UNIVERSIDAD NACIONAL DEL LITORAL
DOCTORADO EN INGENIERÍA

Desarrollo de nuevas arquitecturas neuronales en grafos para la inferencia de propiedades de redes biológicas

Eugenio Borzone

Facultad de Ingeniería y Ciencias Hídricas
Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional

Tesis de Doctorado **2025**

Doctorado en Ingeniería
mención en Inteligencia Computacional Señales y Sistemas

Título de la obra:

Desarrollo de nuevas arquitecturas neuronales en grafos para la inferencia de propiedades de redes biológicas

Autor: Eugenio Borzone

Lugar: Santa Fe, Argentina

Palabras clave:
Redes Neuronales en Grafos, Tareas Centradas en Aristas, Mecanismos de Atención, Aprendizaje Auto-Supervisado

UNIVERSIDAD NACIONAL DEL LITORAL
Facultad de Ingeniería y Ciencias Hídricas
Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional

Desarrollo de nuevas arquitecturas neuronales en grafos para la inferencia de propiedades de redes biológicas

Eugenio Borzone

Tesis remitida al Comité Académico del Doctorado
como parte de los requisitos para la obtención
del grado de
DOCTOR EN INGENIERÍA
Mención en Inteligencia Computacional Señales y Sistemas
de la
UNIVERSIDAD NACIONAL DEL LITORAL
2025

Secretaría de Posgrado, Facultad de Ingeniería y Ciencias Hídricas, Ciudad Universitaria,
Paraje "El Pozo", S3000, Santa Fe, Argentina

UNIVERSIDAD NACIONAL DEL LITORAL
Facultad de Ingeniería y Ciencias Hídricas
Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional

Desarrollo de nuevas arquitecturas neuronales en grafos para la inferencia de propiedades de redes biológicas

Eugenio Borzone

Lugar de Trabajo:
sinc(i) - Instituto de Investigación en Señales, Sistemas e Inteligencia Computacional

Facultad de Ingeniería y Ciencias Hídricas - UNIVERSIDAD NACIONAL DEL LITORAL

Director: Matías Gerard, sinc(i)
Co-director: Leandro Di Persia, sinc(i)

ACTA DE DEFENSA DE TESIS DE DOCTORADO

En la sede de la Facultad de Ingeniería y Ciencias Hídricas de la Universidad Nacional del Litoral, a los veinte días del mes de julio del año dos mil veintiséis, se reúnen en forma online sincrónica los miembros del Jurado designado para la evaluación de la Tesis de Doctorado en Ingeniería, Mención "Inteligencia Computacional, Señales y Sistemas" titulada ***"Desarrollo de nuevas arquitecturas neuronales en grafos para la inferencia de propiedades de redes biológicas."***, desarrollada por el Ing. Eugenio BORZONE, DNI: 38.898.056, bajo la dirección del Dr. Matías Gerard y la codirección del Dr. Leandro Di Persia. Ellos son: Dr. Ignacio Ponzoni, el Dr. Pablo Granitto, la Dra. Ana Carolina Olivera y el Dr. Flavio Spetale.----------

La Presentación oral y defensa de la Tesis se efectúan bajo la modalidad online sincrónica según lo establecido por Resolución CS N° 382/21.

Luego de escuchar la Defensa Pública y de evaluar la Tesis, el Jurado considera:
Que la tesis está claramente escrita y presenta una cantidad importante de aportes tanto en la metodología planteada como en el área de aplicación.
Que los resultados obtenidos son destacables y han sido adecuadamente validados en base a los datos disponibles y que los trabajos futuros propuestos son relevantes para el estudio de redes biológicas.
Que la presentación fue dinámica con un nivel adecuado de profundidad y que respondió satisfactoriamente a todas las preguntas del jurado.
Por lo tanto, el jurado aprueba la tesis con calificación 10 (Diez) Sobresaliente.

Sin más, se da por finalizado el Acto Académico con la firma de los miembros del Jurado al pie de la presente. ----------

Dr. Ignacio Ponzoni	Dr. Pablo Granitto	Dra. Ana Carolina Olivera

Dr. Flavio Spetale

FACULTAD DE INGENIERÍA Y CIENCIAS HÍDRICAS * SANTA FE *

Dr. José Luis Macor
Director de Posgrado
FICH - UNL

Universidad Nacional del Litoral
Facultad de Ingeniería y
Ciencias Hídricas

Secretaría de Posgrado

Ciudad Universitaria
C.C. 217
Ruta Nacional Nº 168 – Km. 472,4
(3000) Santa Fe
Tel: (54) (0342) 4575 229
Fax: (54) (0342) 4575 224
E-mail: posgrado@fich.unl.edu.ar

DEDICATORIA

A mis padres, por su amor incondicional, su paciencia infinita y por enseñarme, desde siempre, a confiar en mis sueños. Esta tesis es también de ustedes.

Agradecimientos

En primer lugar, este trabajo no hubiera sido posible sin el amor, esfuerzo y sacrificio de mis padres, Daniela Serrao y Rogelio Borzone, quienes han sido mi apoyo incondicional en cada etapa de mi vida. Su dedicación y confianza en mí han sido la base sobre la cual he podido construir este camino. Les agradezco profundamente por enseñarme el valor del esfuerzo y la perseverancia.

A mis directores, Matías Gerard y Leandro Di Persia, expreso mi más sincero reconocimiento por su guía, paciencia y compromiso. Su experiencia y generosidad intelectual han sido fundamentales en mi formación y en la realización de esta investigación. Gracias por compartir su pasión por la ciencia y por inspirarme a ser un mejor investigador.

A mis amigos, quienes han llenado mis días de alegría y han sido un pilar fundamental para mantener el equilibrio y la motivación en los momentos más desafiantes. Su compañía y palabras de aliento siempre serán invaluables.

A mis compañeros del instituto y al sinc(i), por crear un entorno colaborativo y estimulante, donde cada día representa una oportunidad para aprender y crecer. A la Universidad Nacional del Litoral, la Facultad de Ingeniería Química y la Facultad de Ingeniería y Ciencias Hídricas, les agradezco por el apoyo y los recursos brindados para llevar a cabo esta investigación.

Mi más profundo agradecimiento al CONICET y a todo el pueblo argentino, cuya contribución y compromiso con la educación y la ciencia han permitido que esta investigación sea posible.

Finalmente, quiero dedicar este logro a todos aquellos que, de manera directa o indirecta, han influido positivamente en mi camino. A ustedes, gracias por creer en mí y por acompañarme en este viaje.

Índice general

Índice de figuras

Índice de tablas

RESUMEN

Las redes neuronales en grafos (GNNs) han demostrado un rendimiento notable en tareas de clasificación y regresión a nivel de nodo y grafo completo. Sin embargo, las tareas centradas en aristas —donde el objetivo es predecir propiedades asociadas a pares de nodos conectados— han recibido comparativamente menos atención, a pesar de su relevancia en aplicaciones bioinformáticas como la estimación de similaridad molecular, la predicción de interacciones proteína-proteína y la anotación funcional de genes. Las arquitecturas tradicionales de GNNs priorizan la agregación de características de nodos, subutilizando la información contenida en las aristas que conectan entidades biológicas.

Esta tesis aborda esta limitación mediante el desarrollo de NEAConv (Node-Edge Attention Convolution), una arquitectura de red neuronal de paso de mensajes específicamente diseñada para tareas centradas en aristas. La contribución principal consiste en un mecanismo de atención que integra simultáneamente características de nodos y aristas mediante una formulación query-key-value inspirada en la arquitectura transformer, permitiendo que el modelo aprenda adaptativamente qué conexiones son más informativas para cada predicción. La arquitectura se complementa con una función de pérdida híbrida que combina aprendizaje supervisado con términos auto-supervisados basados en la similaridad coseno entre embeddings, organizando coherentemente el espacio latente y permitiendo predicciones robustas incluso para entidades con información incompleta o ausente.

El desarrollo metodológico siguió una progresión incremental de complejidad. Inicialmente, se estableció una línea base utilizando perceptrones multicapa con codificación one-hot, alcanzando un MAE de 0.081 (RMSE = 0.1019) en la predicción del coeficiente de Tanimoto entre compuestos metabólicos. La incorporación de embeddings topológicos mediante Node2Vec mejoró el rendimiento en un 8.9 % (MAE = 0.074, RMSE = 0.0928), demostrando que la estructura del grafo metabólico contiene información relevante para la similaridad estructural. Finalmente, la arquitectura NEAConv alcanzó un error absoluto medio de 0.01013 (1.01 % de error), representando una mejora del 87.5 % respecto al enfoque inicial.

Un hallazgo crítico del proceso de optimización fue que el filtrado selectivo del grafo de entrada —reteniendo únicamente las dos aristas de mayor centralidad (betweenness centrality) por nodo— tuvo un impacto mayor sobre el rendimiento que las decisiones arquitecturales convencionales. Este resultado sugiere que, en redes biológicas anotadas, la curaduría estructural del grafo puede ser tan determinante como la sofisticación del modelo, eliminando ruido proveniente de conexiones de baja confiabilidad.

El estudio de ablación demostró que tanto el mecanismo de atención como la fun-

ción de pérdida híbrida son componentes esenciales con efectos complementarios: el modelo baseline alcanzó MAE = 0.01590, mientras que agregar únicamente atención redujo el error a 0.01523 (mejora del 4.2 %), agregar únicamente pérdida híbrida lo redujo a 0.01426 (mejora del 10.3 %), y el modelo completo alcanzó MAE = 0.01013 (mejora del 36.3 %).

La versatilidad de la arquitectura fue validada en dos dominios bioinformáticos adicionales. En predicción de interacciones proteína-proteína, el modelo superó significativamente al estado del arte (test de Friedman $p = 7{,}8 \times 10^{-30}$) en cinco organismos: HPRD (F1 = 0.977), Human (F1 = 0.961), *E. coli* (F1 = 0.967), *Drosophila* (F1 = 0.991) y *C. elegans* (F1 = 0.985). En anotación funcional con Gene Ontology, el modelo igualó el rendimiento del método especializado exp2GO ($p = 0{,}237$) mientras superó significativamente a DeepGOPlus y NMF-GO ($p = 10^{-3}$) en tres organismos y tres subontologías.

En síntesis, esta tesis demuestra que las tareas centradas en aristas en grafos biológicos se benefician de arquitecturas que integran explícitamente información de nodos y aristas mediante mecanismos de atención, entrenadas con funciones de pérdida híbridas que alinean geométricamente el espacio latente con métricas del dominio. Los principios de diseño desarrollados son transferibles entre dominios bioinformáticos distintos, sugiriendo su potencial aplicabilidad más allá de los contextos específicos evaluados.

ABSTRACT

Graph neural networks (GNNs) have demonstrated remarkable performance in node-level and graph-level classification and regression tasks. However, edge-centric tasks —where the objective is to predict properties associated with pairs of connected nodes— have received comparatively less attention, despite their relevance in bioinformatics applications such as molecular similarity estimation, protein-protein interaction prediction, and functional gene annotation. Traditional GNN architectures prioritize node feature aggregation, underutilizing the information contained in edges connecting biological entities.

This thesis addresses this limitation through the development of NEAConv (Node-Edge Attention Convolution), a message-passing neural network architecture specifically designed for edge-centric tasks. The main contribution consists of an attention mechanism that simultaneously integrates node and edge features through a query-key-value formulation inspired by the transformer architecture, enabling the model to adaptively learn which connections are most informative for each prediction. The architecture is complemented by a hybrid loss function that combines supervised learning with self-supervised terms based on cosine similarity between embeddings, coherently organizing the latent space and enabling robust predictions even for entities with incomplete or missing information.

The methodological development followed an incremental progression of complexity. Initially, a baseline was established using multilayer perceptrons with one-hot encoding, achieving an MAE of 0.081 (RMSE = 0.1019) in predicting the Tanimoto coefficient between metabolic compounds. The incorporation of topological embeddings via Node2Vec improved performance by 8.9 % (MAE = 0.074, RMSE = 0.0928), demonstrating that metabolic graph structure contains relevant information for structural similarity. Finally, the NEAConv architecture achieved a mean absolute error of 0.01013 (1.01 % error), representing an 87.5 % improvement over the initial approach.

A critical finding from the optimization process was that selective filtering of the input graph —retaining only the two edges with the highest betweenness centrality per node— had a larger impact on performance than conventional architectural decisions. This result suggests that, in annotated biological networks, structural graph curation may be as determinant as model sophistication, eliminating noise from low-reliability connections.

The ablation study demonstrated that both the attention mechanism and the hybrid loss function are essential components with complementary effects: the baseline model achieved MAE = 0.01590, while adding attention only reduced error to 0.01523 (4.2 % improvement), adding hybrid loss only reduced it to 0.01426 (10.3 % impro-

vement), and the complete model achieved MAE = 0.01013 (36.3 % improvement). The architecture's versatility was validated in two additional bioinformatics domains. In protein-protein interaction prediction, the model significantly outperformed state-of-the-art methods (Friedman test $p = 7{,}8 \times 10^{-30}$) across five organisms: HPRD (F1 = 0.977), Human (F1 = 0.961), *E. coli* (F1 = 0.967), *Drosophila* (F1 = 0.991), and *C. elegans* (F1 = 0.985). In Gene Ontology functional annotation, the model matched the performance of the specialized method exp2GO ($p = 0{,}237$) while significantly outperforming DeepGOPlus and NMF-GO ($p = 10^{-3}$) across three organisms and three sub-ontologies.

In summary, this thesis demonstrates that edge-centric tasks in biological graphs benefit from architectures that explicitly integrate node and edge information through attention mechanisms, trained with hybrid loss functions that geometrically align the latent space with domain metrics. The design principles developed are transferable across distinct bioinformatics domains, suggesting their potential applicability beyond the specific contexts evaluated.

Capítulo 1

Introducción

1.1 El Concepto de Similaridad en la Era de los Datos Masivos

La explosión de datos digitales en la última década ha transformado radicalmente la forma en que se aborda la gestión, la organización y el análisis de la información. En este contexto de grandes volúmenes de datos, cuantificar la relación entre entidades se ha convertido en un requisito fundamental para la inteligencia artificial y la minería de datos. Estas entidades pueden ser documentos textuales, perfiles de usuario, o muestras de un conjunto de datos. Este proceso se articula matemáticamente a través del concepto de *similaridad* (o su complemento, la *distancia*), que constituye una piedra angular de la Minería de Datos y el Aprendizaje Automático.

1.1.1 Similaridad, Distancia y Proximidad: Perspectivas Fundamentales

Definición Filosófica y Formalización Matemática

Conceptualmente, la similaridad puede entenderse como la inversa de una distancia: cuanto más cercanos están dos objetos en un espacio de representación, mayor es su similaridad. Esta noción permite formalizar matemáticamente la proximidad entre entidades mediante funciones de distancia geométrica o topológica.
En el contexto de la ingeniería de datos, la similaridad se formaliza como una función escalar que asigna un valor real, típicamente normalizado en el rango $[0, 1]$, a un par de objetos. Un valor de cero indica que los objetos son completamente disímiles, mientras que un valor de uno implica que son prácticamente idénticos. En el análisis de texto, por ejemplo, esto permite evaluar la cercanía entre documentos mediante la representación vectorial tradicional (Vijaymeena & Kavitha, 2016).

Propiedades Métricas y Desafíos Prácticos

Las medidas de distancia ideales que sustentan la similaridad se definen formalmente al adherirse a las siguientes propiedades fundamentales (Deza & Deza, 2009):

1. **No negatividad e identidad**: $d(A, B) \geq 0$, y $d(A, B) = 0$ si y solo si $A = B$.
2. **Simetría**: $d(A, B) = d(B, A)$.
3. **Desigualdad triangular**: $d(A, C) \leq d(A, B) + d(B, C)$.

La desigualdad triangular es una condición rigurosa que asegura que la distancia más corta entre dos puntos es una línea recta.
Sin embargo, en el ámbito de la ciencia de datos aplicada, es un hecho reconocido que muchas de las medidas de distancia y similaridad utilizadas comúnmente en la práctica no cumplen estrictamente estas condiciones, especialmente cuando se trabaja con datos complejos como los de alta dimensión o semánticamente ricos (Deza & Deza, 2009). Esta tensión entre el rigor matemático y la utilidad pragmática obliga a un compromiso: en la Computación Científica, la eficacia predictiva y la robustez de una función ante la dimensionalidad o el ruido a menudo tienen prioridad sobre el rigor geométrico absoluto, lo que justifica el uso de cuasi-métricas cuando demuestran un rendimiento superior en tareas de *machine learning*.

1.2 Taxonomía de las Métricas de Similaridad

La selección de la métrica de similaridad debe ser una decisión informada, ya que determina la noción de cercanía o agrupación en cualquier algoritmo de aprendizaje. Las métricas pueden clasificarse según su enfoque matemático: distancia geométrica, orientación vectorial o superposición de conjuntos.

1.2.1 Métricas de Distancia Geométrica (ℓ_p) y Sus Adaptaciones

Las métricas de distancia basadas en la norma ℓ_p son fundamentales para medir la proximidad en espacios numéricos. Dado un vector $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, la norma ℓ_p se define como:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p} \tag{1.1}$$

donde $p \geq 1$. Los casos más comunes son $p = 1$ (norma Manhattan o suma de valores absolutos), $p = 2$ (norma Euclidiana) y $p = \infty$ (máximo valor absoluto). La distancia ℓ_p entre dos puntos $\mathbf{u}$ y $\mathbf{v}$ se calcula como $\|\mathbf{u} - \mathbf{v}\|_p$.

Distancia Euclidiana y Mahalanobis

La Distancia Euclidiana (ℓ_2) es la métrica más utilizada para medir la proximidad absoluta en espacios numéricos (Ramli & Ahmad, 2014). Sin embargo, su principal limitación es su sensibilidad a la escala y la suposición de que las dimensiones son independientes y tienen varianza similar. En la práctica, esto puede llevar a que características irrelevantes o altamente correlacionadas dominen el cálculo de la distancia.

Para abordar la interdependencia entre características, se emplea la Distancia de Mahalanobis. Esta es una extensión crítica de la Euclidiana que incorpora la matriz de covarianza de los datos, lo que le permite medir la distancia ajustando automáticamente por la correlación entre las variables (De Maesschalck y col., 2000). Los análisis en tareas de *clustering* sugieren que la Mahalanobis puede ser más efectiva y proporcionar ventajas de precisión sobre la Euclidiana, especialmente en escenarios donde la interdependencia de las características es alta (De Maesschalck y col., 2000). Esto pone de manifiesto que la distancia efectiva debe adaptarse a la estructura estadística interna del conjunto de datos.

1.2.2 Métricas de Orientación y el Desafío de la Dimensionalidad

La Preponderancia de la Similaridad de Coseno

La Similaridad de Coseno (CS) es una métrica de orientación que calcula el coseno del ángulo θ entre dos vectores $\mathbf{u}$ y $\mathbf{v}$ en un espacio de producto interior (Bajusz y col., 2015). Se define formalmente como:

$$\mathrm{CS}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$$

A diferencia de las distancias euclidianas, la similaridad del coseno se centra en la orientación de los vectores más que en su magnitud (Bajusz y col., 2015). Esta propiedad le confiere una robustez excepcional frente a la variación en la longitud del vector. Su eficacia en espacios de datos dispersos, como los generados por el modelo *bag-of-words* en análisis documental, la ha establecido como la métrica predominante en recuperación de información y *clustering* de documentos (Saini y col., 2023; Vijaymeena & Kavitha, 2016). En concreto, ha demostrado ser la medida estadística más efectiva en tareas de PLN (Procesamiento del Lenguaje Natural), incluyendo el resumen de texto multi-documento y el *clustering* basado en densidad (Gahman & Elangovan, 2023).

1.2.3 Métricas de Intersección y la Similaridad Semántica Avanzada

Índice de Jaccard (Tanimoto) y Superposición

El Índice de Jaccard, también conocido como coeficiente de Tanimoto en el contexto de quimioinformática (Bajusz y col., 2015), mide la similaridad entre dos conjuntos, A y B, como la razón entre la intersección de los ítems y su unión. Formalmente:

$$\text{Jaccard}(A, B) = T(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1.2}$$

Es particularmente eficaz para datos binarios o para medir la superposición de características de presencia/ausencia (Bajusz y col., 2015). En el ámbito de la química computacional, cuando se aplica a vectores binarios que representan fingerprints moleculares, esta métrica se denomina habitualmente coeficiente de Tanimoto y es ampliamente utilizada para cuantificar la similaridad estructural entre compuestos (Bajusz y col., 2015). El índice de Jaccard ha sido tradicionalmente contrastado con la fórmula del Coseno para medir la relación entre documentos.

1.3 Aplicaciones de la similaridad

La medida de similaridad entre objetos es una herramienta central en múltiples dominios de la ciencia y la ingeniería. En recuperación de información y motores de búsqueda, las funciones de similaridad permiten ordenar resultados por relevancia y mejorar la precisión de las consultas mediante comparaciones entre representaciones vectoriales de documentos y consultas, así como mediante métodos clásicos basados en distancias y coeficientes de correlación. Estas aplicaciones y sus fundamentos metodológicos se discuten en profundidad en la literatura técnica sobre similaridad y métricas (Deza & Deza, 2009).

En procesamiento de lenguaje natural y sistemas basados en embeddings, la similaridad entre vectores (por ejemplo, mediante coseno o distancias métricas) es la base para tareas como búsqueda semántica, agrupamiento de documentos, detección de temas y expansión de consultas. Los avances recientes en representación densa han incrementado la aplicabilidad de estas medidas a problemas de recuperación semántica y clasificación de texto (El Amrani y col., 2020; Lee y col., 2023).

En detección de plagio y análisis forense textual, numerosas herramientas comerciales y académicas emplean medidas de similaridad para comparar fragmentos de texto y detectar solapamientos o reescrituras. Plataformas ampliamente usadas para control de originalidad se apoyan en variantes heurísticas y técnicas de hashing

junto con medidas de similaridad más finas para reducir falsos positivos.
En visión por computadora, se usan medidas de similaridad entre descriptores locales o embeddings extraídos por redes convolucionales para tareas de recuperación de imágenes, reconocimiento de objetos y emparejamiento de características entre imágenes. La robustez de la similaridad frente a transformaciones y ruido es un aspecto de gran interés práctico (Formica & Taglino, 2021).
En estadística y aprendizaje automático, la elección de una métrica de distancia afecta directamente a algoritmos como k-NN, clustering jerárquico y métodos basados en kernels. Métricas más sofisticadas, como la distancia de Mahalanobis, permiten incorporar estructura de covarianza y escalamiento de variables para mejorar la discriminación entre clases (De Maesschalck y col., 2000).
En sistemas de recomendación, la similaridad entre usuarios o ítems (colaborativa y basada en contenido) alimenta la generación de recomendaciones personalizadas, ya sea mediante puntuaciones agregadas o modelos basados en embeddings latentes que capturan preferencias implícitas. La combinación de diferentes medidas de similaridad y la integración de señales contextuales es una práctica común para mejorar la calidad de las sugerencias (Gahman & Elangovan, 2023; Vijaymeena & Kavitha, 2016).
Finalmente, la similaridad tiene aplicaciones en tareas de análisis de redes y grafos (p. ej., comparación de subgrafos o alineamiento), en la integración y limpieza de datos (matching de registros y deduplicación) y en sistemas conversacionales y agentes conversacionales donde se requiere hallar intenciones o respuestas cercanas en un espacio semántico. Estas aplicaciones ilustran la versatilidad del concepto y la necesidad de seleccionar medidas y representaciones apropiadas al dominio (Torres y col., 2009).
Entre todas estas aplicaciones, el dominio de la bioinformática y la química computacional presenta desafíos particulares que motivan el desarrollo de nuevas metodologías. En particular, la estimación de similaridad entre compuestos químicos en redes metabólicas combina aspectos estructurales (propiedades moleculares) y topológicos (conectividad en el grafo de reacciones), lo que la convierte en un problema complejo que requiere enfoques especializados. Este problema constituye el foco central de la presente investigación y se desarrolla en la siguiente sección.

1.4 Una aplicación de interés: Estimación de similaridad de compuestos

El estudio del metabolismo celular puede abordarse desde una perspectiva sistémica, considerando las reacciones bioquímicas como parte de un entramado complejo de

transformaciones moleculares (Barabási & Oltvai, 2004; Palsson, 2006). Estas reacciones forman rutas metabólicas que, en conjunto, constituyen una red altamente interconectada de procesos celulares (Jeong y col., 2000; Wagner & Fell, 2001). Los metabolitos —moléculas que participan como sustratos, productos o intermediarios— son los elementos fundamentales que conectan estas reacciones (Ma & Zeng, 2003). Debido a que un mismo compuesto puede intervenir en múltiples rutas, la representación natural de estos sistemas es mediante grafos, en donde los nodos representan a los compuestos, y las aristas indican relaciones bioquímicas, usualmente mediadas por una reacción común (Arita, 2012; Jeong y col., 2000; Ma & Zeng, 2003). La Figura 1.1 ilustra esta representación para el camino metabólico de la glucólisis.

