte: elaboración propia

Como se aprecia en la serie temporal de color azul, entre las 6 de la mañana y las 11 de la mañana, los valores reales de intensidad de tráfico son diferentes que el día anterior (color naranja, modelo base). El modelo ajustado T-GCN, al estar realizando predicciones a 24 horas, tampoco es capaz de ajustar la irregularidad. En algunas series temporales conectadas que en el grafo espacial con el sensor $X6120$ de ventas, se observa como existen patrones irregulares en horas cercanas al atasco (figura 5.8). En las predicciones del modelo T-GCN a 24 horas, también ocurre lo mismo que en la serie temporal anterior.

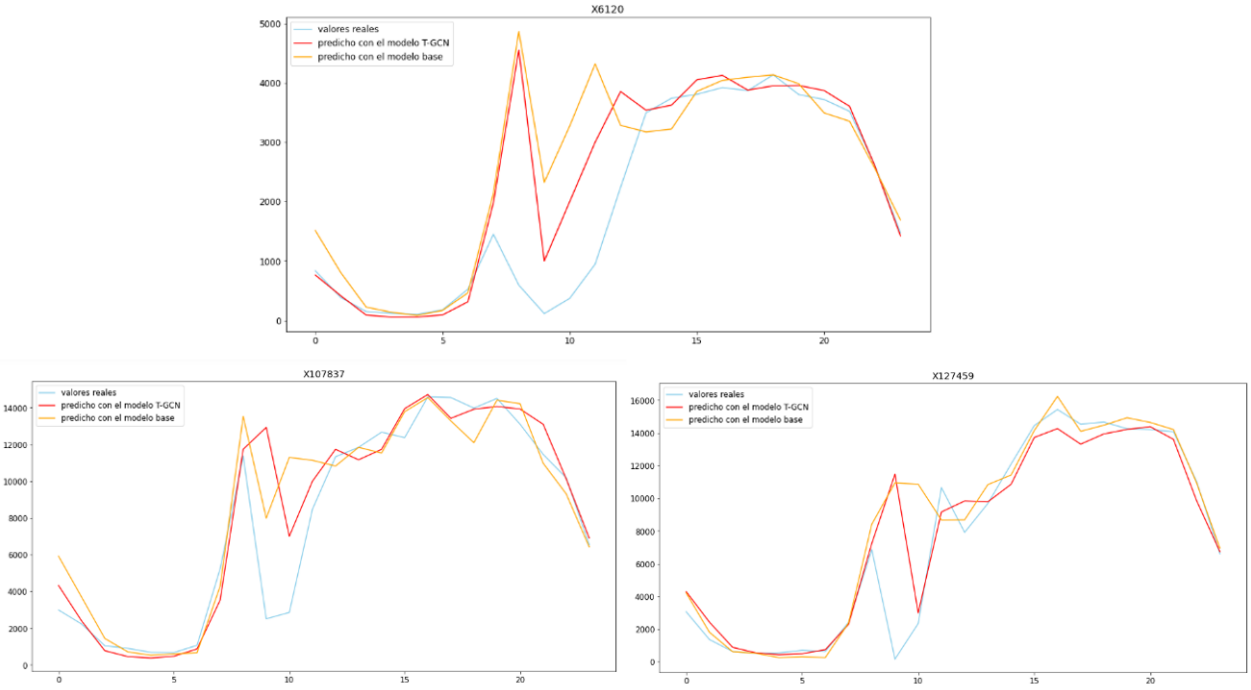
Figura 5.8: Retenciones en puente de ventas



Fte: elaboración propia

A continuación, se predice el modelo T-GCN actualizando para cada hora el valor real de la hora anterior y viendo qué predice el modelo en la hora siguiente para las series de las figuas 5.7 y 5.8. De esta forma, se evalúa la capacidad del algoritmo de predecir un atasco y la propagación espacialmente de él. Los resultados en las 3 series temporales son (figura 5.9):

Figura 5.9: Retenciones en puente de ventas



Fte: elaboración propia

Como se observa en la figura 5.9, el modelo obtiene mejores resultados actualizando los valores de manera horaria y es capaz de propagar el atasco en los dos sensores conectados, aunque no lo suficiente. Esto puede ser debido al peso que se ha introducido en las conexiones del grafo, el estudio de la razón y su modificación queda expuesto para su realización en futuros proyectos.

