MÁSTER UNIVERSITARIO EN INGENIERÍA INDUSTRIAL

TRABAJO FIN DE MÁSTER

Estimación de volúmenes de tráfico en vías no monitorizadas

Autor: Jaime Navajas Diez

Director: Luis Francisco Sánchez Merchante

Madrid

Julio de 2025

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Estimación de volúmenes de tráfico en vías no monitorizadas en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2024/2025 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

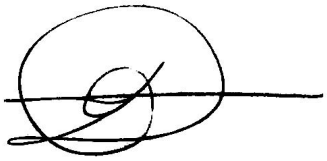
El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Jaime Navajas Diez Fecha: 28/07/2025

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO



Fdo.: Luis Francisco Sánchez Merchante Fecha: 28/07/2025

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

MÁSTER UNIVERSITARIO EN INGENIERÍA EN INDUSTRIALES

TRABAJO FIN DE MÁSTER

Estimación de volúmenes de tráfico en vías no monitorizadas

Autor: Jaime Navajas Diez

Director: Luis Francisco Sánchez Merchante

Madrid

Julio de 2025

Agradecimientos

Agradecimiento a mis padres y a mi director de proyecto. Sin su constante apoyo, orientación y colaboración, este trabajo no habría sido posible.

Por último, un especial agradecimiento a mi abuelo, quien me transmitió desde pequeño el interés y la pasión por las matemáticas, la física y la ingeniería.

Gracias

ESTIMACIÓN DE VOLÚMENES DE TRÁFICO EN VÍAS NO MONITORIZADAS

Autor: Navajas Diez, Jaime
Director: Sánchez Merchante, Luis Francisco
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Este proyecto consiste en el desarrollo de un sistema capaz de estimar el volumen de tráfico en vías urbanas que no disponen de sensores de monitoreo, empleando técnicas de Inteligencia Artificial aplicadas sobre grafos. Para ello se han aplicado y comparado diferentes modelos de aprendizaje, incluyendo arquitecturas avanzadas de redes neuronales gráficas como GraphSAGE y GIN, así como un método de postprocesado denominado Correct & Smooth (C&S), utilizando como caso de estudio la red viaria real de la ciudad de Barcelona. El enfoque más eficaz resultó ser la combinación de un modelo de red neuronal simple (MLP) con el postprocesado C&S, con el que se logró reducir el error absoluto medio de estimación hasta en un 60 % frente a métodos de referencia. Estos resultados demuestran la viabilidad de predecir el tráfico en calles no monitorizadas mediante modelos de IA, lo que permitiría optimizar la gestión de la movilidad urbana ahorrando costos en la instalación de sensores adicionales.

Palabras clave:

- Tráfico
- Grafo
- GNN (*Graph Neural Network*)
- Predicción
- Redes neuronales
- Baseline
- MAE (*Mean Absolute Error*)

1. Introducción

El tráfico urbano es un problema creciente que afecta tanto a la economía como al medioambiente de las ciudades. El tiempo perdido en atascos, el consumo excesivo de

combustible y el aumento de emisiones contaminantes suponen costes considerables para ciudadanos y administraciones. Aunque muchas ciudades instalan sensores para medir el flujo de vehículos, hacerlo de forma generalizada resulta inviable por motivos económicos y logísticos. En la práctica, la mayoría de las calles, especialmente las secundarias, no están monitorizadas.

Ante esta falta de información, surge la necesidad de desarrollar herramientas capaces de estimar el tráfico en vías no instrumentadas, aprovechando los datos disponibles en otras partes de la red. En este contexto, las técnicas de Inteligencia Artificial (Deep Learning) y el uso de grafos, estructuras que representan la red urbana conectando calles e intersecciones, ofrecen una solución prometedora. Dado que el tráfico en una calle está relacionado con el de sus calles vecinas, es posible emplear modelos que propaguen la información desde las zonas con sensores hacia las que no los tienen.

Este proyecto se enmarca precisamente en esa idea: aplicar modelos de aprendizaje sobre grafos para estimar el tráfico urbano en zonas sin datos, con el objetivo de ampliar virtualmente la cobertura de la red de sensores. A través de esta aproximación, se pretende contribuir a una movilidad más eficiente, sostenible y basada en datos, facilitando así una mejor planificación urbana sin necesidad de desplegar sensores en toda la ciudad, resultando en un opción más económica y escalable.

2. Definición del proyecto

Este proyecto tiene como objetivo desarrollar y validar un modelo capaz de estimar los volúmenes de tráfico en calles urbanas no monitorizadas, utilizando para ello tanto los datos disponibles de otras partes de la red como la estructura de conectividad urbana. El problema que se plantea es habitual en muchas ciudades: disponer de sensores solo en un subconjunto limitado de calles, lo que impide tener una visión completa del estado del tráfico.

La solución propuesta se basa en representar la red vial como un grafo, donde los nodos y aristas corresponden, respectivamente, a intersecciones y tramos de calle o viceversa. Sobre esta estructura se aplican modelos de aprendizaje automático (IA) que permiten inferir el volumen de tráfico en los tramos sin datos, aprovechando las correlaciones y la información de los tramos vecinos y el grafo en general.

El caso de estudio elegido es sobre un grafo de la ciudad de Barcelona, que ofrece un entorno urbano complejo y suficientemente grande. Se parte de un conjunto parcial de datos de tráfico reales, y el objetivo es estimar con la mayor precisión posible los valores desconocidos. El proyecto se centra en un escenario estático (una foto fija del tráfico en un momento determinado), con vistas a validar la viabilidad del enfoque como paso previo a futuras extensiones del proyecto.

3. Descripción de los experimentos

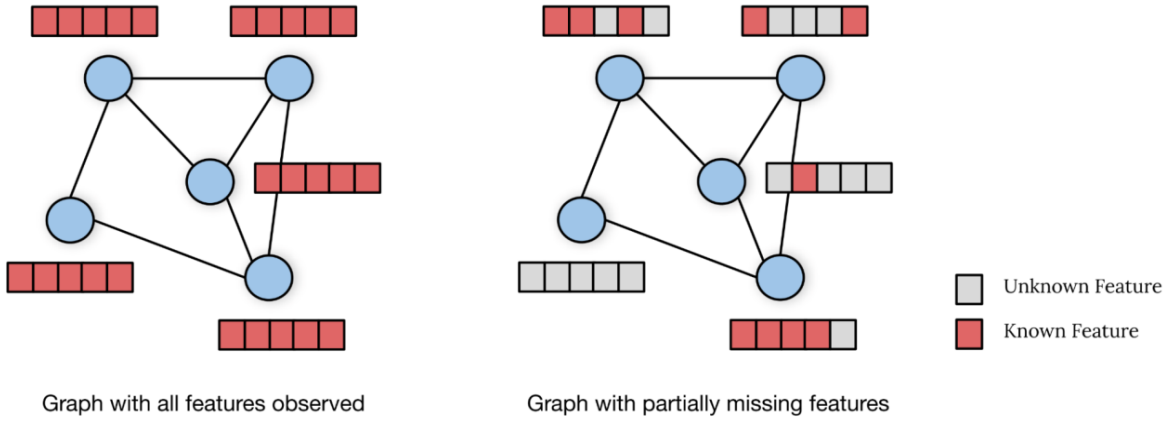


Ilustración 1: Ejemplo de la problemática (https://blog.x.com/engineering)

Para abordar el problema planteado, se diseñó una solución basada en técnicas de aprendizaje automático (Machine Learning o Deep Learning) aplicadas sobre el grafo urbano. El primer paso fue construir un grafo que representa la red vial de Barcelona, donde las conexiones entre calles permiten capturar relaciones espaciales clave para la estimación del tráfico.

Sobre este grafo se probaron tres enfoques principales:

- **Modelos de referencia (baselines):** Se emplearon métodos sencillos como la media global o local para estimar el tráfico en calles sin datos. Estos sirven como punto de comparación o benchmark para evaluar el valor añadido de modelos más avanzados.
- **Redes Neuronales Gráficas (GNN):** Se implementaron modelos como GraphSAGE, GIN, NoGE o DAGI, capaces de aprender relaciones considerando la topología y atributos sobre grafos. Estas redes permiten que cada calle ajuste su predicción combinando su información con la de las calles conectadas y el resto del grafo. [1] [2]

- **Correct & Smooth (C&S):** Se integró también esta técnica de postprocesado, que corrige y suaviza las predicciones iniciales propagando errores conocidos a través del grafo. Al aplicarla sobre modelos como MLP o GNN, se logró una mejora significativa en la precisión de las estimaciones. [11]

Durante el desarrollo se realizaron múltiples pruebas experimentales, en las que se variaron tanto los modelos empleados como el porcentaje de calles sin datos (ocultas), desde un 2 % hasta un 30 %. Para cada configuración, se repitieron los ensayos varias veces utilizando semillas aleatorias distintas, con el fin de simular diferentes escenarios posibles y asegurar que los resultados no dependieran de un único ensayo concreto del grafo, evitando así el factor suerte. Esta estrategia permitió evaluar la robustez de los modelos frente a la escasez de datos y obtener métricas más fiables.

4. Resultados

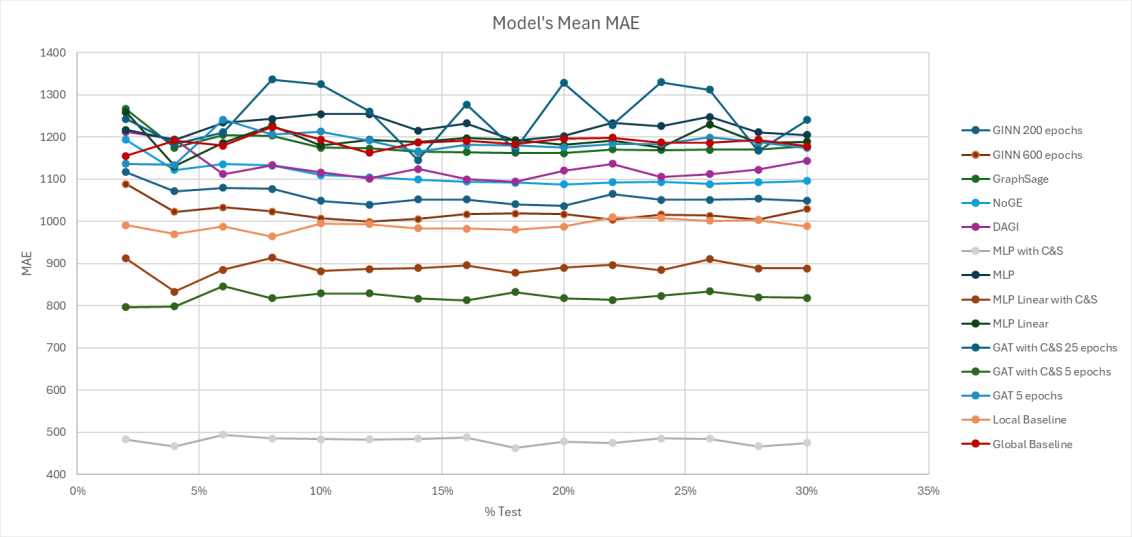


Ilustración 2: Gráfica resumen de resultados obtenidos

Los experimentos realizados demostraron que es posible estimar el tráfico en calles no monitorizadas con un grado razonable de precisión, incluso cuando solo se dispone de datos en una parte reducida de la red urbana. El enfoque que obtuvo mejores resultados fue la combinación de un modelo simple (MLP) con el postprocesado Correct & Smooth (C&S), logrando reducir el error medio (MAE) hasta en un 60 % respecto a los métodos de referencia, como la media global o local.

Por el contrario, modelos más complejos como GraphSAGE, GINN, NoGE o DAGI no ofrecieron mejoras significativas sobre los baselines simples. Aunque en teoría cuentan con una arquitectura más expresiva, su rendimiento se vio limitado por la escasez de atributos en los nodos, lo que dificultó el aprendizaje de patrones útiles. En la práctica, estos modelos no lograron superar de forma consistente al baseline local, lo que pone de relieve que, en este caso, la cantidad y calidad de los datos de entrada resulta más determinante que la complejidad del modelo.

Aplicar C&S como técnica de corrección posterior sí mostró beneficios claros, especialmente cuando se usaba sobre predicciones iniciales generadas por un MLP. Gracias a su sencillez y capacidad para aprovechar la estructura del grafo, esta estrategia híbrida fue la más eficaz para mantener estimaciones precisas incluso en escenarios con un porcentaje elevado de calles sin datos.

5. Conclusiones

Este trabajo ha permitido validar que es posible estimar los volúmenes de tráfico en calles urbanas no monitorizadas utilizando modelos basados en grafos y un conjunto parcial de datos. La aproximación propuesta, que combina una red neuronal simple (MLP) con un postprocesado Correct & Smooth (C&S), ha demostrado ser especialmente eficaz, superando ampliamente a métodos de referencia como la media global o local, incluso en escenarios con niveles altos de ocultación de datos. El error medio se redujo hasta en un 60 %, lo que evidencia que esta estrategia logra reconstruir la información faltante con un nivel de precisión adecuado para su uso operativo.

Este resultado no solo valida la viabilidad técnica del proyecto, sino que demuestra su potencial como herramienta práctica para el monitoreo de ciudades. Gracias a esta capacidad de extrapolar información a partir de datos limitados, se abre la posibilidad de mejorar la planificación y gestión del tráfico sin necesidad de realizar grandes inversiones en infraestructura.

A lo largo del trabajo se evaluaron también modelos más sofisticados, como GraphSAGE, GINN, NoGE y DAGI. Aunque teóricamente más expresivos, estos modelos no lograron mejorar sustancialmente las predicciones, en gran parte debido a la escasez de atributos disponibles. Esta limitación muestra que la calidad de los datos es un factor tan o más

importante que la arquitectura empleada: en ausencia de información, incluso los modelos más avanzados pueden resultar ineficaces, siendo mejor en ocasiones modelos más sencillos.

Este proyecto demuestra que los modelos basados en grafos tienen un enorme potencial como herramientas de simulación urbana. La posibilidad de predecir cómo se comportará el tráfico en zonas sin datos permite explorar escenarios hipotéticos, evaluar futuros impactos en el tráfico: obras, eventos o incluso estimar mapas de emisiones contaminantes.

En conjunto, este trabajo contribuye a sentar las bases de una nueva forma de abordar la monitorización del tráfico: no mediante una red extensa de sensores, sino con un enfoque inteligente que combina datos parciales y modelos estructurales para generar una imagen completa de la red urbana. Permitiendo así seguir avanzando hacia ciudades más eficientes, conectadas y sostenibles.

6. Motivación del proyecto

La motivación de este trabajo surge de una necesidad: muchas ciudades no pueden permitirse monitorizar todo su sistema de calles, pero aun así deben tomar decisiones basadas en datos. Esta limitación genera una brecha informativa que afecta tanto a la planificación como a la gestión diaria del tráfico.

La motivación es doble. Por un lado, económica: una herramienta de estimación fiable puede reducir costes al evitar instalaciones innecesarias y ayudar a priorizar dónde conviene invertir en sensores. Por otro, ambiental y social: conocer mejor los flujos de tráfico facilita la toma de decisiones que reducen la congestión, mejoran la calidad del aire y promueven una movilidad más eficiente y segura. También permite avanzar hacia una movilidad urbana más inteligente y sostenible. Además, el uso de modelos de predicción como herramienta de simulación de escenarios abre nuevas posibilidades en la gestión de eventos futuros, obras o emergencias urbanas.

Por otro lado, desde el punto de vista académico y tecnológico, el problema abordado representa un reto actual entre transporte e inteligencia artificial. El trabajo no solo aborda un caso práctico concreto (Barcelona), sino que sienta las bases para aplicar soluciones similares en otras ciudades y contextos (medicina, Internet, economía…), incluso integrando datos dinámicos y más factores en trabajos futuros.

ESTIMATION OF TRAFFIC VOLUMES IN UNMONITORED ROADS

Author: Navajas Diez, Jaime
Supervisor: Sánchez Merchante, Luis Francisco
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

This project involves the development of a system capable of estimating traffic volume on urban roads that lack monitoring sensors, using Artificial Intelligence techniques applied to graphs. To achieve this, different learning models have been applied and compared, including advanced graph neural network architectures such as GraphSAGE and GIN, as well as a post-processing method called Correct & Smooth (C&S), using the real road network of the city of Barcelona as a case study. The most effective approach turned out to be the combination of a simple neural network model (MLP) with the C&S post-processing method, which achieved a reduction of up to 60% in the mean absolute error compared to baseline methods. These results demonstrate the feasibility of predicting traffic on unmonitored streets using AI models, which could help optimize urban mobility management while reducing the cost of installing additional sensors.

Keywords:

- Traffic
- Graph
- GNN (Graph Neural Network)
- Prediction
- Neural networks
- Baseline
- MAE (Mean Absolute Error)

1. Introduction

Urban traffic is a growing problem that affects both the economy and the environment of cities. Time lost in traffic jams, excessive fuel consumption, and increased pollutant emissions represent significant costs for both citizens and public administrations. Although many cities install sensors to measure vehicle flow, doing so on a large scale is unfeasible

due to economic and logistical reasons. In practice, most streets, especially secondary ones, are not monitored.

Given this lack of information, there is a need to develop tools capable of estimating traffic on non-instrumented roads, leveraging the data available from other parts of the network. In this context, Artificial Intelligence (Deep Learning) techniques and the use of graphs structures that represent the urban network by connecting streets and intersections offer a promising solution. Since traffic on a given street is related to that of its neighboring streets, it is possible to use models that propagate information from sensor-equipped areas to those without sensors.

This project is precisely framed within that idea: applying learning models on graphs to estimate urban traffic in areas without data, with the aim of virtually extending the coverage of the sensor network. Through this approach, the project aims to contribute to more efficient, sustainable, and data-driven urban mobility, facilitating better urban planning without the need to deploy sensors throughout the entire city, resulting in a more economical and scalable option.

2. Project definition

The objective of this project is to develop and validate a model capable of estimating traffic volumes on unmonitored urban streets, using both the available data from other parts of the network and the structure of urban connectivity. The problem addressed is common in many cities: having sensors installed on only a limited subset of streets, which prevents obtaining a complete picture of the traffic conditions.

The proposed solution is based on representing the road network as a graph, where the nodes and edges correspond, respectively, to intersections and street segments or vice versa. Machine learning (AI) models are applied to this structure to infer traffic volumes on the segments without data, leveraging correlations and information from neighboring segments and the overall graph.

The chosen case study focuses on a graph of the city of Barcelona, which provides a complex and sufficiently large urban environment. The project starts with a partial set of real traffic data, and the goal is to estimate the unknown values as accurately as possible. The project is centered on a static scenario (a snapshot of traffic at a given moment), with the aim of

validating the feasibility of the approach as a preliminary step toward future project extensions.

3. Experiments description

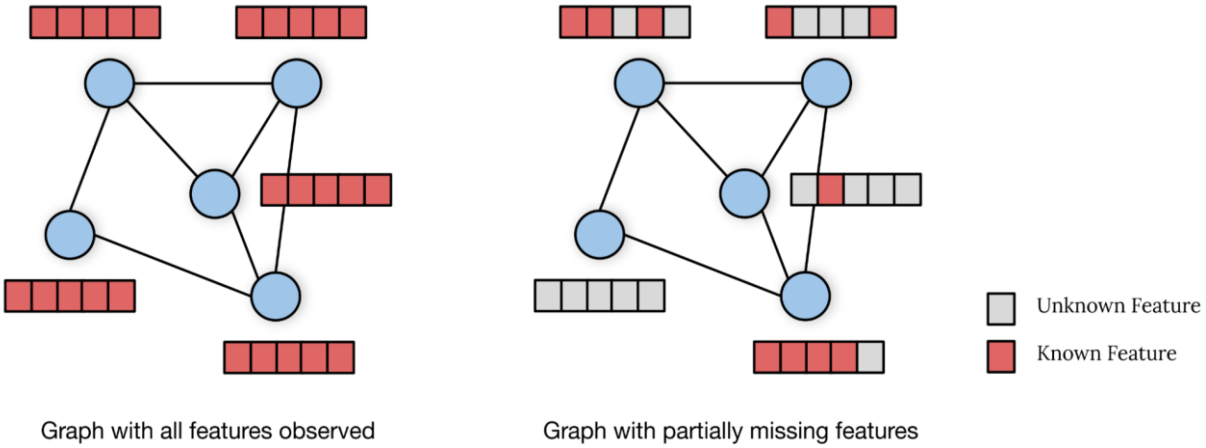


Illustration 1: Example of the problem (https://blog.x.com/engineering)

To address the proposed problem, a solution based on machine learning (ML or Deep Learning) techniques applied to the urban graph was designed. The first step was to construct a graph representing the road network of Barcelona, where the connections between streets capture key spatial relationships for traffic estimation.

Three main approaches were tested on this graph:

- **Baseline models:** Simple methods such as global or local means were used to estimate traffic on streets without data. These serve as a benchmark to assess the added value of more advanced models.
- **Graph Neural Networks (GNNs):** Models like GraphSAGE, GIN, NoGE, and DAGI were implemented. These are capable of learning relationships by considering both the topology and attributes on graphs. Such networks allow each street to adjust its prediction by combining its own information with that of its connected streets and the rest of the graph. [1] [2]
- **Correct & Smooth (C&S):** This post-processing technique was also integrated. It corrects and smooths initial predictions by propagating known errors throughout the graph. When applied on top of models like MLP or GNNs, it led to a significant improvement in estimation accuracy. [11]

Throughout development, multiple experimental trials were conducted, varying both the models used and the percentage of streets without data (hidden), ranging from 2% to 30%. For each configuration, the experiments were repeated several times using different random seeds in order to simulate various possible scenarios and ensure that the results did not depend on a single specific trial, thereby avoiding the influence of luck. This strategy allowed for assessing the robustness of the models under data scarcity and obtaining more reliable metrics.

4. Results

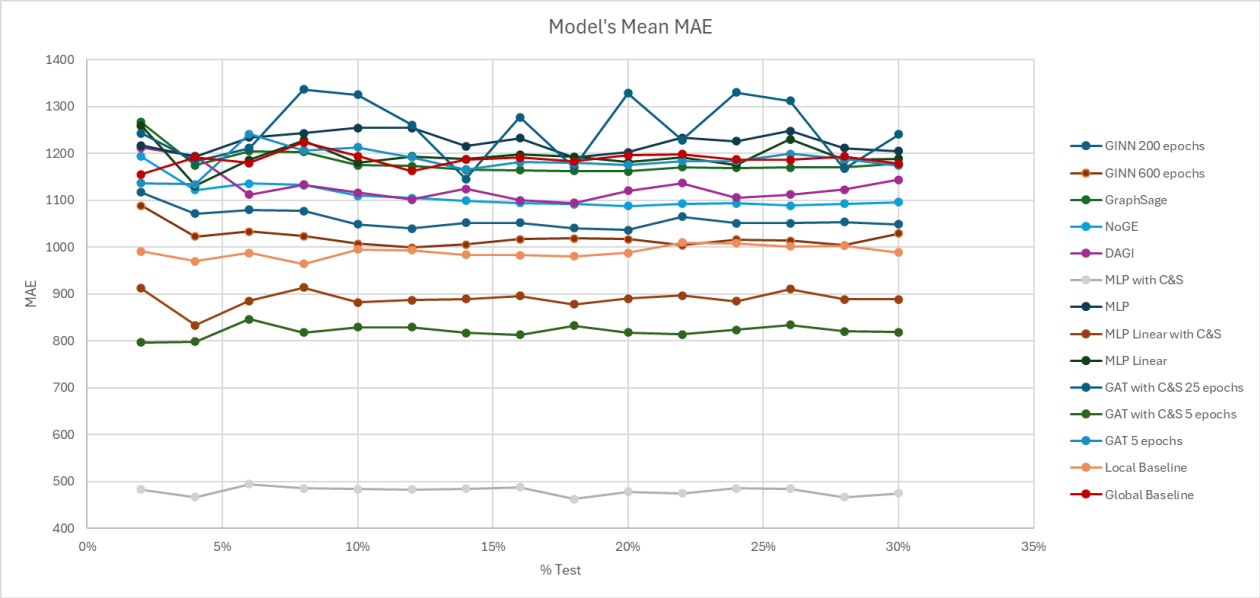


Illustration 2: Summary chart of the obtained results

The experiments carried out demonstrated that it is possible to estimate traffic on unmonitored streets with a reasonable degree of accuracy, even when data is available for only a small portion of the urban network. The approach that achieved the best results was the combination of a simple model (MLP) with the Correct & Smooth (C&S) post-processing method, managing to reduce the mean error (MAE) by up to 60% compared to baseline methods such as global or local averages.