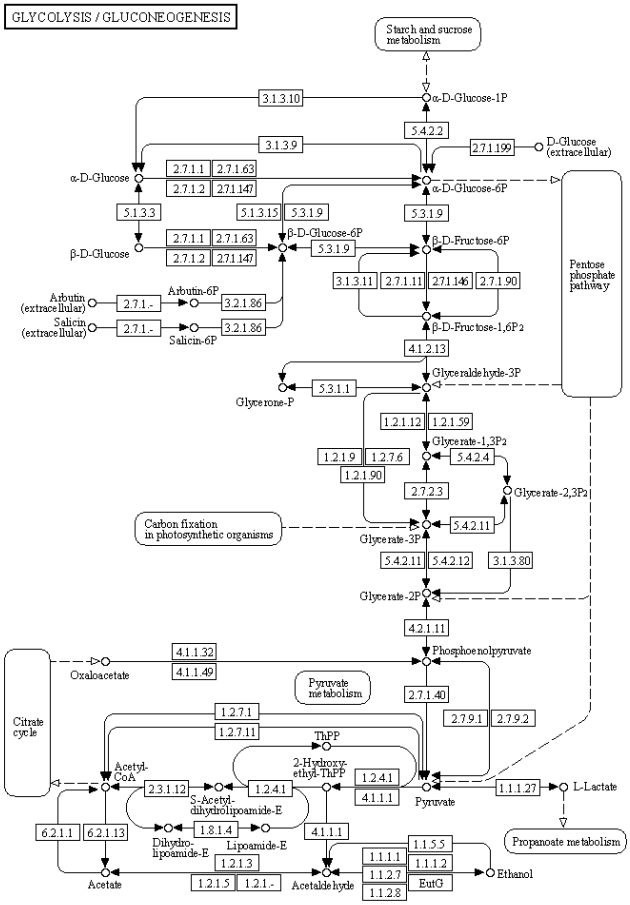

Figura 1.1: Representación del camino metabólico de la glucólisis/gluconeogénesis obtenido de la base de datos KEGG. Los compuestos químicos (metabolitos) aparecen como nodos circulares, mientras que las reacciones enzimáticas —identificadas por sus números EC— conectan sustratos con productos. Esta estructura ilustra cómo las vías metabólicas pueden modelarse naturalmente como grafos, donde la topología de la red refleja las relaciones bioquímicas entre compuestos. Las conexiones con otros caminos metabólicos (ciclo de Krebs, vía de las pentosas fosfato, metabolismo del piruvato) evidencian la naturaleza interconectada del metabolismo celular.

El modelado de redes metabólicas como grafos posibilita el uso de herramientas de teoría de grafos, análisis de redes complejas y aprendizaje automático para inferir propiedades emergentes de los sistemas biológicos. Este enfoque, ampliamente utilizado en biología de sistemas, permite representar las interacciones entre cientos de metabolitos y reacciones, revelando propiedades ocultas del funcionamiento celular (Gillis y col., 2014). La estructura de estas redes es clave para comprender la complejidad metabólica, los flujos de energía y la plasticidad del sistema, aspectos fundamentales tanto para estudiar mecanismos de enfermedades como para guiar el descubrimiento de nuevos fármacos (Noor y col., 2010; Steuer y col., 2006).

Además, los avances en secuenciación y metabolómica han permitido reconstruir redes metabólicas en un gran número de organismos. En este contexto, uno de los desafíos relevantes es la estimación de la similaridad entre compuestos, dado que comprender cuán relacionados están dos metabolitos dentro de la red puede aportar información clave para tareas como la predicción de funciones, la anotación de metabolitos desconocidos o la caracterización de vías metabólicas alternativas.

Este desafío puede formalizarse como la predicción de una propiedad (la similaridad) definida sobre pares de nodos de un grafo. Existen múltiples motivaciones para abordar esta tarea. Por un lado, se ha observado empíricamente que compuestos conectados por reacciones suelen presentar subestructuras químicas similares, lo que implica que las conexiones topológicas reflejan, al menos parcialmente, proximidad estructural o funcional (Arita, 2012; Rahman y col., 2005a). La evaluación de la similaridad permite identificar similitudes estructurales, funcionales o de comportamiento entre entidades, lo que facilita tareas como la clasificación, la agrupación y la toma de decisiones informadas.

En el campo de la quimioinformática y disciplinas relacionadas como la química medicinal y el descubrimiento de fármacos, la similaridad molecular (Muegge & Mukherjee, 2016; Willett y col., 1998) ha sido ampliamente utilizada. Se aplica en la predicción de propiedades (Brown & Martin, 1996), el cribado virtual (Muegge & Mukherjee, 2016), la búsqueda de similitudes (Willett y col., 1998) y el diseño de rutas metabólicas al utilizar la estructura molecular de los compuestos como guía (McShan y col., 2003; Rahman y col., 2005b). El cálculo de similitud se realiza utilizando descriptores moleculares, como las fingerprints, que corresponden a vectores numéricos que capturan las características estructurales de los compuestos (Durant y col., 2002a). Sin embargo, si la estructura molecular no está disponible, o contiene elementos que no pueden ser analizados por los métodos clásicos, no es posible realizar el cálculo de la similitud.

En el contexto de las redes biológicas, el análisis suele basarse en información experimental, que a menudo es incompleta o presenta variabilidad (Thieffry, 2007; X. Wang & Chen, 2003). El modelado de estas redes mediante grafos permite estudiar

la conectividad entre sus componentes y las propiedades individuales a diferentes escalas (Arita, 2012). Los nodos y los arcos en los grafos pueden contener información característica de las entidades biológicas y sus interacciones, que puede emplearse para predecir nuevas propiedades o inferir información faltante.
El análisis tradicional de grafos se ha centrado en métricas de conectividad y agrupamiento de nodos (Liu y col., 2018). Los avances en aprendizaje profundo han introducido métodos basados en representaciones aprendidas. Los métodos de aprendizaje de representaciones han facilitado la construcción automática de descriptores en espacios euclídeos para grafos y sus componentes, lo que ha mejorado el análisis e inferencia de propiedades (Goyal & Ferrara, 2018). Sin embargo, estos métodos generan representaciones de forma independiente para cada nodo, sin aprovechar la estructura completa del grafo durante el aprendizaje.
En este contexto, las redes neuronales en grafos han surgido como un enfoque altamente efectivo para el procesamiento de datos no euclídeos (Jiang y col., 2021). Estas redes permiten aprovechar la estructura local y global de las vecindades de los nodos, generando representaciones que capturan naturalmente la estructura del grafo (Ciortan & Defrance, 2021; Y. Wang y col., 2020). Al combinar estas redes con otras arquitecturas, es posible construir modelos robustos para la inferencia de propiedades en redes biológicas y sus componentes (J. Wang y col., 2021). Por lo tanto, el desarrollo de nuevas arquitecturas neuronales en grafos se presenta como una herramienta prometedora para la inferencia de similitud entre compuestos en redes biológicas, al considerar tanto la estructura de la red como las características moleculares de los compuestos.

1.5 Modelado con grafos

Los grafos representan una estructura flexible y ampliamente utilizada para modelar relaciones entre entidades, especialmente en contextos donde los datos carecen de una estructura euclidiana. En estos modelos, los nodos representan entidades y las aristas describen relaciones entre pares de nodos. Tanto nodos como aristas pueden incluir características que describen propiedades asociadas, haciendo de los grafos herramientas versátiles para una gran variedad de aplicaciones. Ejemplos de estas aplicaciones incluyen sistemas de recomendación (Covington y col., 2016; Steck y col., 2021), redes sociales, algoritmos de clustering (Xie y col., 2016) y bioinformática, donde se emplean para modelar interacciones biológicas complejas (Z. Wu y col., 2019).

1.5.1 Taxonomía de Tareas de Aprendizaje en Grafos

Para abordar los problemas en grafos de manera estructurada, las tareas se clasifican generalmente en tres niveles principales (Zhang y col., 2020): tareas a nivel de grafo completo, tareas a nivel de nodo y tareas a nivel de arista. Cada una de estas categorías presenta objetivos específicos y aplicaciones particulares.

Las tareas a nivel de grafo se enfocan en predecir características o propiedades que abarcan al grafo en su totalidad. En química computacional, por ejemplo, una tarea a nivel de grafo podría consistir en predecir si una molécula, representada como un grafo de sus átomos constitutivos, tiene la capacidad de unirse a un receptor asociado a una enfermedad o si tiene propiedades que la convierten en un candidato prometedor para un nuevo fármaco.

Las tareas a nivel de nodo se centran en predecir propiedades específicas asociadas a nodos individuales dentro de un grafo. Por ejemplo, en redes sociales, una tarea a nivel de nodo podría consistir en predecir el grupo social al que probablemente se unirá un nuevo usuario, basándose en sus conexiones y en las características de los amigos que ya pertenecen a ese grupo. Este tipo de tareas es común en contextos con datos no etiquetados, como identificar si un individuo tiene un hábito específico, como fumar.

Las tareas a nivel de arista, también conocidas como tareas de predicción de enlaces, se enfocan en analizar relaciones entre pares de nodos en un grafo. Un ejemplo clásico es evaluar la probabilidad de que dos usuarios establezcan una conexión en una plataforma de citas o matchmaking. Otro caso frecuente es la recomendación de contenido en plataformas como Netflix, donde la tarea consiste en predecir qué video recomendar a un usuario en función de su historial de visualización y preferencias. Estas tareas son fundamentales en contextos donde las conexiones entre entidades contienen información crucial, como la predicción de interacciones proteína-proteína o la identificación de similitudes moleculares. Las tareas a nivel de arista presentan desafíos únicos, como la integración simultánea de características de nodos y aristas, y la representación robusta de las relaciones en un espacio común (Gilmer y col., 2017; Vaswani y col., 2017).

1.5.2 Evolución de Métodos para el Análisis de Grafos

Los primeros enfoques para el análisis de grafos se centraron en el uso de métricas como la conectividad, la centralidad y otras propiedades estructurales para describir las características de un grafo (Newman, 2003). Aunque útiles para analizar propiedades estáticas, estos métodos no capturan relaciones complejas ni patrones intrínsecos presentes en los datos.

Uno de los primeros métodos desarrollados para generar representaciones vectoriales

de los nodos, conocidas como embeddings, fue DeepWalk (Perozzi y col., 2014). Este método utiliza recorridos aleatorios (random walks) para explorar la vecindad local de cada nodo y luego aplica un modelo skip-gram para convertir estos recorridos en embeddings densos. Sin embargo, DeepWalk no considera los pesos de las aristas, lo que limita su aplicabilidad en casos donde las relaciones entre nodos varían en intensidad o importancia.

Posteriormente, se introdujo LINE (Tang y col., 2015), un enfoque que se centra en preservar tanto la proximidad de primer orden (conexiones directas) como de segundo orden (vecindad compartida) en las representaciones. A pesar de sus avances, LINE tampoco incorpora características de las aristas, limitando su capacidad para modelar relaciones complejas.

El método Node2Vec (Grover & Leskovec, 2016) amplió las capacidades de los recorridos aleatorios al ajustar la estrategia de exploración, permitiendo un muestreo más flexible de la vecindad. Sin embargo, este enfoque utiliza pesos de aristas fijos y no aprendibles, lo que restringe su adaptabilidad a diferentes tipos de grafos y escenarios.

A pesar de los avances logrados por estos métodos, presentan limitaciones importantes: no comparten parámetros entre nodos, lo que reduce su eficiencia en grafos grandes, y no generalizan bien a grafos dinámicos o nuevos nodos no observados durante el entrenamiento (Hamilton y col., 2017). Estas limitaciones resaltan la necesidad de métodos más avanzados que puedan capturar simultáneamente la información de los nodos y las aristas de manera flexible y escalable, allanando el camino para el desarrollo de modelos más poderosos.

Las redes neuronales en grafos (*Graph Neural Networks*, GNNs) han surgido como un enfoque poderoso para superar las limitaciones de los métodos tradicionales, permitiendo aprender de la estructura de las vecindades en un grafo (Zhou y col., 2020). Estos modelos buscan generar nuevas representaciones para las entidades basadas en sus conexiones dentro del grafo, aprovechando las relaciones inherentes entre nodos y aristas. Existen dos enfoques principales para construir GNNs: métodos espectrales y métodos espaciales, cada uno con ventajas, desventajas y aplicaciones específicas.

Los métodos espectrales definen convoluciones en grafos utilizando la teoría espectral, donde el grafo se transforma al dominio espectral, generalmente empleando la transformada de Fourier en grafos (Bruna y col., 2013; Defferrard y col., 2016). Al utilizar los componentes espectrales de la matriz Laplaciana, estos métodos capturan patrones globales en el grafo. Sin embargo, pueden ser computacionalmente costosos, especialmente para grafos de gran escala, debido a la necesidad de descomposición en valores propios de la matriz Laplaciana (Shuman y col., 2013; Zhu & Rabbat, 2012). Además, los modelos basados en espectros suelen tener dificultades para generalizar a grafos con estructuras diferentes, lo que limita su aplicabilidad en tareas

de aprendizaje inductivo (Bronstein y col., 2016; Z. Wu y col., 2019). Para abordar estas limitaciones, se propuso el método GCN/ChebNet, que utiliza aproximaciones polinomiales de los filtros espectrales para reducir los costos computacionales (Defferrard y col., 2016). Sin embargo, estos métodos requieren que el grafo permanezca inmutable durante el proceso de aprendizaje, lo que limita su flexibilidad en escenarios de grafos dinámicos.

Por otro lado, los métodos espaciales definen convoluciones directamente sobre el grafo al agregar información de los nodos vecinos (Zhou y col., 2020). Estos métodos propagan información localmente, permitiendo que cada nodo pase mensajes a sus vecinos y refine sus representaciones iterativamente (Bronstein y col., 2016). Los métodos espaciales ofrecen ventajas clave sobre los métodos espectrales, particularmente en eficiencia computacional y flexibilidad. A diferencia de los métodos espectrales, que requieren costosos cálculos de descomposición de valores propios, los métodos espaciales operan directamente sobre la estructura del grafo, haciéndolos más escalables para grafos grandes con millones de nodos y adaptables a grafos dinámicos sin necesidad de recalcular los valores propios. También generalizan mejor a nodos o grafos no observados durante el entrenamiento, un proceso conocido como aprendizaje inductivo, lo cual es especialmente importante en escenarios donde se incorpora constantemente nueva información (Zhang y col., 2020). Además, los métodos espaciales permiten integrar características de las aristas en el proceso de aprendizaje, proporcionando información relevante para modelar relaciones complejas entre nodos.

Uno de los enfoques más recientes en este campo es la Red Neuronal de Paso de Mensajes (*Message Passing Neural Network*, MPNN) (Gilmer y col., 2020), que ha demostrado ser efectiva en tareas que van desde la predicción de interacciones proteína-proteína hasta el análisis de redes de regulación génica (Duvenaud y col., 2015; Kearnes y col., 2016). Las MPNNs son una clase de GNN diseñadas para manejar tareas basadas en grafos mediante el intercambio iterativo de mensajes entre nodos y la actualización de las representaciones de los nodos en función de los mensajes recibidos de su vecindad local. Este enfoque resulta particularmente poderoso para capturar dependencias estructurales dentro de los grafos, haciéndolas altamente versátiles. La definición formal de las MPNNs y su formulación matemática se presentan en detalle en la Sección 2.4.4.

1.5.3 Desafíos y Motivación

A pesar del éxito de las MPNNs en tareas a nivel de nodo, existen desafíos importantes al trabajar con grafos de gran escala y al abordar tareas centradas en aristas, como la regresión y clasificación de propiedades asociadas a las conexiones del grafo.

Muchos modelos tradicionales de GNN, incluidas las MPNNs, se centran principalmente en características de los nodos, subutilizando la información contenida en las aristas, lo cual es especialmente relevante en áreas como la bioinformática, donde las interacciones entre entidades suelen ocurrir en el nivel de las aristas (Z. Wu y col., 2019).
Este trabajo propone un modelo basado en MPNNs, cuya complejidad va creciendo a lo largo de la tesis, incorporando más características que le permiten mejorar algunas de las limitaciones descriptas. El modelo propuesto integra características de nodos y aristas en el proceso de predicción. Se entrena con una función de costo híbrida que combina aprendizaje supervisado y auto-supervisado, lo que permite manejar relaciones complejas entre entidades del grafo. Además, emplea un mecanismo de atención inspirado en transformers (Vaswani y col., 2017), que mejora su capacidad para capturar dependencias complejas dentro del grafo. La arquitectura resulta robusta incluso con codificaciones *one-hot* simples para las características de los nodos, permitiendo un desempeño eficaz en escenarios con información limitada o incompleta.

1.6 Objetivos

El objetivo general de esta tesis es el desarrollo de nuevas arquitecturas neuronales en grafos para la inferencia de propiedades relacionadas con aristas de grafos biológicos. Estas arquitecturas estarán diseñadas para analizar las propiedades de las redes biológicas y las entidades que las componen, generando representaciones robustas basadas en la estructura de vecindades de los nodos y sus características.

1.6.1 Objetivos específicos

A continuación se detallan los objetivos específicos que guiarán el desarrollo de esta investigación:

- Estudiar y evaluar diferentes estrategias para construir descriptores vectoriales que permitan estimar la similaridad entre compuestos que participan en vías metabólicas, considerando tanto información estructural como topológica.
- Analizar y comparar métodos para la generación de embeddings en grafos que capturen información relevante de los nodos y/o aristas, incluyendo métodos clásicos (por ejemplo, *Node2Vec*) y modelos basados en paso de mensajes (*Message Passing Neural Networks*).
- Desarrollar modelos neuronales en grafos capaces de predecir la similaridad entre compuestos dentro de la red metabólica, explotando explícitamente la

estructura del grafo y las propiedades locales y globales de sus nodos.

- Investigar estrategias de entrenamiento que permitan incorporar información incompleta o ausente sobre ciertos compuestos, explorando enfoques autosupervisados y de aprendizaje semisupervisado sobre grafos.
- Evaluar la capacidad de generalización de los modelos desarrollados trasladándolos a otros dominios biológicos relevantes, tales como la predicción de interacciones proteína-proteína (PPI) o la asignación de funciones en la Ontología de Genes (GO), utilizando conjuntos de datos de acceso público.

1.7 Organización de la Tesis

Esta tesis se organiza en siete capítulos que presentan una progresión metodológica incremental, partiendo de enfoques simples basados en redes neuronales clásicas hasta arquitecturas especializadas en grafos con mecanismos de atención y aprendizaje híbrido. Esta estructura responde a una estrategia deliberada: cada capítulo identifica limitaciones del enfoque anterior y propone mejoras específicas, permitiendo al lector comprender la evolución natural de la investigación y la justificación de cada contribución.

El **Capítulo 2** establece los fundamentos teóricos necesarios para el resto de la tesis. Se presentan métodos clásicos de análisis de grafos (métodos espectrales, caminatas aleatorias) y se introduce el marco de redes neuronales en grafos (GNNs), con énfasis en las redes de paso de mensajes (MPNNs). Este capítulo incluye la formulación matemática formal de las MPNNs, que servirá como base para las arquitecturas desarrolladas en capítulos posteriores. La ubicación de este capítulo al inicio es estratégica: proporciona el contexto teórico necesario antes de presentar las contribuciones experimentales, evitando interrumpir el flujo narrativo de los resultados con explicaciones técnicas extensas.

El **Capítulo 3** presenta el primer enfoque experimental de la tesis. Se investiga si un perceptrón multicapa (MLP) simple puede predecir el índice de Tanimoto entre pares de compuestos metabólicos utilizando codificación one-hot como entrada. Este capítulo establece un *baseline* fundamental que demuestra la viabilidad del aprendizaje supervisado para la tarea, alcanzando un RMSE de 0.1019. Aunque este enfoque no explota la estructura del grafo metabólico, su simplicidad permite evaluar el desempeño mínimo esperado y motiva las mejoras presentadas en capítulos subsiguientes. La decisión de comenzar con un enfoque simple responde a principios de investigación rigurosa: establecer primero una línea base clara antes de introducir complejidad adicional.

El **Capítulo 4** aborda las limitaciones identificadas en el Capítulo 3. La codificación

one-hot, aunque efectiva, no captura información topológica del grafo ni permite representar compuestos con estructura molecular desconocida. Para superar estas restricciones, este capítulo propone utilizar embeddings de nodos generados mediante Node2Vec como descriptores alternativos. Los embeddings condensan la información de vecindad en vectores densos de menor dimensionalidad, capturando la estructura local y global del grafo metabólico. Se realiza una exploración exhaustiva de hiperparámetros mediante optimización bayesiana (Optuna con TPE), demostrando que los embeddings mejoran significativamente el desempeño y permiten generalizar a compuestos sin estructura conocida. Este capítulo ilustra cómo la incorporación de información estructural del grafo, aunque mediante un método pre-entrenado separado (Node2Vec), representa un avance conceptual importante.

El **Capítulo 5** presenta la contribución principal de esta tesis: la arquitectura NEAConv. Este capítulo integra las lecciones aprendidas de los capítulos anteriores y aborda sus limitaciones mediante tres innovaciones clave: (1) una arquitectura MPNN que trata nativamente características de nodos y aristas con un mecanismo de atención compartido inspirado en transformers, (2) una función de pérdida híbrida que combina objetivos supervisados y auto-supervisados, permitiendo aprendizaje robusto incluso con características dispersas, y (3) la generación de embeddings de manera *end-to-end* durante el entrenamiento, en contraste con el enfoque de embeddings pre-calculados del Capítulo 4. Se presenta un estudio de ablación exhaustivo que valida la importancia de cada componente del modelo (atención, términos auto-supervisados). La ubicación de este capítulo en el centro de la tesis es deliberada: consolida las contribuciones metodológicas después de haber establecido tanto los fundamentos teóricos (Capítulo 2) como las líneas base experimentales (Capítulos 3 y 4).

El **Capítulo 6** demuestra la versatilidad y capacidad de generalización de la arquitectura NEAConv propuesta en el Capítulo 5. El modelo se evalúa en dos tareas bioinformáticas adicionales que involucran predicción a nivel de aristas: predicción de interacciones proteína-proteína (PPI) y predicción de términos del Gene Ontology (GO). Estas aplicaciones validan que la arquitectura —con su mecanismo de atención para nodos y aristas, y su función de pérdida híbrida— puede adaptarse efectivamente a dominios distintos sin modificaciones sustanciales. Los resultados demuestran rendimiento competitivo o superior al estado del arte, reforzando el argumento de que las tareas centradas en aristas se benefician de arquitecturas especializadas que integran información de nodos y aristas explícitamente. Este capítulo se ubica después de la presentación de la arquitectura principal porque su propósito es validar la generalización del modelo, no introducir nuevas contribuciones metodológicas.

Finalmente, el **Capítulo 7** sintetiza las contribuciones de la tesis, analiza críticamente las limitaciones identificadas durante la investigación, y propone direcciones

específicas para trabajo futuro. Se discuten tanto mejoras arquitecturales (arquitecturas más profundas, mecanismos de atención sofisticados) como extensiones a nuevos dominios de aplicación (redes sociales, sistemas de recomendación).

Esta organización sigue una lógica de *complejidad incremental*: cada capítulo construye sobre los anteriores, identificando limitaciones específicas y proponiendo soluciones justificadas. El lector puede así comprender no solo *qué* se propone, sino *por qué* cada enfoque es necesario y cómo se relaciona con el trabajo previo. Esta estructura narrativa facilita la evaluación crítica de las contribuciones y permite apreciar la evolución metodológica que condujo a la arquitectura NEAConv como solución final a los desafíos planteados.

Capítulo 2

Análisis de Grafos

Este capítulo presenta los fundamentos teóricos necesarios para el análisis y modelado de grafos mediante técnicas de aprendizaje automático. Se introducen primero las definiciones formales de grafos y sus elementos, seguidas por métodos clásicos de análisis y, finalmente, arquitecturas basadas en redes neuronales.

2.1 Definiciones Fundamentales

2.1.1 Grafos, nodos y aristas

Sea $\mathcal{G} = \{v, e\}$ un grafo, donde v es el conjunto de $|v|$ nodos en la red y e denota el conjunto de aristas que los conectan. Sea $\mathbf{X} \in \mathbb{R}^{|v| \times d}$ la matriz de características de los nodos en $\mathcal{G}$, donde d es el número de características de cada nodo. Además, sea $\mathcal{E} \in \mathbb{R}^{|e| \times q}$ la matriz de características de las aristas, donde cada fila representa las características asociadas a una arista específica en el grafo. Estas características de aristas pueden o no estar presentes y ser un vector de tamaño arbitrario q. La Figura 2.1a muestra un ejemplo de un grafo típico y sus elementos.
El subgrafo $\mathcal{G}'_{i,j} = \{v_{i,j}, e_{i,j}\} \subseteq \mathcal{G}$ se define como el inducido por los nodos i y j (Figura 2.1b), donde $v_{i,j} = \{\mathcal{N}(i), \mathcal{N}(j)\}$ son los nodos incluidos y $e_{i,j}$ son las conexiones entre ellos. El conjunto $\mathcal{N}(k)$ representa los vecinos directos (conexiones de primer orden) que comparten una arista con el nodo k. Además, la matriz de características de las aristas $\mathbf{E}'_{\mathcal{G}'_{i,j}} \in \mathbb{R}^{|e_{i,j}| \times q}$ está asociada con las $|e_{i,j}|$ conexiones en el subgrafo $\mathcal{G}'_{i,j}$. La matriz de características $\mathbf{X}'_{\mathcal{G}'_{i,j}} \in \mathbb{R}^{|v_{i,j}| \times d}$ también se define para los $|v_{i,j}|$ nodos en el subgrafo $\mathcal{G}'_{i,j}$.

2.1.2 Representaciones Matriciales

Los grafos pueden representarse mediante diversas estructuras matriciales que facilitan el análisis computacional y teórico.