Capítulo 6

Conclusiones y futuros estudios

Esta investigación estudia la aplicación de grafos para modelar las calles de una ciudad conservando su topología y desarrolla un enfoque novedoso para la predicción de datos espacio-temporales basado en redes neuronales (T-GCN), que combina capas GCN y capas LSTM. Los datos utilizados corresponden a sensores geolocalizados en la Comunidad de Madrid con datos abiertos de intensidad de tráfico.

El grafo realizado contiene en los nodos la información de los sensores situados en las calles y las aristas representan calles directamente conectadas. La información del camino mínimo entre un sensor y otro se utiliza como peso de las aristas. El modelo T-GCN se utiliza para predecir la intensidad de tráfico en cada sensor utilizando las relaciones topológicas de cada sensor. Las capas GCN se utilizan para capturar la estructura topológica del grafo y cubrir la dependencia espacial, mientras que se introducen las capas LSTM para modelar la dependencia temporal.

Respecto a los objetivos iniciales del proyecto enunciados en el capítulo 1, se han cubierto los siguientes aspectos:

1. Familiarización con datos geoespaciales y búsqueda de representaciones en el lenguaje de programación de python. Se han probado diferentes librerías para datos geoespaciales en python como *ArcPi, GeoPandas* o *OsMnx*. Finalmente, se ha utilizado la librería *OsMnx* [36] para la creación de un grafo inicial que se ajuste las calles de Madrid y geoposicionar los sensores sobre el grafo.

2. Investigación de estructuras de grafos que conserven las características topológicas de las calles con sus longitudes y direcciones. Se ha realizado un enfoque novedoso transformando el grafo inicial a su dual para tener las características de los sensores en los nodos y poder aplicar una red neuronal para predecir la información de los nodos.

Finalmente, las aristas del grafo conservan direcciones y longitudes de las calles. Este objetivo ha sido el que más tiempo a abarcado del proyecto, probando diferentes enfoques.

3. Evaluación de la complejidad computacional de los algoritmos, analizando cual es el más apropiado en cada aspecto del proyecto. En la creación del grafo con propiedades espaciales, se ha necesitado calcular el camino minimo de un sensor a otro. Para ello, se han probado diferentes enfoques evaluando su complejidad computacional. Por último, en la selección de hiperparámeteros del modelo, se ha tenido en cuenta la complejidad computacional además de las métricas de evaluación del modelo.

4. Investigación de modelos de IA en estado del arte para la predicción de datos espacio-temporales. Se han investigado los algoritmos empleados para la resolución del problema de la predicción de tráfico evaluando sus ventajas e inconvenientes. En concreto, se ha analizado la literatura de modelos tradicionales, enfoques de aprendizaje profundo con estructuras de capas más tradicionales y con capas aplicadas a grafos (GCN).

5. Estudio matemático de los modelos realizados. Se ha analizado la estructura del modelo ajustado T-GCN descomponiéndolo en sus dos capas: GCN y LSTM.

6. Aplicación de modelos de IA basados en grafos mediante librerías en python. Se ha aplicado un enfoque que se encuentra en estado del arte utilizando capas de grafos convolucionales (GCN) y capas recurrentes (LSTM), ajustando el modelo T-GCN. Para aplicarlo, los datos procesados se han dividido en entrenamiento, validación y test, se han normalizando y se han aplicado los redimensionamientos necesarios para conseguir la estructura de entrada correcta a la red neuronal.

7. Evaluación del modelo predictivo. Se ha evaluado el modelo de manera general y en regiones de Madrid en particular. Al comparar con el modelo base (repetir los valores de periodos anteriores) se han obtenido resultados satisfactorios para el caso de predicciones a corto plazo, donde el modelo T-GCN obtiene mejores resultados. Para la predicción de situaciones irregulares como atascos, se ha concluído que el modelo debe proporcionar predicciones a corto plazo.

Como continuación de esta investigación, se podría introducir al modelo predictivo características de datos meteorológicos, datos de accidentes de tráfico geolocalizados o días festivos que complementen a las variables explicativas del modelo. Además, se pueden aplicar otros modelos para comparar la idoneidad de los resultados, incluyendo un histórico de datos más extenso en el entrenamiento del modelo o granularidades diferentes.