In contrast, more complex models like GraphSAGE, GIN, NoGE, or DAGI did not provide significant improvements over the simple baselines. Although these models theoretically have a more expressive architecture, their performance was limited by the lack of features in the nodes, which hindered the learning of useful patterns. In practice, these models failed

to consistently outperform the local baseline, highlighting that, in this case, the quantity and quality of input data are more decisive than model complexity.

Applying C&S as a post-hoc correction technique did show clear benefits, especially when used on initial predictions generated by an MLP. Thanks to its simplicity and ability to leverage the graph structure, this hybrid strategy was the most effective in maintaining accurate estimates, even in scenarios with a high percentage of streets lacking data.

5. Conclusions

This work has validated that it is possible to estimate traffic volumes on unmonitored urban streets using graph-based models and a partial dataset. The proposed approach, which combines a simple neural network (MLP) with Correct & Smooth (C&S) post-processing, proved to be particularly effective, significantly outperforming reference methods such as global or local averages even in scenarios with high levels of data occlusion. The mean error was reduced by up to 60%, demonstrating that this strategy can reconstruct missing information with a level of accuracy suitable for operational use.

This result not only confirms the technical feasibility of the project but also highlights its potential as a practical tool for city monitoring. Thanks to its ability to extrapolate information from limited data, it opens new possibilities for improving traffic planning and management without requiring large investments in infrastructure.

Throughout the project, more sophisticated models such as GraphSAGE, GIN, NoGE, and DAGI were also evaluated. Although theoretically more expressive, these models did not substantially improve the predictions largely due to the limited availability of features. This limitation underscores the fact that data quality is as important as, or even more important than, the architecture employed: in the absence of sufficient information, even the most advanced models may prove ineffective, with simpler models sometimes delivering better results.

This project demonstrates that graph-based models hold great potential as urban simulation tools. The ability to predict traffic behavior in areas without data allows for the exploration of hypothetical scenarios and the assessment of future traffic impacts such as construction works, events, or even the estimation of pollution emission maps.

Overall, this work contributes to laying the groundwork for a new way of approaching traffic monitoring not through an extensive network of sensors, but through an intelligent approach that combines partial data and structural models to generate a complete picture of the urban network. This enables continued progress toward more efficient, connected, and sustainable cities.

6. Project Motivation

The motivation behind this work arises from a clear need: many cities cannot afford to monitor their entire street network, yet they must still make data-driven decisions. This limitation creates an information gap that affects both traffic planning and daily management.

The motivation is twofold. On the one hand, economic: a reliable estimation tool can reduce costs by avoiding unnecessary sensor installations and help prioritize where it makes the most sense to invest in monitoring infrastructure. On the other hand, environmental and social: better understanding of traffic flows supports decision-making that reduces congestion, improves air quality, and promotes safer and more efficient mobility. It also enables progress toward smarter and more sustainable urban mobility. Furthermore, the use of predictive models as tools for scenario simulation opens new possibilities for managing future events, construction works, or urban emergencies.

From an academic and technological perspective, the problem addressed represents a current challenge at the intersection of transportation and artificial intelligence. The work not only tackles a specific practical case (Barcelona) but also lays the groundwork for applying similar solutions in other cities and contexts (e.g., healthcare, the Internet, economics…), potentially integrating dynamic data and additional factors in future studies.

Indice de la memoria

Indice de figuras

Indice de tablas

Capítulo 1. INTRODUCCIÓN

1.1 CONTEXTO GENERAL

El crecimiento constante del tráfico urbano y el aumento de la densidad, así como la expansión de las grandes ciudades, han hecho que la gestión del tráfico se convierta en uno de los grandes desafíos de las ciudades en la actualidad. Cada vez resulta más necesario contar con información detallada, en tiempo real sobre cómo se distribuyen los flujos de vehículos (el tráfico), ya que esto influye directamente en la movilidad, la sostenibilidad y la calidad de vida de los ciudadanos... Disponer de estos datos permite mejorar la planificación del transporte público, reducir la congestión, acortar los tiempos de viaje y disminuir las emisiones contaminantes. [1]

Sin embargo, monitorizar de forma exhaustiva toda la red vial de una ciudad es costoso y poco escalable. La instalación de sensores físicos como cámaras, sensores inductivos, capacitivos o estaciones de conteo conlleva una inversión demasiado grande, tanto en infraestructura como en mantenimiento. Por ello, lo habitual es que solo se recoja información de tráfico en una parte limitada del total de las calles, normalmente en vías principales, lugares concurridos o puntos estratégicos en general. Esto deja sin datos una gran cantidad de calles secundarias o zonas menos transitadas, lo que dificulta tener una visión global y precisa del tráfico urbano.

Ante esta limitación, surge la necesidad de explorar soluciones que permitan estimar el tráfico en aquellas calles que no cuentan con dispositivos de medición. Una de las opciones, es la presentada en este proyecto. Esta consiste en modelar la ciudad como un grafo, es decir, una red donde los nodos representan intersecciones y las aristas las calles que las conectan o al revés. Esta representación permite capturar de forma natural la estructura topológica de la red urbana, lo que es clave para entender cómo se propaga el tráfico de una calle a otra.

En este contexto, los avances en técnicas de aprendizaje automático, y en particular las redes neuronales orientadas a grafos (Graph Neural Networks o GNN), ofrecen una vía muy interesante a explorar. Estas redes están diseñadas para trabajar directamente sobre estructuras de tipo grafo y han demostrado una gran capacidad para aprender patrones complejos de dependencia entre nodos y aristas. Aplicadas al caso del tráfico, permiten entrenar modelos que, a partir de información parcial (por ejemplo, cámaras instaladas en un porcentaje de las calles), sean capaces de predecir el volumen de vehículos en el resto de la red con un margen de error aceptable.

Más allá de la predicción puntual, este tipo de modelos también abre la puerta a nuevas aplicaciones. Por ejemplo, pueden utilizarse para simular diferentes escenarios y apoyar la toma de decisiones en situaciones como la organización de eventos masivos (maratones, carreras ciclistas…), obras o cortes temporales de calles. También sería posible generar mapas de emisiones contaminantes en tiempo real, estimando el impacto ambiental de cada zona según la intensidad del tráfico. Esta información sería especialmente útil en el contexto de políticas de movilidad sostenible, como las zonas de bajas emisiones o los sistemas de etiquetas ambientales, como es el caso de Madrid.

En definitiva, este proyecto se sitúa en un área de investigación que combina tecnologías emergentes como son las redes neuronales, con necesidades reales de la gestión urbana. Su objetivo es contribuir al desarrollo de herramientas inteligentes que, sin necesidad de cubrir toda la ciudad con sensores, permitan estimar el tráfico de forma precisa, eficiente y útil para la toma de decisiones tanto operativas como estratégicas para el futuro.

1.2 MOTIVACIÓN DEL TRABAJO

La motivación de este trabajo parte de una observación simple pero muy real: muchas ciudades no pueden permitirse el lujo de medir el tráfico en toda su red vial, pero aun así necesitan tomar decisiones basadas en datos. Esto genera una contradicción que obliga a buscar alternativas más inteligentes y sostenibles. Este proyecto nace precisamente con la intención de cubrir ese hueco, proponiendo una solución que combine herramientas modernas de análisis con una base sólida en la estructura urbana.

En ciudades grandes, aunque se cuenta con miles de dispositivos de medición, la cobertura no es constante ni completa. Muchos sensores solo funcionan durante ciertas franjas horarias, y muchos barrios carecen totalmente de datos fiables. Esto dificulta la planificación y complica la gestión diaria del tráfico, especialmente cuando ocurren imprevistos o se organizan eventos puntuales (Carreras ciclistas, maratones, partidos importantes de futbol, conciertos ...) que alteran la circulación habitual.

En este contexto, los modelos basados en grafos pueden representar una forma eficaz de reconstruir la información que falta. Gracias a su capacidad para modelar las relaciones entre calles conectadas, permiten hacer inferencias allí donde no hay datos suficientes. Las redes neuronales gráficas, en particular, pueden aportan una mejora significativa respecto a métodos más tradicionales, ya que aprenden directamente de la estructura de la red y son capaces de generalizar a nuevas situaciones, así como aprender patrones o estructuras ocultas.

Esta motivación técnica y económica se complementa además con un fuerte interés por la dimensión ambiental del problema. El tráfico no solo afecta a la movilidad, sino también a la calidad del aire, al ruido urbano y al consumo energético. Si conseguimos estimar el tráfico en toda la red con suficiente precisión, se podría también estimar las emisiones contaminantes asociadas, y así disponer de mapas dinámicos que ayuden a implementar medidas más eficaces, como restricciones temporales, rediseño de rutas o campañas de transporte público.

Por otro lado, estos modelos tienen un enorme potencial como herramientas de simulación. Imaginemos que una ciudad quiere cortar varias calles por obras durante varios días: poder simular cómo se redistribuiría el tráfico con distintos planes alternativos permitiría elegir la mejor opción antes de actuar. Incluso en escenarios más cotidianos, como ajustar los horarios de transporte público o planificar eventos deportivos, esta capacidad predictiva resulta muy valiosa.

Un caso práctico puede ilustrar mejor este planteamiento. Supongamos una intersección en la que confluyen varias calles, y disponemos de datos de tráfico de todas excepto una. Si observamos que todas esas calles tienen un flujo conjunto de entrada hacia la última

calle no monitorizada, es lógico suponer que esa calle restante tendrá un volumen de salida igual a la suma del resto, salvo pequeñas variaciones por parkings en esas calles u otras razones. Este tipo de deducciones, cuando se extienden a toda la red con ayuda de modelos avanzados y/o algoritmos, permiten cubrir las lagunas informativas y mejorar la eficiencia general del sistema, sería como "realizar un sudoku a gran escala".

En definitiva, la motivación de este trabajo es doble: por un lado, técnica y científica, en cuanto al desarrollo y validación de modelos predictivos sobre grafos; por otro, práctica, social y económica, en cuanto a su aplicabilidad real en entornos urbanos que buscan ser más inteligentes, eficientes y sostenibles. [2]

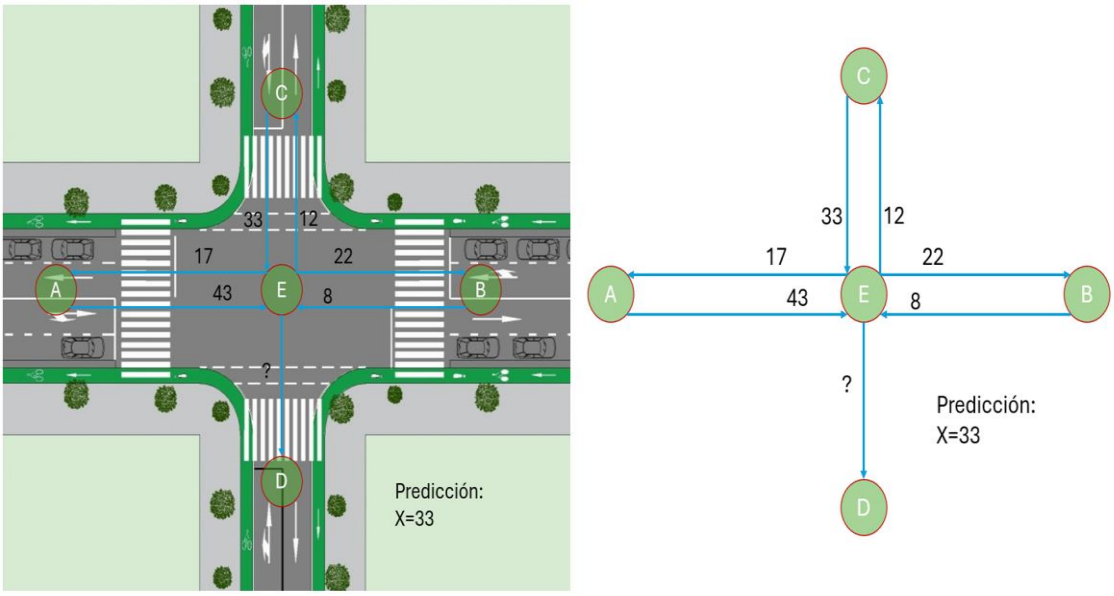


Figura 1: Predicción una intersección una incógnita

1.3 OBJETIVOS DEL PROYECTO

El objetivo principal del proyecto es desarrollar un sistema que permita estimar de forma lo suficientemente precisa el volumen de tráfico en aquellas vías no monitorizadas, utilizando para ello modelos basados en grafos urbanos y técnicas de aprendizaje automático, especialmente redes neuronales gráficas. O, dicho de otro modo, explorar diferentes modelos de GNN y evaluar que tan buenos son a la hora de predecir un caso real de tráfico. [1] [2]

La idea no es solo construir un modelo que funcione en condiciones ideales, sino evaluar su comportamiento en distintos escenarios, comparar su rendimiento con métodos simples (como baselines globales o locales) y estudiar hasta qué punto es posible generalizar sus resultados. A través de esta aproximación, se busca comprobar si realmente es viable reducir el número de sensores físicos necesarios sin comprometer en exceso la calidad de la información. [3]

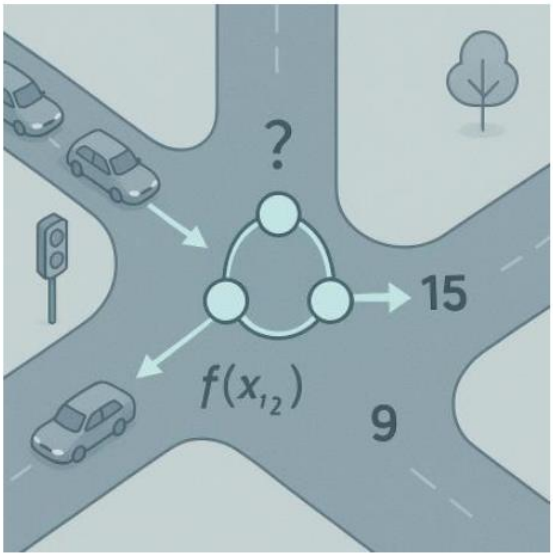


Figura 2: Ejemplo Intersección de calles y predicción de tráfico

1.3.1 OBJETIVO GENERAL

Desarrollar y evaluar un sistema computacional basado en grafos para estimar el tráfico en calles urbanas no monitorizadas, utilizando información parcial obtenida de sensores reales y aplicando algoritmos avanzados de predicción como GraphSAGE, GIN o Correct and Smooth.

1.3.2 OBJETIVOS ESPECÍFICOS

1. **Aplicar y comparar técnicas avanzadas de predicción**: Se implementarán varios algoritmos para grafos usando redes neuronales, como GIN, GraphSAGE y Correct and Smooth, y se compararán con métodos tradicionales sencillos (baseline global y local), para evaluar su precisión.
2. **Analizar la robustez frente a datos incompletos**: Uno de los puntos clave será estudiar cómo se comporta cada modelo a medida que aumenta el porcentaje de calles sin datos. Se buscará determinar un umbral razonable para mantener una estimación fiable.
3. **Optimizar recursos y reducir costes**: Se pretende demostrar que es posible obtener resultados aceptables sin necesidad de desplegar sensores por toda la red urbana, lo que implicaría un importante ahorro económico y una mejora en la escalabilidad de los sistemas de movilidad.
4. **Explorar aplicaciones prácticas**: Se valorarán diferentes trabajos futuros orientados a aplicaciones reales como mapas de emisiones, simulación de eventos o planes de movilidad urbana. Esto ayudará a valorar la utilidad del sistema más allá del entorno puramente académico.

En conjunto, estos objetivos buscan no solo validar la viabilidad técnica del modelo propuesto, sino también su aplicabilidad en el contexto de ciudades reales, como herramienta de apoyo a la toma de decisiones orientadas a una movilidad más eficiente, sostenible y basada en datos.

1.4 ALINEACIÓN CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE (ODS)

La sostenibilidad y el bienestar urbano son prioridades reconocidas internacionalmente a través de los Objetivos de Desarrollo Sostenible (ODS) de las Naciones Unidas. Este proyecto contribuye directamente a tres de estos objetivos.

1.4.1 ODS 3: SALUD Y BIENESTAR

El transporte urbano eficiente tiene un impacto directo en la salud y el bienestar de la población. Al mejorar la planificación del tráfico mediante técnicas de predicción avanzadas, este proyecto puede reducir los tiempos de viaje, disminuir el estrés asociado a la congestión y tráfico, y promover el uso del transporte público. Una mejor gestión del tráfico también disminuye la exposición de la población a emisiones contaminantes, lo que repercute en una mejora en la calidad del aire y, por ende, en la salud pública.

1.4.2 ODS 11: CIUDADES Y COMUNIDADES SOSTENIBLES

El proyecto aborda la necesidad de hacer las ciudades más inclusivas, seguras, resilientes y sostenibles. Al ofrecer herramientas para una mejor organización del tráfico y reducir la dependencia de dispositivos de medición, se fomenta una infraestructura urbana más eficiente y menos costosa. Esto facilita la planificación del transporte público y su integración con otras soluciones de movilidad sostenible, mejorando la accesibilidad y reduciendo la huella ambiental.

1.4.3 ODS 13: ACCIÓN POR EL CLIMA

El transporte urbano es una de las principales fuentes de emisiones de gases de efecto invernadero. Al optimizar el tráfico y promover el uso del transporte público, este proyecto contribuye a la reducción de emisiones de CO2 por el tráfico, apoyando la lucha contra el cambio climático. La capacidad de estimar volúmenes de tráfico en calles no monitorizadas también permite generar mapas de emisiones que pueden ser utilizados por las autoridades para implementar medidas de mitigación más efectivas.

En conjunto, este proyecto no solo busca resolver un problema técnico de predicción de tráfico, sino también promover un desarrollo urbano más sostenible, equitativo y respetuoso con el medio ambiente.

Capítulo 2. ESTADO DEL ARTE Y TECNOLOGÍAS

2.1 INTRODUCCIÓN A LA PREDICCIÓN DE TRÁFICO: EVOLUCIÓN Y ENFOQUES ACTUALES

La estimación del tráfico urbano es un área de investigación con una evolución marcada por las propias necesidades de las ciudades modernas. Inicialmente, el problema se abordaba mediante métodos estadísticos y técnicas de interpolación espacial, como el kriging, que permitían inferir datos en zonas con poca o nula monitorización (Wang & Kockelman, 2009). Aunque útiles en redes pequeñas u homogéneas, estos métodos demostraban ser insuficientes cuando se aplicaban a ciudades con estructuras complejas o con tráfico altamente variable. [4] [5]

A partir de los años 2000, el desarrollo del aprendizaje automático abrió nuevas posibilidades. Por ejemplo, Chang (1999) aplicó redes neuronales al análisis de tráfico en autopistas, logrando predecir con éxito patrones de corto plazo a partir de series temporales. Más adelante, Ishak, Kotha y Alecsandru (2003) demostraron que redes neuronales dinámicas superaban a los modelos estadísticos clásicos en horizontes de predicción más amplios, resaltando la importancia de incorporar el componente temporal al análisis. [20] [21]

Con la llegada del Big Data y el aumento del poder computacional, comenzaron a aplicarse redes neuronales recurrentes (RNN) para captar patrones temporales, y redes convolucionales (CNN) para extraer información espacial de la red vial. Estas técnicas demostraron buenos resultados, como se vio en trabajos como el de Choi y Kim (2018) con modelos RNN para predicción de trayectorias de vehículos, o en el uso de redes convolucionales de grafos con LSTM por Bogaerts et al. (2020), que lograron mantener buena precisión en predicciones multifase hasta con cuatro horas de horizonte temporal de 4 horas en el futuro. [3] [4] [22]

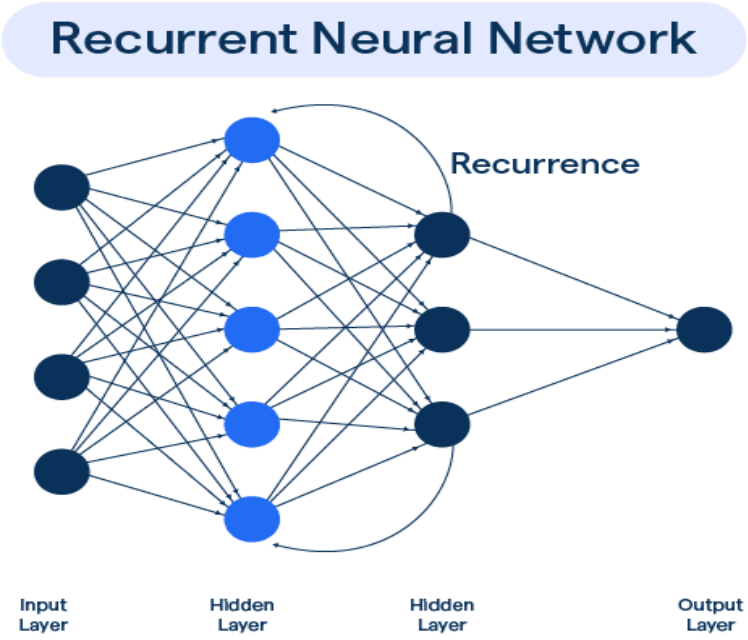


Figura 3: Red Neuronal Simple RNN

No obstante, tanto las RNN como CNN presentan una limitación clave: no tienen en cuenta la estructura topológica real de la red urbana. Las ciudades no son imágenes ni secuencias lineales, sino redes interconectadas con relaciones complejas entre sus componentes. Por ello, en los últimos años se ha estado dando un giro hacia el uso de grafos como estructura de datos más adecuada para representar ciudades y flujos de tráfico. [5] [6] [7]

En este contexto emergen las redes neuronales gráficas (GNN), diseñadas específicamente para trabajar sobre grafos. Modelos como GraphSAGE (Hamilton et al., 2017) permiten generalizar a nodos no vistos, aprendiendo funciones de agregación basadas en las características de los vecinos. También se han realizado muchos otros enfoques, como la propagación de etiquetas (Label Propagation) o el método MrAP (Propagación Multirelacional de Atributos), que se centran en completar los valores faltantes mediante el uso estructurado de la conectividad del grafo. [19]

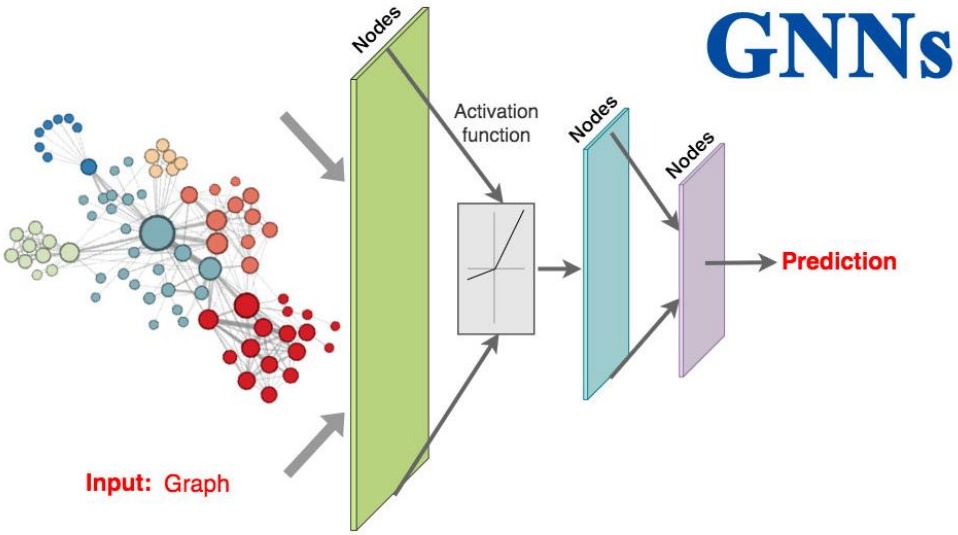


Figura 4: Red Neuronal Gráfica GNN

Recientemente, han aparecido técnicas como Correct and Smooth (C&S) (Huang et al., 2020), que plantea una combinación de predicción inicial seguida de una fase de corrección y suavizado mediante propagación. Este enfoque permite ajustar las predicciones teniendo en cuenta errores locales y pretende añadir mejoras en nodos con predicciones especialmente inexactas. [11]

Durante esta evolución y surgimiento de nuevos métodos de predicción, siempre se ha tratado de usar métodos de evaluación para validar los nuevos modelos frente a baselines simples, como la media global o la media local de vecinos. Estas comparaciones son esenciales para cuantificar si los modelos avanzados realmente aportan valor añadido o si su complejidad no aporta lo suficiente como para ser realmente rentable, y compensar el coste computacional asociado.