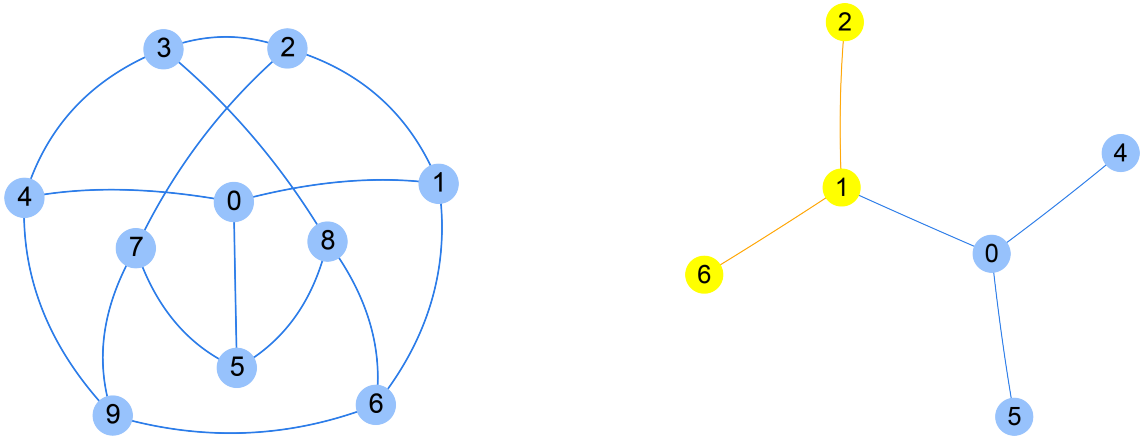

Figura 2.1: (a) Ejemplo de un grafo $\mathcal{G}$. (b) Ejemplo de un subgrafo patrón $\mathcal{G}'_{0,1}$ inducido por los nodos 0 y 1. Los nodos y aristas relacionados con el nodo 1 están resaltados en amarillo, mientras que los relacionados con el nodo 0 se muestran en azul. Estos representan los vecinos de primer grado de cada nodo dentro del subgrafo. Estos grafos representan solo la estructura; en esta representación, cada nodo está asociado con un vector de características, y cada arista tiene características correspondientes.

La **matriz de adyacencia** $\mathbf{A} \in \{0,1\}^{|v|\times|v|}$ codifica la estructura del grafo, donde $A_{ij} = 1$ si existe una arista entre los nodos i y j, y $A_{ij} = 0$ en caso contrario. Para grafos no dirigidos, esta matriz es simétrica.
La **matriz de grados** $\mathbf{D} \in \mathbb{R}^{|v|\times|v|}$ es una matriz diagonal donde cada elemento D_{ii} representa el grado del nodo i, es decir, el número de aristas conectadas a dicho nodo. Formalmente, $D_{ii} = \sum_j A_{ij}$.
La **matriz Laplaciana** $\mathbf{L} \in \mathbb{R}^{|v|\times|v|}$ se define como:

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{2.1}$$

Esta matriz captura propiedades espectrales fundamentales del grafo y es central en métodos de análisis espectral. Una versión normalizada comúnmente utilizada es:

$$\tilde{\mathbf{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} \tag{2.2}$$

donde $\mathbf{I}$ es la matriz identidad. La matriz Laplaciana normalizada tiene propiedades espectrales útiles para el análisis de grafos y es la base de muchas arquitecturas de redes neuronales en grafos.

2.2 Métodos Clásicos de Análisis y Predicción de Propiedades

Los métodos clásicos de análisis de grafos se dividen en dos categorías principales según el dominio en el que operan. Los **enfoques espectrales** se basan en la descomposición de matrices asociadas al grafo (particularmente la matriz Laplaciana) para extraer información estructural mediante sus valores y vectores propios. Estos métodos operan en el dominio espectral del grafo y son efectivos para capturar propiedades globales como la conectividad y la estructura de comunidades. Por otro lado, los **enfoques espaciales** trabajan directamente sobre la estructura del grafo, explorando la vecindad de los nodos mediante caminatas aleatorias o algoritmos de propagación de información. A continuación se describen ambos enfoques, sus fundamentos teóricos, aplicaciones y limitaciones.

2.2.1 Métodos Espectrales

Los métodos espectrales se fundamentan en la teoría espectral de grafos, utilizando la descomposición en valores y vectores propios de matrices asociadas al grafo para extraer información estructural. La descomposición espectral de la matriz Laplaciana $\mathbf{L} = \mathbf{U}\Lambda\mathbf{U}^\top$ descompone el grafo en sus componentes fundamentales, donde Λ es una matriz diagonal que contiene los valores propios $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$, y $\mathbf{U}$ contiene los correspondientes vectores propios. Los valores propios capturan propiedades globales del grafo como conectividad, número de componentes conexas y estructura de comunidades, mientras que los vectores propios proporcionan coordenadas para representar nodos en espacios de baja dimensión preservando la estructura del grafo (Chung, 1997).

Aplicaciones. Los métodos espectrales han sido ampliamente utilizados en diversas tareas de análisis de grafos. En *clustering espectral*, los eigenvectores asociados a los eigenvalores más pequeños de la matriz Laplaciana se utilizan para particionar el grafo en comunidades, minimizando el número de aristas que cruzan entre particiones. En *visualización de grafos*, los primeros eigenvectores no triviales proporcionan coordenadas bidimensionales o tridimensionales para representar nodos preservando su proximidad en el grafo. En *reducción de dimensionalidad*, métodos como *Laplacian Eigenmaps* (Belkin & Niyogi, 2003) proyectan nodos a espacios de baja dimensión preservando relaciones locales.

Formalmente, *Laplacian Eigenmaps* busca preservar relaciones locales minimizando la distancia entre nodos conectados en el espacio de incrustación. El problema de optimización se plantea como:

$$\min_{\mathbf{Y}} \sum_{i,j} w_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|^2,$$

sujeto a $\mathbf{Y}^\top \mathbf{D}\mathbf{Y} = \mathbf{I}$, donde $\mathbf{Y}$ es la matriz de representaciones embebidas, w_{ij} son pesos de las aristas, y $\mathbf{I}$ es la matriz identidad. La solución se obtiene mediante la descomposición en valores propios del Laplaciano normalizado del grafo.

Complejidad computacional y limitaciones. El principal desafío de los métodos espectrales es su alta complejidad computacional. La descomposición en valores propios de una matriz $n \times n$ tiene complejidad $O(n^3)$ en tiempo y $O(n^2)$ en memoria, donde n es el número de nodos del grafo. Esta complejidad hace que estos métodos sean inviables para grafos grandes: un grafo con $n = 1{,}000{,}000$ nodos requeriría almacenar una matriz de 10^{12} elementos ($\approx$ 8 TB en precisión doble) y realizar aproximadamente 10^{18} operaciones, lo cual es computacionalmente intratable incluso en hardware moderno.

Además de la complejidad computacional, los métodos espectrales presentan otras limitaciones importantes. Son inherentemente *transductivos*: requieren recalcular toda la descomposición espectral si se agrega un nuevo nodo al grafo, lo que impide su uso en escenarios con grafos dinámicos. También dependen fuertemente de la estructura global del grafo completo, lo que dificulta su aplicación a grafos que evolucionan en el tiempo o en configuraciones de aprendizaje inductivo donde se requiere generalización a nuevos nodos no vistos durante el entrenamiento (Bronstein y col., 2016; Goyal & Ferrara, 2018).

2.2.2 Métodos Basados en Caminatas Aleatorias

Los métodos basados en caminatas aleatorias exploran la estructura del grafo mediante trayectorias estocásticas, generando secuencias de nodos que luego se procesan mediante técnicas de aprendizaje de representaciones inspiradas en el procesamiento de lenguaje natural. A diferencia de los métodos espectrales que requieren descomposición de matrices completas, estos enfoques operan localmente mediante muestreo, lo que los hace más escalables.

Skip-Gram: Fundamento de embeddings en grafos. El modelo Skip-Gram, originalmente propuesto para aprendizaje de embeddings de palabras en procesamiento de lenguaje natural (Mikolov y col., 2013), ha sido adaptado exitosamente para grafos. La idea central es aprender representaciones vectoriales de nodos tal que nodos que co-ocurren frecuentemente en caminatas aleatorias tengan embeddings similares.

Formalmente, dado un vocabulario de nodos V y una secuencia de nodos observada mediante caminatas aleatorias, Skip-Gram maximiza la probabilidad de observar el

contexto de un nodo dado el nodo central:

$$\max_{\theta} \sum_{v \in V} \sum_{c \in \mathcal{C}(v)} \log P(c|v;\theta),$$

donde $\mathcal{C}(v)$ denota el contexto del nodo v (nodos que aparecen dentro de una ventana en las caminatas), y θ representa los parámetros del modelo (las matrices de embeddings). La probabilidad condicional se modela típicamente mediante softmax:

$$P(c|v;\theta) = \frac{\exp(\mathbf{u}_c^\top \mathbf{u}_v)}{\sum_{n \in V} \exp(\mathbf{u}_n^\top \mathbf{u}_v)},$$

donde $\mathbf{u}_v$ y $\mathbf{u}_c$ son los vectores de embedding del nodo central y del nodo de contexto, respectivamente. En la práctica, se utiliza *negative sampling* para aproximar eficientemente la normalización del softmax, evitando el costo computacional de sumar sobre todos los nodos del grafo (Mikolov y col., 2013).

DeepWalk. DeepWalk (Perozzi y col., 2014) fue el primer método en aplicar Skip-Gram a grafos mediante caminatas aleatorias uniformes. El algoritmo consta de dos fases: (1) generación de caminatas aleatorias, y (2) optimización de embeddings mediante Skip-Gram.

En la fase de generación, se realizan múltiples caminatas aleatorias de longitud T iniciando desde cada nodo del grafo. Una caminata de longitud T genera una secuencia de T nodos $[v_1, v_2, ..., v_T]$, donde cada nodo v_{i+1} se selecciona uniformemente al azar entre los vecinos de v_i. Para capturar adecuadamente la estructura del grafo, se generan típicamente γ caminatas por nodo.

En la fase de optimización, cada secuencia generada se procesa con una ventana deslizante de tamaño w. Para cada nodo v_i en la posición i de la secuencia (donde $w + 1 \leq i \leq T - w$), se consideran como contexto los nodos en las posiciones $[i - w, ..., i - 1, i + 1, ..., i + w]$. El objetivo es maximizar:

$$\max \prod_{i=w+1}^{T-w} \prod_{-w \leq j \leq w, j \neq 0} P(v_{i+j}|v_i),$$

donde T determina cuántos nodos se visitan en cada caminata (afectando el alcance de la exploración), y w determina el tamaño del contexto local considerado alrededor de cada nodo en la secuencia. Intuitivamente, T controla cuán lejos se explora desde el nodo inicial, mientras que w controla cuán locales o globales son las relaciones capturadas: un w pequeño captura relaciones muy locales, mientras que un w grande captura relaciones más globales dentro de la caminata.

Node2Vec. Node2Vec (Grover & Leskovec, 2016) extiende DeepWalk introduciendo caminatas aleatorias sesgadas que permiten controlar la estrategia de exploración del grafo mediante dos parámetros: p (parámetro de retorno) y q (parámetro de

entrada/salida).
El sesgo de transición se define sobre aristas (t, v, x), donde t es el nodo previamente visitado, v es el nodo actual, y x es un candidato para el próximo nodo. La probabilidad de transición no normalizada es:

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & \text{si } d_{tx} = 0 \text{ (retorno a } t) \\ 1 & \text{si } d_{tx} = 1 \text{ (vecino común)} \\ \frac{1}{q} & \text{si } d_{tx} = 2 \text{ (alejamiento)} \end{cases}$$

donde d_{tx} es la distancia en el grafo entre t y x. La probabilidad normalizada de transición es entonces:

$$P(x|v,t) = \frac{\alpha_{pq}(t,x) \cdot w_{vx}}{\sum_{y \in \mathcal{N}(v)} \alpha_{pq}(t,y) \cdot w_{vy}},$$

donde w_{vx} es el peso de la arista (v, x).
Los parámetros p y q controlan el balance entre exploración en amplitud (BFS-like) y en profundidad (DFS-like) (ver Figura 2.2):

- Un valor bajo de p ($p < 1$) favorece el retorno a nodos visitados recientemente, promoviendo exploración local y capturando similaridad estructural (nodos con roles similares en el grafo).
- Un valor bajo de q ($q < 1$) favorece alejarse de nodos visitados, promoviendo exploración tipo DFS y capturando comunidades (nodos conectados en la misma vecindad).
- Valores altos de p y q promueven exploración tipo BFS, enfocándose en la vecindad inmediata.

Una vez generadas las caminatas sesgadas, Node2Vec aplica el mismo proceso de optimización Skip-Gram que DeepWalk.
Limitaciones. A pesar de su eficiencia relativa comparada con métodos espectrales, los métodos basados en caminatas aleatorias presentan limitaciones importantes. Requieren generar un número suficientemente grande de caminatas para capturar adecuadamente la estructura del grafo, lo que puede ser costoso en grafos densos. Son inherentemente transductivos: los embeddings se calculan para un grafo fijo y no pueden generalizarse directamente a nuevos nodos sin recalcular caminatas. Además, la calidad de los embeddings depende críticamente de la selección de hiperparámetros (T, w, γ, y en Node2Vec, p y q), que pueden requerir búsqueda exhaustiva para cada grafo específico (Grover & Leskovec, 2016; Perozzi y col., 2014).

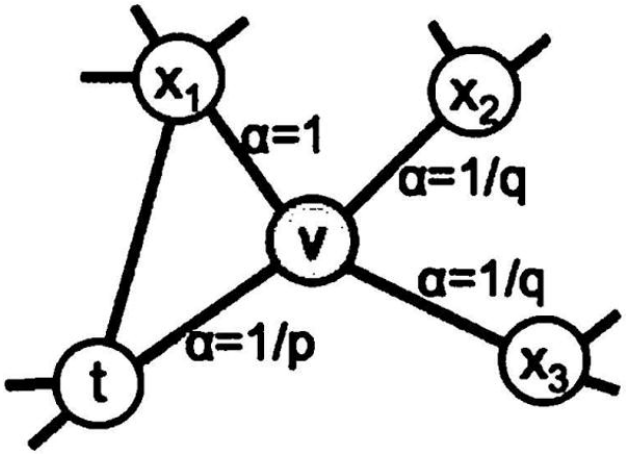

Figura 2.2: Ilustración del procedimiento de caminata aleatoria en Node2Vec. La caminata acaba de transicionar desde el nodo t hacia el nodo v y ahora está evaluando su próximo paso saliendo del nodo v. Las etiquetas en las aristas indican los sesgos de búsqueda α, donde $\alpha = 1/p$ para retorno a t, $\alpha = 1$ para vecinos comunes, y $\alpha = 1/q$ para alejamiento. Adaptado de (Grover & Leskovec, 2016).

2.3 Aprendizaje Supervisado: Fundamentos

Antes de introducir las redes neuronales en grafos, es necesario establecer los conceptos fundamentales del aprendizaje supervisado y las arquitecturas neuronales básicas que constituyen sus bloques de construcción. Esta sección presenta los elementos esenciales: perceptrones multicapa, funciones de pérdida y métricas de evaluación.

2.3.1 Perceptrón Multicapa (MLP)

El perceptrón multicapa (MLP, por sus siglas en inglés) es la arquitectura neuronal más fundamental en aprendizaje profundo. Un MLP consiste en múltiples capas de neuronas completamente conectadas organizadas en tres tipos de capas: capa de entrada, capas ocultas y capa de salida.

Estructura y operación. Cada neurona en una capa recibe como entrada una combinación lineal ponderada de las salidas de todas las neuronas de la capa anterior, seguida por la aplicación de una función de activación no lineal. Formalmente, la salida de la capa l se calcula como:

$$\mathbf{h}^{(l)} = \sigma\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right), \tag{2.3}$$

donde $\mathbf{h}^{(l-1)} \in \mathbb{R}^{d_{l-1}}$ es la entrada de la capa, $\mathbf{W}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ es la matriz de pesos, $\mathbf{b}^{(l)} \in \mathbb{R}^{d_l}$ es el vector de sesgos, $\sigma(\cdot)$ es la función de activación, y $\mathbf{h}^{(l)} \in \mathbb{R}^{d_l}$ es la salida de la capa. La primera capa recibe las características de entrada $\mathbf{h}^{(0)} = \mathbf{x}$, y la última capa produce la predicción $\hat{y} = \mathbf{h}^{(L)}$.

Funciones de activación. Las funciones de activación introducen no linealidad en

el modelo, permitiendo que la red aprenda relaciones complejas. Las funciones más utilizadas incluyen:

- **ReLU** (Rectified Linear Unit): $\sigma(z) = \text{máx}(0, z)$. Es la más utilizada en capas ocultas debido a su simplicidad computacional y efectividad para mitigar el problema de gradientes desvanecientes.
- **Sigmoid**: $\sigma(z) = 1/(1 + e^{-z})$. Mapea valores a $(0, 1)$, útil para probabilidades en clasificación binaria.
- **Softmax**: $\sigma(\mathbf{z})_i = e^{z_i} / \sum_j e^{z_j}$. Generaliza sigmoid para clasificación multiclase, produciendo una distribución de probabilidad sobre clases.
- **Tanh**: $\sigma(z) = \tanh(z)$. Similar a sigmoid pero centrada en cero, con rango $(-1, 1)$.

Concepto de *capa* en redes neuronales. Una capa representa una transformación paramétrica de la información. En redes profundas, múltiples capas se apilan secuencialmente, donde cada capa refina progresivamente las representaciones aprendidas. La profundidad (número de capas) determina la capacidad del modelo para aprender jerarquías de características: capas iniciales típicamente capturan patrones simples, mientras que capas profundas combinan estos patrones en representaciones abstractas más complejas. Este concepto de procesamiento jerárquico es fundamental también en las redes neuronales en grafos, donde múltiples capas de agregación permiten capturar información de vecindades de mayor alcance en el grafo.

2.3.2 Funciones de Pérdida y Optimización

El entrenamiento supervisado de redes neuronales consiste en ajustar los parámetros $\theta = \{\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, ..., \mathbf{W}^{(L)}, \mathbf{b}^{(L)}\}$ para minimizar una función de pérdida que cuantifica la discrepancia entre las predicciones del modelo y los valores verdaderos.

Funciones de pérdida para regresión. En problemas de regresión, donde el objetivo es predecir valores continuos, las funciones de pérdida más comunes son:

- **Error Cuadrático Medio (MSE)**: $\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$. Penaliza cuadráticamente los errores, siendo sensible a valores atípicos.
- **Error Absoluto Medio (MAE)**: $\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|$. Más robusta a outliers que MSE.

Funciones de pérdida para clasificación. En clasificación, donde el objetivo es asignar instancias a categorías discretas, las más usadas son:

- **Entropía Cruzada Binaria**: Para clasificación binaria, $\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$, donde $y_i \in \{0,1\}$ es la etiqueta verdadera y $\hat{y}_i \in (0,1)$ es la probabilidad predicha.

- **Entropía Cruzada Categórica**: Para clasificación multi-clase con C clases, $\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic}\log(\hat{y}_{ic})$, donde y_{ic} es un indicador one-hot de la clase verdadera y $\hat{y}_{ic}$ es la probabilidad predicha para la clase c.

Optimización mediante descenso de gradiente. Los parámetros se optimizan mediante variantes de descenso de gradiente, que iterativamente ajustan los parámetros en la dirección opuesta al gradiente de la pérdida:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta_t),$$

donde η es la tasa de aprendizaje y $\nabla_\theta \mathcal{L}$ es el gradiente de la pérdida respecto a los parámetros. En la práctica, se utilizan optimizadores adaptativos como Adam o RMSprop que ajustan automáticamente las tasas de aprendizaje por parámetro.
Métricas de evaluación. Además de la función de pérdida utilizada durante el entrenamiento, se emplean diversas métricas para evaluar el desempeño del modelo:

- Para regresión: RMSE (raíz del MSE), MAE, coeficiente de determinación R^2.

- Para clasificación binaria: Accuracy, Precision, Recall, F1-score (media armónica de precision y recall), AUC-ROC.

- Para clasificación multi-clase: Accuracy, F1-score macro/micro, matriz de confusión.

Estos fundamentos de aprendizaje supervisado son directamente aplicables a redes neuronales en grafos, donde la arquitectura se extiende para operar sobre estructuras de grafos en lugar de datos euclidianos, pero los principios de optimización y evaluación permanecen esencialmente los mismos.

2.4 Métodos Basados en Aprendizaje Profundo

Las redes neuronales en grafos (*Graph Neural Networks*, GNNs) han surgido como un enfoque poderoso para superar las limitaciones de los métodos tradicionales, permitiendo aprender de la estructura de las vecindades en un grafo (Zhou y col., 2020). Estos modelos generan nuevas representaciones para las entidades basadas en sus conexiones dentro del grafo, aprovechando las relaciones inherentes entre nodos y aristas. A continuación, se describen en detalle los enfoques principales y recientes avances en este campo.

2.4.1 Métodos Espectrales para GNNs

Spectral CNN: Convoluciones en el dominio espectral.

A diferencia de los métodos espectrales clásicos que utilizan la descomposición espectral como técnica de análisis estática (como Laplacian Eigenmaps), los métodos espectrales para GNNs definen *filtros espectrales parametrizados* que se aprenden mediante backpropagation. La idea fundamental es realizar convoluciones en grafos mediante operaciones en el dominio espectral, inspiradas en convoluciones en señales sobre dominios euclidianos (Bruna y col., 2013; Defferrard y col., 2016).
Formalmente, una señal en el grafo $\mathbf{x} \in \mathbb{R}^n$ puede transformarse al dominio espectral mediante los eigenvectores de la matriz Laplaciana: $\hat{\mathbf{x}} = \mathbf{U}^\top \mathbf{x}$, donde $\mathbf{U}$ contiene los eigenvectores. Un filtro espectral parametrizado g_θ se aplica en el dominio espectral y luego se transforma de vuelta al dominio espacial:

$$g_\theta \star \mathbf{x} = \mathbf{U} g_\theta(\Lambda) \mathbf{U}^\top \mathbf{x},$$

donde Λ es la matriz diagonal de eigenvalues, y $g_\theta(\Lambda)$ es una función de los eigenvalues con parámetros aprendibles θ. Los parámetros θ se optimizan mediante descenso de gradiente para minimizar una función de pérdida supervisada, lo que diferencia fundamentalmente este enfoque de los métodos espectrales clásicos que simplemente calculan embeddings mediante descomposición espectral.
Limitaciones. Este enfoque directo hereda las mismas limitaciones computacionales que los métodos espectrales clásicos: requiere calcular la descomposición completa de la matriz Laplaciana ($O(n^3)$), y los filtros aprendidos dependen de la base de eigenvectores del grafo específico, lo que impide la transferencia a grafos con estructuras diferentes (Shuman y col., 2013; Z. Wu y col., 2019).

ChebNet: Aproximación mediante polinomios de Chebyshev.

Para abordar estas limitaciones, Defferrard et al. (Defferrard y col., 2016) propusieron aproximar los filtros espectrales mediante polinomios de Chebyshev, evitando la necesidad de calcular explícitamente los eigenvectores. La aproximación se basa en expandir $g_\theta(\Lambda)$ como un polinomio de grado K:

$$g_\theta(\mathbf{L})\mathbf{x} \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{\mathbf{L}})\mathbf{x},$$

donde T_k son los polinomios de Chebyshev de orden k, $\tilde{\mathbf{L}} = 2\mathbf{L}/\lambda_{max} - \mathbf{I}$ es la matriz Laplaciana normalizada escalada al intervalo $[-1, 1]$, y $\theta = [\theta_0, ..., \theta_K]$ son los parámetros aprendibles. Los polinomios de Chebyshev se calculan recursivamente: $T_0(\tilde{\mathbf{L}}) = \mathbf{I}$, $T_1(\tilde{\mathbf{L}}) = \tilde{\mathbf{L}}$, y $T_k(\tilde{\mathbf{L}}) = 2\tilde{\mathbf{L}} T_{k-1}(\tilde{\mathbf{L}}) - T_{k-2}(\tilde{\mathbf{L}})$.

Esta aproximación tiene complejidad $O(K|\mathcal{E}|)$, lineal en el número de aristas, y permite operar directamente sobre la estructura del grafo sin calcular eigenvectores. Un filtro ChebNet con K parámetros tiene un receptive field de K saltos: captura información hasta K vecindades de distancia.

GCN (Graph Convolutional Network). Kipf y Welling (Kipf & Welling, 2016a) simplificaron ChebNet limitando $K = 1$ y aproximando $\lambda_{max} \approx 2$, resultando en una capa convolucional extremadamente eficiente que se ha convertido en una de las arquitecturas GNN más utilizadas. Esta simplificación se discutirá en detalle en la siguiente subsección sobre métodos espaciales, ya que la formulación resultante puede interpretarse tanto desde una perspectiva espectral como espacial.

2.4.2 Métodos Espaciales

Los métodos espaciales definen las operaciones de convolución directamente sobre la estructura del grafo, agregando información de los nodos vecinos (Zhou y col., 2020). A diferencia de los métodos espectrales, estos modelos operan en el dominio espacial, calculando representaciones para cada nodo mediante un proceso iterativo de agregación.

Concepto de capa en GNNs. En redes neuronales en grafos, el concepto de *capa* adquiere un significado estructural específico relacionado con la topología del grafo. Cada capa de una GNN realiza una operación de agregación que combina la representación de un nodo con las representaciones de sus vecinos directos. La notación $\mathbf{h}_i^{(l)}$ denota la representación (embedding) del nodo i después de l capas de agregación. En la capa inicial ($l = 0$), $\mathbf{h}_i^{(0)}$ corresponde a las características originales del nodo (por ejemplo, atributos de entrada). En cada capa subsiguiente, la representación se refina incorporando información de la vecindad:

- **Capa 1** ($l = 1$): $\mathbf{h}_i^{(1)}$ agrega información de los vecinos inmediatos (1-hop neighbors) de i.
- **Capa 2** ($l = 2$): $\mathbf{h}_i^{(2)}$ agrega información de vecinos a 2 saltos, es decir, los vecinos de los vecinos.
- **Capa** l: $\mathbf{h}_i^{(l)}$ captura información de todos los nodos dentro de una distancia de l saltos desde i.