Respecto a las aplicaciones de las predicciones obtenidas en el modelo, se podría trabajar en la planificación de rutas o utilizar sensores de contaminación, disponibles como los datos abiertos de la Comunidad de Madrid de sensores de contaminación. Actual-

mente, aplicaciones como *Google Maps* planifican la ruta optima en base a los atascos o congestiones de tráfico que están ocurriendo en el instante. Sin embargo, no se planifican las rutas óptimas en base a lo que va a ocurrir. Por ejemplo, si en la carretera M-30 de Madrid se va a 10 por hora es evidente que hay un atasco, pero a qué otras carreteras habrá afectado el atasco en 1 hora puede aportar valor para predecir la ruta óptima con tiempo de antelación. En este aspecto, las predicciones del modelo pueden aportar valor a usuarios que desempeñan funciones de logística para planificar trayectos o ahorrar energía o conductores de taxi. Por otro lado, congestión de tráfico puede ser una de las causas del aumento de la contaminación en Madrid y, por tanto, las predicciones de la intensidad de tráfico pueden ayudar a programar las limitaciones de tráfico o contribuir a una mayor precisión de las predicciones relacionadas con la contaminación.

Por último, para concluir que el algoritmo empleado en este proyecto es generalizable para cualquier tipo de datos tabulares con relación espacio-temporal como valores de consumo de electricidad o precios de activos financieros.

Bibliografía

[1] Zhang, Y., Jin, Z., & Sikand, M. (2021). The top-of-atmosphere, surface and atmospheric cloud radiative kernels based on isccp-h datasets: Method and evaluation. *J. Geophys. Res. Atmos.*, *126*(24), e2021JD035053. https://doi.org/10.1029/2021JD035053

[2] Firouraghi, N., Mohammadi, A., Hamer, D., Bergquist, R., Mostafavi, S., Shamsoddini, A., Raouf-Rahmati, A., Fakhar, M., Moghaddas, E., & Kiani, B. (2021). Spatio-temporal visualisation of cutaneous leishmaniasis in an endemic, urban area in iran. *Acta Tropica*, *225*, 106181. https://doi.org/10.1016/j.actatropica.2021.106181

[3] de Medrano, R., & Aznarte, J. (2021). A new spatio-temporal neural network approach for traffic accident forecasting. *Applied Artificial Intelligence*, *35*, 1–20. https://doi.org/10.1080/08839514.2021.1935588

[4] de Medrano, R., & Aznarte, J. (2020). A spatio-temporal spot-forecasting framework forurban traffic prediction.

[5] Vazquez Giménez, J. J. (2019). Traffic forecasting in smart cities. *Universidad Politécnica de Cataluña*.

[6] Rico, J., Barateiro, J., & Oliveira, A. (2021). Graph neural networks for traffic forecasting.

[7] Diehl, F., Brunner, T., Le, M. T., & Knoll, A. (2019). Graph neural networks for modelling traffic participant interaction. *2019 IEEE Intelligent Vehicles Symposium (IV)*, 695–701. https://doi.org/10.1109/IVS.2019.8814066

[8] Zheng, C., Fan, X., Wang, C., & Qi, J. (2019). Gman: A graph multi-attention network for traffic prediction.

[9] Url de datos de sensores de madrid [Última visualización: 2022-07-16]. (2016). https://datos.madrid.es/portal/site/egob/

[10] Jiang, W., & Luo, J. (2021). Graph neural network for traffic forecasting: A survey.

[11] Haitao, Y., & Guoliangand, L. (2021). A survey of traffic prediction: From spatio-temporal data to intelligent transportation. *Data Science and Engineering*, *6*, 63–85. https://doi.org/https://doi.org/10.1007/s41019-020-00151-z

[12] Alghamdi, T., Elgazzar, K., Bayoumi, M., Sharaf, T., & Shah, S. (2019). Forecasting traffic congestion using arima modeling, 1227–1232. https://doi.org/10.1109/IWCMC.2019.8766698

[13] Duan, P., Mao, G., Yue, W., & Wang, S. (2018). A unified starima based model for short-term traffic flow prediction, 1652–1657. https://doi.org/10.1109/ITSC.2018.8569964

[14] Li, J., Wang, S., Zhang, H., & Zhou, A. (2020). A multi-objective evolutionary algorithm based on knn-graph for traffic network attack. *Electronics*, *9*, 1589. https://doi.org/10.3390/electronics9101589

[15] Rasaizadi, A., Sherafat, E., & Seyedabrishami, S. (2021). Short-term prediction of traffic state, statistical approach versus machine learning approach. *Scientia Iranica*, -. https://doi.org/10.24200/sci.2021.57906.5469

[16] Dong, X., Lei, T., Jin, S., & Hou, Z. (2018). Short-term traffic flow prediction based on xgboost. *2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS)*, 854–859.