En definitiva, la predicción de tráfico ha pasado de basarse en extrapolaciones locales a utilizar modelos que capturan la estructura urbana en su conjunto y tratan de aprender patrones escondidos. El uso de grafos y aprendizaje profundo parece estar siendo cada vez más una solución más robusta y escalable para estimar tráfico en calles no monitorizadas, especialmente cuando se busca un equilibrio entre precisión, coste y aplicabilidad real.

2.2 MODELADO URBANO MEDIANTE GRAFOS: DEFINICIÓN Y APLICACIONES

Uno de los principales retos a la hora de modelar el tráfico urbano es representar de forma estructurada la red vial y sus interacciones. Las ciudades no están compuestas por puntos aislados, sino por calles, cruces, sentidos de circulación, prioridades y relaciones espaciales. Para capturar esta complejidad, una de las soluciones más naturales es el uso de grafos, como ya hemos comentado previamente.

¿Pero qué es exactamente un grafo?

Un grafo es una estructura matemática que representa entidades (llamadas nodos) y sus relaciones (llamadas aristas). En el caso del tráfico urbano, los nodos pueden representar intersecciones, y las aristas, las calles que las conectan o viceversa, además se puede representar la dirección de una arista (en el ejemplo de tráfico si una calle es de un solo sentido). Esta representación no solo es intuitiva, sino que además permite aplicar algoritmos que explotan la conectividad del sistema para hacer inferencias sobre partes no observadas.

Un grafo puede utilizarse para mucho más que representar calles y rutas. A menudo estos se usan para analizar una amplia variedad de sistemas complejos donde existen relaciones entre entidades. Por ejemplo, en redes sociales, los nodos pueden representar personas y las aristas sus relaciones de amistad o interacción. En biología, se pueden emplear para modelar redes de proteínas o interacciones genéticas. En informática, se utilizan en estructuras de datos, análisis de rutas en redes de comunicación, o en motores de recomendación, donde los productos y usuarios forman una red de preferencias. También son fundamentales en logística y transporte para optimizar rutas, en inteligencia artificial para representar conocimiento (mediante grafos semánticos), o en finanzas para detectar patrones en redes de transacciones. Gracias a su flexibilidad, los grafos se han convertido en una herramienta clave para abordar problemas de conectividad, influencia, propagación o agrupamiento en múltiples dominios.

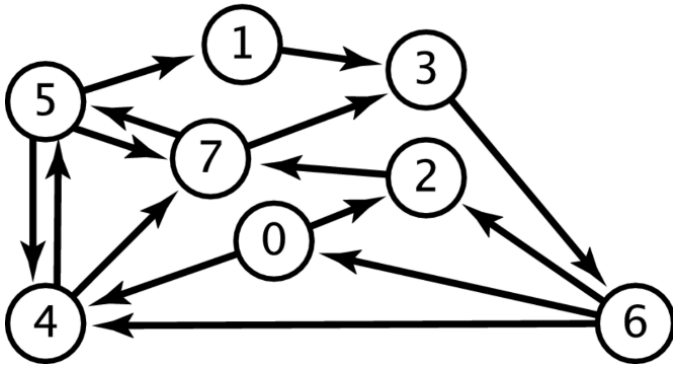


Figura 5: Ejemplo de un Grafo simple direccional

Para construir un grafo existen distintas formas, en nuestro caso donde representamos una red vial. Una de las formas más utilizadas consiste en considerar como nodos las intersecciones y como aristas las calles que unen dichas intersecciones. Sin embargo, también se pueden encontrar variantes más elaboradas, como los grafos extendidos o segmentados, donde cada segmento de calle se trata como un nodo y las intersecciones actúan como conectores entre ellos. Esta segunda opción, aunque más compleja, ofrece un mayor nivel de detalle y se puede adaptar mejor a ciertos algoritmos de predicción.

Otra de las ventajas de utilizar grafos es que permiten introducir información adicional en sus componentes. Por ejemplo, las aristas pueden tener atributos como longitud, tipo de vía, límite de velocidad o número de carriles; mientras que los nodos pueden estar etiquetados con datos sobre flujo de tráfico, densidad o tiempo medio de paso. Esta flexibilidad convierte al grafo en una estructura ideal para enriquecer el análisis sin perder la relación entre los elementos de la red.

Para este proyecto el modelado urbano mediante grafos también permite realizar tareas de predicción más avanzadas. En particular, el uso de redes neuronales gráficas (GNN) que permiten aprender representaciones internas de cada nodo o arista a partir de su vecindario. Esto significa que, incluso si no hay datos de tráfico directos en una calle concreta, es posible estimar su volumen en función de la información de las calles adyacentes.

Otra ventaja importante del modelado mediante grafos es su aplicabilidad a múltiples escalas. Se puede trabajar con grafos de ciudades completas, de distritos o de zonas

específicas. Además, se pueden simular escenarios: por ejemplo, ver qué ocurre si se cierra una calle o si se modifica el sentido de circulación en una zona determinada. Gracias a la estructura del grafo, los efectos de estos cambios se pueden propagar de forma coherente a través de la red y de manera relativamente sencilla.

En resumen, el uso de grafos para representar redes urbanas proporciona una base sólida y flexible para el análisis del tráfico. Esta estructura no solo refleja de forma fiel la conectividad del sistema, sino que permite incorporar múltiples capas de información y aplicar modelos de predicción avanzados, así como enfoques más tradicionales.

2.3 APRENDIZAJE SOBRE GRAFOS CON REDES NEURONALES

El aprendizaje sobre grafos (Graph Learning) es una rama del aprendizaje automático que trabaja con datos estructurados en forma de relaciones, como redes sociales, sistemas biológicos o redes urbanas de tráfico. A diferencia de otros modelos que usan tablas o imágenes, este enfoque permite capturar cómo se influencian entre sí los elementos conectados, algo muy útil en contextos como el tráfico, donde lo que pasa en una calle depende de las calles cercanas. Las redes neuronales gráficas (Graph Neural Networks, GNN) resultan especialmente efectivas, ya que permiten que cada nodo aprenda una representación propia o *embedding* combinando su información con la de sus vecinos.

Antes de seguir conviene preguntarse, ¿qué es un embedding?

Un embedding es una representación numérica (habitualmente un vector) que resume las características más relevantes de un objeto complejo, como un nodo en un grafo, de manera que pueda ser procesado por modelos de aprendizaje automático. En el caso de las GNN, el embedding de un nodo captura no solo su información interna (por ejemplo, atributos del nodo), sino también la influencia de sus vecinos en la estructura del grafo.

Es decir, es una forma compacta de codificar tanto el contenido como el contexto de cada nodo.

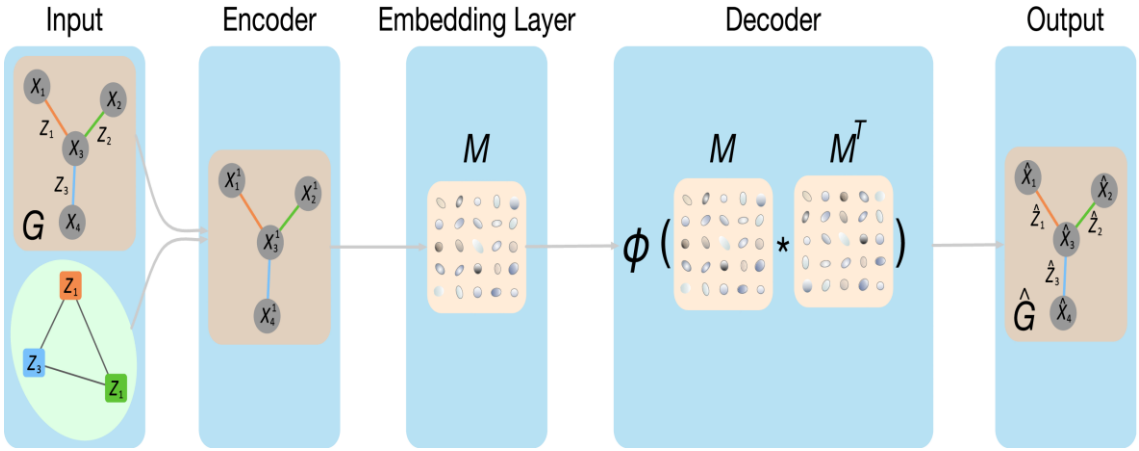


Figura 6: Esquema funcionamiento del embedding en grafo (https://www.computer.org/csdl/journal)

Por otro lado, también conviene definir una red neuronal:

Una red neuronal es un modelo de aprendizaje automático inspirado en el funcionamiento del cerebro humano. Está compuesta por capas de nodos o "neuronas", que procesan la información de forma jerárquica. Cada neurona recibe entradas, las transforma mediante funciones matemáticas y transmite el resultado a las siguientes capas. A través del entrenamiento con datos, la red ajusta sus conexiones internas (pesos) para reconocer patrones y hacer predicciones o outputs.

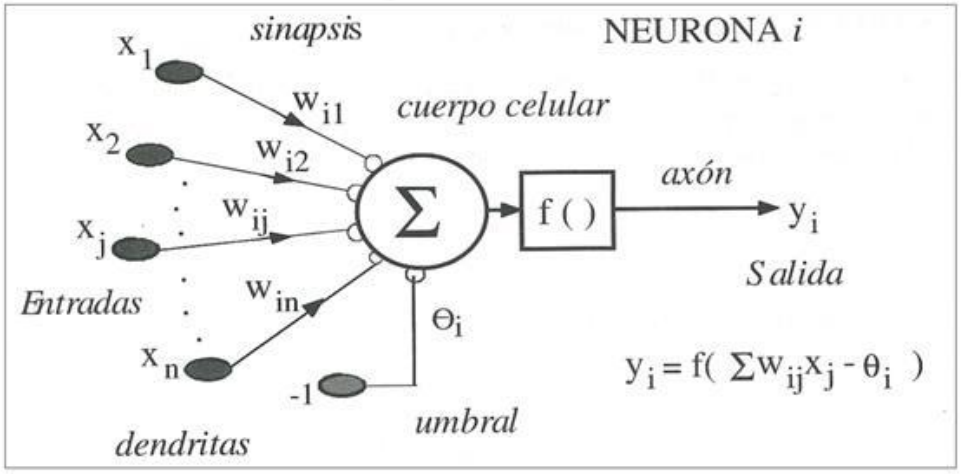


Figura 7: Ejemplo de una neurona simple

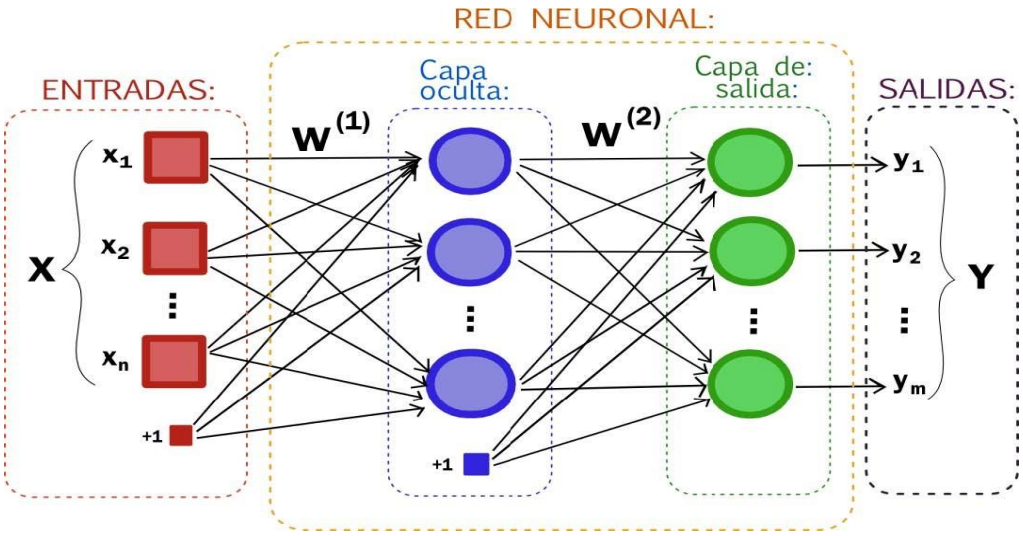


Figura 8: Esquema de una red neuronal básica

Cuando se construye la arquitectura de un GNN, en principio cuantas más capas tenga la red neuronal, mayor puede ser la profundidad del contexto capturado, permitiendo que un nodo incorpore información incluso de calles lejanas dentro del grafo. Pero por otro lado estará incrementando su complejidad, y quizás pierda capacidad de generalizar, pudiendo llevar a lo que se conoce como overfitting os sobreentranamiento. [12]

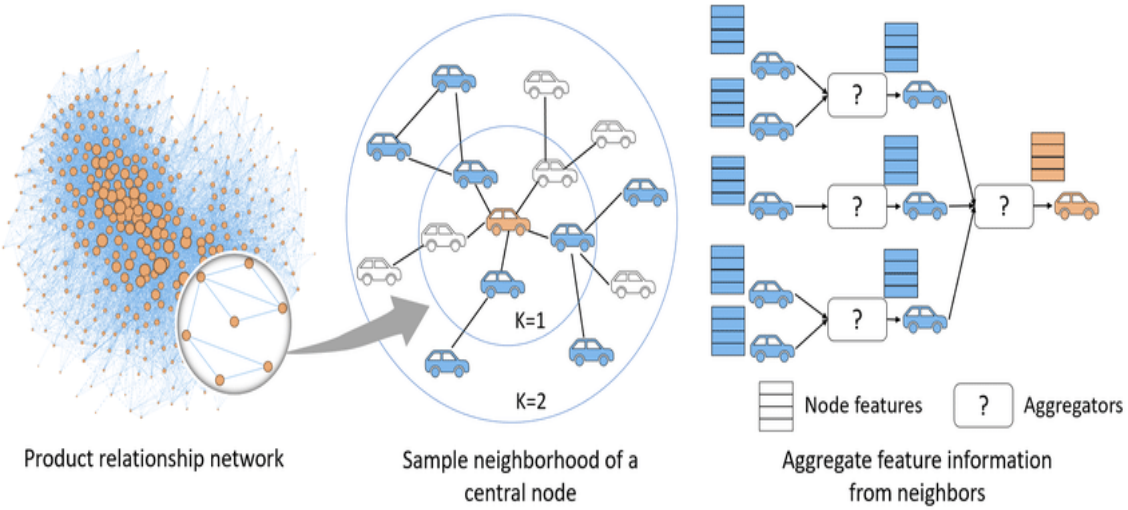


Figura 9: GraphSAGE ejemplo básico (https://www.researchgate.net/)

Uno de los aspectos clave en este tipo de modelos es cómo se representa la información. Cada nodo (por ejemplo, una calle o intersección) puede tener atributos como el volumen de tráfico, la longitud, el tipo de vía o su densidad media. Las aristas también pueden

incluir características como la dirección del flujo o la conectividad. Esta información se incorpora al modelo como entrada (input), y el objetivo es predecir valores continuos (como el volumen de tráfico, labels).

Dentro del campo de las GNN, existen múltiples arquitecturas. GraphSAGE es una de las más populares por su capacidad inductiva, lo que le permite generalizar a nodos no vistos durante el entrenamiento. Utiliza funciones de agregación como "mean", "LSTM" o "pooling" para combinar la información del vecindario y actualizar los embeddings de cada nodo de forma eficiente [19]. Sin embargo, una de las limitaciones de GraphSAGE es que su capacidad para distinguir estructuras similares puede ser limitada si no se eligen bien las funciones de agregación.

En respuesta a este problema, surge GIN (Graph Isomorphism Network), una arquitectura que, aunque también se basa en la agregación del vecindario, utiliza una suma directa de las características de los nodos vecinos sin normalización, lo que incrementa su capacidad expresiva. Esta propiedad hace que GIN sea más potente a la hora de diferenciar subgrafos estructuralmente distintos, una característica particularmente útil cuando se trabaja con grafos viales heterogéneos. [18]

Aparte de las GNN más conocidas, también existen técnicas más ligeras que no requieren un proceso de entrenamiento como tal. Es el caso de Label Propagation o Label Spreading, que se basan en la idea de que la información puede propagarse a lo largo del grafo siguiendo la conectividad estructural. Estas técnicas son útiles cuando se dispone de pocos datos etiquetados, ya que infieren los valores de los nodos desconocidos a partir de los conocidos, teniendo en cuenta su proximidad en el grafo. [9]

Un enfoque reciente que ha demostrado ser altamente eficaz en tareas de predicción en grafos parcialmente observados es Correct and Smooth (C&S). Este método no es una GNN en sí, sino una técnica de postprocesado que parte de una predicción base (por ejemplo, generada por una GNN o incluso una MLP, una red neuronal básica), y la corrige mediante dos fases: *corrección* (ajuste local de errores) y *suavizado* (propagación de la corrección por el grafo). Es una técnica simple y eficiente, que ha logrado resultados

competitivos en benchmarks estándar de aprendizaje sobre grafos, siendo además escalable y fácil de implementar. [11]

Otros trabajos recientes han propuesto arquitecturas más especializadas. Es el caso de DAGI (Deep Aggregation with Graph Isomorphism and Imputation), un modelo diseñado para la imputación de atributos ausentes en grafos, que no solo considera las características de los nodos, sino también relaciones estructurales entre ellos y factores externos relevantes. La arquitectura combina una red neuronal sobre grafos (GIN) con un módulo auxiliar de clasificación, integrando toda esta información en un proceso de aprendizaje conjunto, al que llaman DAGI (Demographic Aware Graph-based Imputation). Como resultado, se genera un grafo con atributos completos, coherente y enriquecido, adaptado a distintos contextos de entrada [13]. Por otro lado, Nguyen plantea un enfoque basado en cuaterniones duales para representar nodos y relaciones con mayor precisión geométrica, llamado NoGE (Node Co-occurrence-based Graph Neural Network) [17]. Su ventaja principal radica en capturar dependencias multidimensionales, lo que lo hace especialmente útil en grafos con gran riqueza de atributos y estructuras complejas.

En conclusión, el aprendizaje sobre grafos y las GNN proporciona un marco flexible y potente para abordar problemas como la estimación de tráfico urbano. Desde modelos simples basados en propagación de valores vecinos hasta arquitecturas profundas como GIN, pasando por híbridos como Correct and Smooth, las herramientas disponibles permiten adaptarse a distintos escenarios de datos, topologías y requisitos de precisión. Este trabajo tratará de explorar cuáles son las combinaciones más efectivas en el contexto de una red urbana parcialmente monitorizada, con un caso real como es la ciudad de Barcelona.