Este proceso define el *receptive field* (campo receptivo) de un nodo: una GNN con L capas tiene un receptive field de L saltos, lo que significa que la representación final de un nodo depende de todos los nodos alcanzables en L pasos. Apilar múltiples capas permite que la red capture dependencias estructurales a mayor distancia en el grafo, de manera análoga a cómo redes convolucionales profundas en imágenes capturan patrones jerárquicos a diferentes escalas espaciales.

GCN: Graph Convolutional Network. La arquitectura GCN (Kipf & Welling, 2016a) es uno de los métodos espaciales más utilizados (ver Figura 2.3). Su formulación surge como simplificación de ChebNet (discutida en la sección anterior), pero puede interpretarse directamente desde una perspectiva espacial como una regla de agregación normalizada. La actualización de representaciones en una capa GCN se define como:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}(i)\cup\{i\}} \frac{1}{\sqrt{\tilde{d}_i\tilde{d}_j}}\mathbf{W}^{(l)}\mathbf{h}_j^{(l)}\right),$$

donde $\mathbf{h}_i^{(l)}$ representa la característica del nodo i en la capa l, $\mathcal{N}(i)\cup\{i\}$ incluye tanto los vecinos de i como el propio nodo (mediante el uso de self-loops en el grafo), $\tilde{d}_i$ es el grado del nodo i en el grafo con self-loops agregados, $\mathbf{W}^{(l)}$ son los parámetros aprendibles de la capa l, y σ es una función de activación no lineal (típicamente ReLU). El término de normalización $\frac{1}{\sqrt{\tilde{d}_i\tilde{d}_j}}$ proviene de la normalización simétrica de la matriz Laplaciana y asegura que la escala de las representaciones se mantenga estable durante el entrenamiento.

La operación GCN puede expresarse en forma matricial para todo el grafo como:

$$\mathbf{H}^{(l+1)} = \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right),$$

donde $\mathbf{H}^{(l)} \in \mathbb{R}^{n\times d}$ contiene las representaciones de todos los nodos en la capa l, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ es la matriz de adyacencia con self-loops, y $\tilde{\mathbf{D}}$ es la matriz de grados correspondiente. Esta formulación tiene complejidad $O(|\mathcal{E}| \cdot d \cdot d')$, donde d es la dimensión de entrada y d' la dimensión de salida, haciéndola altamente escalable para grafos grandes siempre que sean dispersos (sparse).

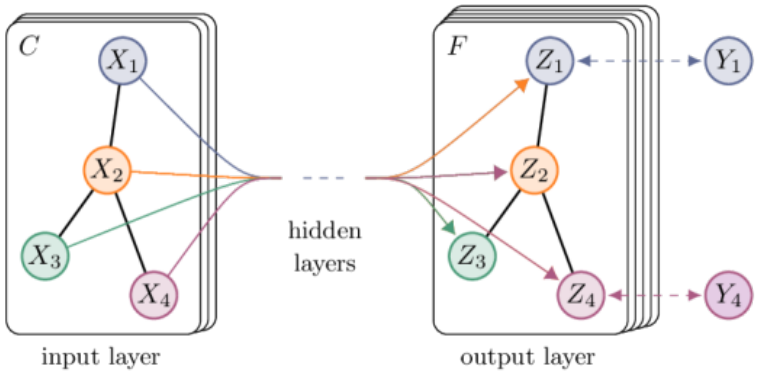

Figura 2.3: Representación esquemática de una Graph Convolutional Network (GCN) multi-capa para aprendizaje semi-supervisado, con C canales de entrada y F mapas de características en la capa de salida. Las líneas negras representan las conexiones del grafo que se mantienen consistentes a través de todas las capas convolucionales. Adaptado de (Kipf & Welling, 2016a).

GraphSAGE: Inductive learning con diferentes agregadores. GraphSAGE (SAmple and aggreGatE) (Hamilton y col., 2017) extiende el enfoque espacial introduciendo muestreo de vecindades y múltiples funciones de agregación, habilitando aprendizaje inductivo eficiente. A diferencia de GCN, que agrega información de todos los vecinos, GraphSAGE muestrea un número fijo de vecinos en cada capa, controlando la complejidad computacional. La actualización de representaciones sigue un esquema de dos pasos:

$$\mathbf{m}_{\mathcal{N}(i)}^{(l)} = \text{AGGREGATE}^{(l)}\left(\{\mathbf{h}_j^{(l)} : j \in \mathcal{N}(i)\}\right), \tag{2.4}$$

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{W}^{(l)} \cdot \text{CONCAT}\left(\mathbf{h}_i^{(l)}, \mathbf{m}_{\mathcal{N}(i)}^{(l)}\right)\right), \tag{2.5}$$

donde AGGREGATE puede ser una de varias funciones:

- **Mean aggregator**: $\mathbf{m}_{\mathcal{N}(i)}^{(l)} = \frac{1}{|\mathcal{N}(i)|}\sum_{j\in\mathcal{N}(i)} \mathbf{h}_j^{(l)}$
- **Max aggregator**: $\mathbf{m}_{\mathcal{N}(i)}^{(l)} = \text{máx}_{j\in\mathcal{N}(i)}\left\{\sigma\left(\mathbf{W}_{\text{pool}}\mathbf{h}_j^{(l)} + \mathbf{b}\right)\right\}$
- **LSTM aggregator**: Aplica una LSTM sobre una permutación aleatoria de los vecinos.

La operación CONCAT concatena la representación del nodo central con la información agregada de sus vecinos, preservando explícitamente la información del nodo antes de la agregación. GraphSAGE normaliza las representaciones usando ℓ_2 normalization después de cada capa: $\mathbf{h}_i^{(l)} \leftarrow \mathbf{h}_i^{(l)}/\|\mathbf{h}_i^{(l)}\|_2$, lo que mejora la estabilidad del entrenamiento.

Ventajas de métodos espaciales sobre espectrales. Los métodos espaciales presentan ventajas significativas frente a los enfoques espectrales clásicos:

- **Escalabilidad**: Complejidad lineal en el número de aristas $O(|\mathcal{E}|)$, frente a $O(n^3)$ de descomposición espectral.
- **Aprendizaje inductivo**: Los pesos aprendidos pueden aplicarse a grafos nuevos sin re-entrenamiento, ya que las operaciones se definen localmente sobre vecindades.
- **Flexibilidad**: Permiten incorporar características de nodos y aristas directamente, sin requerir propiedades espectrales específicas del grafo.
- **Paralelización**: Las agregaciones en diferentes nodos son independientes y pueden ejecutarse en paralelo eficientemente en GPUs.

Estas propiedades han convertido a los métodos espaciales en la base de la mayoría de las arquitecturas GNN modernas utilizadas en producción.

2.4.3 Graph Attention Networks (GAT)

Las *Graph Attention Networks* (GAT) (Veličković y col., 2018) introducen mecanismos de atención en GNNs, permitiendo que cada nodo asigne pesos adaptativos a sus vecinos según su relevancia. A diferencia de GCN y GraphSAGE, que utilizan esquemas de agregación fijos (normalización por grado o promedios uniformes), GAT aprende dinámicamente la importancia relativa de cada vecino mediante un mecanismo de atención paramétrico. Esta capacidad de ponderar selectivamente las conexiones resulta particularmente útil en grafos donde diferentes vecinos tienen diferentes niveles de relevancia para la tarea objetivo.

Mecanismo de atención en GAT. El núcleo de GAT es su mecanismo de *self-attention* aplicado a las vecindades de los nodos (ilustrado en la Figura 2.4). Para cada nodo i, se calculan coeficientes de atención α_{ij} que cuantifican la importancia del nodo vecino j para el nodo central i. El proceso consta de tres pasos principales:

1. Cálculo de coeficientes de atención. Primero, se transforma linealmente la representación de cada nodo mediante una matriz de pesos compartida $\mathbf{W} \in \mathbb{R}^{d' \times d}$:

$$\mathbf{h}'_i = \mathbf{W}\mathbf{h}_i$$

Luego, se calcula un *attention score* no normalizado e_{ij} entre el nodo i y cada vecino $j \in \mathcal{N}(i)$ usando un mecanismo de atención paramétrico $a : \mathbb{R}^{d'} \times \mathbb{R}^{d'} \to \mathbb{R}$:

$$e_{ij} = a\left(\mathbf{h}'_i, \mathbf{h}'_j\right)$$

En la formulación original de GAT, a se implementa como una capa feed-forward de una sola neurona con activación LeakyReLU:

$$e_{ij} = \text{LeakyReLU}\left(\mathbf{a}^\top \left[\mathbf{h}'_i \,\text{CONCAT}\, \mathbf{h}'_j\right]\right),$$

donde $\mathbf{a} \in \mathbb{R}^{2d'}$ es un vector de pesos aprendible.

2. Normalización mediante softmax. Los coeficientes de atención se normalizan usando la función softmax para que sumen 1 sobre todos los vecinos:

$$\alpha_{ij} = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})}$$

Esta normalización permite interpretar α_{ij} como la importancia relativa del nodo j para el nodo i. Es importante notar que la atención es asimétrica: $\alpha_{ij} \neq \alpha_{ji}$ en general, reflejando que la importancia de j para i puede diferir de la importancia de i para j.

3. Agregación ponderada. Finalmente, la nueva representación del nodo i se calcula como una combinación lineal ponderada de las representaciones transformadas

de sus vecinos:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}\mathbf{W}^{(l)}\mathbf{h}_j^{(l)}\right)$$

donde σ es una función de activación no lineal (típicamente ELU o ReLU).

Multi-head attention. Para estabilizar el proceso de aprendizaje y capturar diferentes aspectos de las relaciones entre nodos, GAT emplea *multi-head attention*, análogo al mecanismo usado en Transformers (Vaswani y col., 2017). Se computan K mecanismos de atención independientes en paralelo (llamados "cabezas" o heads), cada uno con sus propios parámetros $\mathbf{W}^k$ y $\mathbf{a}^k$:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^k\mathbf{W}^{k,(l)}\mathbf{h}_j^{(l)}\right)$$

o alternativamente, mediante concatenación de las salidas de cada cabeza:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\Big\|_{k=1}^{K}\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^k\mathbf{W}^{k,(l)}\mathbf{h}_j^{(l)}\right),$$

donde $\|$ denota concatenación. La versión con concatenación se utiliza típicamente en capas intermedias, mientras que el promedio se usa en la última capa para producir embeddings de tamaño fijo.

Ventajas de GAT. La introducción de mecanismos de atención proporciona varios beneficios importantes:

- **Agregación adaptativa**: Los pesos de atención se aprenden específicamente para la tarea, permitiendo que el modelo identifique qué vecinos son más informativos.
- **Interpretabilidad**: Los coeficientes α_{ij} proporcionan información sobre qué conexiones son relevantes, facilitando la interpretación del modelo.
- **Generalización a grafos inductivos**: Al igual que GraphSAGE, GAT puede aplicarse a grafos no vistos durante el entrenamiento, ya que la atención se calcula dinámicamente basándose en las características de los nodos.
- **Grafos con grados variables**: A diferencia de GCN que normaliza uniformemente por grado, GAT puede manejar naturalmente nodos con muy diferentes números de vecinos sin requerir normalización explícita.

Comparación con agregación uniforme. Para ilustrar la diferencia entre GAT y métodos con agregación uniforme, consideremos un nodo i con tres vecinos. En

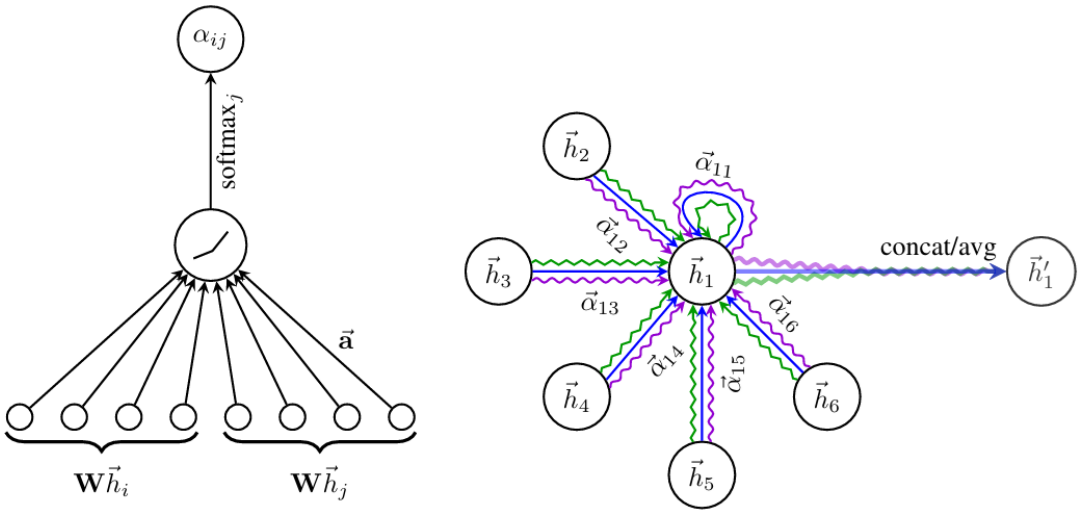

Figura 2.4: Mecanismo de atención en Graph Attention Networks (GAT). **Izquierda:** El mecanismo de atención $a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j)$ empleado por el modelo, parametrizado por un vector de pesos $\mathbf{a} \in \mathbb{R}^{2F'}$, aplicando activación LeakyReLU. Las características transformadas $\mathbf{W}\vec{h}_i$ y $\mathbf{W}\vec{h}_j$ se concatenan y pasan por la red de atención para producir coeficientes normalizados α_{ij} mediante softmax. **Derecha:** Ilustración de multi-head attention con $K = 3$ cabezas independientes aplicadas por el nodo 1 sobre su vecindad. Cada cabeza (representada con diferentes colores/estilos de flechas) computa sus propios coeficientes de atención, y las salidas se concatenan o promedian para producir la representación final $\vec{h}'_1$. Adaptado de (Veličković y col., 2018).

GCN, cada vecino contribuye igualmente (normalizado por grado): si i tiene grado 3, cada vecino aporta $\frac{1}{\sqrt{3 \cdot d_j}}$ de su información. En GraphSAGE con mean aggregator, cada vecino contribuye exactamente $\frac{1}{3}$. En contraste, en GAT los coeficientes de atención podrían ser, por ejemplo, $\alpha_{i1} = 0{,}7$, $\alpha_{i2} = 0{,}2$, $\alpha_{i3} = 0{,}1$, reflejando que el vecino 1 es mucho más relevante para el nodo i que los otros dos. Esta flexibilidad es especialmente valiosa en grafos heterogéneos o en tareas donde ciertas conexiones son ruido estructural.

GAT ha sido extendida en trabajos posteriores, incluyendo GATv2 (Brody y col., 2021), que mejora la expresividad del mecanismo de atención permitiendo que el attention score sea una función más general, y Graph Transformers que generalizan la atención a grafos completos mediante codificaciones posicionales especializadas (Rampášek y col., 2022; Ying y col., 2021).

2.4.4 Redes Basadas en Paso de Mensajes (MPNNs)

El enfoque de *Message Passing Neural Networks* (MPNNs) unifica varias arquitecturas bajo un marco general que consta de tres fases principales: paso de mensajes, actualización de nodos y lectura (Gilmer y col., 2017, 2020). Las MPNNs son una clase de GNN diseñadas para manejar tareas basadas en grafos mediante el inter-

cambio iterativo de mensajes entre nodos y la actualización de las representaciones de los nodos en función de los mensajes recibidos de su vecindad local. Este enfoque resulta particularmente poderoso para capturar dependencias estructurales dentro de los grafos, haciéndolas altamente versátiles en aplicaciones que van desde la predicción de interacciones proteína-proteína hasta el análisis de redes de regulación génica (Duvenaud y col., 2015; Kearnes y col., 2016).
Formalmente, las MPNNs operan a través de tres fases principales: paso de mensajes, actualización de nodos y lectura. En el paso t, las características $\mathbf{h}_p^{(t)}$ del nodo p se actualizan al agregar mensajes de sus nodos vecinos según:

$$m_p^{(t+1)} = \sum_{q \in \mathcal{N}(p)} M_t \left(\mathbf{h}_p^{(t)}, \mathbf{h}_q^{(t)}, e_{pq} \right) \tag{2.6}$$

donde $\mathcal{N}(p)$ denota el conjunto de vecinos del nodo p, y M_t es la función de mensajes que depende tanto de las características de los nodos como de las características de las aristas e_{pq}. Las características del nodo central p se actualizan utilizando la información de sus nodos vecinos:

$$\mathbf{h}_p^{(t+1)} = U_t \left(\mathbf{h}_p^{(t)}, m_p^{(t+1)} \right) \tag{2.7}$$

donde U_t es la función de actualización del nodo, que típicamente combina las características de los nodos vecinos con los mensajes agregados, utilizando esquemas como suma, media o máximo para actualizar la representación de cada nodo. Después de varias iteraciones del paso de mensajes, los estados finales de los nodos pueden agregarse utilizando una función de lectura:

$$\hat{y} = R \left(\{ \mathbf{h}_p^{(K)} | p \in \mathcal{G} \} \right) \tag{2.8}$$

La función de lectura R agrega información de todos los nodos para producir una representación a nivel de grafo o subgrafo $\hat{y}$, que puede utilizarse en tareas de clasificación o regresión relacionadas con el grafo, los nodos o las aristas. Esta arquitectura ha demostrado ser altamente efectiva en muchas aplicaciones, aunque presenta limitaciones al abordar tareas centradas en aristas, que implican aprender relaciones entre pares de nodos en lugar de características individuales de los nodos.

2.4.5 Aprendizaje Auto-Supervisado en Grafos

El aprendizaje auto-supervisado ha emergido como un paradigma fundamental para entrenar modelos robustos en grafos, especialmente en escenarios donde las etiquetas supervisadas son escasas o costosas de obtener (L. Wu y col., 2022). Este enfoque aprovecha la estructura intrínseca del grafo y las relaciones entre nodos para crear

tareas de aprendizaje auxiliares que no requieren anotaciones manuales.
Los métodos de aprendizaje contrastivo han mostrado particular éxito en este ámbito. You et al. (You y col., 2020) propusieron Graph Contrastive Learning (GraphCL), que utiliza aumentaciones de grafos para crear vistas positivas y negativas, permitiendo que el modelo aprenda representaciones invariantes bajo transformaciones estructurales preservando la información semántica. Este enfoque ha demostrado mejoras significativas en tareas de clasificación de nodos y grafos.
El trabajo de Thakoor et al. (Thakoor y col., 2022) introdujo BGRL (Bootstrapped Graph Latents), un método que evita la necesidad de ejemplos negativos mediante el uso de técnicas de bootstrapping. Esta aproximación entrena dos redes que se supervisan mutuamente, donde una red genera objetivos para la otra, creando un proceso de aprendizaje auto-reforzante que ha mostrado escalabilidad excepcional en grafos de gran tamaño.
Los paradigmas híbridos que combinan aprendizaje supervisado y auto-supervisado han demostrado particular efectividad (Goodfellow y col., 2016). Estos enfoques utilizan señales auto-supervisadas para regularizar el entrenamiento supervisado, mejorando la generalización y robustez del modelo, especialmente en presencia de características de nodos limitadas o ruidosas. Esta estrategia es particularmente relevante en aplicaciones bioinformáticas donde la información estructural del grafo puede compensar la ausencia de características detalladas de los nodos.

2.4.6 Desafíos y Perspectivas Futuras

A pesar de los avances significativos en las redes neuronales en grafos, persisten desafíos importantes, particularmente en tareas centradas en aristas (*edge-centric tasks*). Estas tareas, que incluyen la regresión y clasificación de aristas, así como la predicción de enlaces, son fundamentales en aplicaciones como bioinformática, donde las interacciones entre entidades ocurren principalmente en las aristas (por ejemplo, interacciones proteína-proteína o relaciones en redes metabólicas) (Fout y col., 2017; Zitnik y col., 2017). Sin embargo, muchos modelos tradicionales de GNN, incluidas las MPNNs y las GATs, están diseñados principalmente para capturar características a nivel de nodo, subutilizando la información contenida en las aristas (Gilmer y col., 2017; Veličković y col., 2018).
Un desafío crítico es diseñar arquitecturas que integren de manera efectiva las características de las aristas en el proceso de aprendizaje, permitiendo modelar relaciones complejas entre nodos. Además, las tareas centradas en aristas enfrentan problemas relacionados con la escalabilidad, especialmente en grafos densos, donde el número de aristas puede crecer cuadráticamente con respecto al número de nodos.
Otro desafío relevante es la falta de bibliografía y benchmarks específicos para ta-

reas centradas en aristas, lo que dificulta la comparación sistemática de modelos. Mientras que existen abundantes recursos para tareas a nivel de nodo o grafo, las tareas a nivel de arista siguen siendo menos exploradas, lo que representa una oportunidad para desarrollar metodologías y evaluaciones estandarizadas (Rossi y col., 2020; Zitnik y col., 2017).

Esta tesis aborda estos desafíos al proponer modelos que priorizan las aristas como unidades centrales de análisis, integrando tanto características estructurales como atributos de nodos y aristas. En este contexto, se exploran las combinaciones de mecanismos de atención y aprendizaje auto-supervisado para mejorar la representación y predicción en estas tareas.

2.5 Comentarios Finales

Este capítulo presentó los fundamentos teóricos del análisis de grafos mediante técnicas de aprendizaje automático, organizados en tres categorías principales: métodos clásicos, fundamentos de aprendizaje supervisado, y arquitecturas neuronales en grafos.

Los **métodos clásicos** (Sección 2.2) se dividen en enfoques espectrales, basados en descomposición de matrices Laplacianas con complejidad $O(n^3)$, y métodos de caminatas aleatorias como DeepWalk y Node2Vec, que operan mediante muestreo local. Estos métodos enfrentan limitaciones de escalabilidad y generalización a nuevos nodos, motivando el desarrollo de enfoques basados en aprendizaje profundo.

Los **fundamentos de aprendizaje supervisado** (Sección 2.3) establecieron conceptos esenciales: arquitecturas de perceptrón multicapa, funciones de pérdida (MSE, MAE, entropía cruzada), y el concepto de capa como unidad de transformación paramétrica. Estos elementos son fundamentales para comprender las arquitecturas GNN subsiguientes.

Las **redes neuronales en grafos** (Sección 2.4) se clasificaron según su dominio de operación. Los métodos espectrales para GNNs (ChebNet, GCN desde perspectiva espectral) definen filtros parametrizados aprendibles mediante backpropagation, diferenciándose de los métodos espectrales clásicos estáticos. Los métodos espaciales (GCN, GraphSAGE, GAT) operan directamente sobre vecindades locales, ofreciendo escalabilidad lineal $O(|\mathcal{E}|)$ y capacidad inductiva. Las Graph Attention Networks introducen agregación adaptativa mediante mecanismos de atención, aprendiendo ponderaciones específicas para cada conexión. El marco de Message Passing Neural Networks unifica estas arquitecturas bajo un esquema general de paso de mensajes y actualización de estados.

El **aprendizaje auto-supervisado** (Sección 2.4.5) ha emergido como paradigma complementario al aprendizaje supervisado, utilizando señales de la estructura del

grafo para regularizar el entrenamiento. Los enfoques híbridos que combinan objetivos supervisados y auto-supervisados son particularmente relevantes en aplicaciones bioinformáticas donde las características de nodos pueden ser limitadas.

Los **desafíos en tareas centradas en aristas** (Sección 2.4.6) constituyen el problema central de esta tesis. Las arquitecturas tradicionales de GNN priorizan representaciones de nodos, subutilizando información en las aristas. Las tareas de regresión y clasificación de aristas, fundamentales en bioinformática (interacciones proteína-proteína, redes metabólicas), requieren arquitecturas especializadas que integren efectivamente características de aristas en el proceso de aprendizaje.

La progresión metodológica de los capítulos experimentales sigue una complejidad creciente. El Capítulo 3 establece una línea base con perceptrones multicapa y codificación one-hot, sin explotar la estructura del grafo. El Capítulo 4 incorpora embeddings Node2Vec que capturan topología de la red metabólica. Los capítulos subsiguientes desarrollan arquitecturas GNN especializadas que integran paso de mensajes, mecanismos de atención, y aprendizaje auto-supervisado para tareas centradas en aristas. Este desarrollo incremental permite evaluar sistemáticamente la contribución de cada componente arquitectural al desempeño predictivo.

Capítulo 3

Construcción de Ground Truth para Similaridad Molecular

3.1 Introducción

El desarrollo de métodos de aprendizaje automático para predecir similaridad molecular requiere, como primer paso fundamental, contar con datos de similaridad conocidos y confiables que sirvan como ground truth durante el entrenamiento. La estimación de similaridad molecular es un problema central en bioinformática con aplicaciones en ingeniería metabólica, predicción de función metabólica y análisis de redes bioquímicas (Willett y col., 1998). Sin embargo, existen múltiples formas de calcular esta similaridad, cada una con diferentes propiedades y sesgos, lo que plantea la pregunta: *¿qué representación molecular es más apropiada para construir el dataset de entrenamiento?*

Tradicionalmente, la similaridad molecular se calcula mediante descriptores estructurales conocidos como *fingerprints moleculares* (Willett y col., 1998). Estos métodos codifican características químicas de los compuestos en vectores binarios que pueden compararse mediante índices como el coeficiente de Tanimoto. Existen diversas familias de fingerprints (estructurales, basadas en hash, topológicas) que capturan diferentes aspectos de la estructura molecular, y la elección del método puede afectar significativamente los valores de similaridad obtenidos. Adicionalmente, como se discutió en el Capítulo 1, un desafío importante surge cuando la estructura molecular de algunos compuestos no está completamente disponible o contiene elementos genéricos (por ejemplo, sustituyentes representados como "-R"), situación en la cual los métodos tradicionales de fingerprints no pueden aplicarse directamente.