[17] Xia, H., Wei, X., Gao, Y., & Lv, H. (2019). Traffic prediction based on ensemble machine learning strategies with bagging and lightgbm. *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, 1–6. https://doi.org/10.1109/ICCW.2019.8757058

[18] Menon, R., M, S., Rajashree, & Thammaiah, R. (2020). Traffic congestion prediction system using k-nearest neighbour algorithm. *International Research Journal of Engineering and Technology (IRJET)*, *7*.

[19] Sharma, S., Meenakshi, V., Apurva, Chakrasali, S., & Satish, S. V. (2013). Real-time traffic congestion detection using combined svm.

[20] Ranjan, N., Bhandari, S., Zhao, H. P., Kim, H., & Khan, P. (2020). City-wide traffic congestion prediction based on cnn, lstm and transpose cnn. *IEEE Access*, *8*, 81606–81620.

[21] Ma, Z., Yu, H., Xia, J., Wang, C., Yan, L., & Zhou, X. (2021). Network traffic prediction based on seq2seq model. *2021 16th International Conference on Computer Science Education (ICCSE)*, 710–713. https://doi.org/10.1109/ICCSE51940.2021.9569477

[22] Velez, D., Alvaro-Meca, A., Sebastián-Huerta, F., & Vélez-Serrano, J. (2021). Spatio-temporal traffic flow prediction in madrid: An application of residual convolutional neural networks. *Mathematics*, *9*, 1068. https://doi.org/10.3390/math9091068

[23] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, *1*, 57–81. https://doi.org/https://doi.org/10.1016/j.aiopen.2021.01.001

[24] Karagiannakos, S. (2021). Best graph neural networks architectures: Gcn, gat, mpnn and more. *https://theaisummer.com/*. https://theaisummer.com/gnn-architectures/.

[25] Hu, N., Zhang, D., Xie, K., Liang, W., & Hsieh, M.-Y. (2021). Graph learning-based spatial-temporal graph convolutional neural networks for traffic forecasting. *Connection Science*, *0*(0), 1–20. https://doi.org/10.1080/09540091.2021.2006607

[26] Guo, S., Lin, Y., Feng, N., Song, C., & Wan, H. (2019). Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. *AAAI*.

[27] Stellargraph [Última visualización: 2022-07-16]. (2018). https://stellargraph.readthedocs.io/en/stable/

[28] Librería de python tensorflow [Última visualización: 2022-07-16]. (2016). https://www.tensorflow.org/?hl=es-419

[29] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

[30] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

[31] Introducción a las redes neuronales con grafos [Última visualización: 2022-07-16]. (n.d.). https://ichi.pro/es/una-introduccion-a-graph-neural-network-gnn

[32] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

[33] Capas de un grafo [Última visualización: 2022-07-16]. (2016). https://tkipf.github.io/graph-convolutional-networks/

[34] Apaydin, H., Feizi, H., Sattari, M. T., Colak, M. S., Shamshirband, S., & Chau, K.-W. (2020). Comparative analysis of recurrent neural network architectures for reservoir inflow forecasting. *Water*, *12*(5). https://www.mdpi.com/2073-4441/12/5/1500

[35] Arias Botey, A. (2022). Repositorio de github con el código de la parte práctica [Última visualización: 2022-08-18]. https://github.com/AnaBotey/T-GNN

[36] Osmnx [Última visualización: 2022-07-16]. (2019). https://osmnx.readthedocs.io/en/stable/

[37] Ananta, M., Jiang, J.-R., & Muslim, M. (2014). Multicasting with the extended dijkstra's shortest path algorithm forsoftware defined networking. *9*, 21017–21030.

[38] Wang, Y., Yao, H., Zhao, S., & Zheng, Y. (2015). Dimensionality reduction strategy based on auto-encoder, 1–4. https://doi.org/10.1145/2808492.2808555

[39] Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., & Li, H. (2020). T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, *21*(9), 3848–3858. https://doi.org/10.1109/tits.2019.2935152