2.3.1 PRINCIPALES TÉCNICAS APLICADAS EN LA LITERATURA

A modo de resumen, se presentan las técnicas más relevantes que se han aplicado en la literatura tanto para proyectos similares como para este trabajo:

- **GraphSAGE (supervisado, inductivo o transductivo)**: Este modelo permite generar embeddings de nodos basándose en el vecindario local mediante funciones de agregación (media, LSTM, max pooling). Puede aplicarse en modo supervisado y en aprendizaje inductivo o transductivo, según se quiera generalizar a nuevas calles o no. Es una de las GNN más utilizadas por su flexibilidad y escalabilidad. [12]
- **GIN (Graph Isomorphism Network)**: Esta red se basa en una suma simple de las características de los nodos vecinos, seguida de una MLP. Está diseñada para captar con mayor precisión las diferencias estructurales entre subgrafos, siendo especialmente adecuada cuando la topología local influye fuertemente en la predicción. [18]
- **MRAP (Multi-Relational Attribute Propagation)**: Técnica centrada en la propagación de atributos a través de relaciones múltiples en el grafo. Puede incorporar aspectos como dirección, tipo de calle o peso, y es útil cuando se dispone de información adicional sobre las conexiones. [10]
- **Correct and Smooth (C&S)**: Método de postprocesamiento que parte de una predicción inicial y la ajusta en dos fases: primero corrige localmente los errores, y luego suaviza los resultados mediante propagación por el grafo. Es eficiente y eficaz, especialmente en grafos parcialmente etiquetados. [11]
- **Label Propagation y Label Spreading**: Métodos clásicos que propagan etiquetas o valores desde nodos conocidos a desconocidos siguiendo la estructura del grafo. No requieren entrenamiento y son útiles cuando se cuenta con poca información adicional. [9]
- **Algoritmo iterativo basado en la hipótesis de equilibrio**: Parte de la idea de que el tráfico que entra a una intersección debe ser igual al que sale. Mediante un

proceso iterativo, ajusta los volúmenes de tráfico hasta alcanzar un equilibrio. Existen variantes según cómo se calcule el ajuste: ICEBA, VEBA, con pesos o normalizaciones. [10]

- **Algoritmo de valores basado en equilibrio**: En lugar de iterar, este enfoque resuelve directamente un sistema de ecuaciones que refleja el equilibrio de flujos en intersecciones. Es computacionalmente más directo, aunque puede ser sensible a errores en las mediciones. [10]
- **Algoritmo de mínimos cuadrados (NNLSA)**: Propone estimar los flujos en calles no monitorizadas mediante una regresión por mínimos cuadrados no negativos. Busca minimizar el error cuadrático entre los valores estimados y los medidos. [10]
- **DAGI (Demographic Aware Graph-based Imputation):** Este modelo usa la imputación de atributos faltantes en grafos mediante una arquitectura basada en redes neuronales sobre grafos. Cada nodo representa una entidad con atributos observados (parciales), y los bordes definen relaciones estructurales entre nodos vecinos. Utiliza una Graph Isomorphism Network (GIN) para codificar la información estructural y de atributos compartidos entre nodos. El modelo predice los atributos faltantes de nodos completos, generando un grafo enriquecido y coherente con la estructura original. [13]
- **NoGE (Node Co-occurrence-based Graph Neural Network):** NoGE está diseñado para completar grafos de conocimiento mediante predicción de enlaces entre entidades y relaciones (tripletas). Construye un grafo en el que entidades y relaciones son nodos conectados según su coocurrencia dentro del conjunto de datos. Introduce una arquitectura llamada Dual Quaternion GNN (DualQGNN), que utiliza álgebra cuaterniónica para representar mejor las relaciones complejas. Finalmente, aplica una función de puntuación (QuatE) para evaluar la validez de nuevas tripletas, pudiendo alcanzar un alto rendimiento en tareas predicción. [17]

Capítulo 3. RECURSOS Y COSTES

3.1 HERRAMIENTAS UTILIZADAS

1. El desarrollo del proyecto se ha realizado íntegramente en Python, se ha escogido este lenguaje principalmente por su flexibilidad, legibilidad y extenso ecosistema científico. Se ha utilizado para implementar los algoritmos, procesar los datos, manipular estructuras de grafos y visualizar los resultados obtenidos.
2. Para el trabajo específico con grafos, se han empleado bibliotecas especializadas. NetworkX ha permitido la creación, modificación y análisis estructural de grafos base. PyTorch Geometric ha sido la herramienta central para la implementación de modelos de aprendizaje profundo sobre grafos, facilitando el uso de arquitecturas como Graph Isomorphism Networks (GIN) o Correct and Smooth (C&S). Adicionalmente, se han considerado otras librerías como StellarGraph, DGL y TensorFlow GNN para exploración y en algunos ensayos.
3. El entorno de trabajo se ha gestionado mediante Jupyter Notebooks, usando Visual Studio Code, lo que ha permitido mantener una documentación integrada con código y resultados, así como comentarios. También se han utilizado librerías auxiliares típicas de Python, como pandas, numpy, random, pickle y copy, así como herramientas de visualización como Matplotlib. Esta combinación de herramientas ha proporcionado un entorno robusto y reproducible para el desarrollo completo del proyecto.
4. Para todo el proyecto se ha utilizado únicamente un ordenador pórtatil, para ejecutar los programas, así como para entrenar y analizar los diferentes modelos.

3.2 COSTE ECONÓMICO DEL PROYECTO

El principal coste asociado al desarrollo del proyecto corresponde a la mano de obra técnica. Suponiendo que el perfil del ingeniero responsable del desarrollo tuviera un salario bruto anual de 26.000 €, y considerando un 30 % adicional en concepto de costes de empresa (seguros, cotizaciones, etc.), el coste total anual ascendería a 33.800 €. Al dividir esta cifra entre las 1.800 horas laborales anuales estándar, se obtiene un coste por hora de aproximadamente 18,78 €/h.

Dado que el Trabajo de Fin de Máster tiene una carga de 12 créditos ECTS (equivalente a 300 horas de trabajo), el coste salarial estimado sería de un total de 5.634€ en mano de obra.

$$26.000€ + 7800€ = 33.800€ /año$$

$$33.800(€ /año) /1800 (h/año) = 18,78 (€ /h)$$

$$18,78 (€ /h) \text{ x } 300 \text{ h} = 5.634€$$

A este importe se añaden los costes de hardware y software utilizados durante el desarrollo. El ordenador personal empleado tuvo un coste de 800 € hace 3 años, y se ha supuesto una amortización lineal en cinco años, lo que corresponde a 160 € por año. Además, se han utilizado herramientas de Windows con licencias de pago:

- **Windows 10**: 120 €
- **Microsoft Office Home**: 140 €

Por tanto, el coste total aproximado del proyecto en un entorno particular sería de en torno a 6.054€.

$$5.634€ + 160€ + 120€ + 140€ = 6.054€$$

No obstante, si se estima el coste desde una perspectiva puramente empresarial, con un coste estándar de 25 €/hora para un perfil técnico especializado, el coste de personal se elevaría a 7500€.

En cuanto al hardware, un equipo equivalente tendría un valor de mercado actual de 1.200 €, pero aplicando un descuento medio del 21 % (derivado de condiciones comerciales y deducciones fiscales), su coste real para la empresa sería de 948€ (con licencias ya incluidas). Amortizado en 5 años, el coste anual sería de 189.60 €.

En ese caso, el coste total del proyecto en un contexto empresarial medio sería de:

$$7.500\,€\ (\text{salario}) + 237\,€\ (\text{hardware}) = 7.737\,€$$

En resumen, el coste total estimado del desarrollo del sistema piloto oscila entre 6.054 € y 7.737 €, dependiendo del contexto de ejecución (personal o empresarial). Esta cifra refleja los recursos técnicos y humanos necesarios para realizar este proyecto de predicción de tráfico usando grafos y aprendizaje automático avanzado.

3.2.1 HARDWARE Y SOFTWARE EMPLEADOS

El desarrollo del proyecto se ha llevado a cabo utilizando un equipo informático de gama media-alta, suficientemente potente para ejecutar modelos de aprendizaje profundo sobre grafos de tamaño medio, como es el grafo de Barcelona utilizado. El ordenador portátil utilizado en cuestión presenta las siguientes especificaciones técnicas:

- **Procesador**: Intel Core i7 de 11ª generación
- **Memoria RAM**: 16 GB
- **Almacenamiento**: 1 TB de memoria SSD
- **Sistema operativo**: Windows 10
- **Tarjeta gráfica**: Intel Iris Xe integrada

En cuanto al software, se ha optado por herramientas de código abierto, lo que garantiza la reproducibilidad, escalabilidad y sostenibilidad del desarrollo. El stack principal ha estado compuesto como se mencionó anteriormente por:

- Python como lenguaje base.
- PyTorch Geometric para la implementación de arquitecturas GNN.
- NetworkX y StellarGraph para la manipulación y visualización de grafos.
- DGL, TensorFlow GNN y otras librerías auxiliares para pruebas comparativas. [16]
- Herramientas comunes como pandas, numpy, random, pickle y matplotlib para el procesamiento y análisis de datos.
- Jupyter Notebooks como entorno de desarrollo interactivo. [15]

La elección de estos recursos ha permitido una implementación ágil y flexible, sin comprometer el rendimiento ni la escalabilidad del proyecto.

Capítulo 4. METODOLOGÍA GENERAL

4.1 CONSTRUCCIÓN Y VARIANTES DEL GRAFO DE TRABAJO

El punto de partida del proyecto es la representación de la red urbana de tráfico de la ciudad de Barcelona mediante un grafo urbano dirigido y ponderado con atributos como volumen y longitud de las calles, diseñado específicamente para abordar tareas de predicción de tráfico sobre estructuras parcialmente monitorizadas. Para ello, se han utilizado los datos disponibles públicamente en el repositorio de *Transportation Networks* [14], que recopila ficheros de tráfico realistas de distintas ciudades del mundo.

En concreto, se ha empleado el fichero Barcelona_flow.tntp, el cual contiene cuatro columnas: nodo origen, nodo destino, volumen de tráfico y coste (en minutos) asociado a cada tramo o "distancia del tramo". A partir de esta información se ha construido un grafo dirigido, donde cada nodo representa una intersección o punto de la red, y cada arista corresponde a un tramo de vía con tráfico asociado. El fichero original incluye un total de 930 nodos y 2522 aristas.

Como el fichero no proporciona coordenadas geográficas, la estructura del grafo no refleja visualmente la disposición real de la ciudad, pero sí conserva la topología funcional de la red de transporte. Además, al detectarse discontinuidades en el etiquetado de los nodos (por ejemplo, pasar del nodo 50 al 100 sin que haya nodos entre esos números), se optó por renombrar todos los nodos secuencialmente del 0 al 929, garantizando la compatibilidad con las principales librerías de grafos utilizadas en el proyecto: NetworkX, DGL y StellarGraph.

A lo largo del desarrollo experimental se han empleado diferentes estructuras de grafos, tanto en lo que respecta a su forma conceptual (es decir, cómo se representan los elementos reales del entorno urbano) como en el formato computacional utilizado para procesarlos con distintas librerías de aprendizaje automático sobre grafos.

Por un lado, la forma del grafo hace referencia a la estructura lógica del mismo: si los nodos representan intersecciones o tramos, si existen nodos intermedios para modelar direcciones, etc. Se ha trabajado principalmente con tres formas: grafo original, grafo de segmentos y grafo extendido. Estas variaciones responden a la necesidad de adaptar el problema de regresión a la estructura que cada modelo puede procesar de forma eficiente.

Por otro lado, el formato del grafo corresponde al tipo de objeto Python y estructura de datos que se requiere según la librería empleada (PyTorch Geometric, DGL, StellarGraph o NetworkX, entre otras). Aunque todos ellos representan grafos, lo hacen con modelos internos distintos que condicionan tanto la carga como el entrenamiento de los modelos de predicción.

A partir de esta base se han construido tres variantes del grafo en forma, orientadas a diferentes enfoques experimentales y requisitos arquitectónicos:

- **Grafo original**: Representación directa del fichero base, donde los nodos corresponden a intersecciones y las aristas a los tramos entre ellas. Esta versión se ha utilizado principalmente para análisis estructural y como base para derivar otras formulaciones.
- **Grafo de segmento**: En esta variante, cada nodo representa un tramo de vía (es decir, una arista del grafo original). Dos nodos (segmentos) están conectados si comparten un nodo en común en el grafo original, lo cual permite modelar la sucesión de tramos en una ruta. Esta conversión se ha realizado utilizando la función line_graph() de NetworkX. El resultado es un grafo con 2522 nodos y 6679 aristas.
- **Grafo extendido**: Diseñado para modelos que requieren predecir valores sobre los nodos, pero trabajan originalmente con información de aristas. En este caso, se han introducido nodos intermedios (uno por dirección) en cada arista del grafo de segmento, los cuales se conectan con los nodos correspondientes del tramo. El objetivo es transformar una tarea de regresión sobre aristas en una tarea de regresión sobre nodos. Esta versión final del grafo cuenta con 3452 nodos y 5044 aristas.

A continuación, se detallan los formatos de grafo disponibles para el proyecto:

- **Grafo en formato DGL:** Utiliza la librería DGL (Deep Graph Library), que está especialmente diseñada para ejecutar modelos GNN de forma eficiente sobre GPUs. Un grafo en este formato se representa como un objeto dgl (DGLGraph), que almacena explícitamente los nodos, aristas y sus atributos. Este formato se emplea en los modelos de Correct & Smooth (C&S) combinados con MLP, MLPLinear y GAT, ya que DGL facilita el manejo de nodos con múltiples features y el procesamiento por lotes.
- **Grafo en formato StellarGraph:** Propio de la librería StellarGraph, orientada a problemas de machine learning sobre grafos heterogéneos, grandes y con atributos complejos. Los grafos se representan como objetos StellarGraph, que internamente se construyen a partir de pandas. DataFrame o NetworkX. Este formato ha sido utilizado principalmente para ensayos con GraphSAGE, aprovechando sus herramientas integradas para sampling eficiente y soporte de modelos inductivos.
- **Grafo en formato NetworkX:** NetworkX es una librería ampliamente utilizada en el análisis de grafos por su simplicidad, aunque no está optimizada para entrenamiento con GPUs ni grandes volúmenes de datos. Los grafos se representan como objetos networkx Graph o networkx DiGraph, con nodos y aristas indexados mediante claves. Este formato fue usado como base para representar el grafo de Barcelona y generar variantes estructurales (como el grafo de segmentos), y fue también empleado directamente en los modelos como GINN, NoGE y DAGI.

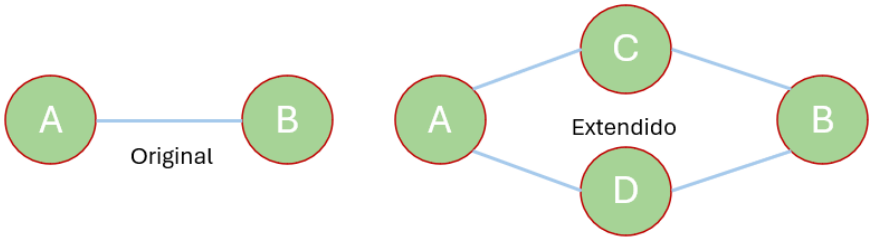


Figura 10: Ejemplo grafo original VS extendido

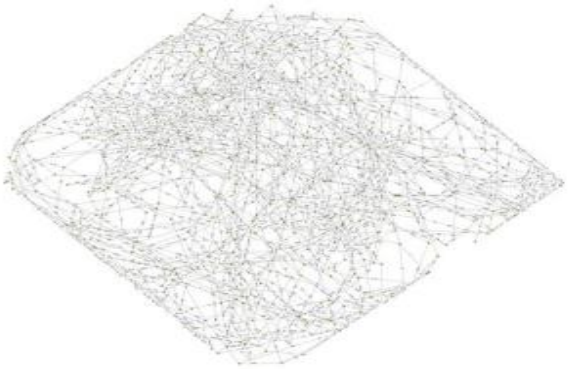


Figura 11: Grafo Barcelona en Python si geolocalización

Figura 12: Grafo de Barcelona en la realidad

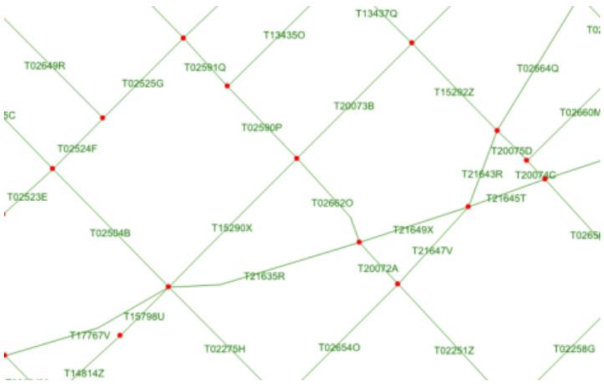


Figura 13: Fragmento ampliado del grafo de Barcelona

En este trabajo, se ha utilizado para los modelos el grafo de segmento y los formatos DGL y NetworkX. No obstante, el resto de los grafos son útiles para futuros y anteriores trabajos. A continuación, se presenta a modo de resumen, como quedaría cada forma de grafo en cuanto a nodos y aristas se refiere:

Tabla 1: Nodos y aristas de cada tipo de Grafo

Forma del Grafo	Nodos	Aristas
Grafo Original	930	2522
Grafo de Segmento	2522	6679
Grado Extendido	3452	5044

4.2 PREPROCESADO Y FORMATO DE LOS DATOS

El preprocesado de los datos para este tipo de proyectos es una etapa clave para garantizar la coherencia estructural del grafo, la calidad de los atributos y la compatibilidad con los modelos de aprendizaje utilizados, así como para obtener unos resultados óptimos. En primer lugar, se llevó a cabo una limpieza inicial para eliminar nodos aislados, duplicados o sin información válida, así como para corregir inconsistencias en los identificadores de tramos y coordenadas geográficas.

Posteriormente, los datos brutos han sido transformados en estructuras compatibles con los frameworks de aprendizaje sobre grafos. Para ello, se ha convertido la información

tabular en tensores de atributos por nodo (x), listas de aristas (edge_index) y vectores de etiquetas o valores a predecir (y). En los grafos dirigidos, se ha respetado la dirección de circulación, y en los casos donde ha sido necesario, se han añadido las aristas inversas para garantizar conectividad.

La segmentación de los datos se ha realizado dividiendo el conjunto de volúmenes a predecir en 2 subconjuntos: entrenamiento, test. Además, para simular condiciones reales de monitorización parcial y ver la robustez de los modelos, se han generado versiones del grafo con distintos porcentajes de volúmenes de tráfico ocultos y varias versiones sobre un mismo porcentaje, pero cambiando que volúmenes se ocultan, es decir, sin valores disponibles para la variable objetivo, lo que permite evaluar la capacidad de generalización de los modelos. Esta ocultación se ha aplicado de forma aleatoria, pero con una semilla para permitir la reproducibilidad del ensayo.

4.3 MÉTRICA DE EVALUACIÓN: ERROR ABSOLUTO MEDIO (MAE)

Para evaluar el rendimiento de los modelos de predicción desarrollados en este proyecto, se ha utilizado como métrica principal el Error Absoluto Medio (MAE, por sus siglas en inglés). Esta métrica permite cuantificar la precisión de las predicciones realizadas sobre los volúmenes ocultos del grafo, comparándolas con los valores reales (originales).

El MAE se define como el valor medio de las diferencias absolutas entre las predicciones y los valores reales:

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

Donde y_i representa el valor real del nodo i, $\hat{y}_i$ su valor predicho, y n el número total de nodos evaluados.

Esta métrica ha sido seleccionada por su interpretabilidad directa (está en las mismas unidades que la variable predicha) y por ser robusta frente a valores atípicos, en

comparación con otros indicadores como el error cuadrático medio (RMSE), que da mucho peso a los errores grandes (outliers). Además, al centrarse en la magnitud del error sin penalizarlo cuadráticamente, es especialmente adecuada en escenarios urbanos donde el objetivo es reducir de forma homogénea las desviaciones de predicción en toda la red.

El cálculo del MAE se ha realizado tras cada experimento e iteración, en los subconjuntos de test, permitiendo comparar de forma objetiva los distintos modelos y configuraciones utilizadas a lo largo del proyecto. Tras iterar sobre un mismo ensayo (mismo % de volumen oculto), 10 veces sea hecho la media del MAE entre todas las iteraciones, evitando así que justo por azar una de las iteraciones haya sido muy favorable, aunque el modelo pueda en realidad no ser tan bueno para otros casos al azar.

4.4 MÉTODOS DE REFERENCIA O BASELINES

Para comparar el rendimiento de los modelos avanzados utilizados en este proyecto, se han definido dos métodos de referencia o **baselines** simples, que permiten establecer un comparativa clara frente a técnicas más sofisticadas. Ambos métodos se basan en reglas estadísticas locales o globales, sin aprendizaje ni entrenamiento de parámetros, y sirven como punto de partida para interpretar la mejora relativa obtenida mediante modelos de GNN. [8]

4.4.1 BASELINE GLOBAL

Este primer baseline consiste en imputar el mismo valor para todos los nodos ocultos del grafo, utilizando la media global de los valores observados en los nodos con información disponible. Se trata de una estrategia simple que no considera la estructura del grafo ni diferencias locales, y actúa como una aproximación constante. Aunque rudimentario, este método permite comprobar si los modelos complejos logran superar un enfoque simplista sin contexto espacial.

4.4.2 BASELINE LOCAL

El segundo baseline incorpora información estructural del grafo. En este caso, el valor predicho para un nodo oculto se calcula como la media de los valores observados de sus vecinos adyacentes (es decir, aquellos con los que está conectado directamente). Este enfoque es más realista en contextos urbanos, donde los tramos cercanos suelen compartir características de flujo similares. Es especialmente útil para evaluar hasta qué punto las redes neuronales sobre grafos logran capturar relaciones más allá del vecindario inmediato.

Ambos métodos se han aplicado de forma sistemática a los mismos conjuntos de prueba utilizados en los experimentos con modelos GNN, permitiendo una comparación justa y cuantitativa basada en la métrica de error absoluto medio (MAE). Su uso es fundamental para validar la utilidad real de los enfoques basados en aprendizaje profundo. A continuación, se muestra una tabla con sendos resultados:

Tabla 2: MAE Baselines Global y Local

Test	Mean MAE Local Baseline	Mean MAE Global Baseline
2%	991	1155
4%	970	1192
6%	988	1180
8%	964	1224
10%	995	1195
12%	993	1162
14%	984	1187
16%	983	1191
18%	981	1183
20%	988	1196
22%	1010	1199
24%	1008	1187
26%	1002	1186
28%	1003	1194
30%	989	1178

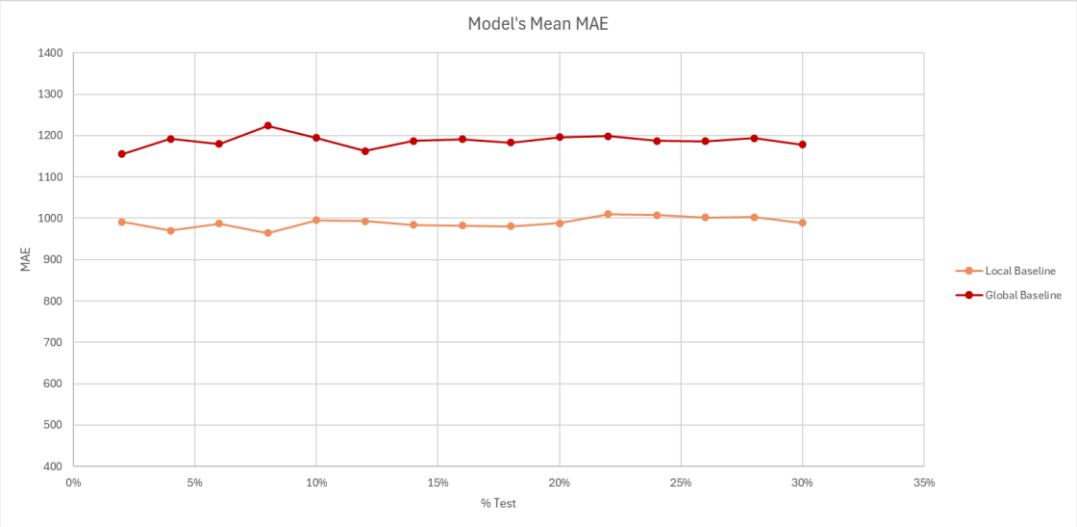


Figura 14: Gráfica MAE Baselines Local y Global

4.5 DISEÑO EXPERIMENTAL

El diseño experimental se ha concebido para evaluar la precisión y consistencia de los modelos de predicción sobre grafos urbanos en escenarios con diferentes niveles de observabilidad. El objetivo es simular condiciones realistas donde no se dispone de información completa sobre todos los volúmenes del grafo, y analizar cómo se comportan los modelos bajo distintas proporciones de datos faltantes. Para ello se ha mantenido consistencia en cada ensayo, en cuanto a la semilla de aleatoriedad a la hora de dividir el dataset en entrenamiento y test, así como los porcentajes de datos ocultos y el número de repeticiones, para eliminar el azar a la hora de que un modelo prediga mejor solo por el hecho de que se hayan ocultado unos volúmenes determinados. Cabe destacar que, debido al formato de cada modelo, no todos han usado el mismo modelo de grafo, pero en sí el grafo usado siempre fue el mismo, aunque no tuviese el mismo formato, su información (volúmenes) si siguió siendo la misma. [10]

4.5.1 PORCENTAJES DE DATOS OCULTOS

Para simular distintos grados de monitorización, se han generado versiones del grafo con una parte de los volúmenes sin valor conocido. Concretamente, se han evaluado porcentajes de ocultación desde el 2 % hasta el 30 % en intervalos de 2 puntos porcentuales. Es decir, se han considerado 15 niveles de ocultación: 2 %, 4 %, 6 %, ..., hasta 30 %.