Este capítulo aborda el problema de **seleccionar la representación molecular más adecuada para construir el ground truth de similaridad**, que será utilizado como objetivo de aprendizaje en los capítulos subsiguientes. El trabajo se

organiza en dos etapas complementarias:

1. **Estudio comparativo de fingerprints moleculares**: Se evalúan cualitativamente 6 métodos de fingerprints ampliamente utilizados (2 estructurales y 4 basados en hash) para determinar cuál produce valores de similaridad más apropiados según criterios de distribución, separabilidad y coherencia química. Este análisis permitirá fundamentar empíricamente la elección del método para construir el dataset de entrenamiento.

2. **Validación con aprendizaje supervisado**: Una vez seleccionada la fingerprint más apropiada, se valida que los valores de similaridad obtenidos son útiles para entrenar modelos de aprendizaje automático. Para esto, se utiliza un perceptrón multicapa (MLP) simple que recibe como entrada representaciones básicas de los compuestos (codificación one-hot) y aprende a predecir el índice de Tanimoto calculado con la fingerprint seleccionada. Este experimento demuestra la viabilidad del aprendizaje supervisado para la tarea y establece una línea base de desempeño para métodos más sofisticados.

Es importante enfatizar que el objetivo de este capítulo **no es desarrollar un sistema de predicción de similaridad que reemplace las fingerprints tradicionales**, sino más bien establecer la infraestructura metodológica necesaria para los experimentos de los capítulos posteriores. El modelo MLP con codificación one-hot se utiliza como herramienta de validación: si este modelo simple logra aprender a predecir similaridad a partir de identificadores básicos de compuestos, esto confirma que el ground truth seleccionado contiene patrones aprendibles mediante redes neuronales. Los capítulos subsiguientes explorarán representaciones más sofisticadas basadas en la topología de la red metabólica (Capítulo 4) y arquitecturas neuronales especializadas en grafos (Capítulo 5), pero todos utilizarán como objetivo de aprendizaje la medida de similaridad establecida en este capítulo.
La organización de este capítulo es la siguiente:

- **Descripción de fingerprints**: Se presentan los 6 métodos de fingerprints moleculares evaluados, explicando sus principios de funcionamiento y diferencias fundamentales.

- **Selección de fingerprint**: Se realiza un análisis comparativo cualitativo mediante histogramas de distribución y matrices de similaridad para seleccionar el método más apropiado.

- **Modelo de validación**: Se describe el perceptrón multicapa utilizado para validar la utilidad del ground truth seleccionado.

- **Construcción del dataset**: Se detalla la construcción del conjunto de datos de entrenamiento a partir de la vía metabólica de la Glucólisis.
- **Resultados**: Se presentan los resultados del análisis comparativo de fingerprints y del experimento de validación con el MLP.
- **Conclusiones**: Se discuten las implicancias de los hallazgos y se establece la conexión con los capítulos subsiguientes.

3.1.1 Descripción de fingerprints

Las fingerprints moleculares son representaciones vectoriales que codifican características estructurales de compuestos químicos. Para entrenar el modelo neuronal, es necesario calcular valores ground truth de similaridad entre compuestos, lo cual requiere seleccionar un método de cálculo de fingerprints apropiado. En este trabajo se evaluaron 6 métodos de construcción de fingerprints ampliamente utilizados en la literatura, clasificados en dos familias principales: estructurales (2 métodos) y basadas en códigos de hash (4 métodos).

Fingerprints Estructurales

En las fingerprints estructurales, diferentes aspectos de los compuestos llamados "claves" son codificados en un vector binario: cada bit corresponde a una subestructura o propiedad molecular predefinida. Si la molécula contiene la característica correspondiente, el bit de la posición asociada se establece en 1; caso contrario, en 0. Como se observa en la Figura 3.1a, este enfoque permite representar la presencia o ausencia de subestructuras específicas de manera binaria.
Cada clave representa una característica química específica, que puede ser:

- **Grupos funcionales**: Grupos carboxilo (-COOH), amino (-NH_2), hidroxilo (-OH), carbonilo (C=O), éster, amida, etc.
- **Anillos**: Anillos aromáticos de 5 o 6 miembros, anillos saturados, anillos fusionados, heterociclos (conteniendo N, O, S).
- **Heteroátomos y enlaces**: Presencia de nitrógeno, oxígeno, azufre, halógenos; enlaces dobles, triples, aromáticos.
- **Propiedades estructurales**: Número de enlaces rotatorios, número de donantes/aceptores de puentes de hidrógeno, peso molecular, número de átomos de carbono.

Por ejemplo, si un compuesto como el ácido pirúvico ($C_3H_4O_3$, con estructura CH_3-CO-COOH) se procesa con MACCS keys, se activarían claves correspondientes a:

grupo carboxilo, grupo carbonilo, presencia de oxígeno, número de carbonos = 3, entre otras. Cada una de estas características activa su bit correspondiente en el vector de 166 posiciones.
Los métodos estructurales evaluados incluyen:

- **MACCS keys** (Durant y col., 2002b): Molecular ACCess System keys, versión pública con 166 claves predefinidas que codifican fragmentos estructurales y propiedades moleculares comunes. Este conjunto fue diseñado para capturar las características más relevantes en química medicinal y metabólica, balanceando especificidad y generalidad. La cantidad relativamente pequeña de claves (166) reduce la dimensionalidad pero puede limitar la representación de compuestos muy complejos.

- **PubChem Fingerprint (PC)**: Conjunto de 880 claves que codifican un rango significativamente más amplio de características estructurales y propiedades moleculares (Kim y col., 2016). Incluye patrones subestructurales más específicos, propiedades fisicoquímicas adicionales, y combinaciones de características. La mayor cantidad de claves permite capturar mayor diversidad química, pero resulta en representaciones de mayor dimensionalidad. Es particularmente útil para datasets grandes y heterogéneos donde la especificidad fina es importante.

La diferencia fundamental entre MACCS (166 claves) y PC (880 claves) radica en el balance entre complejidad y especificidad: MACCS prioriza características generales y frecuentes en moléculas biológicas, mientras que PC captura patrones más específicos y raros. Esta diferencia influye en la distribución de valores de similaridad obtenidos, como se analizará en la Sección 3.2.1.

Fingerprints Basadas en Códigos de Hash

Las fingerprints basadas en códigos de hash no requieren la definición y listado previo de claves. En cambio, se generan mediante un algoritmo que enumera subestructuras locales alrededor de cada átomo y les aplica funciones de hash de forma iterativa. Este enfoque permite capturar una mayor diversidad de subestructuras sin estar limitado a un conjunto predefinido, y puede adaptarse automáticamente a la complejidad química del dataset.
El principio de funcionamiento consiste en: (1) inicializar cada átomo con un identificador basado en sus propiedades (tipo de átomo, hibridación, carga), (2) iterar expandiendo la vecindad de cada átomo en capas circulares (radio 1, 2, 3, etc.), (3) en cada iteración, actualizar el identificador de cada átomo combinando su valor actual con los identificadores de sus vecinos, (4) aplicar una función hash para convertir esta información en un valor entero, y (5) mapear estos valores a un vector

de bits de longitud fija. Como se ilustra en la Figura 3.1b, este proceso captura el entorno químico local de cada átomo hasta una distancia determinada.
Los métodos basados en hash evaluados incluyen:

- **Extended-Connectivity Fingerprints (ECFP)** (Rogers & Hahn, 2010): Fingerprints de estructura radial que mapean características subestructurales centradas en átomos (excluyendo hidrógenos) registradas en múltiples capas circulares hasta un diámetro determinado. El parámetro de radio (típicamente 2 o 4) define cuántos enlaces de distancia se consideran alrededor de cada átomo. Por ejemplo, ECFP4 (radio 4) captura el entorno químico hasta 4 enlaces de distancia desde cada átomo. Estas fingerprints son particularmente efectivas para detectar similitudes estructurales locales y son ampliamente utilizadas en búsqueda de similaridad y modelos QSAR (Quantitative Structure-Activity Relationship).

- **MORGAN**: Implementación específica del algoritmo de conectividad extendida equivalente a ECFP con radio 3, nombrada en honor al algoritmo de Morgan original para numeración canónica de átomos. Es ampliamente utilizada en química computacional por su balance entre especificidad estructural y eficiencia computacional.

- **MinHash Fingerprint (MHFP)** (Probst & Reymond, 2018): Codifican subestructuras utilizando el principio de conectividad extendida pero empleando MinHash, una técnica de hashing sensible a la localidad que permite estimaciones eficientes de similaridad de Jaccard. Esta aproximación es particularmente útil para búsquedas de vecinos más cercanos en grandes bases de datos químicas, ya que permite calcular similaridades aproximadas de manera muy eficiente sin necesidad de comparar todos los bits explícitamente.

- **SMILES Extended Connectivity Fingerprint (SECFP)**: Variante que combina el esquema de hashing circular con información derivada de la representación SMILES (Simplified Molecular Input Line Entry System) de las moléculas. Utiliza la misma operación de módulo que ECFP para mapear los valores hash a posiciones de bits, pero incorpora información adicional del string SMILES que puede capturar aspectos de la conectividad molecular complementarios a las ECFP tradicionales.

La ventaja principal de las fingerprints basadas en hash es su capacidad para representar automáticamente cualquier subestructura molecular sin necesidad de especificación previa, lo que las hace más flexibles que las fingerprints estructurales. Sin embargo, como se verá en la Sección 3.2.1, esta flexibilidad puede resultar en

distribuciones de similaridad muy sesgadas hacia valores bajos, especialmente en datasets pequeños donde la diversidad de subestructuras puede ser limitada.
La selección del método más apropiado se realizó mediante un análisis comparativo detallado cuyos resultados se presentan en la Sección 3.2.1.

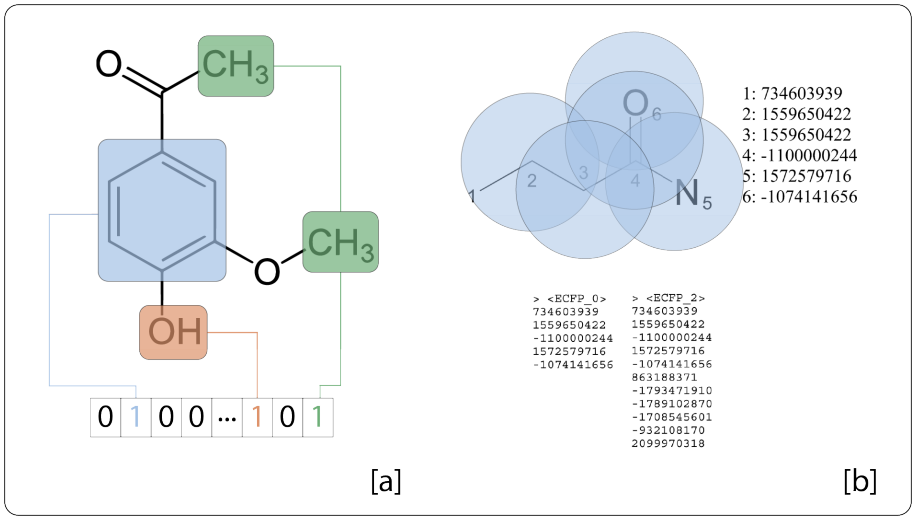

Figura 3.1: Esquema de construcción de fingerprints. [a] Fingerprint de tipo estructural. [b] Fingerprints basadas en código de hash (ECFP).

3.1.2 Modelo de validación con perceptrón multicapa

Para validar que los valores de similaridad obtenidos con la fingerprint seleccionada son útiles para aprendizaje supervisado, se empleó un perceptrón multicapa (MLP) simple. Los fundamentos teóricos de esta arquitectura fueron presentados en detalle en la Sección 2.3.1 del Capítulo 2, donde se describieron la estructura de capas, funciones de activación (ReLU para capas ocultas, Sigmoid/Softmax para la capa de salida), y el proceso de optimización mediante descenso de gradiente.
El modelo utilizado en este capítulo presenta las siguientes características específicas para la tarea de predicción de similaridad:
Arquitectura y entrada: El modelo recibe como entrada representaciones de pares de compuestos mediante codificación one-hot. Cada compuesto i se representa con un vector binario de dimensión N (donde $N = 47$ compuestos en la vía de Glucólisis), conteniendo un único valor 1 en la posición correspondiente al identificador del compuesto, y 0 en las demás posiciones. Para predecir la similaridad entre dos compuestos, se concatenan sus vectores one-hot, resultando en un vector de entrada de dimensión $2N = 94$. Como se ilustra en la Figura 3.2, la primera mitad del vector $\mathbf{x}[0 : N]$ codifica el primer compuesto, y la segunda mitad $\mathbf{x}[N : 2N]$ codifica el segundo compuesto.
Función de activación de salida: La capa de salida emplea una función sigmoidea

para acotar las predicciones al intervalo $[0, 1]$, consistente con el rango del índice de Tanimoto que se busca predecir.

Regularización: Se utilizó Dropout (Wager y col., 2013) como técnica de regularización con una probabilidad p constante aplicada a todas las capas ocultas, para reducir el riesgo de sobreajuste dado el tamaño relativamente pequeño del dataset (1,081 pares de compuestos).

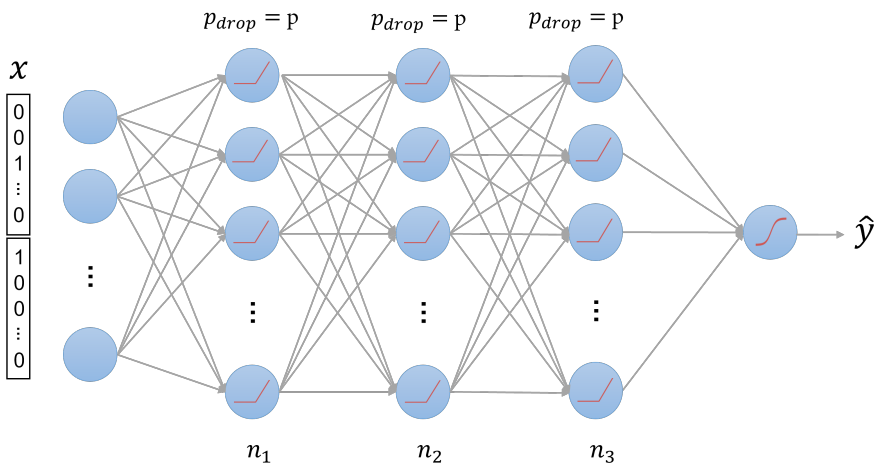

Figura 3.2: Arquitectura neuronal. El vector de características x contiene las representaciones one-hot de los compuestos a comparar [0, N] y [N + 1, 2N] respectivamente.

3.1.3 Construcción del dataset

Se utilizó la totalidad de la vía metabólica de la Glucólisis[1]. Esta vía está compuesta por 52 reacciones, que involucran a un total de 60 compuestos. De estos solo 47 tienen estructura conocida, lo que se necesita para calcular las fingerprints, por lo tanto, ese será el conjunto de compuestos a usar en este trabajo. Los datos fueron extraídos de la base de datos KEGG[2] (versión 95.2). Para cada compuesto se descargó la estructura molecular en formato SMILES (Weininger, 1988), cuando estaba disponible de la base de datos de PubChem[3]. Con base en estas estructuras, se generaron las diferentes fingerprints empleando la librería RDKit[4]; se eligió esta representación debido a su sencillez para representar la información de los compuestos. Luego se calculó la similaridad entre pares de compuestos empleando el coeficiente de Tanimoto (Ecuación 1.2), aplicado sobre las representaciones binarias de las fingerprints moleculares (Bajusz y col., 2015). Este coeficiente toma valores en el intervalo [0, 1] y calcula la proporción de características compartidas entre ambas estructuras.

[1]https://www.genome.jp/pathway/map00010
[2]https://www.genome.jp/kegg/
[3]https://pubchem.ncbi.nlm.nih.gov/
[4]https://www.rdkit.org

En total se definieron 1081 patrones que resultan de la combinatoria de a pares de los 47 compuestos con estructura conocida, a los cuales se los codificó con vectores one-hot: se enumeran los 47 compuestos de la vía metabólica de la Glucólisis que poseen estructura conocida, y se codifica el compuesto con un vector de ceros de dimensión 47 con un uno en la posición del compuesto correspondiente y se los concatena de a pares para formar los datos de entrada x como se explicó en la sección anterior.

3.2 Resultados

En esta sección se presentan los resultados obtenidos. En primer lugar, se realiza la selección de la fingerprint más apropiada según los criterios ya detallados. Una vez definido este aspecto, se calcula la salida deseada para la similaridad de todos los pares de compuestos. Luego se presenta la optimización de los hiperparámetros del modelo y construcción del modelo definitivo.

3.2.1 Selección de fingerprint

Las seis fingerprints presentadas en la Sección 3.1.1 fueron seleccionadas por ser las más frecuentemente utilizadas según la bibliografía consultada. Para determinar cuál de ellas resulta más adecuada para generar los valores a predecir con el modelo neuronal, se realizó un análisis basado en tres criterios ordenados: en primer lugar, la frecuencia de uso en el ámbito científico (estas son las seis que se describieron anteriormente); en segundo lugar, la separabilidad de resultados (resolución), examinada mediante el análisis de distribución de histogramas; y finalmente, la predicción de similaridad, evaluada a través del experimento de correspondencia entre compuestos. Con el objetivo de determinar cuán bien se distribuyen los valores de similaridad a lo largo del rango de posibles valores (intervalo [0, 1]), se construyeron los histogramas que se muestran en la Figura 3.3. Cada gráfico presenta la distribución del índice de Tanimoto para cada una de las fingerprints estudiadas. Como puede apreciarse en la Figura 3.3a, b, c, e y f, las distribuciones están concentradas en valores bajos. Se observa que las fingerprints MACCS (Figura 3.3d) presentan una distribución de coeficientes de Tanimoto más uniforme a lo largo del intervalo; esto es deseable para que el modelo entrenado produzca predicciones con similar calidad en todo el rango de variación del índice.

Por otro lado, se realizó un nuevo estudio para profundizar en la comparación de las fingerprints. Para esto se seleccionó un subconjunto de compuestos de la Glucólisis, sobre el que se realizó un análisis subjetivo que determinó la bondad de cada similaridad calculada. Se construyó una matriz de similaridad entre los compuestos cuyas KEGG ids son: C00103, C00668, C01172, C05345, C05378, C00118, C00197,

C00074, C00068, C00024, C05125, C00022, C00033, C00084, C00469. Los mismos fueron seleccionados por tener estructuras de diferentes complejidades, y con distinto grado de parecido al compararlas de a pares.

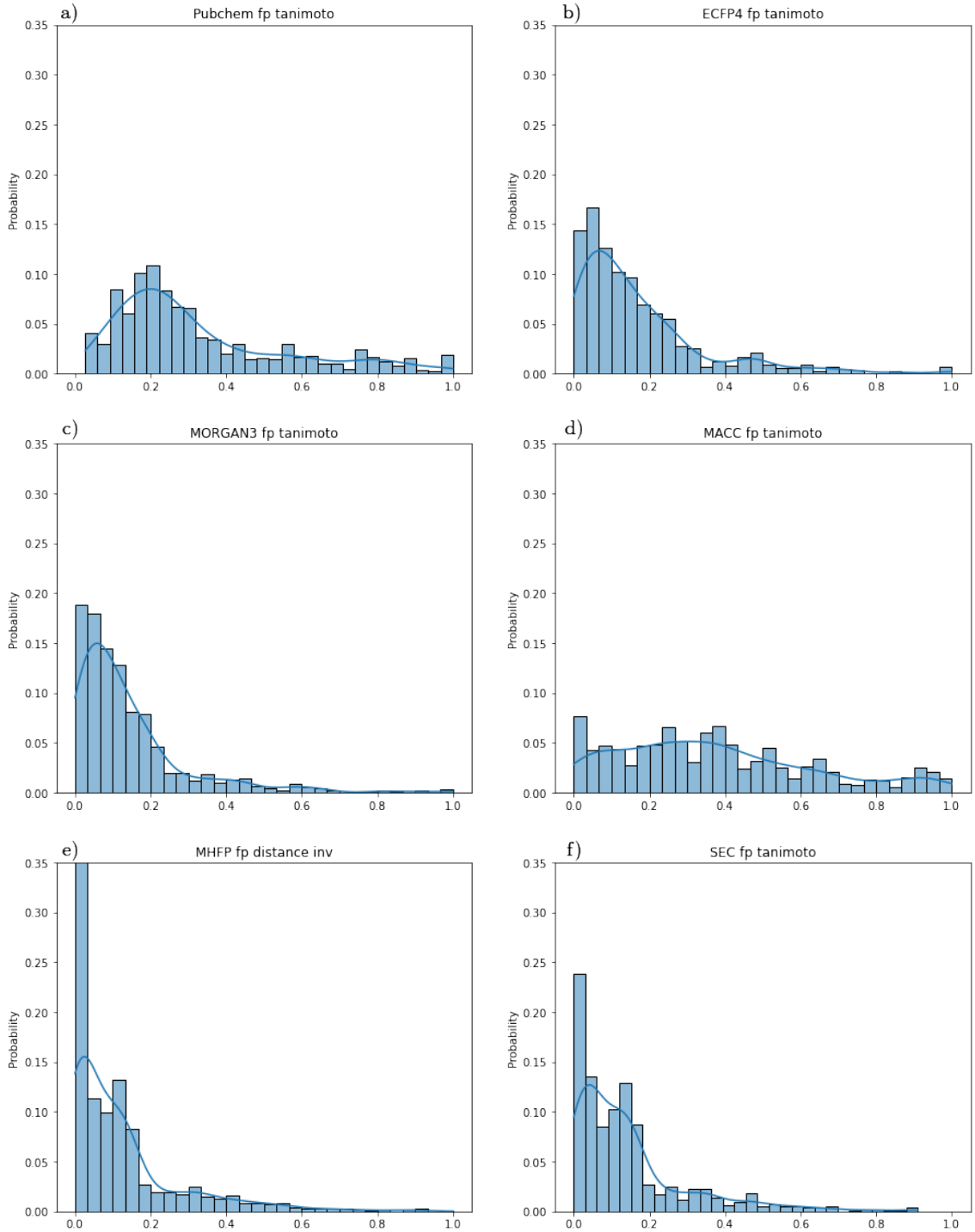

Figura 3.3: Histograma de la distribución de los valores de similaridad calculado con el índice de Tanimoto para las diferentes fingerprints consideradas sobre el camino metabólico de la Glucólisis: a) Pubchem, b) ECFP de radio 4, c) MORGAN (radio 3), d) MACCS, e) MHFP, f) SEC

En la Figura 3.4 se presenta la matriz de similaridad (matriz triangular inferior) pa-

ra un subconjunto de los compuestos seleccionados para la prueba. Dentro de cada cuadro se muestra la similaridad calculada con las fingerprints PC, ECFP4 (ECFP de radio 4), MOR3 (MORGAN de radio 3), MACCS, MHFP y SEC respectivamente. Como resulta esperable, todas las medidas devuelven similaridad 1 cuando la estructura de un compuesto se compara consigo mismo. Además, se ve claramente cómo las fingerprints ECFP, MORGAN, MHFP y SEC tienden a otorgar valores bajos de similaridad, incluso cuando las estructuras presentes son muy parecidas, como es el caso de C00668 y C00103. Esto es consistente con el estudio anterior. Hay que señalar que las MACCS keys y las PC fueron las que mejor desempeño tuvieron al generar los índices. Se observó que las fingerprints de PubChem tienden a otorgar valores altos de similaridad al comparar compuestos pequeños por más que sean diferentes.
Dadas las comparativas realizadas, se determinó la utilización de las MACCS keys como fingerprint para construir el dataset para entrenar el modelo neuronal.

3.2.2 Búsqueda de hiperparámetros y resultados de la red neuronal

En esta sección se evalúa la capacidad de un modelo neuronal basado en MLP para inferir la similaridad. Para calcular los valores targets y se utilizó el coeficiente de Tanimoto calculado a partir de las MACCS keys de los compuestos. Se emplearon como valores de entrada x representaciones one-hot de los mismos, como ya se explicó. Para esto, se realizó el entrenamiento del modelo neuronal utilizando cross-validation con 5 folds con un porcentaje de validación del 10 % separado del conjunto de training de cada fold. El modelo fue implementado utilizando PyTorch 1.9. El entrenamiento se realizó empleando el optimizador Adam y el error cuadrático medio (MSE) como función de costo. Se consideraron dos arquitecturas diferentes: una con tres capas ocultas y otra con cuatro, todas ellas con funciones de activación ReLU. Para la capa de salida se utilizó una sigmoidea. Los experimentos realizados se llevaron a cabo empleando 6000 como número máximo de épocas de entrenamiento y 1000 épocas de paciencia para la detención temprana, evitando que la red continúe el entrenamiento cuando no hay mejora en el error de validación. En los casos donde se supera el umbral de paciencia, el algoritmo es detenido y se recupera el modelo con el menor error de validación hasta el momento.
Dado que existen múltiples hiperparámetros que pueden afectar el desempeño del modelo, como primer paso se realizó una exploración de los mismos. En particular, se realizó una búsqueda en grilla detallada en la Tabla 3.1. Esto implica que se entrenaron 16384 modelos, uno para cada combinación de los hiperparámetros descriptos.