En cada caso, los nodos a ocultar se seleccionan de forma aleatoria. Los nodos no ocultos conservan su valor real y actúan como base de entrenamiento y contexto para la predicción de los valores faltantes.

4.5.2 REPETICIONES POR EXPERIMENTO

Para cada configuración (modelo y porcentaje de ocultación), se han realizado 10 repeticiones independientes. En cada repetición se utiliza una semilla aleatoria distinta (pero siempre la misma para cada modelo), lo que modifica la partición del grafo (los volúmenes que se ocultan cambian en cada ensayo), garantizando así una evaluación más robusta frente a la variabilidad aleatoria.

Esta estrategia permite obtener resultados estables y estadísticamente representativos. Las métricas presentadas en el análisis de resultados reflejan la media del MAE obtenida en las 10 ejecuciones.

4.5.3 CONSIDERACIÓN ADICIONAL: ELIMINACIÓN DE VALORES NEGATIVOS

Como medida adicional de validación, en todas las ejecuciones se ha aplicado una corrección posterior al modelo que elimina posibles valores negativos en las predicciones. Aunque los modelos no imponen restricciones explícitas sobre el rango de salida, en algunos casos se han generado predicciones menores que cero, lo cual no es físicamente coherente en el contexto del tráfico. Por tanto, cualquier valor negativo ha sido automáticamente truncado a cero antes de calcular la métrica de error (MAE).

4.6 AJUSTE DE HIPERPARÁMETROS

El ajuste de hiperparámetros ha sido una fase clave para optimizar el rendimiento de los modelos de predicción utilizados en este trabajo. Dado que cada modelo presenta una arquitectura y comportamiento distinto, se ha definido un conjunto de hiperparámetros relevantes y se han explorado distintas configuraciones dentro de rangos razonables, seleccionando aquellas que ofrecían aparentemente el mejor rendimiento medio sobre los volúmenes ocultos del grafo.

Entre los hiperparámetros ajustados se encuentran:

- Arquitectura de red: Número de capas y tamaño de las capas (número de neuronas) ocultas del modelo
- Tasa de aprendizaje (learning rate)
- Número de épocas de entrenamiento
- Porcentaje de dropout aplicado para prevenir sobreajuste
- Tipo de optimizador (optimizer), para entrenar a la red neuronal
- Tipo de criterio (criterion), para entrenar a la red neuronal
- Y otros hiperparámetros referente a cada modelo

Para llevar a cabo este ajuste, se ha empleado una búsqueda sistemática, donde en cada repetición del experimento se probaban distintas combinaciones y se iba ajustando según los resultados para optimizar el modelo. Se seleccionaba la combinación que obtenía un error (MAE) promedio bajo sobre los volúmenes para un caso específico. Este enfoque ha permitido mantener un equilibrio entre coste computacional y calidad de predicción. Cabe destacar que la complejidad de cada modelo debía de tratarse de hacer relativamente sencilla dadas las limitaciones computacionales, aunque para un proyecto como este tampoco se vio necesario una complejidad mucho mayor para los modelos, dada la problemática y cantidad de datos del proyecto.

4.7 CRONOGRAMA Y PLANIFICACIÓN DEL TRABAJO

El desarrollo del proyecto se ha planificado a lo largo de un periodo de once meses, abarcando desde septiembre hasta julio. En la siguiente figura se muestra un cronograma detallado con las principales fases del trabajo, desde la asignación inicial del tema hasta la presentación final del TFM. Las tareas se han organizado en paralelo cuando ha sido posible, permitiendo compaginar la revisión bibliográfica con las fases de implementación y experimentación. La redacción del documento se ha concentrado en los últimos cuatro meses, coincidiendo con la etapa final de análisis de resultados. Esta planificación ha facilitado una ejecución progresiva y estructurada, asegurando el cumplimiento de los objetivos dentro del calendario académico establecido.

Tabla 3: Cronograma del Proyecto

Fases	Septiembre	Octubre	Noviembre	Diciembre	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto
Asignación de proyecto												
Revisión literaria												
Implementación Algoritmo C&S												
Implementación Algoritmo GraphSage												
Implementación Algoritmo GINN												
Implementación Algoritmo NoGE												
Implementación Algoritmo DAGI												
Redacción del TFM												
Presentación del TFM												

Capítulo 5. ENSAYOS Y MODELOS

En este capítulo se describen los principales modelos de aprendizaje sobre grafos evaluados en el marco del proyecto, así como los resultados obtenidos en los distintos escenarios experimentales definidos. Los modelos seleccionados representan una combinación de enfoques clásicos y recientes en el campo del Graph Machine Learning, abarcando tanto arquitecturas basadas en agregación local como técnicas más avanzadas de corrección posterior o representación geométrica.

Los cinco modelos evaluados son los siguientes: Correct and Smooth (C&S), GraphSAGE, GINN (Graph Isomorphism Network for Nodes), NoGE (Node Co-occurrence-based GNN) y DAGI (Demographic Aware Graph-based Imputation). Esta selección permite comparar el rendimiento de modelos con diferentes mecanismos de inferencia, capacidad expresiva y requisitos de entrada, cubriendo tanto escenarios de imputación directa como métodos originalmente diseñados para tareas de clasificación o predicción de enlaces.

Cada apartado detalla la estructura del modelo, su adaptación al contexto del grafo urbano y su comportamiento frente a diferentes niveles de información disponible. La evaluación se basa en la métrica de Error Absoluto Medio (MAE) y se realiza sobre volúmenes con valor oculto, conforme a lo descrito anteriormente en el diseño experimental.

5.1 CORRECT AND SMOOTH (C&S)

5.1.1 FUNDAMENTOS DEL MODELO

Correct and Smooth (C&S) es un método de mejora posterior a la predicción que se aplica sobre los resultados iniciales generados por un modelo base (como un MLP, una red GAT u otra arquitectura supervisada). Su principal objetivo es refinar las predicciones de los nodos de un grafo mediante un proceso puramente algorítmico, sin necesidad de volver a entrenar el modelo base. [11]

Este enfoque se basa en dos etapas secuenciales: Correction y Smoothing. En la fase de *correction*, se propagan los errores conocidos de los nodos observados hacia sus vecinos en el grafo, con el objetivo de reducir el sesgo en las predicciones iniciales. Posteriormente, en la fase de *smoothing*, se aplica un suavizado sobre todas las predicciones incluidos los nodos ocultos, reforzando la coherencia local entre nodos vecinos en términos de valores predichos. Ambas fases se implementan mediante técnicas de propagación lineal, inspiradas en métodos clásicos de inferencia sobre grafos como la difusión de etiquetas.

El principal valor de C&S reside en que no sustituye al modelo base, sino que actúa como una capa de postprocesado independiente que mejora su rendimiento sin requerir una arquitectura adicional ni reentrenamiento. Esto lo hace especialmente atractivo en contextos donde ya se dispone de predicciones preliminares, pero se desea explotar la estructura del grafo para corregir errores sistemáticos y homogeneizar los resultados finales.

En este proyecto, el enfoque C&S se ha utilizado para mejorar los resultados de tres modelos base distintos:

- **MLP clásico**: El MLP es una red neuronal feedforward compuesta por varias capas densas totalmente conectadas. En su versión más común, cada capa aplica una transformación lineal seguida de una función de activación no lineal (como ReLU). En este caso, el MLP actúa como un modelo supervisado clásico, entrenado exclusivamente con los atributos disponibles de los nodos observados para predecir su valor objetivo (por ejemplo, el flujo de tráfico). Este enfoque no utiliza la estructura del grafo durante el proceso de entrenamiento, lo que lo convierte en una referencia útil para evaluar hasta qué punto el grafo y sus relaciones pueden mejorar el rendimiento. Una vez entrenado, el MLP se utiliza para generar predicciones iniciales para todos los nodos, incluyendo los no observados, y dichas predicciones son refinadas por C&S.
- **MLP Linear**: La versión MLP Linear corresponde a un perceptrón multicapa sin activaciones no lineales. Es decir, cada capa aplica únicamente una

transformación lineal (y = Wx + b), lo que convierte al modelo en una cadena de proyecciones lineales sin capacidad de modelar relaciones no lineales complejas. Este modelo es deliberadamente más limitado que el MLP tradicional, y se incluye con el objetivo de estudiar si el método Correct and Smooth es capaz de compensar la falta de expresividad del modelo base. Al no contar con activaciones, las representaciones intermedias no capturan interacciones complejas entre atributos, lo que hace que sus predicciones iniciales sean más simples y, potencialmente, más dependientes del postprocesado posterior.

- **GAT (Graph Attention Network):** La red GAT introduce un mecanismo de atención sobre grafos, lo que permite a cada nodo ponderar la importancia de sus vecinos al construir su representación interna. A diferencia de GCN o GIN, que agregan de forma uniforme o fija, GAT aprende pesos de atención específicos para cada par de nodos conectados, dependiendo de sus atributos y contexto. Esta capacidad permite que la red se enfoque más en vecinos relevantes y reduzca el impacto de nodos ruidosos o menos informativos. A diferencia del MLP y el MLP Linear, GAT explota explícitamente la estructura del grafo desde el inicio del aprendizaje, integrando relaciones topológicas durante el proceso de inferencia. Esto permite generar predicciones iniciales que ya incorporan contexto estructural, y ofrece un punto de comparación útil para evaluar si C&S sigue aportando valor en modelos que ya modelan el grafo de forma nativa.

Esta diversidad permite analizar hasta qué punto el mecanismo de corrección y suavizado logra reducir el error medio independientemente de la arquitectura previa, y si su efecto es más notorio en modelos simples que en arquitecturas ya adaptadas a grafos como GAT.

5.1.2 ADAPTACIÓN E IMPLEMENTACIÓN DEL MODELO

El enfoque Correct and Smooth (C&S) fue originalmente propuesto como un método de post-procesado capaz de mejorar las predicciones de modelos base poco expresivos mediante una combinación de propagación de errores (correction) y suavizado (smoothing) sobre la estructura del grafo. Aunque su aplicación se ha popularizado en

tareas de clasificación, este trabajo lo adapta y explora en un contexto de regresión sobre un grafo urbano segmentado de la ciudad de Barcelona, donde el objetivo es estimar el volumen de tráfico en calles no monitorizadas.

Estructura general del pipeline

La implementación general sigue un flujo modular:

- **Entrenamiento del modelo base**: Se entrena un predictor (MLP, MLPLinear o GAT) sobre las características de los nodos disponibles. Este modelo se entrena únicamente con nodos observados y se guarda para posteriores usos. A modo de ejemplo podemos ver el error en cada epoch (iteración de entrenamiento) del entrenamiento para cada caso y como se va reduciendo a medida que el modelo se va entrenando (Se puede ver además como el entrenamiento del modelo linear es mucho más lento aprendiendo debido a que este no es capaz de capturar interacciones complejas):

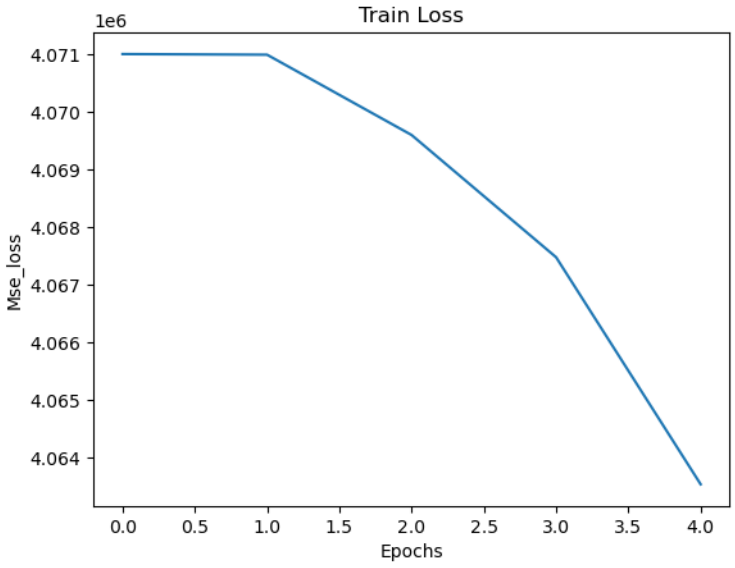


Figura 15: Ejemplo Training GAT

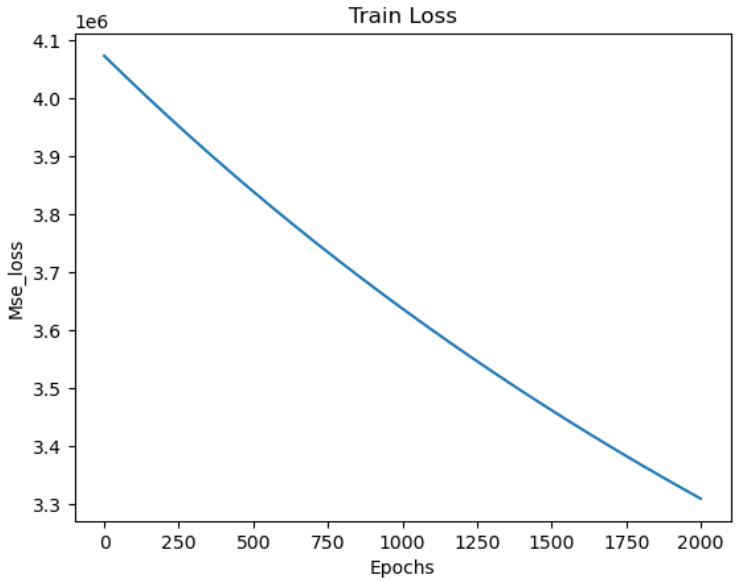


Figura 16: Ejemplo Training MLP Linear

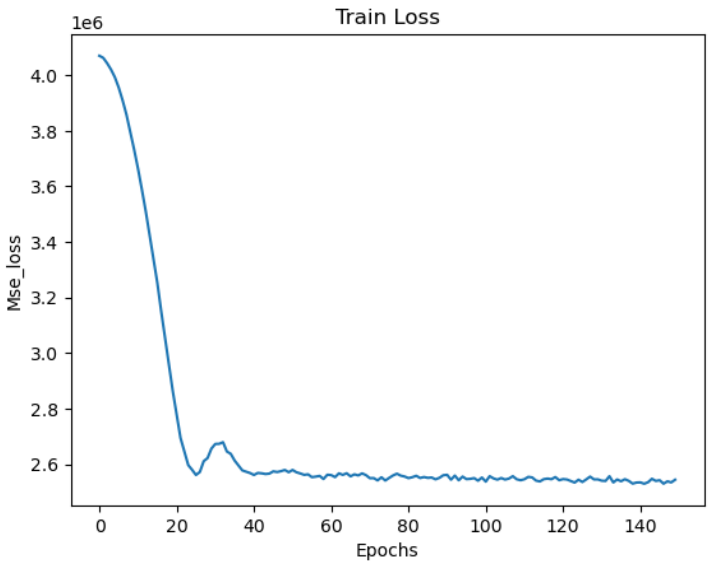


Figura 17: Ejemplo MLP No Linear

- **Corrección (Correction)**: Tras el entrenamiento, se aplica una propagación del error entre los valores predichos y los verdaderos de los volúmenes etiquetados. Este error es suavizado y reinyectado en las predicciones iniciales, permitiendo ajustar las salidas de los nodos no observados.

- **Suavizado (Smoothing)**: Finalmente, se ejecuta una propagación de los valores corregidos para suavizar las predicciones y reforzar la coherencia topológica entre nodos adyacentes.

Código y consideraciones prácticas

- El grafo es tratado con la librería DGL, y se normalizan las características de los nodos costes (weight) antes del entrenamiento.
- Se realiza una estandarización de los nodos, esencial para garantizar una convergencia estable de los modelos.
- Las máscaras de entrenamiento, validación y test se generan aleatoriamente para cada repetición experimental, permitiendo simular distintos grados de ocultamiento de datos (test ratio).
- En todos los casos, se evalúan los errores antes y después de aplicar C&S, registrando el MAE (Mean Absolute Error) sobre el conjunto de test.
- Se ajustan manualmente hiperparámetros específicos para el C&S como el número de capas de corrección y suavizado, el coeficiente de propagación (alpha) y el tipo de normalización (DAD, AD, DA), adaptando el método a un contexto de regresión (sin activaciones ni proyecciones softmax).

Integración de C&S con modelos base

- **C&S con MLP**

El modelo MLP utilizado consta de 4 capas, 224 unidades ocultas por capa y una tasa de dropout del 25%. Su activación principal es ReLU. Tras su entrenamiento, se aplica C&S con los parámetros: 46 capas de corrección y 45 de suavizado, con alpha=0.85 y normalización DAD. Esta configuración logra una mejora sustancial en los resultados, reduciendo el MAE alrededor de un 60% frente a las predicciones iniciales.

- **C&S con MLPLinear**

En este caso, el modelo base consiste en una única capa lineal sin activaciones ni no linealidades. El objetivo es evaluar la capacidad de C&S para corregir un predictor extremadamente limitado. A pesar de esta simplicidad, el postprocesado mejora de forma notable los resultados, reduciendo consistentemente el MAE en todos los niveles de test alrededor de un 26%.

- **C&S con GAT**

La red GAT incorpora atención multi-cabeza (4 heads) y 4 capas profundas con 234 unidades cada una. Aunque ya incorpora información estructural durante el entrenamiento, se observó que C&S aún es capaz de mejorar las predicciones, aunque en menor proporción comparado con modelos más simples. El GAT se entrena con RMSprop, y el grafo requiere la adición explícita de self-loops. Para 25 epochs el GAT se puede ver que sufre de overfitting, pero el C&S es capaz de reducirlo considerablemente en un 60% más o menos. En cambio, para el GAT con 5 epochs el C&S lo mejora solo en alrededor de un 30%, sin embargo, esto se debe también a que en este caso el GAT a predicho mucho mejor que en el anterior y por lo tanto no hay tanto error sobre el que mejorar.

5.1.3 RESULTADOS

A continuación, se presenta la tabla resumen con el MAE medio obtenido para cada nivel de test:

Tabla 4:MAE C&S Resultados

Test	Mean MAE (MLP)		Mean MAE (MLP Linear)		Mean MAE (GAT 25 epochs)		Mean MAE (GAT 5 epochs)	
	After C&S	Before C&S	After C&S	Before C&S	After C&S	Before C&S	After C&S	Before C&S
2%	483	1217	912	1259	1243	3166	797	1137
4%	467	1194	833	1131	1184	3416	798	1134
6%	494	1234	885	1186	1211	3265	846	1242
8%	485	1243	914	1228	1337	4170	818	1206
10%	483	1255	882	1181	1326	4091	830	1213
12%	483	1255	887	1194	1262	3614	829	1192
14%	484	1216	890	1188	1145	3137	817	1166
16%	487	1232	896	1198	1278	3845	813	1181
18%	462	1192	878	1193	1171	2846	832	1180
20%	478	1203	890	1181	1329	4032	818	1176
22%	474	1234	896	1192	1229	3397	814	1184
24%	485	1226	884	1176	1330	4188	823	1184
26%	485	1248	911	1230	1312	3932	834	1199
28%	467	1211	889	1186	1167	2888	820	1188
30%	474	1205	889	1189	1241	3576	819	1174

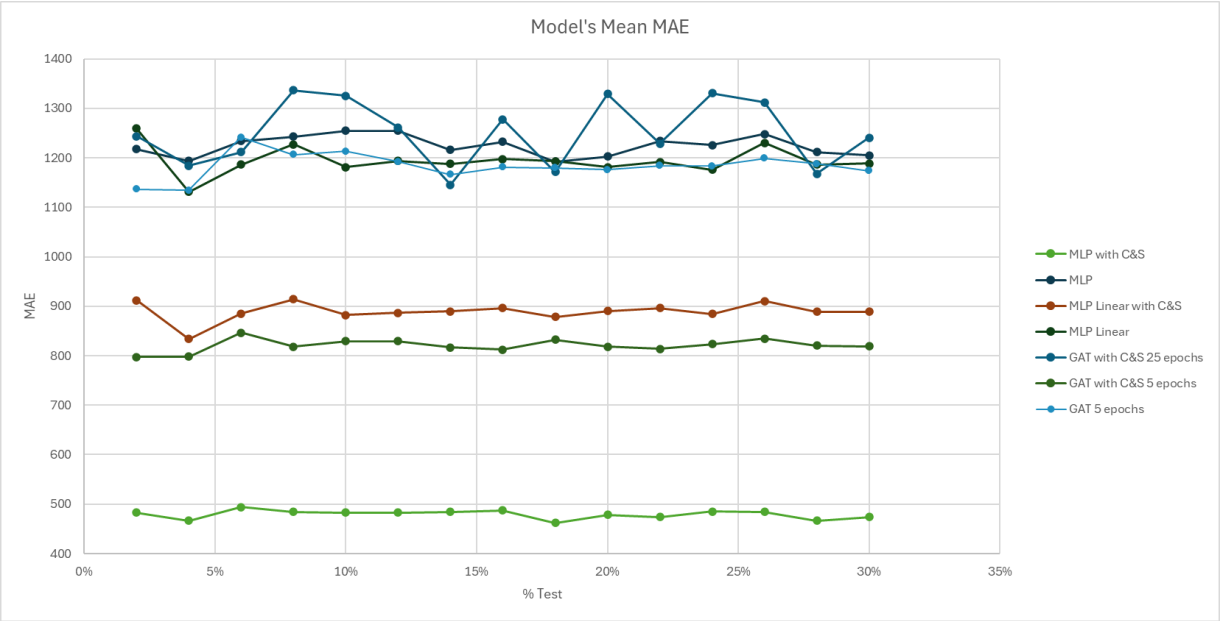


Figura 18: MAE del modelo C&S

Los resultados obtenidos muestran una mejora significativa en todos los casos tras aplicar el postprocesado C&S, especialmente en combinación con el modelo MLP. En esta configuración, el error absoluto medio (MAE) se redujo desde aproximadamente 1240 unidades hasta 490, lo que supone una mejora del 60 % respecto a la predicción inicial. Este comportamiento se mantuvo de forma consistente incluso en los niveles más altos de ocultación (30 % de nodos sin información), lo que indica una buena capacidad de generalización del sistema.

En el caso del MLP Linear, la mejora fue más moderada, pero igualmente relevante. El modelo base partía de un MAE cercano a 1190, y tras aplicar C&S logró reducirlo hasta 880. Este descenso de en torno al 26 % es destacable teniendo en cuenta que se trata de un modelo lineal sin capacidad para capturar relaciones no lineales entre las variables, lo que sugiere que el postprocesado es capaz de corregir parte de las limitaciones estructurales del predictor base.

Para GAT, se observó un comportamiento más heterogéneo según el número de épocas de entrenamiento. Cuando el modelo se entrenó durante 25 épocas, sufrió claramente de sobreajuste, generando un MAE inicial muy elevado (superior a 3700). En este escenario, C&S permitió reducir el error en más de 2300 unidades, mejorando en torno a un 60 %. Sin embargo, con solo 5 épocas de entrenamiento, el GAT ofreció ya resultados razonables (MAE en torno a 850) y C&S aportó una mejora más contenida, de unos 250 puntos, equivalente a un 30 % de mejora relativa.

En todos los casos, los modelos mejorados con C&S superaron con claridad tanto al baseline global como al local, lo cual valida su capacidad de aprendizaje y justifica su aplicación. De hecho, esta técnica resultó especialmente útil para modelos con errores sistemáticos o expresividad limitada, y demostró ser capaz de adaptarse a diferentes niveles de datos disponibles, manteniendo la calidad de las predicciones incluso cuando la información era escasa. Estos resultados refuerzan la idea de que C&S constituye una solución eficaz, sencilla y robusta para mejorar modelos en entornos parcialmente monitorizados, como es el caso del tráfico urbano.

5.2 GRAPHSAGE

5.2.1 FUNDAMENTOS DEL MODELO

GraphSAGE (Graph Sample and Aggregate) es un modelo de aprendizaje sobre grafos que genera representaciones vectoriales o embeddings para cada nodo combinando su propia información con la de sus vecinos inmediatos. A diferencia de arquitecturas como GCN, que requieren conocer todo el grafo de antemano, GraphSAGE puede llegar a ser inductivo, puede generalizar a nodos o grafos no vistos durante el entrenamiento, lo que lo podría hacer adecuado para entornos dinámicos o parcialmente observables. [12]

El modelo funciona en dos fases principales: primero muestra un subconjunto de vecinos de cada nodo; luego, agrega sus características mediante funciones como la media, una LSTM, o un operador de pooling. Estas representaciones agregadas se combinan con las del nodo central para producir una nueva representación. Esta operación se realiza capa por capa, lo que permite capturar información estructural de forma progresiva.

En este proyecto, GraphSAGE se aplica sobre un grafo urbano segmentado, donde cada nodo representa un tramo de calle. Las características de entrada de cada nodo incluyen tanto atributos propios como información supervisada parcial (por ejemplo, si se conoce o no su volumen de tráfico). Estas variables permiten al modelo tener un contexto más enriquecido para aprender a inferir los valores ocultos. Además, se ha optado por extender las características con un indicador binario que informa si el nodo pertenece al conjunto de entrenamiento, lo que ayuda al modelo a distinguir entre ejemplos reales y puntos a predecir.

La arquitectura empleada incluye dos capas convolucionales SAGE con activación ReLU, seguidas de una capa lineal de salida para producir una única predicción por nodo. La estructura del grafo se utiliza directamente para definir qué nodos están conectados, y por tanto, quién influye en quién durante la agregación. En resumen, GraphSAGE aprovecha tanto la topología del grafo como los atributos de nodo conocidos para generar estimaciones robustas en escenarios con información parcial.

5.2.2 ADAPTACIÓN E IMPLEMENTACIÓN DEL MODELO

La implementación del modelo GraphSAGE se ha desarrollado en entorno PyTorch Geometric, adaptando su estructura para ajustarse a las particularidades del grafo de tráfico utilizado en este trabajo. El proceso parte de un grafo segmentado, cargado desde NetworkX, en el que cada nodo representa un tramo de vía y contiene atributos propios como el peso (weight) y el volumen de tráfico (volume), siendo este último la variable objetivo a predecir.

Una de las principales particularidades de esta implementación es el enriquecimiento de las características de los nodos. Para ello, además de los atributos originales, se incorporan dos variables adicionales:

- Un valor supervisado (label_value), que es el volumen real si el nodo pertenece al conjunto de entrenamiento, y cero en caso contrario.
- Un indicador binario (label_known), que señala si ese nodo ha sido observado durante el entrenamiento.

Estas dos variables se concatenan a las características originales de entrada, permitiendo al modelo conocer explícitamente qué nodos disponen de información verificada y cuáles no, lo que puede reforzar su capacidad de inferencia en escenarios de imputación.

La red implementada consta de dos capas SAGEConv seguidas de una capa densa final para la predicción regresiva. Durante el entrenamiento, se emplean máscaras binarias (train_mask y test_mask) generadas mediante particiones aleatorias reproducibles, controladas por una semilla. De este modo, se asegura la comparabilidad de los resultados entre repeticiones y se simulan condiciones de información parcial, ocultando entre un 2% y un 30% de los nodos. Además, para evitar overfitting se determinó usar 80 epochs para cada repetición. Y se optó por un learning rate estándar de 0.01.

Finalmente, el modelo se entrena utilizando una función de pérdida MSE y un optimizador Adam. Para reforzar la coherencia física de las predicciones, se ha introducido un postprocesado que elimina valores negativos en la salida del modelo, dado que los flujos de tráfico no pueden ser inferiores a cero.

5.2.3 RESULTADOS

La tabla siguiente recoge el MAE medio obtenido para cada nivel de ocultación:

Tabla 5: MAE GraphSage Resultados

Test	Mean MAE
2%	1267
4%	1175
6%	1205
8%	1203
10%	1175
12%	1173
14%	1165
16%	1164
18%	1163
20%	1162
22%	1171
24%	1169
26%	1170
28%	1170
30%	1178

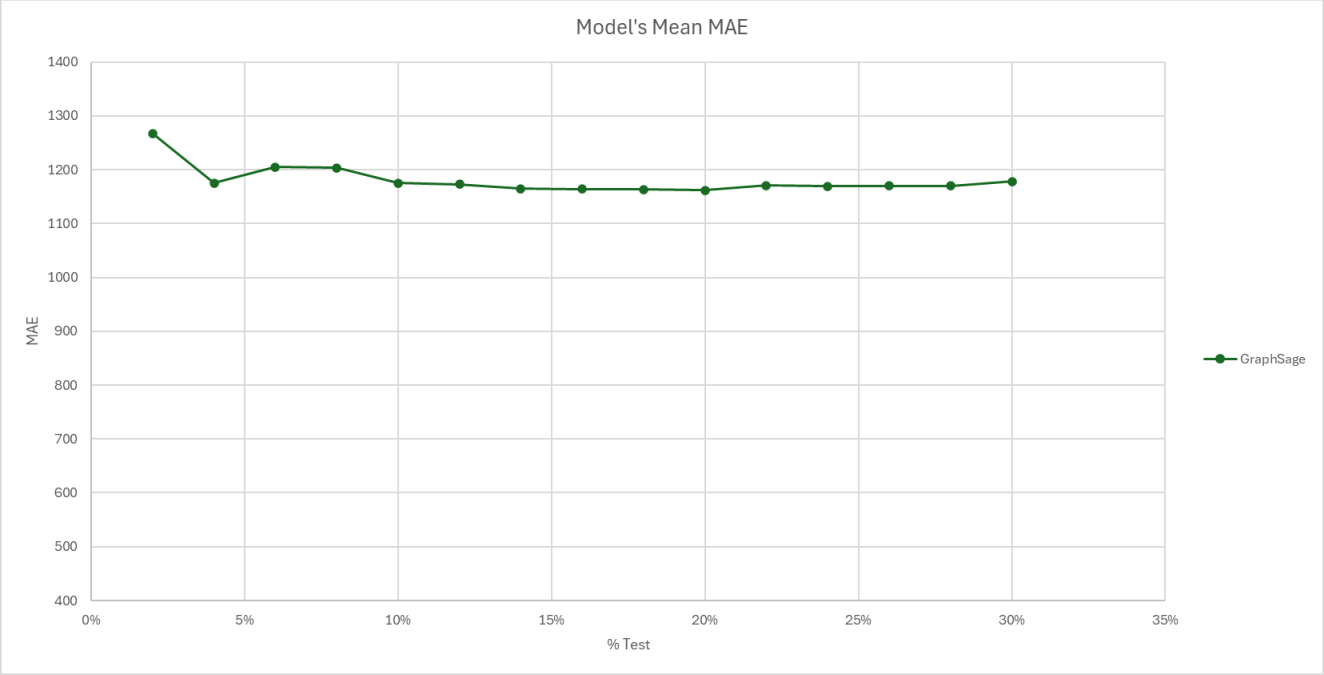


Figura 19: MAE del modelo GraphSage

Los resultados obtenidos con el modelo GraphSAGE muestran un comportamiento estable en términos de error absoluto medio (MAE) a lo largo de los distintos niveles de ocultación de datos, con valores que oscilan entre aproximadamente 1160 y 1175 unidades. Esta estabilidad sugiere que el modelo mantiene una capacidad de generalización razonable incluso cuando se reduce la cantidad de datos disponibles durante el entrenamiento.

Sin embargo, al comparar estos resultados con los métodos de referencia definidos como baselines, se observa que GraphSAGE apenas logra superar el baseline global y no mejora de forma consistente el baseline local. En algunos porcentajes de nodos ocultos, el rendimiento del modelo es incluso inferior al del enfoque local basado en la media de los vecinos. Este hecho plantea dudas sobre la eficacia del modelo en este contexto concreto.

La incapacidad de superar de forma clara a los baselines sugiere que GraphSAGE no está extrayendo una información significativa de la estructura del grafo más allá de lo que se puede obtener con métodos simples. La arquitectura utilizada, basada en dos capas con función de agregación media, puede haber sido insuficiente para capturar patrones más complejos o de mayor alcance. Además, la información de entrada limitada también puede haber condicionado la capacidad del modelo para distinguir entre situaciones locales similares.

En conjunto, aunque GraphSAGE presenta un comportamiento predecible y robusto frente a variaciones en la proporción de datos ocultos, su utilidad real queda comprometida al no ofrecer mejoras sustanciales sobre los métodos de referencia más simples. En este caso, el modelo no muestra un aprendizaje suficientemente diferencial como para justificar su complejidad adicional frente a una estrategia basada en promedios locales.

5.3 GINN (GRAPH ISOMORPHISM NETWORK FOR NODES)

5.3.1 FUNDAMENTOS DEL MODELO

El modelo GINN se basa en la arquitectura Graph Isomorphism Network (GIN), propuesta como una de las redes neuronales gráficas más potentes en términos de capacidad expresiva. A diferencia de otros métodos más simples como GCN o GraphSAGE, que realizan una agregación promedio o ponderada de las características de los nodos vecinos, GIN incorpora una función de agregación basada en un MLP (Multilayer Perceptron), lo que le permite modelar relaciones más complejas o no lineales entre los nodos y su entorno estructural. [18]

La motivación teórica detrás de GIN es alcanzar una expresividad comparable a la del test de isomorfismo de Weisfeiler-Lehman, es decir, que sea capaz de distinguir estructuras de grafos diferentes en función de la distribución local de sus nodos. Para lograr esto, cada nodo actualiza su representación combinando su estado anterior con el de sus vecinos, y pasando la suma resultante por un MLP que introduce una transformación no lineal para el aprendizaje.

La arquitectura GIN típicamente incluye varias capas de este estilo, donde cada capa consta de:

- Una operación de agregación (suma) entre las características del nodo y las de sus vecinos.
- Un bloque MLP que actúa como función de actualización.

Este diseño permite que la red no solo recopile información de la vecindad de cada nodo, sino que también aprenda a interpretar dicha información de forma no lineal y adaptativa.

En este proyecto, GIN ha sido adaptado a una tarea de regresión con el objetivo de predecir el volumen de tráfico. A diferencia de un modelo de clasificación, donde se predicen categorías discretas, aquí la salida del modelo es un valor escalar continuo para cada nodo.

Además, la implementación específica de GINN utilizada en este proyecto incorpora una ampliación de las características de cada nodo (como en el caso de GraphSage): cada nodo incluye, junto a sus atributos originales, dos valores adicionales que indican si su valor objetivo es conocido (label_known) y cuál es ese valor (label_value) en caso de ser un nodo observado. Esta estrategia permite al modelo distinguir entre nodos con y sin supervisión directa durante el entrenamiento, lo que mejora la capacidad de generalización hacia los nodos "ocultos".

Por su diseño, GINN es especialmente adecuado para este tipo de tareas, ya que no solo capta la estructura local del grafo (quién está conectado con quién), sino que también puede aprender patrones más complejos gracias a la flexibilidad de los MLPs internos. Esto lo convierte en una posible buena opción para problemas en los que las relaciones entre nodos sean clave para la inferencia, como ocurre en la predicción de flujos de tráfico en redes urbanas.

5.3.2 ADAPTACIÓN E IMPLEMENTACIÓN DEL MODELO

La implementación del modelo GINN en este proyecto se ha llevado a cabo utilizando la librería PyTorch Geometric, específicamente mediante el módulo GINConv, que permite construir capas GIN con funciones de agregación definidas por redes MLP. La arquitectura final del modelo se ha estructurado en dos capas convolucionales GIN, seguidas de una capa lineal que proyecta la representación final a un único valor escalar por nodo, correspondiente al volumen de tráfico.

Una de las particularidades clave de la implementación reside en el preprocesamiento de las características de entrada. Para cada nodo, se han concatenado dos atributos adicionales (como en el caso de Graphsage):

- **label_value**: el valor objetivo (volumen) si el nodo pertenece al conjunto de entrenamiento, o cero en caso contrario.
- **label_known:** un indicador binario que vale 1 si el nodo es conocido (entrenamiento) y 0 si es desconocido (test).

En cuanto al flujo de entrenamiento, se ha utilizado la función de pérdida MSELoss (Error Cuadrático Medio), común en tareas de regresión, y la métrica de evaluación principal ha

sido el MAE (Error Absoluto Medio). Adicionalmente, se introdujo una modificación post-proceso para forzar a cero aquellas predicciones negativas, ya que los valores negativos de tráfico no tienen sentido físico.

Una decisión relevante en la experimentación fue la evaluación del efecto del número de épocas de entrenamiento. Por ello, se realizaron pruebas con dos configuraciones distintas:

- 200 épocas, para evaluar una convergencia estándar en un tiempo razonable.
- 600 épocas, con el objetivo de estudiar si una mayor persistencia en el entrenamiento permitía al modelo capturar patrones más profundos o si, por el contrario, derivaba en "overfitting".

Este análisis comparativo permite valorar no solo la capacidad de predicción de GINN, sino también su sensibilidad a parámetros clave del proceso de optimización. El modelo mostró una mejora sostenida en el MAE al aumentar las épocas (epochs).

5.3.3 RESULTADOS

Los ensayos realizados con GINN se centraron en analizar su rendimiento bajo dos escenarios distintos: con 200 épocas y con 600 épocas de entrenamiento. En ambos casos, se mantuvieron constantes el resto de los parámetros, incluyendo el porcentaje de nodos ocultos (desde el 2 % hasta el 30 %, con incrementos del 2 %) y la repetición de cada experimento 10 veces con semillas distintas para garantizar robustez estadística.

A continuación, se presentan los valores medios del MAE obtenidos para cada configuración:

Tabla 6: MAE GINN Resultados

Test	Mean MAE (200 epochs)	Mean MAE (600 epochs)
2%	1117	1089
4%	1072	1023
6%	1080	1033
8%	1077	1024
10%	1049	1007
12%	1040	999
14%	1052	1006
16%	1052	1017
18%	1041	1019
20%	1037	1017
22%	1065	1004
24%	1051	1016
26%	1051	1014
28%	1054	1004
30%	1049	1029

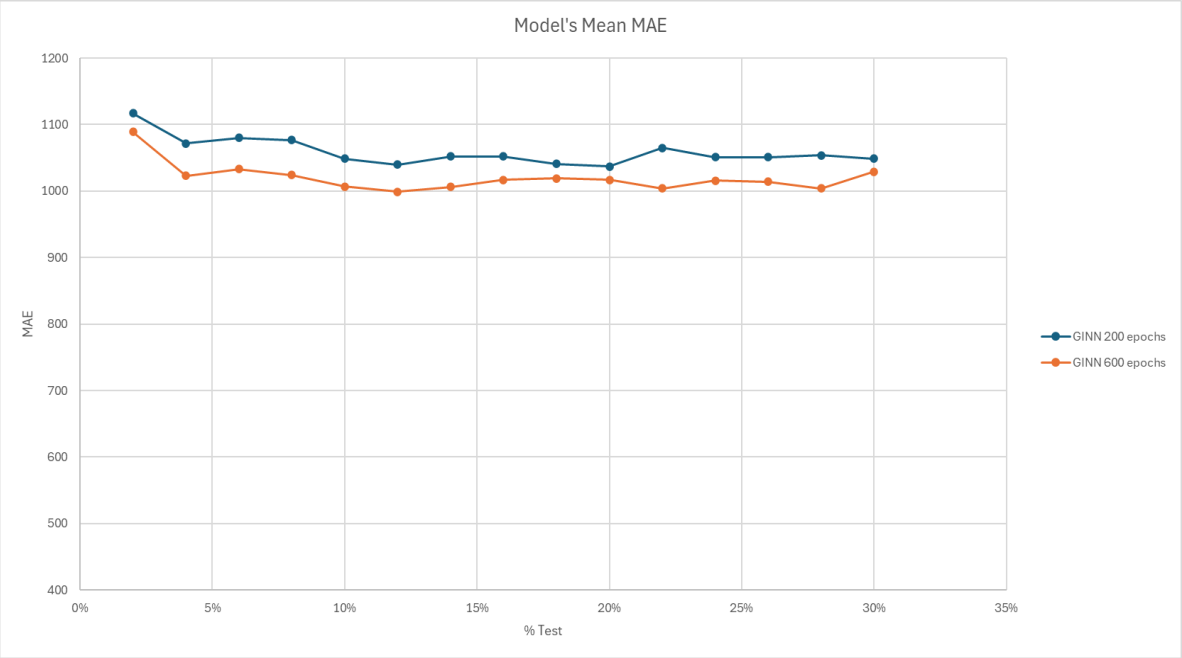


Figura 20: MAE del modelo GINN

El modelo GINN mostró una evolución más positiva en comparación con GraphSAGE, especialmente cuando se incrementó el número de épocas de entrenamiento. En concreto, el MAE medio se redujo progresivamente al pasar de 200 a 600 épocas, consolidando una mejora sostenida y sin evidencia clara de sobreajuste. Esta capacidad de ajuste sugiere que el modelo se beneficia del mayor número de iteraciones, logrando representar patrones estructurales más complejos a lo largo del grafo.

A pesar de esta mejora, los valores absolutos de MAE obtenidos por GINN, en torno a 1075–1150 según el porcentaje de nodos ocultos, no son suficientes para superar con claridad el baseline local. Aunque sí mejora con respecto al baseline global, el margen respecto al método local sigue siendo escaso. Esto implica que, aunque GINN logra aprender ciertos patrones a partir de la topología y atributos del grafo, su ventaja frente a aproximaciones simples basadas en la media de vecinos sigue siendo limitada.

Es posible que la falta de diversidad en los atributos de entrada, escasez de variables o la simplicidad relativa del grafo de entrada limiten el potencial del modelo para predecir.

En resumen, GINN se comporta de forma más eficaz que GraphSAGE, mostrando sensibilidad al número de épocas y una mejora clara al prolongar el entrenamiento. No obstante, la diferencia con respecto al baseline local sigue siendo reducida, lo que sugiere que, aunque el modelo aprende, no alcanza aún una ventaja suficientemente clara como para justificar su uso frente a métodos más simples.

5.4 NoGE (Node Co-occurrence based Graph Neural Network)

5.4.1 Fundamentos del modelo

El modelo NoGE representa una propuesta avanzada en el ámbito del aprendizaje sobre grafos, orientada específicamente a tareas de predicción de enlaces en grafos de conocimiento. Su innovación clave radica en integrar explícitamente información de coocurrencia entre entidades y relaciones dentro del proceso de aprendizaje, lo que permite modelar con mayor precisión estructuras semánticas complejas presentes en los datos. [17]

En lugar de tratar únicamente las entidades como nodos y las relaciones como bordes como ocurre en muchos enfoques clásicos NoGE construye un grafo unificado en el que tanto entidades como relaciones se consideran nodos de primer orden. Esta transformación (conocida como *Levi transformation*) permite capturar la coocurrencia entre cualquier par de nodos, ya sean entidades entre sí, relaciones entre sí o entidades con relaciones. A partir de la frecuencia con la que estos elementos coaparecen en triples (h, r, t), NoGE define pesos para los enlaces del grafo que alimentan una matriz de adyacencia enriquecida.

Una de las contribuciones más significativas del modelo es la introducción de una nueva arquitectura denominada Dual Quaternion Graph Neural Network (DualQGNN). Esta red extiende la representación de vectores clásicos a través de cuaterniones duales, una estructura matemática que permite modelar simultáneamente rotaciones y traslaciones de manera más rica y expresiva que los cuaterniones tradicionales. En este contexto, cada nodo es representado como un cuaternión dual $h = q + \epsilon p$, donde q y p son cuaterniones, y ϵ es una unidad dual tal que $\epsilon^2 = 0$. Las operaciones de propagación de mensajes se llevan a cabo mediante productos de Hamilton extendidos a esta estructura, permitiendo una transformación geométrica más precisa de las representaciones de nodos.

Durante el entrenamiento, la representación resultante de los nodos (tanto entidades como relaciones) se concatena y se alimenta a una función de puntuación basada en QuatE para calcular la verosimilitud de un triple. Esta combinación de encoder geométricamente expresivo (DualQGNN) y decoder especializado (QuatE) ha demostrado, según los experimentos del artículo original, resultados de última generación en conjuntos de datos exigentes como CoDEx-S, M y L.

Aunque originalmente diseñado para grafos de conocimiento, en este trabajo se ha adaptado la estructura de NoGE a la tarea de regresión sobre grafos urbanos de tráfico, eliminando la necesidad de un decoder de tipo QuatE y enfocando la arquitectura en un flujo directo desde los atributos de los nodos hacia una predicción continua, manteniendo la potencia expresiva de la propagación basada en cuaterniones duales.

5.4.2 ADAPTACIÓN E IMPLEMENTACIÓN DEL MODELO

Para adaptar NoGE al contexto del presente trabajo centrado en la predicción de volumen de tráfico en grafos urbanos fue necesario realizar una serie de transformaciones significativas sobre su diseño original, ya que este modelo estaba inicialmente orientado a tareas de enlace en grafos de conocimiento.

En primer lugar, se sustituyó el *decoder* de tipo QuatE encargado de puntuar triples (h, r, t) en tareas de enlace por una capa final de regresión, que toma como entrada la representación final de cada nodo y produce un valor escalar que estima su volumen de tráfico. Esta salida se ajusta mediante una función de pérdida tipo MSELoss, habitual en contextos de regresión.

El modelo conserva, no obstante, la arquitectura interna basada en cuaterniones, en concreto mediante la clase QGNN_layer, que permite realizar una propagación de mensajes avanzada sobre una matriz de adyacencia densa. Cada capa de esta red emplea un mecanismo de multiplicación de Hamilton extendido, lo que incrementa la capacidad de representación geométrica del modelo en comparación con GNNs convencionales.