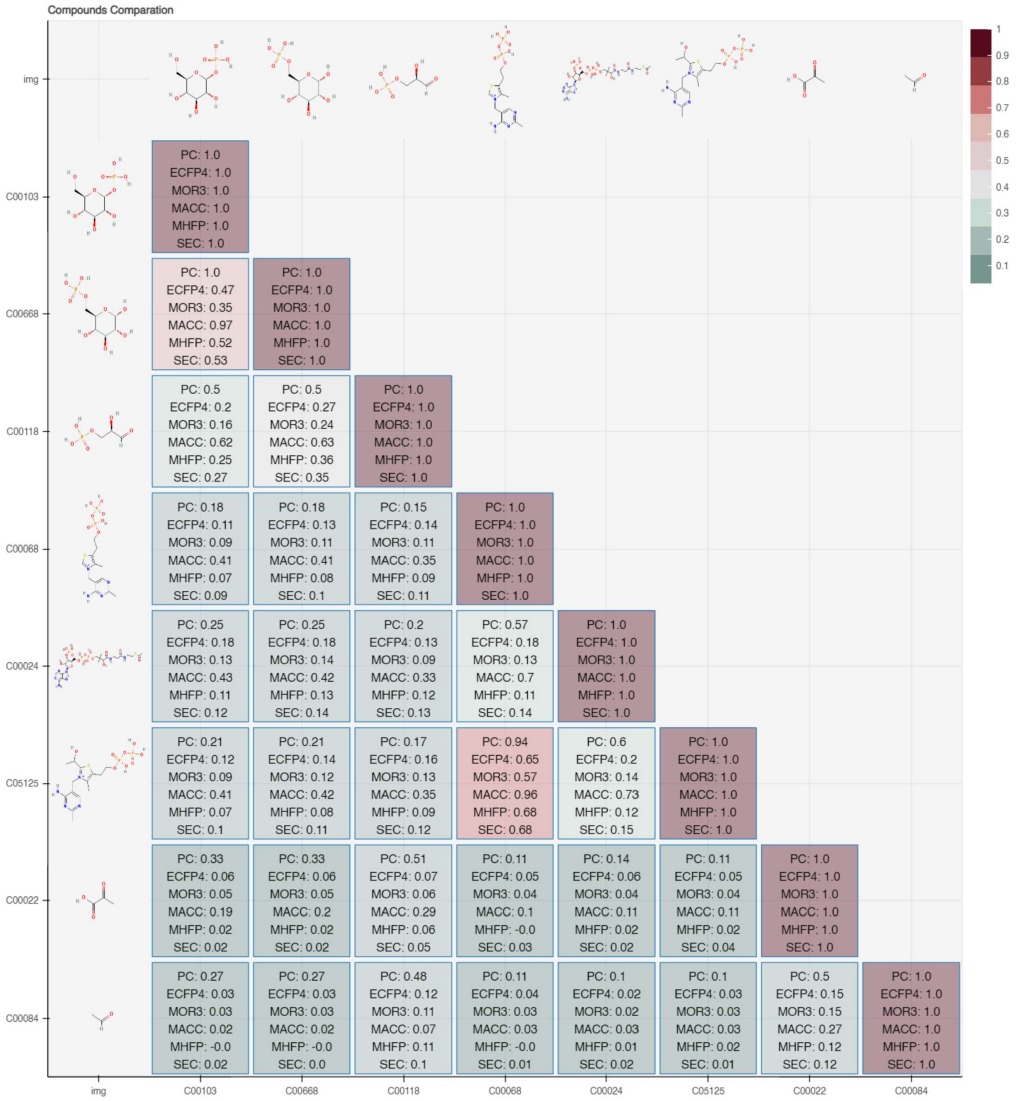

Figura 3.4: Parte de la comparativa de fingerprints: en verde los compuestos que poseen similaridad promedio baja, en rojo alta.

Tabla 3.1: Grilla de búsqueda para el modelo base

Hiperparámetro	Rango	Paso
Tasa de aprendizaje	$5 \cdot 10^{-4}$, $3 \cdot 10^{-4}$	
Tamaño de batch	10, 100	
P dropout	0.3, 0.5	
Tamaño capa 2	16 - 256	16
Tamaño capa 3	16 - 256	16
Tamaño capa 4	16 - 256	16

La Tabla 3.2 presenta los 5 mejores conjuntos de hiperparámetros obtenidos de esta exploración. El modelo fue entrenado con una función de costo del error cuadrático medio, sin embargo, se presenta la raíz cuadrada del mismo porque esta se encuentra en el mismo orden de magnitud del índice a predecir. Como puede apreciarse, se obtuvo el mejor conjunto de hiperparámetros: tasa de aprendizaje de $5{\cdot}10^{-4}$, tamaño de batch de 10, la probabilidad de dropout de 0.3 y un número de neuronas de [94,256,16,16,1] en cada capa respectivamente. En cuanto al error, se obtuvo un MAE de 0.081 (RMSE = 0.1019) en validación cruzada y un coeficiente de determinación R^2 de test de 0.9, determinado a partir de los valores reales y los predichos por la red. Este resultado se condice con lo observado al comparar los valores predichos vs los reales. En la Figura 3.5 se representan los índices de Tanimoto targets (y) comparados con los predichos por la red ($\hat{y}$) para uno de los folds, con resultados similares para los demás folds. Adicionalmente, se agregaron líneas de tendencia de cada set. En azul se muestran los valores de entrenamiento, en naranja los de test y en verde los de validación. Lo primero que se observa es que la mayoría de los valores de similaridad se ubican sobre la diagonal, lo que indica una buena predicción de los valores. A su vez, si se observan las rectas de regresión para cada partición se aprecia que están muy cercanas a la recta de 45°. Por último son presentadas las densidades de distribución del índice, que son similares tanto para los valores reales como los predichos.

Tabla 3.2: Mejores resultados de la búsqueda de grilla. Batch se refiere al tamaño del batch, Lr es la tasa de aprendizaje y Capa se refiere al número de neuronas de esa capa

MAE	RMSE	Batch	Lr	Capa 2	Capa 3	Capa 4
0.0814	0.1020	10	0.0005	256	16	16
0.0815	0.1022	10	0.0005	256	16	16
0.0821	0.1029	10	0.0005	256	256	16
0.0822	0.1029	10	0.0005	128	16	16
0.0822	0.1030	10	0.0003	256	16	16

3.3 Conclusiones

Este capítulo cumplió exitosamente su objetivo de establecer la infraestructura metodológica necesaria para los capítulos subsiguientes de esta tesis. Se logró: (1) seleccionar empíricamente la representación molecular más apropiada para construir el ground truth de similaridad, y (2) validar que este ground truth es útil para entrenar modelos de aprendizaje automático.

Selección fundamentada de fingerprint para ground truth. Se realizó una

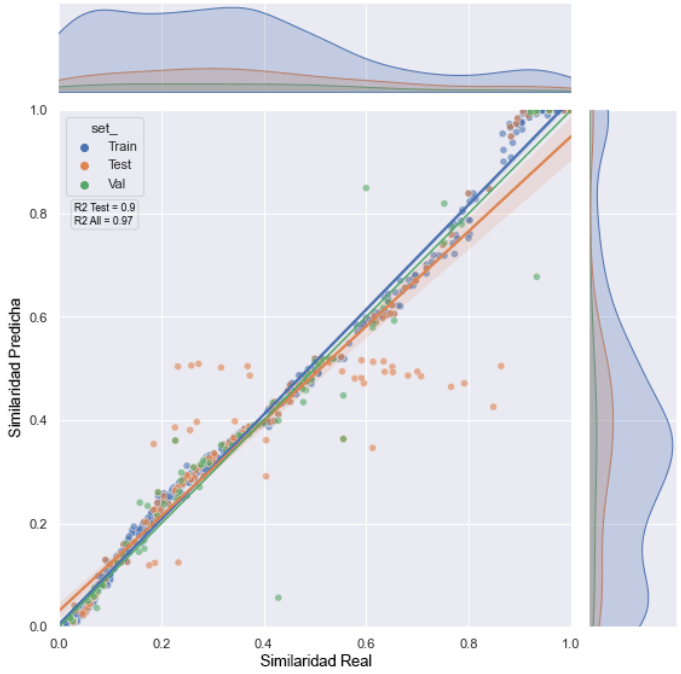

Figura 3.5: Índice de Tanimoto real vs Índice de Tanimoto predicho por la red

comparación exhaustiva y sistemática de 6 métodos de fingerprints moleculares, clasificados en dos familias: estructurales (MACCS keys con 166 claves y PubChem con 880 claves) y basados en hash (ECFP, MORGAN, MHFP y SECFP). El análisis basado en tres criterios complementarios—distribución de valores de similaridad, separabilidad, y frecuencia de uso en la literatura—determinó que las MACCS keys son la representación más apropiada para construir el dataset de entrenamiento. El análisis de histogramas reveló que MACCS keys presenta la distribución más uniforme del índice de Tanimoto en el intervalo [0,1], evitando el sesgo hacia valores bajos observado en las fingerprints basadas en hash. La matriz de similaridad cualitativa confirmó que MACCS keys produce valores coherentes con la intuición química para compuestos metabólicos. Esta selección fundamentada proporciona una base sólida y reproducible para todos los experimentos de los capítulos posteriores.

Validación exitosa mediante aprendizaje supervisado. El experimento de validación con un perceptrón multicapa simple demostró que los valores de similaridad calculados con MACCS keys son efectivamente útiles para entrenar modelos neuronales. El modelo logró un RMSE de 0.1019 y un coeficiente de determinación $R^2 = 0{,}9$, prediciendo con alta precisión el índice de Tanimoto a partir únicamente de identificadores básicos (codificación one-hot) de los compuestos. Este resultado es particularmente notable considerando que las representaciones one-hot no contienen información estructural química explícita: el modelo aprendió patrones de similaridad a partir de los identificadores y las etiquetas de similaridad calculadas con las fingerprints. La exploración exhaustiva de hiperparámetros mediante búsqueda

en grilla (16,384 configuraciones evaluadas) garantizó que el modelo fue optimizado apropiadamente, identificando una arquitectura con tasa de aprendizaje de 5×10^{-4}, batch de tamaño 10, dropout 0.3, y configuración [94, 256, 16, 16, 1] neuronas por capa.

Observaciones relevantes y oportunidades de mejora. El análisis de los resultados reveló varios aspectos que orientan los capítulos subsiguientes:

- **Viabilidad del aprendizaje supervisado**: Los resultados confirman que las redes neuronales pueden aprender efectivamente patrones de similaridad molecular, incluso con representaciones simples y datasets relativamente pequeños (1,081 pares de compuestos de la vía de Glucólisis). Esto valida la aproximación general de la tesis y motiva la exploración de arquitecturas más sofisticadas.

- **Potencial de incorporación de información topológica**: Las representaciones one-hot, a pesar de su simplicidad, permitieron obtener predicciones con $R^2 = 0{,}9$ sin utilizar información sobre la estructura del grafo metabólico. Este resultado sugiere que la incorporación de información topológica de la red—mediante embeddings de nodos que capturan vecindades y contexto funcional—podría mejorar significativamente el desempeño. Esta hipótesis se explorará en el Capítulo 4 utilizando algoritmos como Node2Vec (Grover & Leskovec, 2016).

- **Oportunidad de compresión dimensional**: La concatenación de vectores one-hot genera representaciones de dimensión $2N$ (94 elementos para 47 compuestos), que escalan linealmente con el tamaño del dataset. Los embeddings de nodos, que se explorarán en el capítulo siguiente, permiten representaciones densas de dimensionalidad fija (típicamente 64-256 dimensiones) independientemente del número de compuestos, facilitando la escalabilidad a redes metabólicas más grandes.

- **Posibilidad de representar compuestos con estructura parcial**: El enfoque one-hot requiere que todos los compuestos estén presentes en el conjunto de entrenamiento. Los métodos basados en embeddings de grafos que se desarrollarán posteriormente permitirán generar representaciones para todos los compuestos presentes en la topología de la red, incluyendo aquellos con estructura molecular desconocida o parcialmente conocida (compuestos con sustituyentes genéricos "-R"), ampliando significativamente la aplicabilidad práctica.

Contribuciones y próximos pasos. Este capítulo establece dos contribuciones fundamentales para la tesis: (1) un protocolo reproducible y fundamentado para

seleccionar representaciones moleculares apropiadas para ground truth, y (2) la confirmación empírica de que el aprendizaje supervisado es viable para predicción de similaridad molecular en redes metabólicas. El ground truth establecido con MACCS keys se utilizará consistentemente en los capítulos subsiguientes, permitiendo comparar de manera justa diferentes arquitecturas neuronales y representaciones de entrada. Los capítulos siguientes explorarán: embeddings basados en la topología del grafo metabólico (Capítulo 4), arquitecturas de redes neuronales en grafos especializadas en tareas centradas en aristas (Capítulo 5), y aplicaciones a problemas de interacción proteína-proteína y anotación funcional (Capítulo 6). La metodología establecida en este capítulo constituye la base sobre la cual se construirán todos estos desarrollos.

Capítulo 4

Embeddings como descriptores de nodos

4.1 Introducción

En el Capítulo 3 se demostró la viabilidad de predecir la similaridad entre compuestos metabólicos utilizando un perceptrón multicapa con codificación one-hot de los compuestos como entrada. Este enfoque alcanzó un RMSE de 0.1019 sobre la vía de la Glucólisis, demostrando que es posible aprender patrones de similaridad estructural a partir de representaciones discretas simples. Sin embargo, la codificación one-hot presenta limitaciones inherentes: (1) no captura información sobre la vecindad topológica del compuesto en la red metabólica, (2) genera vectores de alta dimensionalidad y dispersos ($2N$ elementos para comparar N compuestos), y (3) no permite representar de manera diferenciada compuestos con estructura molecular desconocida o parcialmente conocida.

Para superar estas limitaciones, en este capítulo se propone el uso de **embeddings de nodos** como descriptores alternativos que reemplazan la codificación one-hot. Los embeddings, generados mediante el algoritmo Node2Vec (Grover & Leskovec, 2016), condensan la información de vecindad de cada nodo en vectores densos de menor dimensionalidad, capturando tanto la estructura local como global del grafo metabólico. A diferencia del enfoque anterior basado en one-hot, estos embeddings se construyen directamente desde la topología de la red, sin requerir conocimiento previo de las estructuras moleculares de los compuestos.

Este enfoque presenta ventajas conceptuales importantes frente a la codificación one-hot:

- **Generalización**: Los embeddings pueden generarse para todos los compuestos presentes en la red metabólica, incluyendo aquellos sin estructura molecular conocida o con sustituyentes genéricos no procesables por métodos tradiciona-

les de fingerprints.

- **Eficiencia**: Los vectores de embeddings son densos y de dimensionalidad configurable (típicamente 47-256 dimensiones), frente a los $2N$ elementos de la concatenación one-hot, donde N es el número de compuestos.
- **Información contextual**: Los embeddings capturan la proximidad funcional y topológica de los compuestos en la red metabólica, información completamente ausente en la codificación one-hot.

El presente capítulo organiza el estudio de la siguiente manera. En la Sección 4.2, se describe el algoritmo Node2Vec utilizado para construir embeddings, el modelo neuronal (que mantiene la arquitectura MLP del Capítulo 3), y las particularidades del conjunto de datos al usar embeddings como entrada. En la Sección 4.3 se presentan los experimentos realizados, incluyendo: (1) una exploración exhaustiva de hiperparámetros del algoritmo de embedding mediante el framework Optuna con muestreador Tree-structured Parzen Estimator (TPE), (2) optimización de la arquitectura neuronal, y (3) análisis de resultados tanto para compuestos con estructuras conocidas como desconocidas. Finalmente, las conclusiones se presentan en la Sección 4.4.

4.2 Materiales y métodos

Esta sección presenta los conceptos teóricos y materiales utilizados en este trabajo. Primero, se describe el algoritmo de embedding. Posteriormente, se muestra el modelo neuronal utilizado y sus componentes. Finalmente, se describe el proceso para construir el conjunto de datos.

4.2.1 Construcción de embeddings

Para construir los embeddings, se utilizó el algoritmo Node2Vec. Este es un algoritmo basado en grafos que genera embeddings condensando la información de vecindad de cada nodo. Se trata de un método de aprendizaje no supervisado: los embeddings se obtienen maximizando la probabilidad de preservar las vecindades de cada nodo, sin utilizar las etiquetas de la tarea. Sus autores lo describen como semi-supervisado porque los parámetros p y q pueden estimarse a partir de una fracción pequeña de datos etiquetados; en este trabajo se fijaron mediante búsqueda de hiperparámetros. El algoritmo combina dos estrategias de muestreo clásicas y opuestas: el muestreo en amplitud (BFS), que restringe la vecindad de cada nodo a aquellos que son vecinos inmediatos del nodo fuente; y el muestreo en profundidad (DFS), que toma la vecindad desde muestras secuenciales de nodos, incrementando las distancias desde

el nodo fuente. Node2Vec utiliza caminatas aleatorias para incorporar información desde una interpolación suave entre BFS y DFS. El balance entre ambos métodos se controla mediante hiperparámetros, los cuales deben ajustarse según cada aplicación específica.

Para generar embeddings, este método realiza un número de caminatas aleatorias (num_walk) comenzando desde cada nodo del grafo. En cada paso, se combina la información de la vecindad, y los nodos que se toman en cuenta para este proceso dependen del tamaño de la ventana ($window$) considerada. Dado un nodo fuente, el algoritmo elige el siguiente basándose en los hiperparámetros p y q, que controlan las probabilidades de seleccionar un nuevo nodo o de retornar a un nodo previamente seleccionado. Finalmente, la información recolectada de todos los pasos (len_walk) se utiliza para generar un embedding de un tamaño especificado ($size$), empleando un enfoque basado en Word2Vec (Grover & Leskovec, 2016). El algoritmo tiene seis hiperparámetros ajustables importantes que necesitan optimizarse para caracterizar adecuadamente los compuestos y calcular la similaridad entre ellos.

4.2.2 Modelo neuronal

Se utilizó la misma arquitectura de perceptrón multicapa (MLP) descrita en el Capítulo 3, manteniendo las funciones de activación ReLU en capas ocultas, sigmoide en la capa de salida, y el error cuadrático medio (MSE) como función de pérdida. La única diferencia fundamental radica en la entrada al modelo: como se muestra en la Figura 4.1, la entrada $\mathbf{x}$ corresponde a un par de **embeddings concatenados** (uno para cada compuesto a comparar), en lugar de la concatenación de vectores one-hot utilizada anteriormente. Cada embedding captura la estructura de vecindad del compuesto en el grafo metabólico mediante un vector denso de dimensión configurable. Se aplicó regularización mediante dropout para prevenir el sobreajuste.

4.2.3 Construcción del conjunto de datos

Se utilizó el mismo conjunto de datos descrito en el Capítulo 3: la vía metabólica de la Glucólisis con 52 reacciones y 60 compuestos, de los cuales 47 tienen estructura molecular conocida. Los valores ground truth de similaridad se calcularon utilizando el coeficiente de Tanimoto (Ecuación 1.2) sobre fingerprints MACCS keys, exactamente como se detalla en dicho capítulo. En total, se definieron los mismos 1081 patrones resultantes de la combinatoria de a pares de los 47 compuestos con estructura conocida.

Alcance del entrenamiento supervisado: El algoritmo Node2Vec genera embeddings para los 60 compuestos presentes en el grafo metabólico, incluyendo los 13 compuestos sin estructura molecular completamente conocida (aquellos con susti-

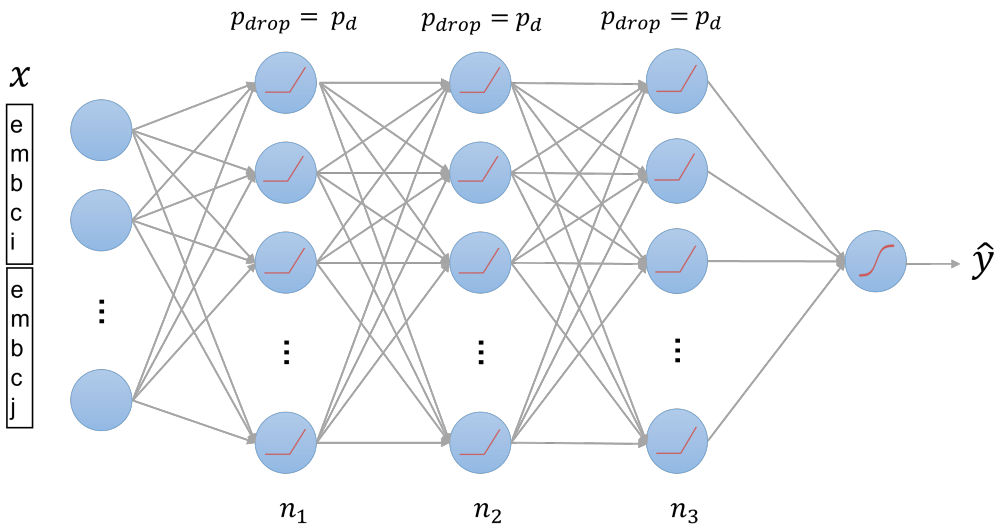

Figura 4.1: Arquitectura del modelo neuronal utilizado para predecir similaridad. La entrada consiste en dos embeddings concatenados, procesados a través de múltiples capas ocultas para producir un valor de similaridad entre 0 y 1.

tuyentes genéricos como "-R"). Esta capacidad constituye una ventaja fundamental sobre la codificación one-hot. Sin embargo, el entrenamiento supervisado del modelo neuronal se realiza exclusivamente sobre los 1081 pares formados por los 47 compuestos con estructura conocida, manteniendo consistencia metodológica con el Capítulo 3 para permitir comparación directa entre enfoques. Esta restricción se debe a que el cálculo del ground truth (coeficiente de Tanimoto, Ecuación 1.2) requiere las fingerprints moleculares completas de ambos compuestos. Los embeddings de los 13 compuestos restantes se utilizarán posteriormente en la Sección 4.3.3 para análisis cualitativo mediante proyección PCA, donde se evaluará si las predicciones de similaridad son consistentes con análisis visual de sus estructuras parciales.

La **diferencia fundamental** con respecto al enfoque anterior radica en la representación de entrada $\mathbf{x}$. La Figura 4.2 ilustra el pipeline de construcción del conjunto de datos, destacando que ahora se utilizan embeddings Node2Vec en lugar de codificación one-hot. El proceso completo se resume de la siguiente manera:

1. Se modela la vía metabólica como un grafo $\mathcal{G} = \{v, e\}$, donde los nodos v representan compuestos y las aristas e conectan compuestos que participan en una misma reacción.

2. Se aplica el algoritmo Node2Vec sobre $\mathcal{G}$ para generar embeddings de dimensión configurable para cada uno de los 60 compuestos (incluyendo aquellos sin estructura conocida).

3. Para cada par de compuestos (i, j) con estructura conocida, se concatenan sus embeddings para formar la entrada $\mathbf{x}_{ij} = [\mathbf{emb}_i; \mathbf{emb}_j]$.

4. La salida objetivo y_{ij} corresponde al coeficiente de Tanimoto (Ecuación 1.2) calculado entre las fingerprints MACCS keys de los compuestos i y j.

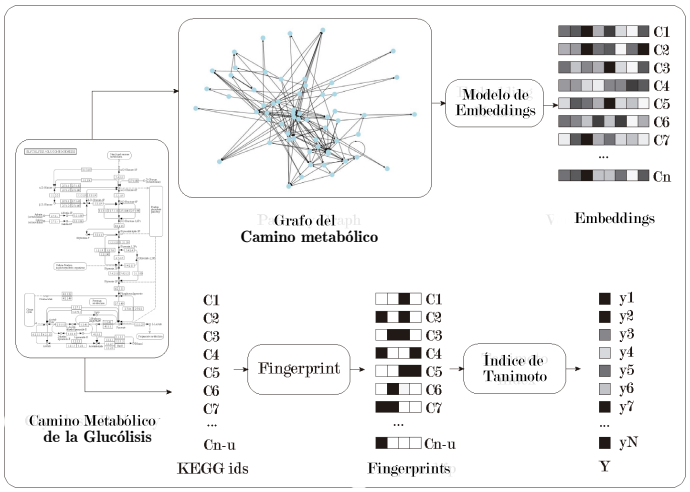

Figura 4.2: Pipeline de construcción del conjunto de datos con embeddings Node2Vec. Donde $n = 60$ es el número total de compuestos en la vía de la Glucólisis, $u = 13$ es el número de compuestos con estructura desconocida, y $N = 1081$ es la combinatoria de a pares de los compuestos con estructura conocida $\binom{n-u}{2} = \binom{47}{2} = 1081$.

Como se muestra en la Figura 4.3, cada entrada $\mathbf{x}$ al modelo neuronal resulta de la concatenación de embeddings Node2Vec para ambos compuestos entre los cuales se desea predecir la similaridad. Esta representación densa y de menor dimensionalidad contrasta con la representación dispersa de vectores one-hot utilizada anteriormente.

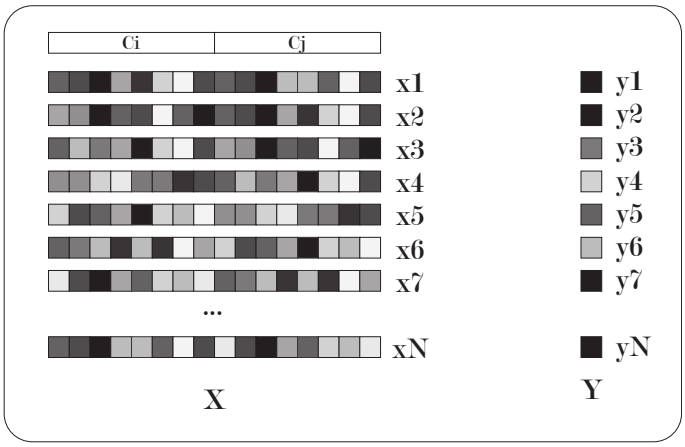

Figura 4.3: Estructura final del conjunto de datos para la red neuronal. Cada patrón consiste en la concatenación de dos embeddings Node2Vec (vectores densos de dimensión d), reemplazando la concatenación de vectores one-hot del enfoque anterior.