El preprocesamiento del grafo se basa en tomar el grafo segmentado de Barcelona, cargado desde un archivo .dat en formato networkx. Se extraen los atributos de cada nodo

(weight) y sus valores de volumen (volume), y se transforma el grafo en una estructura con nodos indexados y matriz de adyacencia simétrica. Posteriormente, los datos se dividen en conjuntos de entrenamiento y test de forma reproducible, mediante semillas fijas y con distintas proporciones de test (del 2% al 30%).

El entrenamiento se realiza durante 300 épocas por experimento, y se evalúa el rendimiento mediante el MAE medio tras 10 repeticiones para cada porcentaje de nodos ocultos. Todos los experimentos para este caso se ejecutarían en GPU (si estuviese disponible), para aprovechar la eficiencia computacional de PyTorch., pero en nuestro caso no se tenía dicha capacidad.

Cabe destacar que, a diferencia de otros modelos implementados en PyTorch Geometric, NoGE se construye de forma más manual, utilizando tensores densos y multiplicaciones explícitas de matrices. Esto permite un control más detallado sobre la lógica interna, aunque también exige una mayor planificación en el manejo de memoria y operaciones en lotes.

5.4.3 RESULTADOS

A continuación, se muestra un resumen de los resultados medios obtenidos por el modelo:

Tabla 7; MAE NoGE Resultados

Test	Mean MAE
2%	1194
4%	1122
6%	1136
8%	1133
10%	1110
12%	1105
14%	1099
16%	1095
18%	1092
20%	1088
22%	1093
24%	1094
26%	1089
28%	1093
30%	1096

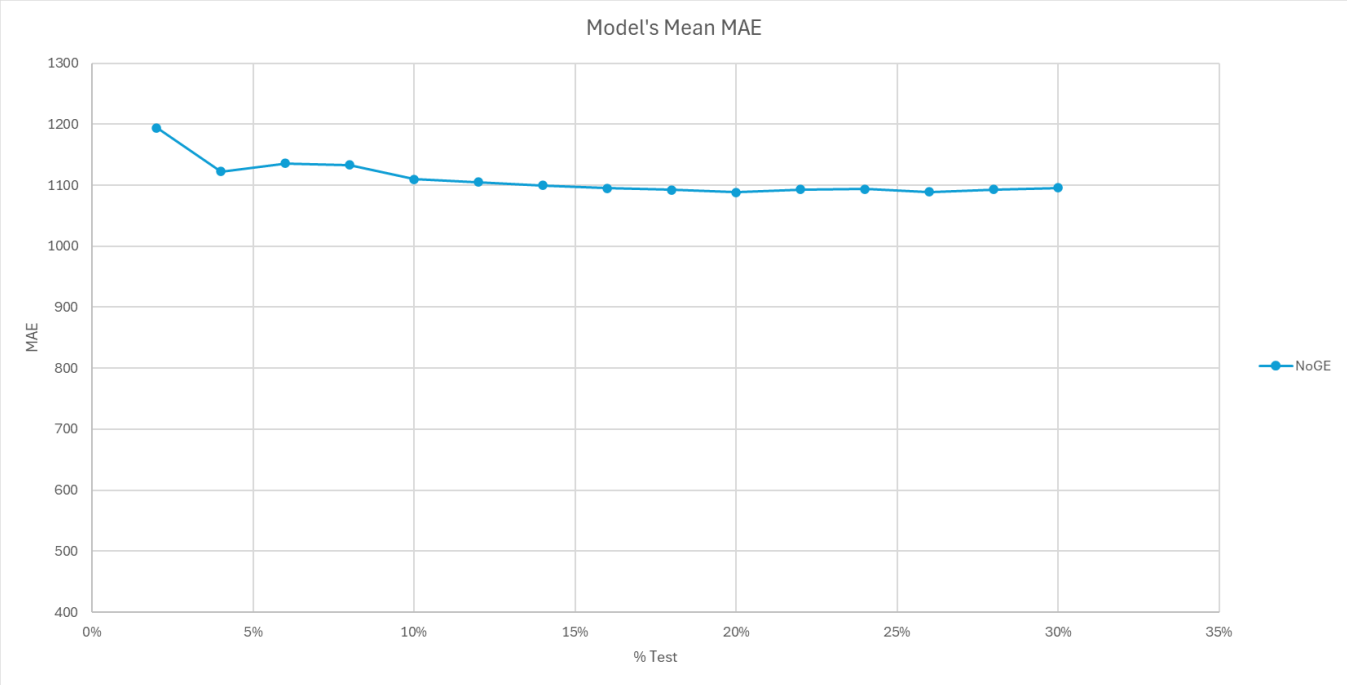


Figura 21: MAE del modelo NoGE

El modelo NoGE ofreció un rendimiento estable a lo largo de los distintos niveles de ocultación, con valores de MAE comprendidos entre aproximadamente 1090 y 1120. Esta estabilidad indica que el modelo fue capaz de mantener una calidad de predicción constante incluso cuando la proporción de nodos sin datos aumentaba, lo que puede atribuirse a la expresividad geométrica de su arquitectura basada en cuaterniones duales.

No obstante, al analizar su rendimiento en términos relativos, se observa que NoGE tampoco logra superar al baseline local. Aunque en ciertas configuraciones se sitúa ligeramente por debajo del MAE del baseline global, la diferencia no es significativa. Esto sugiere que el modelo, pese a su complejidad arquitectónica y su capacidad teórica para capturar estructuras avanzadas, no logra traducir esa sofisticación en una mejora clara de la precisión predictiva.

Cabe destacar que NoGE fue originalmente diseñado para tareas de predicción de enlaces en grafos de conocimiento, y su aplicación aquí ha requerido una adaptación sustancial a un contexto de regresión sobre nodos. Es posible que esta adaptación, aunque funcional, no haya sido suficiente para que el modelo aproveche todo su potencial en el dominio

específico del tráfico urbano. Además, la implementación manual del modelo sin las optimizaciones habituales de librerías como PyTorch Geometric puede haber limitado su rendimiento computacional y su capacidad de ajuste.

En conjunto, los resultados de NoGE reflejan un modelo que aprende algo y generaliza con coherencia, pero que no llega a destacar frente a estrategias más simples. La falta de mejora frente al baseline local, sumada a la complejidad de su diseño y entrenamiento, reduce su atractivo como solución práctica para el problema abordado en este trabajo. Su uso podría explorarse más adelante en contextos con datos más ricos o estructuras más complejas, donde su arquitectura geométrica pueda resultar diferencial.

5.5 DAGI (DEEP AGGREGATION WITH GRAPH ISOMORPHISM AND IMPUTATION)

5.5.1 FUNDAMENTOS DEL MODELO

El modelo DAGI (Deep Aggregation with Graph Isomorphism and Imputation) surge como una arquitectura híbrida que combina la expresividad del modelo GIN (Graph Isomorphism Network) con técnicas de salto de conocimiento (Jumping Knowledge) y un enfoque multitarea centrado en la reconstrucción de atributos. Su diseño responde a la necesidad de obtener representaciones más robustas y generalizables en escenarios de tráfico urbano donde los datos disponibles pueden ser incompletos o escasamente distribuidos. [13]

Base GIN + Jumping Knowledge

DAGI utiliza como núcleo la arquitectura GIN, caracterizada por aplicar operadores convolucionales que imitan la potencia discriminativa del test de isomorfismo de Weisfeiler-Lehman. Esto permite al modelo distinguir eficazmente entre nodos con estructuras locales diferentes. Sin embargo, en lugar de limitarse a una única capa de representación final, DAGI incorpora el mecanismo de Jumping Knowledge (JK), el cual concatena las salidas intermedias de todas las capas convolucionales, ofreciendo así una vista jerárquica y enriquecida del entorno del nodo. Este salto de información favorece la agregación de patrones tanto locales como globales, lo que es especialmente relevante en redes urbanas donde la dinámica del tráfico puede depender de relaciones a múltiples escalas.

Enfoque multitarea: predicción + reconstrucción

Una de las singularidades más importantes de DAGI es su enfoque multitarea. Además de la tarea principal de predicción del volumen de tráfico, el modelo incorpora una tarea secundaria de reconstrucción. Esta segunda tarea intenta predecir (reconstruir) características del nodo a partir de su representación latente, lo cual fuerza al modelo a capturar de forma más informativa los patrones internos de los datos.

El componente de reconstrucción se implementa mediante un segundo bloque de capas convolucionales que toma como entrada la salida concatenada generada por Jumping Knowledge. Este mecanismo impulsa una representación más densa y semánticamente significativa de los nodos, útil incluso para nodos no observados directamente durante el entrenamiento.

Output global agregado

Por último, la arquitectura incluye una operación de pooling global (global mean pooling) sobre la representación de los nodos, que se utiliza como entrada final para la predicción. Esta agregación permite sintetizar la información a nivel de grafo, lo que es especialmente útil en contextos donde se desea obtener una salida supervisada a partir de un grafo completo con características parcialmente observadas.

En conjunto, DAGI representa una propuesta avanzada que aprovecha tanto la estructura del grafo como los atributos de nodos y sus representaciones profundas para tratar de realizar predicciones más precisas y robustas, adaptándose mejor a la heterogeneidad de los datos urbanos reales.

5.5.2 ADAPTACIÓN E IMPLEMENTACIÓN DEL MODELO

La implementación de DAGI se realizó utilizando la librería PyTorch Geometric, tomando como punto de partida una arquitectura modular que combina varias funcionalidades avanzadas de redes neuronales sobre grafos. El modelo fue implementado desde cero y adaptado específicamente al problema de predicción de volumen de tráfico sobre el grafo urbano segmentado de Barcelona.

Estructura general

El modelo se compone de tres bloques principales:

- **Bloque convolucional principal**: Se utilizó una pila de capas GINConv, cada una con su correspondiente MLP interno, seguida de activación ReLU y normalización BatchNorm. Este bloque genera representaciones sucesivas de cada nodo.

- **Jumping Knowledge (JK)**: Las salidas de todas las capas convolucionales intermedias se almacenan y se combinan mediante concatenación (modo "cat"), generando así una representación latente ampliada que captura información multiescala.
- **Cabezal de reconstrucción (decodificador)**: Sobre la representación final obtenida tras JK se aplicó un segundo conjunto de capas convolucionales GIN adicionales, que tienen como objetivo reconstruir los valores originales de los nodos. Esta reconstrucción se interpreta como una tarea de regularización que fuerza al modelo a preservar información útil en el embedding.

El entrenamiento se realizó con pérdida MSE sobre la predicción del volumen, mientras que la reconstrucción actuó como salida secundaria adicional.

Particularidades del diseño

- **Separación de entradas observadas**: El modelo solo recibe como entrada los atributos conocidos del nodo, es decir, no se le entrega directamente el volumen objetivo de testeo. Para asegurar una evaluación justa, se aplicaron máscaras de entrenamiento y test en cada repetición.
- **Reconstrucción como output adicional**: El modelo no solo genera la predicción final del volumen mediante “global mean pool”, sino también una reconstrucción a nivel de nodo que permite comparar la salida con el valor original conocido, en una tarea de imputación auxiliar.

5.5.3 RESULTADOS

A continuación, se presenta la tabla resumen con el MAE medio obtenido en cada nivel de test:

Tabla 8: MAE DAGI Resultados

Test	Mean MAE
2%	1212
4%	1194
6%	1113
8%	1134
10%	1117
12%	1101
14%	1124
16%	1100
18%	1094
20%	1120
22%	1137
24%	1106
26%	1112
28%	1123
30%	1144

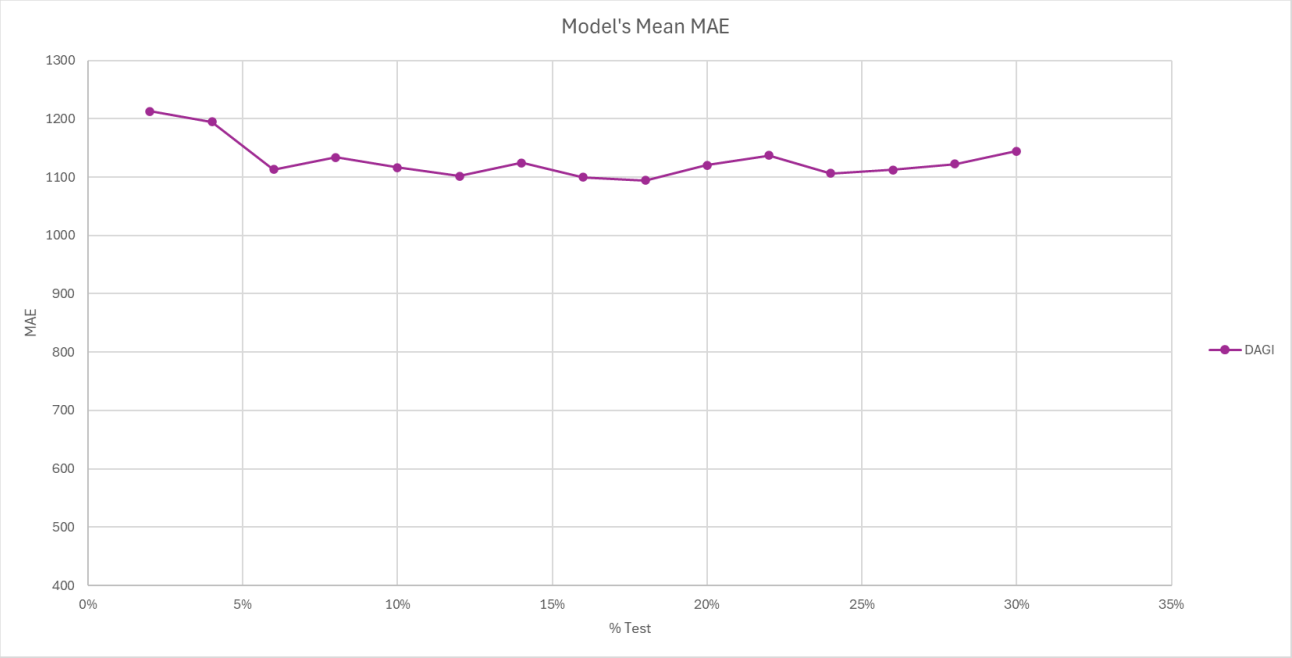


Figura 22: MAE del modelo DAGI

El modelo DAGI mostró un rendimiento competitivo dentro del conjunto de modelos evaluados, con valores de MAE que se situaron mayoritariamente en el rango de 1090 a 1210. Aunque sus resultados fueron algo más variables que en otros casos, en general mantuvo una precisión razonable incluso en niveles altos de ocultación. En algunas configuraciones puntuales, logró aproximarse al rendimiento del baseline local, pero en la mayoría de los casos no lo superó de forma sistemática.

El enfoque multitarea de DAGI, que combina predicción y reconstrucción de atributos, así como el uso del mecanismo Jumping Knowledge para integrar representaciones de distintas capas, apunta a una arquitectura potente y versátil. Sin embargo, los resultados sugieren que esta mayor complejidad estructural no se traduce necesariamente en un mejor rendimiento predictivo en el contexto específico de este trabajo. Es probable que, al igual que ocurre con otros modelos más expresivos, el escaso número de atributos de entrada y la relativa simplicidad de la información disponible hayan limitado su capacidad para explotar todo su potencial.

La variabilidad de los resultados también podría estar relacionada con la sensibilidad del modelo a los splits aleatorios o a la presencia de ciertos nodos clave en el conjunto de entrenamiento. Aunque no se observaron signos claros de sobreajuste, sí se detectaron ligeras oscilaciones no monótonas del MAE a medida que se modificaba el porcentaje de test, lo que sugiere que el modelo puede beneficiarse de ajustes adicionales en su regularización o en la configuración de su tarea secundaria de reconstrucción.

En conjunto, DAGI se sitúa en una posición intermedia: por un lado, muestra un rendimiento aceptable y una arquitectura teóricamente sólida; por otro, no consigue mejorar de forma clara los métodos más simples, como el baseline local. Esto sugiere que su aplicabilidad podría ser más prometedora en escenarios con mayor riqueza de datos o con estructuras más heterogéneas, donde la información multiescala y la reconstrucción auxiliar puedan aportar un valor adicional más tangible.

Capítulo 6. ANÁLISIS DE RESULTADOS

6.1 COMPENDIO DE RESULTADOS

Para facilitar la comparación entre los distintos modelos evaluados, se han representado gráficamente los valores medios del error absoluto medio (MAE) obtenidos en cada experimento, en función del porcentaje de nodos sin información (es decir, aquellas vías de tráfico no monitorizadas). Así se puede observar de forma clara las diferencias de entre modelos a lo largo del rango del test, que varía desde el 2 % hasta el 30 %.

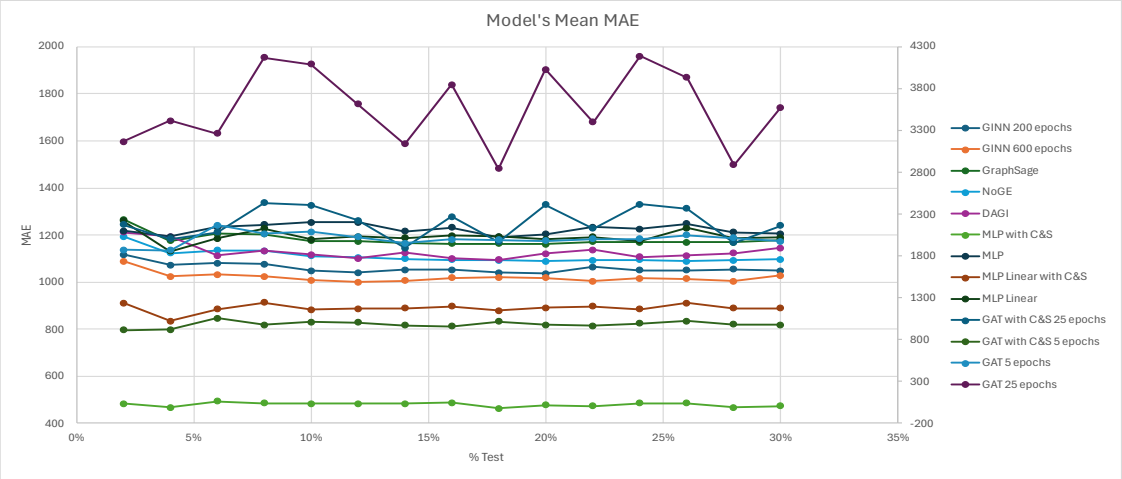


Figura 24: Gráfica comparativa del MAE de los modelos

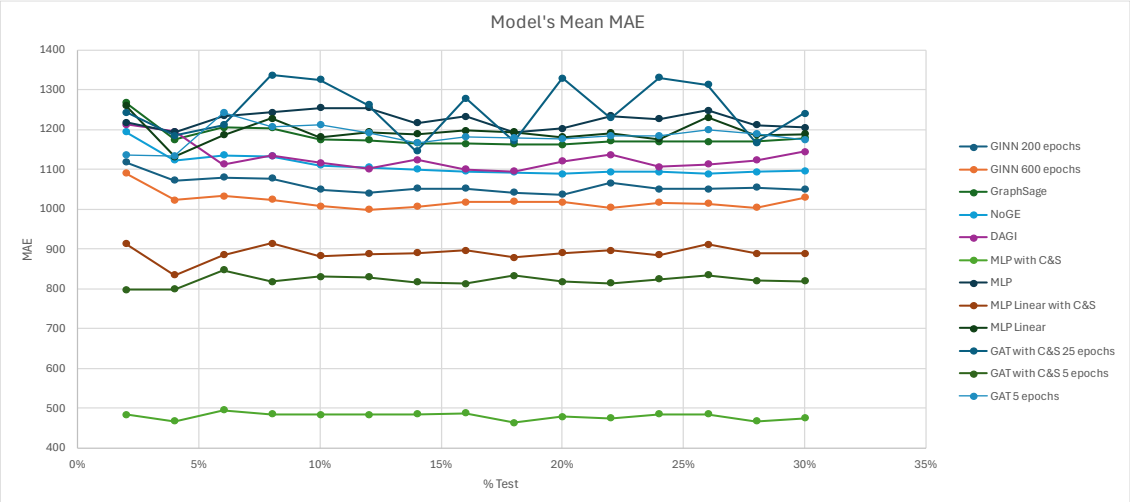


Figura 25: Gráfica comparativa del MAE de los modelos sin GAT 25

Tabla 9: Comparativa, MAE de los modelos

Test	GINN (200 epochs) Mean MAE	GINN (600 epochs) Mean MAE	GraphSage Mean MAE	NoGE Mean MAE	DAGI Mean MAE	C&S MLP Mean MAE		C&S MLP Linear Mean MAE		C&S GAT 25 epochs Mean MAE		C&S GAT 5 epcohs Mean MAE	
						After C&S	Before C&S	After C&S	Before C&S	After C&S	Before C&S	After C&S	Before C&S
2%	1117	1089	1267	1194	1212	483	1217	912	1259	1243	3166	797	1137
4%	1072	1023	1175	1122	1194	467	1194	833	1131	1184	3416	798	1134
6%	1080	1033	1205	1136	1113	494	1234	885	1186	1211	3265	846	1242
8%	1077	1024	1203	1133	1134	485	1243	914	1228	1337	4170	818	1206
10%	1049	1007	1175	1110	1117	483	1255	882	1181	1326	4091	830	1213
12%	1040	999	1173	1105	1101	483	1255	887	1194	1262	3614	829	1192
14%	1052	1006	1165	1099	1124	484	1216	890	1188	1145	3137	817	1166
16%	1052	1017	1164	1095	1100	487	1232	896	1198	1278	3845	813	1181
18%	1041	1019	1163	1092	1094	462	1192	878	1193	1171	2846	832	1180
20%	1037	1017	1162	1088	1120	478	1203	890	1181	1329	4032	818	1176
22%	1065	1004	1171	1093	1137	474	1234	896	1192	1229	3397	814	1184
24%	1051	1016	1169	1094	1106	485	1226	884	1176	1330	4188	823	1184
26%	1051	1014	1170	1089	1112	485	1248	911	1230	1312	3932	834	1199
28%	1054	1004	1170	1093	1123	467	1211	889	1186	1167	2888	820	1188
30%	1049	1029	1178	1096	1144	474	1205	889	1189	1241	3576	819	1174

Tabla 10: MAE de los Baselines

Test	Mean MAE Local Baseline	Mean MAE Global Baseline
2%	991	1155
4%	970	1192
6%	988	1180
8%	964	1224
10%	995	1195
12%	993	1162
14%	984	1187
16%	983	1191
18%	981	1183
20%	988	1196
22%	1010	1199
24%	1008	1187
26%	1002	1186
28%	1003	1194
30%	989	1178

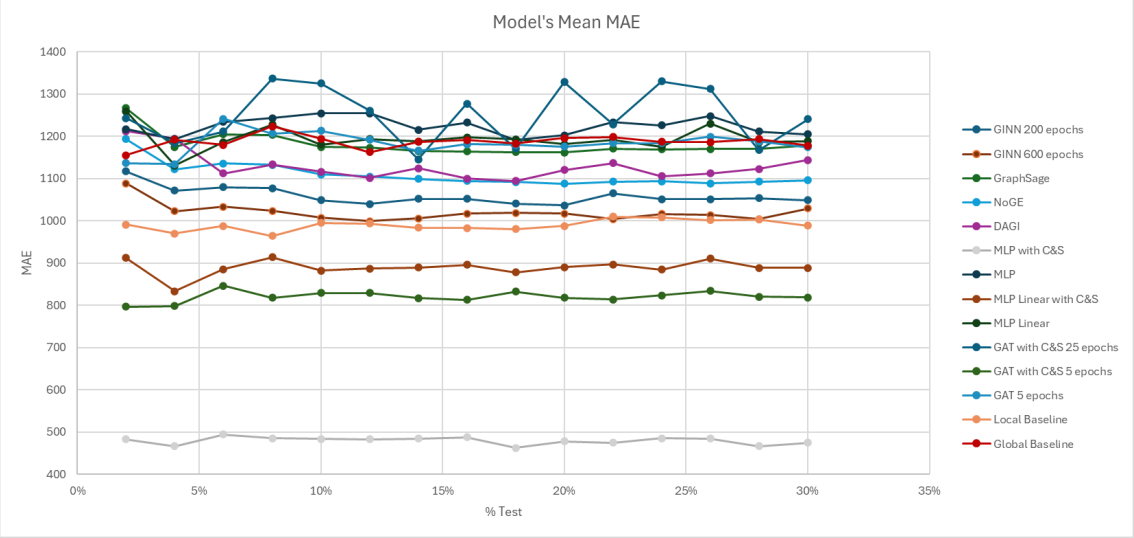


Figura 25: Tabla Comparativa modelos y baselines sin GAT 25 epochs

Nota: El modelo GAT 25 epochs sin C&S, se ha eliminado de varias gráficas para permitir una mejor visualización del resto de modelos, ya que el error de este era muy superior al del resto.

En primer lugar, se aprecia de forma evidente la superioridad del modelo Correct and Smooth (C&S) cuando se combina con un MLP como predictor base. Su curva de MAE se sitúa por debajo del resto de modelos de forma consistente en todos los niveles de ocultación, lo que confirma su robustez y capacidad para generalizar incluso con cantidades limitadas de datos conocidos. La pendiente de su evolución es suave, y los valores obtenidos mantienen un margen de mejora sustancial respecto a los baselines.

En el extremo opuesto, se encuentran los modelos GraphSAGE, GINN, NoGE y DAGI, cuyas curvas se sitúan próximas o incluso por encima de la línea correspondiente al baseline local. Esto implica que, pese a incorporar estructuras complejas o arquitecturas especializadas, no lograron en este caso superar de forma sistemática a una simple imputación por media de vecinos. En particular, GraphSAGE mostró una evolución prácticamente plana que coincide con el baseline global en algunos tramos y apenas mejora el local, lo que sugiere una escasa capacidad de aprendizaje útil en este contexto.

En cuanto a GINN y DAGI, sus resultados fueron más competitivos y con cierta capacidad de adaptación, pero con un margen de mejora limitado frente a los métodos de referencia. NoGE, por su parte, destacó por su estabilidad y consistencia, pero tampoco ofreció mejoras claras en términos de error medio.

Además de las gráficas comparativas, se incluye una tabla resumen con los principales hiperparámetros utilizados en cada modelo, así como el tipo de grafo y formato en cada caso. Esta información permite identificar fácilmente los factores como la arquitectura del modelo, el número de capas, la tasa de aprendizaje o la estructura del grafo pueden haber influido en el comportamiento observado. La tabla sirve también como referencia técnica para futuras implementaciones, así como futuros trabajos o réplicas del experimento.

Tabla 11: Resumen hiperparámetros de los modelos

Hiperparameters	GraphSAGE	GINN	NoGE	DAGI	C&S + MLP	C&S + MLPLinear	C&S + GAT
Tipo de grafo	Grafo de segmento	Grafo de segmento	Grafo de segmento	Grafo de segmento	Grafo de segmento	Grafo de segmento	Grafo de segmento
Formato del grafo	NetworkX	NetworkX	NetworkX	NetworkX	DGL	DGL	DGL
Épocas (epochs)	100	200 / 600	300	1000	100	100	25 / 5
Dimensión oculta	16	32	16	16	224	–	234
Número de capas	2	2	2	2 + rec	4	1	4
Learning rate (LR)	0.01	0.01	0.01	0.01	0.1	0.05	0.05
Dropout	–	–	–	0.5	0.25	–	0.5
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	RMSProp
Función de pérdida	MSE	MSE	MSE	MSE	MSE	MSE	MSE
C&S aplicado	No	No	No	No	Sí	Sí	Sí
Detalles C&S	–	–	–	–	46/45, DAD, esc=20	70/70, AD, esc=20	70/70, AD, esc=20
Extras	Añade label_known y label_value como features	Añade label_known y label_value como features	Usa Dual Quaternion GNN (QGNN)	Con Jumping Knowledge + reconstrucción	–	–	GAT con 4 heads y attn_drop=0.03

6.2 COMPARACIÓN CUANTITATIVA ENTRE ALGORITMOS

Desde un punto de vista estrictamente cuantitativo, el modelo que ha ofrecido un rendimiento más sólido y generalizable ha sido el sistema compuesto por un perceptrón multicapa (MLP) seguido del postprocesado Correct and Smooth (C&S). Este enfoque ha logrado mantener el MAE por debajo de 500 unidades incluso cuando se ocultaba hasta un 30 % de los datos, lo cual supone una mejora muy significativa respecto a todos los demás métodos evaluados. Además, ha demostrado ser robusto frente al incremento del volumen de datos ocultos, con una evolución suave y controlada del error.

En términos absolutos, las variantes de C&S aplicadas a MLP Linear y GAT también mostraron mejoras notables respecto a sus predicciones iniciales. Sin embargo, partían de errores más elevados, especialmente en el caso del GAT entrenado durante demasiadas épocas, lo que acabó generando sobreajuste. Aun así, tras la corrección con C&S, ambas configuraciones lograron reducir el MAE a niveles claramente por debajo de los

baselines, lo que confirma la eficacia del postprocesado independientemente del modelo base.

Por el contrario, los modelos GraphSAGE y GINN, aunque más elaborados a nivel arquitectónico, no consiguieron superar de manera consistente al baseline local. Sus resultados se mantuvieron en torno a los 1120–1175 de MAE, es decir, muy próximos o incluso superiores al error obtenido con una imputación simple basada en la media de vecinos. Este hecho cuestiona su capacidad de aprendizaje efectivo en este escenario concreto, o al menos evidencia que no han sido capaces de aprovechar suficientemente la estructura del grafo con la información de entrada disponible.

El modelo NoGE, a pesar de su complejidad matemática y de su propuesta geométrica basada en cuaterniones, mostró un rendimiento similar al de GINN, con una ligera ventaja en términos de estabilidad, pero sin mejoras sustanciales respecto a los baselines. Algo similar puede decirse del modelo DAGI, que, aunque integraba mecanismos como Jumping Knowledge y tareas auxiliares de reconstrucción, no logró diferenciarse de forma clara del baseline local ni justificar el coste computacional adicional.

En conjunto, la comparación numérica entre modelos confirma que la complejidad arquitectónica no garantiza una mejor capacidad predictiva si no se dispone de información suficiente o adecuadamente representada. En este caso, las técnicas basadas en propagación de errores (como C&S) han resultado ser más eficaces que arquitecturas más profundas o expresivas, debido posiblemente a la naturaleza estática del problema y a la limitada disponibilidad de atributos por nodo.

6.3 ANÁLISIS DE SENSIBILIDAD

Una parte fundamental del análisis de resultados consiste en evaluar hasta qué punto el rendimiento de cada modelo se ve afectado por las condiciones del entorno, especialmente por el grado de información disponible en el grafo. En este trabajo, dicha variación se ha simulado a través de distintos porcentajes de nodos sin información de volumen de tráfico, lo que permite observar la sensibilidad de cada modelo a escenarios con mayor o

menor monitorización. Este enfoque es especialmente relevante en contextos urbanos reales, donde los recursos para monitorear son limitados y es habitual disponer solo de una fracción reducida de datos.

El análisis de sensibilidad permite, por tanto, ir más allá de la comparación puntual de MAEs y centrarse en la estabilidad del modelo: cómo evoluciona su precisión a medida que se reduce la cantidad de información disponible. En este sentido, los modelos más robustos no son necesariamente los que obtienen el mejor resultado absoluto, sino aquellos cuyo del rendimiento allá sido más controlado y estable.

6.3.1 INFLUENCIA DEL PORCENTAJE DE CALLES NO MONITORIZADAS

Los resultados muestran que la mayoría de los modelos mantienen una evolución relativamente estable del MAE conforme se incrementa el porcentaje de nodos ocultos. Sin embargo, existen diferencias relevantes entre ellos en cuanto a la pendiente y regularidad de esa evolución.

El modelo C&S combinado con MLP destaca por su baja sensibilidad al aumento de nodos no observados. Su MAE crece de forma muy moderada, lo que indica una gran capacidad de generalización incluso cuando se dispone de poca información. Este comportamiento contrasta con el de modelos como GraphSAGE o GINN, en los que el incremento del MAE es más acusado a partir de ciertos umbrales (por ejemplo, del 10 % al 20 %), lo que refleja una menor capacidad para mantener la precisión en entornos con alta incertidumbre.

En el caso de NoGE y DAGI, se observa una evolución más irregular. Aunque en general mantienen una cierta estabilidad, presentan oscilaciones puntuales que podrían deberse a la sensibilidad del modelo a la posición específica de los nodos ocultos en cada repetición. Esto sugiere que su rendimiento puede depender en mayor medida de la distribución concreta de la información conocida en el grafo.

6.3.2 IMPACTO DE LA TOPOLOGÍA Y TIPO DE GRAFO

Otro factor determinante en el rendimiento de los modelos es la forma en que se representa la red urbana dentro del grafo, así como el formato computacional utilizado

para procesarlo. A lo largo del proyecto se han empleado distintas versiones del grafo de Barcelona, adaptadas a las necesidades de cada modelo, lo que permite evaluar hasta qué punto la topología elegida y el tipo de representación influyen en los resultados obtenidos.

En términos generales, se tenía principalmente tres estructuras de grafo: el grafo original (intersecciones como nodos), el grafo de segmentos (cada tramo como nodo) y el grafo extendido (donde se añaden nodos intermedios para transformar tareas sobre aristas en tareas sobre nodos).

Los modelos basados en PyTorch Geometric (Networkx), como GraphSAGE, GINN, DAGI y NoGE, trabajaron directamente sobre el grafo de segmentos, donde los nodos representan tramos de calle. Este enfoque permitió asociar directamente atributos como volumen de tráfico y longitud a cada entidad predecible, facilitando las tareas de imputación. No obstante, este tipo de grafo introduce una mayor complejidad topológica, ya que las conexiones entre nodos no siempre corresponden a relaciones funcionales claras (como entrada/salida en una intersección), lo que puede dificultar el aprendizaje estructural para modelos sensibles a la orientación de los enlaces.

Por otro lado, los modelos que utilizaban DGL como librería base, como es el caso de C&S, trabajaron también sobre una versión segmentada, pero adaptada específicamente a la arquitectura del pipeline de predicción y postprocesado. El éxito de estos modelos sugiere que, además de la forma del grafo, resulta fundamental conseguir una buena compatibilidad entre la estructura de entrada y las capacidades de cada modelo, tanto a nivel de conectividad como de atributos asociados.

Es importante destacar que los modelos que partían del grafo original no fueron finalmente empleados en los ensayos comparativos, debido a que la tarea se planteó como una predicción sobre tramos concretos y no sobre intersecciones. No obstante, esta opción podría explorarse en trabajos futuros.

En cuanto al impacto del tipo de grafo en el rendimiento, no se observan diferencias drásticas atribuibles únicamente a la estructura topológica. Aun así, es muy probable que el preprocesado y la selección adecuada de la representación del grafo sean decisiones críticas que pueden condicionar por completo el éxito o el fracaso de un modelo.

Capítulo 7. CONCLUSIONES Y FUTUROS TRABAJOS

7.1 CONCLUSIONES

7.1.1 PRINCIPALES CONCLUSIONES DEL PROYECTO

Este trabajo ha servido para comprobar que es posible estimar el tráfico en calles no monitorizadas utilizando modelos basados en grafos, incluso cuando solo se dispone de datos en una parte de la red. Se han probado varios enfoques, y el que ha dado mejores resultados ha sido la combinación de un perceptrón multicapa (MLP) con la técnica de postprocesado Correct and Smooth (C&S). Este sistema ha conseguido mantener errores bajos en todos los niveles de ocultación de datos, demostrando que puede generalizar bien incluso cuando hay poca información.

En cambio, otros modelos más complejos, como DAGI o NoGE, no han logrado mejorar de forma clara respecto a soluciones más simples como los baselines. Tampoco modelos conocidos como GraphSAGE o GINN han destacado especialmente. Esto sugiere que, en este caso concreto, tener una arquitectura sofisticada no garantiza mejores resultados si no se acompaña de datos suficientes o bien preparados.

También se ha visto que el modo en que se representa el grafo tiene un impacto importante. Aquellos modelos que trabajaban con grafos adaptados a sus necesidades (como el grafo de segmentos o extendido) funcionaron mejor que si se hubiera usado directamente el grafo original. Preparar bien los datos es clave para que los modelos puedan llegar a ser eficaces.

7.1.2 LIMITACIONES ENCONTRADAS

Una de las principales limitaciones del estudio ha sido la poca cantidad de información disponible por nodo. La mayoría de los modelos solo han podido trabajar con uno o dos atributos simples, lo que reduce mucho su capacidad para detectar patrones complejos. También es importante señalar que el tráfico se ha tratado aquí como un fenómeno

estático, sin considerar la dimensión temporal. En la realidad, el tráfico cambia a lo largo del día y la semana, y este modelo no lo tiene en cuenta.

Otra limitación es que solo se ha utilizado un grafo, el de Barcelona. Aunque es suficientemente grande y realista, habría sido útil probar en otras ciudades con distintas estructuras para comprobar si los modelos generalizan bien. Además, algunos de los modelos más avanzados requerían muchos recursos de cómputo y eran difíciles de ajustar, lo que puede ser un problema si se busca una solución sencilla, barata y rápida de aplicar.

Además, cabe destacar como limitación la escasez de atributos de los nodos, al final la mayoría de los modelos basados en redes neuronales, aprenden gracias a una gran cantidad de información. Y por lo tanto al tener información más escasa se puede ver como estos modelos no han sido capaces de aprender lo suficiente como para llegar a ser viables.

7.1.3 VALIDACIÓN DE LA HIPÓTESIS INICIAL

La hipótesis inicial planteaba que se podía estimar de forma razonablemente precisa el tráfico en vías no monitorizadas usando modelos sobre grafos y un conjunto reducido de datos conocidos. Los resultados, especialmente los del modelo MLP con C&S, confirman esta idea. Incluso con solo un 70 % de datos disponibles, se obtuvieron errores bastante bajos. Y demostraron la eficacia que puede tener C&S, como postprocesado de una predicción.

Eso sí, no todos los modelos probados han funcionado igual de bien. Algunos apenas han mejorado respecto a los baselines global y local, lo que demuestra que no basta con aplicar cualquier arquitectura: hace falta que el modelo se ajuste bien al tipo de datos y al problema concreto. En definitiva, este trabajo ha permitido validar la hipótesis, pero también ha ayudado a entender mejor qué modelos funcionan y por qué, lo cual es útil para futuras aplicaciones y mejoras.

7.2 FUTUROS TRABAJOS

El presente estudio ha permitido validar distintas arquitecturas de aprendizaje sobre grafos para la estimación de tráfico en redes urbanas, empleando una representación estructurada y enriquecida de la ciudad de Barcelona. No obstante, la naturaleza del problema y la evolución de las tecnologías asociadas abren numerosas posibilidades para ampliar este trabajo. A continuación, se detallan algunas líneas de investigación que podrían abordarse en futuras fases, tanto desde una perspectiva científica como práctica.

7.2.1 EXTENSIÓN A DATOS TEMPORALES (PREDICCIÓN HORARIA)

Una de las limitaciones del enfoque actual es la ausencia de una dimensión explícita temporal. En aplicaciones reales, el tráfico presenta patrones cíclicos diarios y semanales que pueden capturarse mediante modelos espaciotemporales. La incorporación de secuencias temporales en los nodos o aristas permitiría construir modelos más robustos, capaces de predecir la evolución del tráfico a distintas horas del día o incluso a corto plazo. Esto implicaría integrar series temporales, recurrencia o incluso modelos basados en GNNs temporales como ST-GCN o T-GAT.

7.2.2 APLICACIÓN A NUEVOS GRAFOS URBANOS REALES

Aunque el presente trabajo se ha centrado en una única ciudad, la metodología es perfectamente transferible a otras redes urbanas. Como línea futura, sería interesante aplicar los modelos a nuevas ciudades (como Madrid, París o Medellín), comparando su rendimiento bajo distintas topologías, niveles de congestión y hábitos de movilidad. Esto permitiría validar la generalización de las arquitecturas y analizar cómo influyen las características estructurales del grafo en la precisión predictiva.

7.2.3 INTEGRACIÓN CON SISTEMAS URBANOS INTELIGENTES

El desarrollo de ciudades inteligentes requiere sistemas que no solo predigan el tráfico, sino que interactúen con otras fuentes de datos en tiempo real, como sensores IoT, cámaras, o plataformas de movilidad compartida. Una posible ampliación consistiría en

integrar el modelo predictivo dentro de una infraestructura urbana conectada, utilizando flujos de datos en vivo para actualizar predicciones en tiempo real o asistir en la toma de decisiones, como la optimización semafórica o la gestión de flotas.

7.2.4 ESTIMACIÓN DE MAPAS DE EMISIONES URBANOS

Dado que el volumen de tráfico está estrechamente relacionado con las emisiones contaminantes, otro desarrollo interesante sería combinar las predicciones actuales con modelos de conversión a emisiones de CO_2 o NO_x. Esto permitiría estimar de forma indirecta los niveles de contaminación a lo largo del grafo, generando mapas de emisiones dinámicos y localizados. Estos mapas serían de gran valor para políticas de movilidad sostenible, zonas de bajas emisiones o incentivos medioambientales.

7.2.5 EVALUACIÓN ÓPTIMA DE QUÉ VÍAS CONVIENE MONITORIZAR

Una aplicación práctica de gran interés sería determinar en qué calles o tramos resulta más rentable instalar sensores de tráfico, considerando tanto el coste como la ganancia en precisión del modelo. Se podrían explorar métodos de active learning o algoritmos de optimización para seleccionar subconjuntos de nodos que, al ser monitorizados, minimicen el error total de predicción en el grafo. Esta línea abre la puerta a modelos de observabilidad urbana eficiente y a decisiones informadas sobre inversión en infraestructura.

Capítulo 8. BIBLIOGRAFÍA

[1] «Network-Wide Vehicle Trajectory Prediction in Urban Traffic Networks using Deep Learning», ResearchGate, doi: 10.1177/0361198118794735

[2] E. Bayram, A. Garcia-Duran, y R. West, «Node Attribute Completion in Knowledge Graphs with Multi-Relational Propagation», 10 de noviembre de 2020, arXiv: arXiv:2011.05301. doi: 10.48550/arXiv.2011.05301

[3] W. Feng et al., «Graph Random Neural Network for Semi-Supervised Learning on Graphs», 21 de septiembre de 2021, arXiv: arXiv:2005.11079. doi: 10.48550/arXiv.2005.11079

[4] S. Shirakawa, S. Ogino, y T. Nagao, «Graph structured program evolution», en Proceedings of the 9th annual conference on Genetic and evolutionary computation, en GECCO '07. New York, NY, USA: Association for Computing Machinery, jul. 2007, pp. 1686-1693. doi: 10.1145/1276958.1277290

[5] «Forecasting Network Data Spatial Interpolation of Traffic Counts from Texas Data», ResearchGate, doi: 10.3141/2105-13

[6] R. K. C. Chan, J. M.-Y. Lim, y R. Parthiban, «A neural network approach for traffic prediction and routing with missing data imputation for intelligent transportation system», Expert Syst. Appl., vol. 171, p. 114573, jun. 2021, doi: 10.1016/j.eswa.2021.114573

[7] «Traffic Estimation for Proactive Freeway Traffic Control». Disponible en: https://journals.sagepub.com/doi/epdf/10.3141/1679-11

[8] Y. Djenouri, A. Belhadi, G. Srivastava, y J. C.-W. Lin, «Hybrid graph convolution neural network and branch-and-bound optimization for traffic flow forecasting», Future Gener. Comput. Syst., vol. 139, pp. 100-108, feb. 2023, doi: 10.1016/j.future.2022.09.018

[9] D. Zhou, O. Bousquet, T. Lal, J. Weston, y B. Schölkopf, «Learning with Local and Global Consistency», en Advances in Neural Information Processing Systems, MIT Press, 2003.Disponible:https://proceedings.neurips.cc/paper_files/paper/2003/hash/87682805257e619d49b8e0dfdc14affa-Abstract.html

[10] A. Moreno Grande, «Estimación de tráfico en vías no monitorizadas mediante el uso de grafos y redes neuronales». 2 de junio de 2023

[11] Q. Huang, H. He, A. Singh, S.-N. Lim, y A. R. Benson, «Combining Label Propagation and Simple Models Out-performs Graph Neural Networks», 2 de noviembre de 2020, arXiv: arXiv:2010.13993. doi: 10.48550/arXiv.2010.13993

[12] J. Liu, G. P. Ong, y X. Chen, «GraphSAGE-Based Traffic Speed Forecasting for Segment Network With Sparse Data», IEEE Trans. Intell. Transp. Syst., vol. 23, n.o 3, pp. 1755-1766, mar. 2022, doi: 10.1109/TITS.2020.3026025

[13] Y. Wang, W. Peng, S. F. Tapert, Q. Zhao, y K. M. Pohl, «Imputing Brain Measurements Across Data Sets via Graph Neural Networks», 19 de agosto de 2023, arXiv: arXiv:2308.09907. doi: 10.48550/arXiv.2308.09907

[14] bstabler, Repositorio Github bstabler. [En línea]. Disponible en: https://github.com/bstabler/TransportationNetworks

[15] L. F. Sánchez Merchante, Repositorio Github lmerchante. [En línea]. Disponible en: https://github.com/lmerchante

[16] TensorFlow GNN. [En línea]. Disponible en: https://www.tensorflow.org/gnn

[17] «Papers with Code - Node Co-occurrence based Graph Neural Networks for Knowledge Graph Link Prediction».Disponible en: https://paperswithcode.com/paper/node-co-occurrence-based-graph-neural7

[18] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful are Graph Neural Networks?" in Proc. Int. Conf. Learn. Representations (ICLR), 2019

[19] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 1024–1034

[20] G.-L. Chang, et al., "Traffic forecasting using neural networks," Transportation Research Record, no. 1678, pp. 93–104, 1999

[21] S. Ishak, S. Kothuri, and C. Alecsandru, "Performance evaluation of short-term traffic prediction models," Journal of Transportation Engineering, vol. 129, no. 6, pp. 664–672, 2003

[22] J. Choi and D. Kim, "Deep Learning-Based Vehicle Trajectory Prediction with Recurrent Neural Networks," Sensors, vol. 18, no. 12, pp. 1–15, 2018