4.3 Experimentos y resultados

Objetivo y estrategia experimental: El objetivo central de esta sección es determinar si los embeddings basados en la topología del grafo metabólico pueden capturar información relevante para predecir similaridad molecular, y si esta representación supera el desempeño de la codificación one-hot evaluada en el Capítulo 3. La hipótesis subyacente es que la estructura de vecindad en la red metabólica—qué compuestos participan en reacciones relacionadas—se correlaciona con la similaridad estructural molecular, de modo que embeddings que codifican esta topología deberían producir mejores predicciones que representaciones que ignoran completamente la estructura del grafo.

La optimización del pipeline completo (embeddings + modelo neuronal) presenta un desafío metodológico significativo: el algoritmo Node2Vec tiene seis hiperparámetros (q, p, len_walk, num_walk, $size$, $window$), y la arquitectura MLP tiene seis hiperparámetros adicionales (tasa de aprendizaje, tamaño de batch, dropout, y número de neuronas en cada capa oculta). Una búsqueda exhaustiva en grilla considerando todas las combinaciones posibles requeriría cientos de miles de entrenamientos, lo cual es computacionalmente prohibitivo. Para abordar este problema, se adoptó una estrategia de optimización en dos etapas secuenciales:

1. **Etapa 1 - Optimización de embeddings**: Se exploran 2.500 configuraciones de hiperparámetros de Node2Vec utilizando el muestreador Tree-structured Parzen Estimator (TPE) implementado en Optuna. Para cada configuración de embeddings, se entrena un MLP con arquitectura fija y hiperparámetros razonables definidos manualmente. Los cinco mejores conjuntos de embeddings se identifican basándose en el error de validación cruzada. Esta etapa determina qué configuraciones de Node2Vec capturan mejor la estructura del grafo para la tarea específica de predicción de similaridad.

2. **Etapa 2 - Optimización del modelo neuronal**: Manteniendo fijos los cinco mejores conjuntos de embeddings identificados en la etapa anterior, se realiza una búsqueda en grilla sobre los hiperparámetros del MLP. Esta búsqueda se subdivide en dos pasos: primero se optimizan conjuntamente tasa de aprendizaje, batch size, dropout y tamaños de capa; posteriormente se explora el número de capas ocultas óptimo.

Esta estrategia de optimización secuencial reduce el número de experimentos de $O(10^5)$ a $O(10^3)$, haciéndola computacionalmente factible, aunque potencialmente resulte en una configuración subóptima si existe interacción fuerte entre los hiperparámetros de ambas etapas. Sin embargo, la separación es conceptualmente razonable:

los embeddings codifican la información del grafo independientemente del modelo que los utilizará, y el MLP opera sobre embeddings precalculados sin modificarlos.
Resultados esperados: Se anticipa que, si los embeddings capturan efectivamente información topológica relevante, el modelo entrenado con embeddings Node2Vec debería superar el desempeño del modelo con codificación one-hot (RMSE=0.1019, R^2=0.90) reportado en el Capítulo 3. Adicionalmente, se espera que el análisis de importancia de hiperparámetros revele qué aspectos de la topología del grafo (exploración local vs. global, ciclos, etc.) son más relevantes para la similaridad molecular.
Esta sección presenta los experimentos realizados y sus resultados. Primero, se describe el proceso de selección de embeddings mediante optimización con TPE y análisis de importancia de hiperparámetros. Posteriormente, se detalla la optimización de la arquitectura del modelo neuronal. Finalmente, se comparan los resultados con el enfoque baseline de codificación one-hot, y se analizan las predicciones para compuestos con estructuras parciales o desconocidas mediante proyección PCA.

4.3.1 Selección de embeddings

Dado que el algoritmo de embedding utilizado aquí está diseñado para trabajar con cualquier tipo de grafo, deben determinarse hiperparámetros apropiados para producir embeddings que puedan utilizarse para predecir correctamente el índice de Tanimoto. Para determinarlos, se realizó una búsqueda utilizando el framework Optuna con el muestreador Tree-structured Parzen Estimator[1] (TPE) (Bergstra y col., 2011, 2013). La exploración se realizó en dos pasos: primero una búsqueda en grilla sobre los rangos amplios de la Tabla 4.2, que permitió acotar la región de interés, y luego un refinamiento con Optuna sobre los rangos así delimitados. Se analizaron un total de 2.500 combinaciones de hiperparámetros: hiperparámetros q y p entre 0.1 y 1 con un paso de 0.1; *size* del embedding = {47, 64, 128, 256}; tamaños de *window* = {2, 5, 10, 15}; *num_ walk* entre 100 y 500, con un paso de 100, y *len_ walk* entre 5 y 20, con un paso de 5. Para determinar la calidad de cada embedding, se entrenó un MLP con arquitectura [2**size*, 145, 20, 1] utilizando validación cruzada con 5 particiones y una porción de validación del 10 %. Se utilizó el Error Cuadrático Medio (MSE) como función de pérdida. La Tabla 4.1 muestra las cinco mejores combinaciones de hiperparámetros resultantes de la búsqueda. Se incluyeron el MAE y la raíz cuadrada del MSE (RMSE) ya que tienen el mismo orden de magnitud que el índice de Tanimoto.
Análisis de las configuraciones óptimas: La Tabla 4.1 revela patrones consistentes en los hiperparámetros óptimos que proporcionan información sobre la estructura de la red metabólica. El parámetro q muestra valores consistentemente altos (0.8-1.0)

[1] https://optuna.readthedocs.io/en/stable/reference/generated/optuna.samplers.TPESampler.html

q	p	len_walk	num_walk	size	window	MAE	RMSE
1.0	**0.8**	**10**	**300**	**47**	**10**	**0.090**	**0.1124**
1.0	0.6	10	300	64	15	0.090	0.1125
0.8	0.4	10	300	64	2	0.090	0.1126
0.9	0.5	20	300	47	5	0.090	0.1131
0.9	0.2	15	300	128	5	0.090	0.1131

Tabla 4.1: Valores de hiperparámetros correspondientes a los cinco embeddings con menor error de validación cruzada.

en las cinco mejores configuraciones, indicando que el algoritmo favorece estrategias de exploración tipo BFS (Breadth-First Search). Recordando que q controla la probabilidad de alejarse del nodo previamente visitado, valores $q \geq 1$ implican que el algoritmo prioriza la exploración de vecindades locales sobre caminos largos en el grafo. Este resultado sugiere que la similaridad molecular en redes metabólicas está fuertemente determinada por la topología local: compuestos que participan en reacciones cercanas tienden a tener mayor similaridad estructural.

El parámetro p, que controla la probabilidad de retornar a nodos previamente visitados, muestra mayor variabilidad (0.2-0.8), aunque con tendencia hacia valores medios-bajos. Esto indica que el algoritmo se beneficia de cierta capacidad de revisitar nodos (evitando p muy altos), lo cual permite capturar estructuras cíclicas y caminos alternativos en la red metabólica, pero sin sobreenfatizar esta característica (evitando p muy bajos).

El parámetro *num_walk*=300 aparece consistentemente en todas las configuraciones óptimas, indicando que 300 caminatas aleatorias por nodo proporcionan un muestreo suficientemente representativo de la estructura del grafo sin incurrir en sobremuestreo innecesario. El tamaño de ventana (*window*) muestra más variabilidad (2, 5, 10, 15), sugiriendo que el contexto inmediato capturado en cada caminata es menos crítico que la estrategia de exploración (controlada por p y q).

Resulta notable que la configuración óptima utiliza *size*=47, que coincide exactamente con el número de compuestos con estructura conocida en el dataset. Adicionalmente, tres de las cinco mejores configuraciones utilizan dimensiones pequeñas (47, 64) en lugar de dimensiones mayores (128, 256). Esto sugiere que la estructura del grafo metabólico puede capturarse eficientemente en espacios de baja dimensionalidad, evitando la necesidad de representaciones de alta dimensión que podrían introducir ruido o requerir más datos de entrenamiento para el modelo neuronal subsecuente.

El efecto relativo de estos hiperparámetros sobre la predicción de similaridad se cuantificó mediante análisis de importancia utilizando FAnova (Hutter y col., 2014). Los resultados se presentan en la Figura 4.4.

Interpretación del análisis de importancia: La Figura 4.4 cuantifica la influen-

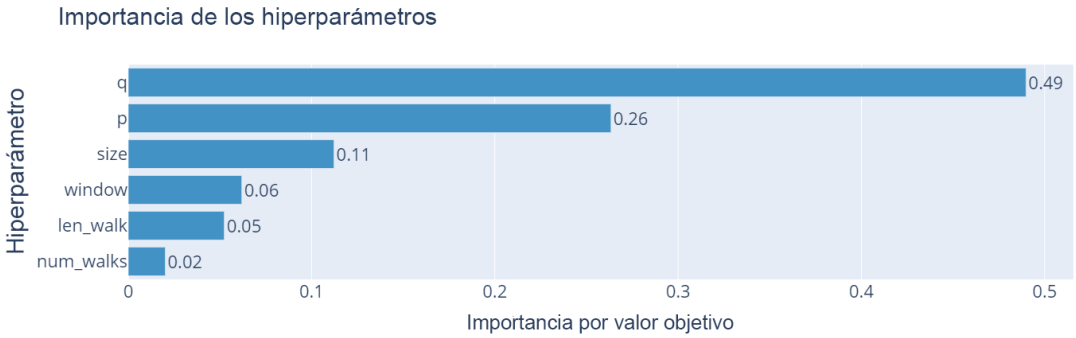

Figura 4.4: Análisis de importancia de hiperparámetros del algoritmo de embedding mediante FAnova. El eje horizontal muestra la importancia relativa (porcentaje de varianza explicada) de cada hiperparámetro sobre el error de predicción del modelo.

cia relativa de cada hiperparámetro sobre el rendimiento del modelo mediante el análisis FAnova, que descompone la varianza del error de predicción atribuible a cada parámetro. El hiperparámetro q emerge como el factor dominante, explicando aproximadamente el 40 % de la varianza en el error de validación cruzada. Este hallazgo confirma cuantitativamente la observación cualitativa de la Tabla 4.1: la estrategia de exploración del algoritmo (BFS vs. DFS) es el aspecto más crítico para capturar información relevante de la red metabólica.

La alta importancia de q tiene implicaciones biológicas significativas. Recordando que q controla la probabilidad de alejarse del nodo previamente visitado en las caminatas aleatorias, su predominancia indica que la **topología local** de la red metabólica—es decir, qué compuestos son vecinos inmediatos en términos de reacciones compartidas—es el factor determinante de la similaridad molecular. Valores altos de q (que favorecen exploración BFS) son óptimos porque los compuestos que participan juntos en reacciones metabólicas consecutivas tienden a tener estructuras químicas relacionadas, reflejando la lógica bioquímica subyacente de las vías metabólicas donde transformaciones enzimáticas secuenciales operan sobre sustratos estructuralmente similares.

El parámetro p, que controla los retornos a nodos visitados, aparece como el segundo factor más influyente (aproximadamente 20-25 % de la varianza), aunque notablemente menos importante que q. Esto sugiere que la capacidad del algoritmo para capturar ciclos y caminos redundantes en la red metabólica tiene relevancia secundaria comparada con la estrategia de exploración fundamental. La presencia de ciclos metabólicos (como el ciclo de Krebs) y vías alternativas es información valiosa, pero subordinada a la estructura de vecindad local.

Los hiperparámetros restantes—*size*, *window*, *len_ walk* y *num_ walk*—muestran importancia considerablemente menor (cada uno contribuyendo menos del 10 % de la varianza). Esto indica que, una vez establecida la estrategia de exploración apro-

piada mediante q y p, los aspectos técnicos de la implementación (dimensionalidad del embedding, tamaño de contexto, longitud y número de caminatas) son relativamente robustos y menos críticos para el desempeño final. Este resultado simplifica futuras aplicaciones del método, sugiriendo que la optimización puede enfocarse primariamente en los parámetros q y p, utilizando valores estándar razonables para los parámetros restantes.

4.3.2 Optimización de la arquitectura neuronal

Justificación metodológica: Se mantuvo la misma configuración experimental del Capítulo 3 para permitir una comparación directa y justa entre los dos enfoques de representación (embeddings Node2Vec vs. codificación one-hot). Al mantener constante la arquitectura MLP, el optimizador, la función de pérdida, y el protocolo de validación cruzada, cualquier diferencia en el rendimiento puede atribuirse exclusivamente al cambio en la representación de entrada, aislando el efecto de los embeddings. Modificar simultáneamente múltiples aspectos del pipeline imposibilitaría determinar qué factor contribuye a las mejoras observadas. Los detalles completos de la configuración se describen en el Capítulo 3.

Protocolo de validación cruzada: Se utiliza validación cruzada estratificada de 5 particiones (5-fold cross-validation). El dataset de 1.081 pares de compuestos se divide en 5 subconjuntos disjuntos de tamaño aproximadamente igual. En cada iteración (fold), uno de estos subconjuntos se reserva como conjunto de prueba (test set, $\sim$20 % de los datos), mientras que los cuatro subconjuntos restantes ($\sim$80 %) se utilizan para entrenamiento y validación. De este 80 %, un 10 % adicional se separa como conjunto de validación, resultando en una división aproximada de 70 % entrenamiento, 10 % validación, y 20 % test. El conjunto de validación se utiliza durante el entrenamiento para: (1) monitorear el error de generalización en cada época, (2) implementar early stopping cuando el error de validación deja de mejorar (evitando overfitting), y (3) seleccionar hiperparámetros óptimos comparando el error de validación entre diferentes configuraciones. El conjunto de test, que permanece completamente aislado durante el entrenamiento, se utiliza únicamente al final para reportar el rendimiento final del modelo. Este protocolo se repite para las 5 particiones, y los resultados se promedian para obtener estimaciones robustas del error de generalización.

La búsqueda de hiperparámetros se realizó en dos etapas. En la primera búsqueda en grilla, se exploraron los hiperparámetros listados en la Tabla 4.2, utilizando los cinco mejores embeddings identificados en el experimento anterior (Sección 4.3.1). Esta búsqueda integrada permite evaluar simultáneamente la calidad de los embeddings y los parámetros del MLP.

Hiperparámetro	Rango	Paso
tamaño de embedding	25 - 250	25
longitud de caminata	5 - 30	5
número de caminatas	5 - 30	5
tamaño de ventana	5 - 20	5
p	0.5 - 2.0	0.5
q	0.5 - 2.0	0.5
tasa de aprendizaje	10^{-4} - 10^{-2}	log
tamaño de batch	50 - 200	50
probabilidad de dropout	0.2 - 0.5	0.1

Tabla 4.2: Rango de valores utilizados en la búsqueda en grilla inicial para la exploración de hiperparámetros tanto del algoritmo de embedding como del modelo neuronal.

Hiperparámetro	Rango	Paso
Tamaño de capa 2	100 - 200	5
Tamaño de capa 3	50 - 200	5
Tamaño de capa 4	50 - 200	5

Tabla 4.3: Rango de valores utilizados en la segunda búsqueda en grilla para la exploración de hiperparámetros de la arquitectura neuronal.

Se evaluaron un total de 2.500 combinaciones diferentes de hiperparámetros para los embeddings. Para cada configuración de embeddings, se entrenó y evaluó el modelo neuronal utilizando validación cruzada de 5 particiones. Los mejores cinco conjuntos de embeddings se identificaron basándose en el error cuadrático medio (RMSE) en el conjunto de validación.

Tras seleccionar los mejores embeddings, se realizó una segunda búsqueda en grilla enfocada específicamente en optimizar la arquitectura de la red neuronal. Se encontró que los embeddings óptimos aparecían en 4 de los 5 mejores modelos, confirmando su calidad. La Tabla 4.3 presenta el espacio de búsqueda para esta segunda etapa.

La Tabla 4.4 muestra los cinco mejores resultados de la búsqueda en grilla. El mejor modelo se obtuvo con el siguiente conjunto de hiperparámetros: tasa de aprendizaje de $5 \cdot 10^{-4}$, tamaño de batch de 100, probabilidad de dropout de 0.3, y [128, 195, 110,

Capa2	Capa3	Capa4	MAE	RMSE
195	**110**	**110**	**0.074**	**0.0928**
185	195	190	0.074	0.0932
200	200	150	0.075	0.0944
200	185	75	0.076	0.0950
185	105	190	0.076	0.0955

Tabla 4.4: Error obtenido en la validación cruzada para la selección de hiperparámetros de la arquitectura neuronal. Se muestran las cinco mejores configuraciones ordenadas por MAE.

110, 1] neuronas en cada capa respectivamente. Se obtuvo un MAE de 0.074 (RMSE = 0.0928) en validación cruzada, y el coeficiente de Pearson R^2 fue de 0.92 para el conjunto de prueba. Estos resultados indican un buen rendimiento del modelo propuesto, dado que el error absoluto promedio fue del 7.4 %, y la distribución de valores de similaridad reales y predichos entre compuestos fue similar.

Análisis de la arquitectura óptima: Los hiperparámetros de la arquitectura neuronal no se encuentran en los límites del espacio de búsqueda explorado (Tabla 4.3). La capa 2 (195 neuronas) está próxima al límite superior de 200, pero las capas 3 y 4 (110 neuronas cada una) se sitúan en la región media del rango explorado (50-200). Esto sugiere que la exploración cubrió adecuadamente el espacio de hiperparámetros, evitando configuraciones subóptimas por restricción del rango. La arquitectura resultante [195, 110, 110] puede caracterizarse como de tamaño medio, con un patrón de reducción progresiva de neuronas desde la capa de entrada hacia la salida. Este diseño favorece la extracción jerárquica de características, donde las primeras capas capturan patrones complejos de los embeddings concatenados, mientras que las capas posteriores consolidan esta información hacia la predicción de similaridad.

Enfoque	**MAE**	**RMSE**	R^2
Codificación one-hot (Capítulo 3)	0.081	0.1019	0.90
Embeddings Node2Vec (Capítulo 4)	**0.074**	**0.0928**	**0.92**
Mejora relativa	**8.6 %**	**8.9 %**	**2.2 %**

Tabla 4.5: Comparación de rendimiento entre codificación one-hot y embeddings Node2Vec. Ambos enfoques utilizan embeddings de dimensión 47, resultando en entradas de dimensión 94 al concatenar los descriptores de dos compuestos. Los embeddings Node2Vec logran una reducción del error del 8.6 % en MAE y una mejora del coeficiente de determinación R^2.

Comparación con el enfoque basado en codificación one-hot

La Tabla 4.5 presenta una comparación cuantitativa entre el enfoque basado en codificación one-hot (Capítulo 3) y el enfoque basado en embeddings Node2Vec propuesto en este capítulo. Ambos modelos utilizaron la misma arquitectura MLP, el mismo conjunto de datos (1081 pares de compuestos de la vía de la Glucólisis) y los mismos valores ground truth calculados mediante el coeficiente de Tanimoto sobre fingerprints MACCS keys. La única diferencia radica en la representación de entrada.

Los resultados demuestran que el modelo neuronal entrenado con embeddings Node2Vec como entrada logra mejor rendimiento que el modelo entrenado con codificación one-hot. La reducción del MAE de 0.081 a 0.074 (RMSE de 0.1019 a 0.0928)

representa una mejora del 8.6 % en la precisión de las predicciones. Esta mejora se atribuye a que los embeddings Node2Vec capturan información de vecindad topológica del grafo metabólico, mientras que la codificación one-hot representa cada compuesto de manera aislada, sin considerar su contexto en la red. Adicionalmente, los embeddings permiten generar representaciones para compuestos sin estructura molecular conocida, aunque estos no puedan utilizarse durante el entrenamiento supervisado debido a la imposibilidad de calcular el ground truth.
La Figura 4.5 presenta los índices de Tanimoto reales (objetivos) comparados con aquellos predichos por la red neuronal para un fold representativo de validación cruzada. Asimismo, se añadieron líneas de tendencia para cada conjunto de partición. Se puede observar que la mayoría de los puntos se ubican sobre la línea de 45° para todos los conjuntos, indicando que el modelo realiza predicciones precisas. Además, se puede apreciar que los valores de Tanimoto reales y predichos son similares.

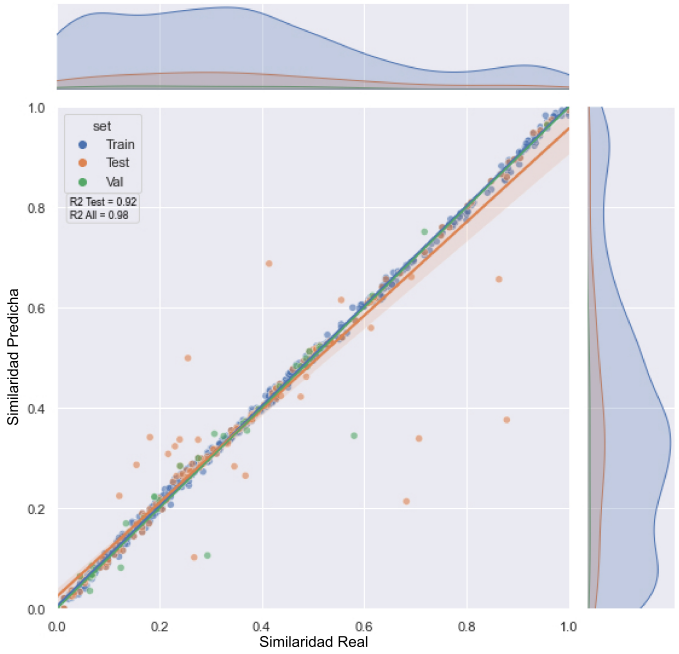

Figura 4.5: Comparación entre la similaridad de Tanimoto real y predicha. R^2 indica el coeficiente de correlación de Pearson al cuadrado para el conjunto de prueba y para todo el conjunto de datos, respectivamente. Los valores de entrenamiento se muestran en azul, los valores de prueba en naranja y los valores de validación en verde.

4.3.3 Predicción de similaridad en compuestos de estructura desconocida

El método de embedding utilizado no requiere la estructura molecular de los compuestos, dado que emplea la estructura del grafo metabólico para generar embeddings de compuestos. Esto permite la construcción de embeddings para todos los

compuestos, incluso aquellos para los cuales la estructura molecular es desconocida. Para evaluar la calidad de los resultados, se realizó un análisis de componentes principales (PCA) utilizando los embeddings correspondientes a la configuración óptima identificada en la Tabla 4.1: $q = 1{,}0$, $p = 0{,}8$, $size = 47$, $window = 10$, $len\_walk = 10$, $num\_walk = 300$.

La Figura 4.6 muestra la proyección de los embeddings en las primeras dos direcciones de componentes principales. En particular, los compuestos pertenecientes a la reacción R03270: C05125 + C15972 $\longleftrightarrow$ C16255 + C00068 se marcan en verde, y aquellos pertenecientes a la reacción R07618: C15973 + C00003 $\longleftrightarrow$ C15972 + C00004 + C00080 en rojo. Se puede observar cómo los compuestos que participan en cada reacción se encuentran efectivamente próximos entre sí. Esto resulta más evidente en la reacción R03270.

Figura 4.6: Gráfico de los primeros dos componentes principales para los embeddings. Se puede observar que los compuestos involucrados en cada reacción aparecen próximos entre sí en el gráfico.

Análisis de los componentes principales: La Figura 4.6 muestra que los compuestos que participan en la misma reacción metabólica se agrupan en regiones próximas del espacio de representación. Este comportamiento indica que los embeddings capturan relaciones topológicas de la red metabólica. En una red metabólica, los compuestos que participan en la misma reacción están conectados por aristas directas. Durante las caminatas aleatorias de Node2Vec, estos compuestos co-ocurren frecuentemente en la misma ventana de contexto, resultando en embeddings similares.

El agrupamiento de la reacción R03270 es más compacto que el de R07618. Esto se debe a diferencias en la conectividad: los compuestos de R07618 incluyen cofactores como C00003 (NAD^+) y C00004 (NADH), que participan en múltiples reacciones redox a lo largo de la vía metabólica. Esta alta conectividad hace que sus embeddings reflejen información de vecindades más amplias, distribuyéndose en una región más extensa del espacio PCA. En contraste, los compuestos de R03270 forman un subgrafo más aislado, generando embeddings más distintivos.

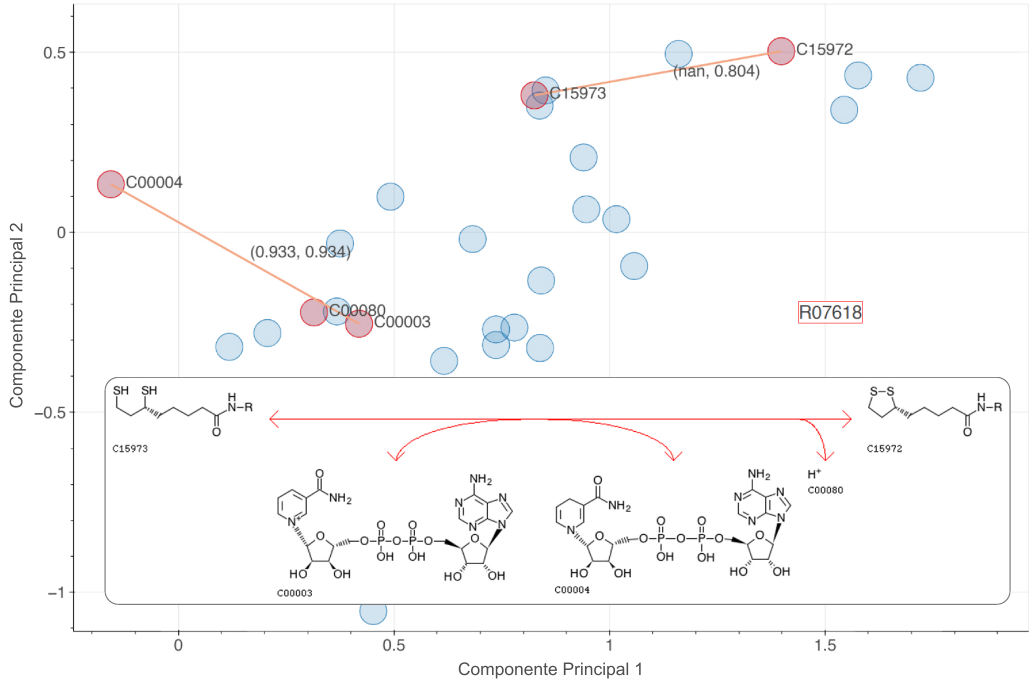

Figura 4.7: Ampliación de una región de la figura alrededor de la reacción R07618. Una línea naranja une ciertos compuestos seleccionados para comparar. A la izquierda se muestran los valores objetivo y a la derecha los valores predichos por el modelo neuronal.

La Figura 4.7 muestra el PCA en el área alrededor de la reacción R07618. A pesar de que la estructura molecular de los compuestos C15972 y C15973 es claramente conocida, nótese que esta incluye el sustituyente genérico "-R" en ambos casos. En consecuencia, la similaridad solo puede estimarse subjetivamente dado que esta información no puede procesarse para construir fingerprints. Esta situación no afecta nuestro enfoque, ya que utiliza embeddings aprendidos a partir de la topología del grafo de la red metabólica para representar compuestos. Como resultado, la similaridad predicha (0.804) está en concordancia con el análisis visual, dado que ambos compuestos comparten una alta proporción de sus estructuras moleculares. Asimismo, la similaridad predicha para los compuestos C00004 y C00003 es 0.934, que está en concordancia con el valor real (0.933).

La Figura 4.8 muestra un extracto de la proyección PCA, en el área alrededor de la reacción R03270. Esta reacción involucra los compuestos C15972 y C16255 sin

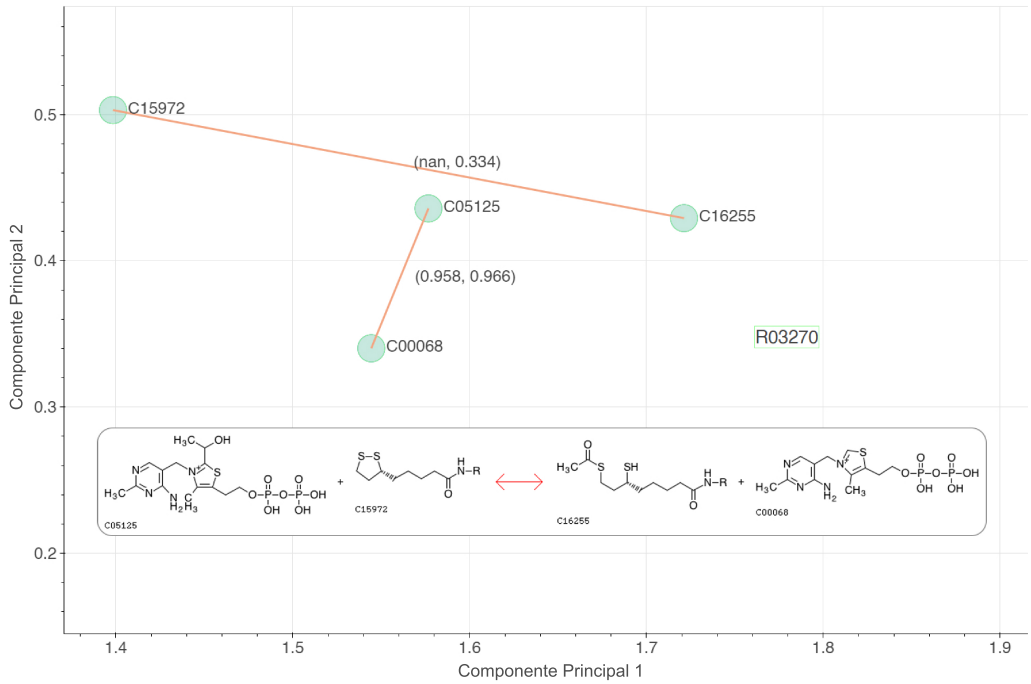

Figura 4.8: Ampliación de una región de la figura alrededor de la reacción R03270. Una línea naranja une ciertos compuestos seleccionados para comparar. A la izquierda se muestran los valores objetivo y a la derecha los valores predichos por el modelo neuronal.

estructura conocida. Esta reacción descompone los sustratos C05125 y C15972 en los productos C16255 y C00068. Como era de esperarse, el par C05125 y C00068 muestra alta similaridad (real: 0.958, predicha: 0.966) dado que comparten una gran parte de sus estructuras. Por otro lado, la similaridad predicha para C15972 y C16255 es 0.334, indicando que no comparten una proporción importante de sus estructuras. Sin embargo, podemos apreciar a partir de las estructuras en la ecuación química que esto no es correcto. Ejemplos como este indican que se necesita incorporar más información en el modelo para obtener predicciones más precisas para estructuras desconocidas.

4.4 Conclusiones

Este capítulo exploró el uso de embeddings generados con Node2Vec como alternativa a la codificación one-hot para la predicción de similaridad entre compuestos metabólicos. Se modeló la vía metabólica de la Glucólisis como un grafo $\mathcal{G} = \{v, e\}$, donde cada nodo representa un compuesto y las aristas conectan compuestos que participan en una misma reacción química. Los embeddings se generaron mediante el algoritmo Node2Vec, capturando información de vecindad topológica ausente en la representación one-hot.

Para optimizar la calidad de los embeddings, se exploraron 2.500 combinaciones de hiperparámetros del algoritmo Node2Vec utilizando el framework Optuna con muestreador Tree-structured Parzen Estimator (TPE). El análisis de importancia mediante FAnova reveló que los parámetros q y p, que controlan el balance entre exploración (BFS) y profundidad (DFS) en las caminatas aleatorias, fueron los más influyentes sobre el rendimiento del modelo. A partir de esta búsqueda, se identificaron los cinco mejores conjuntos de embeddings, que posteriormente se utilizaron para optimizar los hiperparámetros del MLP mediante búsqueda en grilla.

Los resultados experimentales demostraron que el modelo neuronal entrenado con embeddings Node2Vec como entrada logró mejor rendimiento que el modelo entrenado con codificación one-hot, alcanzando un MAE de 0.074 (RMSE = 0.0928) frente a un MAE de 0.081 (RMSE = 0.1019) del enfoque anterior, lo que representó una mejora del 8.6 % en la precisión de las predicciones. El coeficiente de determinación R^2 también mejoró de 0.90 a 0.92. Esta mejora se atribuyó a que los embeddings capturaron la estructura de vecindad del grafo metabólico, mientras que la codificación one-hot representó cada compuesto de manera aislada.

Una ventaja adicional de los embeddings fue su capacidad para representar compuestos con estructura molecular desconocida o parcialmente conocida (como aquellos con sustituyentes genéricos "-R"). El análisis cualitativo mediante PCA demostró que el modelo pudo generar predicciones de similaridad consistentes con análisis visuales para estos compuestos, aunque no pudieron utilizarse durante el entrenamiento supervisado debido a la imposibilidad de calcular el ground truth (coeficiente de Tanimoto).

Los resultados de este capítulo muestran que los embeddings basados en grafos constituyen una representación superior a la codificación one-hot para tareas de predicción de similaridad en redes metabólicas. El siguiente paso natural consiste en explorar arquitecturas neuronales más sofisticadas que integren explícitamente la estructura del grafo durante el aprendizaje de las representaciones de los nodos, de manera de enriquecer estas representaciones y así mejorar las características empleadas para la etapa de predicción de similaridad.

Capítulo 5

Embeddings más informativos, Aprendizaje supervisado y auto-supervisado

5.1 Introducción

Los capítulos previos establecieron dos enfoques complementarios para la predicción de similaridad molecular en redes metabólicas. El Capítulo 3 demostró la viabilidad del aprendizaje supervisado mediante perceptrones multicapa con codificación one-hot. El Capítulo 4 incorporó embeddings precalculados mediante Node2Vec. Sin embargo, ambos enfoques presentan limitaciones fundamentales: el primero no captura la estructura topológica de la red metabólica, mientras que el segundo utiliza embeddings genéricos no optimizados específicamente para la tarea de predicción de similaridad.

Este capítulo presenta NEAConv (*Node-Edge Attention Convolution*), el modelo final y contribución principal de esta tesis, que supera estas limitaciones mediante tres innovaciones clave. Primero, emplea una arquitectura de red neuronal en grafos que genera embeddings de manera *end-to-end* durante el entrenamiento, optimizándolos específicamente para la tarea objetivo. Segundo, incorpora un mecanismo de atención que integra nativamente tanto características de nodos como de aristas, capturando la información enzimática de las reacciones metabólicas. Tercero, utiliza una función de pérdida híbrida que combina aprendizaje supervisado y auto-supervisado, permitiendo realizar predicciones incluso para compuestos con estructura molecular desconocida.

Este trabajo presenta un modelo neuronal compuesto por dos partes principales: un bloque de generación de embeddings y un bloque de predicción. El bloque de predicción produce una estimación de la propiedad para pares de nodos en el grafo. Las redes neuronales en grafos (GNNs) han demostrado ser efectivas en tareas a nivel de nodo; sin embargo, las tareas centradas en aristas —donde el objetivo es predecir

propiedades de conexiones entre pares específicos de nodos— permanecen desafiantes, especialmente en bioinformática donde las interacciones suelen codificarse en las aristas. Los métodos existentes de paso de mensajes (MPNNs) sobresalen en tareas a nivel de nodo pero aún subutilizan los atributos de las aristas y requieren grandes conjuntos de datos etiquetados para supervisión.

Este capítulo aborda estas brechas mediante: (1) una arquitectura MPNN que trata nativamente características de nodos y aristas con un mecanismo de atención compartido inspirado en transformers, (2) una función de pérdida híbrida que combina objetivos supervisados y auto-supervisados, que permite aprender mejores representaciones aun a partir de datos para los que no hay ground truth, y (3) un desempeño competitivo en tareas que requieren regresión y clasificación a nivel de arista, incluyendo predicción de interacciones proteína-proteína.

La organización de este capítulo es la siguiente. La Sección 5.2 describe el conjunto de datos utilizado para validar el modelo. La Sección 5.3 presenta la arquitectura NEAConv, sus componentes y la función de pérdida híbrida. La Sección 5.4 documenta los resultados experimentales, incluyendo la configuración experimental base, el proceso sistemático de optimización de hiperparámetros, la evaluación del desempeño del modelo, y análisis cualitativos de las predicciones y embeddings aprendidos. Finalmente, la Sección 5.5 presenta un estudio de ablación exhaustivo.

5.2 Construcción del dataset de Vías Metabólicas

Basándose en nuestro estudio previo de glucólisis (Borzone, Di Persia & Gerard, 2022; Borzone, Ezequiel y col., 2022), el presente análisis expande la investigación para incluir seis vías metabólicas principales extraídas de la base de datos KEGG[1] (versión 95.2): *Glycolysis* (Glucólisis), *Starch and sucrose metabolism* (Metabolismo del almidón y sacarosa), *Pentose phosphate pathway* (Vía de las pentosas fosfato), *Citrate cycle (TCA cycle)* (Ciclo del citrato o ciclo de Krebs), *Pyruvate metabolism* (Metabolismo del piruvato), y *Propanoate metabolism* (Metabolismo del propanoato).

El conjunto de datos resultante abarca un total de 207 compuestos, de los cuales 174 tienen estructuras SMILES conocidas y 33 carecen de información estructural (*compuestos desconocidos*). Las estructuras moleculares de los compuestos conocidos se descargaron en formato SMILES desde PubChem[2]. Se utilizaron las fingerprints moleculares *MACCS keys* (Durant y col., 2002b) calculadas con la biblioteca RDKit[3] y se aplicó el coeficiente de Tanimoto (Bajusz y col., 2015) (Ecuación 1.2) para

[1] https://www.genome.jp/kegg/
[2] https://pubchem.ncbi.nlm.nih.gov/
[3] https://www.rdkit.org

calcular la similaridad entre pares de compuestos, de la misma forma que se hizo en los capítulos 3 y 4. En total, se construyeron 21,528 patrones, resultantes de la combinación a pares de los 174 compuestos con estructura conocida y sus interacciones con los compuestos de estructura desconocida. De estos patrones, 5390 están relacionados con compuestos de estructura desconocida, y por lo tanto no hay ground truth de similaridad para ellos.

Para este conjunto de datos, se utilizaron vectores one-hot como características de los nodos para representar los compuestos. Adicionalmente, se emplearon vectores one-hot basados en la familia de enzimas que median la reacción como características de las aristas. Las reacciones metabólicas son catalizadas por enzimas, proteínas que aceleran transformaciones químicas específicas. Por ejemplo, la enzima hexoquinasa cataliza la fosforilación de glucosa a glucosa-6-fosfato en la glucólisis. Las enzimas se clasifican mediante números EC (*Enzyme Commission*), un sistema jerárquico estandarizado por la IUBMB/IUPAC donde cada dígito especifica progresivamente la reacción: el primer dígito indica la clase de reacción (1: oxidorreductasas que transfieren electrones, 2: transferasas que transfieren grupos funcionales, 3: hidrolasas, 4: liasas, 5: isomerasas, 6: ligasas, 7: translocasas), los dígitos subsecuentes especifican subclase, sub-subclase y reacción específica. Para este trabajo se utilizó únicamente el primer dígito del número EC, resultando en un vector de 7 dimensiones por arista. La base de datos KEGG organiza estas enzimas mediante identificadores de ortólogos (KOs, *KEGG Orthology*), que agrupan genes de diferentes organismos que realizan la misma función, permitiendo vincular las reacciones metabólicas con datos de secuencia proteica y facilitar análisis comparativos entre especies. En casos donde múltiples enzimas estaban involucradas en una misma arista, los vectores one-hot correspondientes a cada familia enzimática participante se agregaron mediante la operación de unión, resultando en un vector donde cada posición indica si al menos una enzima de esa familia participa en la reacción (es decir, se toma el máximo elemento a elemento de los vectores individuales).

Durante la construcción del grafo, cada compuesto fue identificado como un nodo, y se conectaron pares de nodos cuando los mismos actuaban como sustrato y producto de una misma reacción, respectivamente. Se construyó un grafo integrando todas las reacciones de las seis vías metabólicas especificadas, representando la interacción estructural entre los compuestos metabólicos.

5.3 Modelo propuesto

Nuestro modelo opera en subgrafos $\mathcal{G}'$, estructurados alrededor de dos nodos centrales. Este enfoque es esencial para tareas centradas en aristas, donde el objetivo es predecir la presencia o las propiedades de conexiones entre pares específicos de

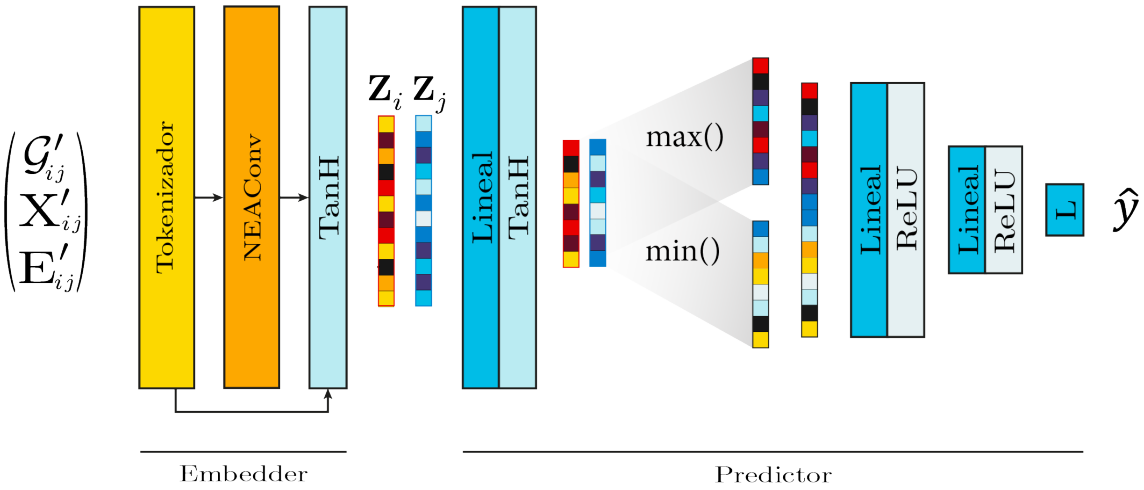

Figura 5.1: Resumen de la arquitectura del modelo. La entrada consta del subgrafo patrón $\mathcal{G}'_{i,j}$, características de los nodos $\mathbf{X}'_{\mathcal{G}'_{i,j}}$, y características de las aristas $\mathbf{E}'_{\mathcal{G}'_{i,j}}$. Estas entradas son procesadas a través del bloque de generación de embeddings: Tokenizador, la capa *NodeEdgeAttentionConv* (*NEAConv*) y la activación Tanh para formar los embeddings Z_i y Z_j. Los embeddings luego pasan por una capa lineal seguida de una activación Tanh, se reordenan y finalmente se alimentan a un perceptrón multicapa de tres capas para producir la salida.

nodos. El subgrafo se construye para enriquecer la representación de estos nodos centrales aprovechando su vecindad local. En cada subgrafo, los nodos centrales y sus vecinos directos, junto con todas las aristas existentes entre estos nodos, forman el contexto para el aprendizaje.

El modelo procesa cada subgrafo en dos etapas: generación de embeddings y predicción. En la fase de generación de embeddings, las características de los nodos dentro del subgrafo se transforman en vectores densos de baja dimensión $\mathbf{Z}$ que capturan información tanto estructural como basada en características. Estos embeddings, particularmente los de los nodos centrales, se concatenan y se pasan al módulo de predicción para estimar la propiedad deseada asociada con su conexión. El modelo completo se entrena de extremo a extremo utilizando una función de pérdida que combina entrenamiento supervisado y auto-supervisado, permitiéndole generar representaciones útiles incluso para nodos con información inicial limitada o ausente. La Figura 5.1 ilustra la arquitectura del modelo, con descripciones detalladas de cada componente proporcionadas en las secciones siguientes.

5.3.1 Generación de embeddings

Este bloque consta de dos componentes: el tokenizador y la capa *NodeEdgeAttentionConv* (*NEAConv*), como se ilustra en la Figura 5.1. Los embeddings de nodos son representaciones compactas y de baja dimensión de los nodos del grafo, que integran tanto las características de los nodos como las de las aristas. Nuestro modelo genera

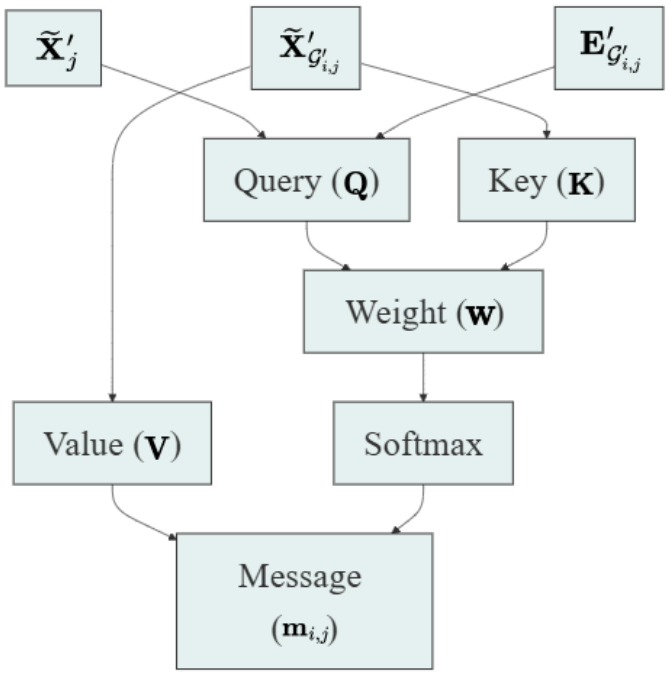

Figura 5.2: Diagrama que ilustra el mecanismo de atención en la función de mensaje. Muestra cómo las características de nodos y aristas se transforman en vectores de clave, consulta y valor para asignar pesos de atención, facilitando la agregación efectiva de información de los nodos vecinos.

estos embeddings capturando la estructura subyacente del grafo y las características de los nodos, incluso cuando se utiliza una codificación one-hot como entrada. Esta capacidad permite la propagación de información estructural y la generación de predicciones en tareas posteriores. Nos enfocamos en la arquitectura y los mecanismos utilizados en nuestro modelo para producir estas representaciones que codifican simultáneamente tres aspectos fundamentales: la estructura subyacente del grafo, las características de los nodos, y la información de las aristas.

El tokenizador toma las características de los nodos $\mathbf{X}'_{\mathcal{G}'_{i,j}}$ como entrada y las proyecta a un espacio más adecuado para la inferencia mediante un Perceptrón Multicapa (MLP), obteniendo una nueva representación $\widetilde{\mathbf{X}}'_{\mathcal{G}'_{i,j}}$. Esta representación es luego procesada por una sola capa *NEAConv*, una red neuronal de paso de mensajes diseñada específicamente para tareas centradas en aristas. En esta capa, se definen funciones personalizadas de paso de mensaje y actualización para alinearse con la arquitectura de nuestro modelo.

En la Figura 5.2, se muestra el proceso de paso de mensajes involucrado en la capa *NEAConv*. La información de cada nodo objetivo $i, j \in \mathcal{G}'_{i,j}$ se actualiza mediante el mecanismo de atención descrito en la Figura 5.2. Para un nodo dado j, el algoritmo proyecta inicialmente las características tokenizadas del nodo $\widetilde{\mathbf{X}}'_j$ y las de todo el patrón $\widetilde{\mathbf{X}}'_{\mathcal{G}'_{i,j}}$ en tres nuevos espacios con la misma dimensión que el original.

Estas proyecciones no lineales son realizadas por tres perceptrones separados, cada

uno compuesto por una capa lineal seguida de una activación sigmoidea. Dado que la información propagada proviene exclusivamente de vecinos de primer orden en lugar de una secuencia ordenada, los mensajes entrantes son intrínsecamente no ordenados; por lo tanto, no se aplica codificación posicional. El vector de consulta $\mathbf{Q}$ se genera a partir de la concatenación de las características del nodo objetivo $\widetilde{\mathbf{X}}'_j$ y, cuando están disponibles, las características de las aristas $\mathbf{E}'_{\mathcal{G}'_{i,j}}$, capturando la relevancia del nodo objetivo en el contexto de su vecindad. El vector de clave $\mathbf{K}$ se deriva de las características de los nodos fuente $\widetilde{\mathbf{X}}'_{\mathcal{G}'_{i,j}}$, codificando información esencial sobre los nodos fuente para ser comparada con el vector de consulta. El vector de valor $\mathbf{V}$, también generado a partir de las características de los nodos fuente $\widetilde{\mathbf{X}}'_{\mathcal{G}'_{i,j}}$, transporta la información que será propagada al nodo objetivo.
El peso vectorial $\mathbf{w}$, que representa la importancia de la información de cada nodo fuente para el nodo objetivo, se calcula como:

$$\mathbf{w} = \text{sum}(\mathbf{Q} \odot \mathbf{K}, \text{dim} = 1) \tag{5.1}$$

donde $\mathbf{Q} \odot \mathbf{K}$ representa el producto elemento a elemento (producto de Hadamard) de los dos tensores. Este vector $\mathbf{w}$ se normaliza luego utilizando una función *softmax*. El vector normalizado $\mathbf{w}$ se utiliza para construir la nueva representación $\widehat{\mathbf{X}}'_j$ del nodo central, a través de la combinación ponderada de las representaciones $\mathbf{V}$ de los nodos vecinos. La función de actualización concatena los mensajes agregados:

$$\widehat{\mathbf{X}}'_j = \sum_{k \in \mathcal{N}(j)} \mathbf{m}_{k,j} \tag{5.2}$$

junto con las características originales tokenizadas del nodo $\widetilde{\mathbf{X}}'_j$, resultando en embeddings actualizados que integran tanto la información agregada como las características originales:

$$\mathbf{Z}_j = \tanh\left(\left[\widehat{\mathbf{X}}'_j, \widetilde{\mathbf{X}}'_j\right]\right) \tag{5.3}$$

5.3.2 Paso de predicción

El módulo de predicción en nuestro modelo es un perceptrón multicapa (MLP) diseñado para procesar los embeddings generados de los nodos y producir predicciones específicas para la tarea. Para garantizar que las predicciones del modelo sean invariantes a la permutación de los nodos, primero se realiza un reordenamiento de las características de los embeddings de los nodos. Este reordenamiento ayuda a capturar las características importantes a través de diferentes nodos mientras se mantiene la propiedad de invarianza a la permutación.
Inicialmente, se extraen los valores mínimos y máximos a lo largo de la dimensión de

las características de los embeddings de los dos nodos para los que se quiere predecir una propiedad. Estas características extraídas se concatenan para formar una nueva representación que encapsula la información crítica de los embeddings de los nodos. La arquitectura del MLP consta de tres capas lineales, cada una seguida por una función de activación, para producir la salida final de predicción.

5.3.3 Función de pérdida

La función de pérdida propuesta está diseñada para optimizar simultáneamente los embeddings de los nodos y las predicciones. Esta es la función de pérdida:

$$\mathcal{L} = \alpha\mathcal{L}_{\text{supervised}} + \beta\mathcal{L}_{\text{cosine}} + \gamma\mathcal{L}_{\text{cosine\_pred}} \tag{5.4}$$

donde α, β y γ son coeficientes de ponderación. Los experimentos preliminares mostraron que los tres componentes de la pérdida tienen el mismo orden de magnitud; por consiguiente, se fijó $\alpha = \beta = \gamma = 1$ para evitar privilegiar cualquier término individual. Estos coeficientes pueden, sin embargo, tratarse como hiperparámetros y ajustarse para conjuntos de datos particulares en trabajos futuros.
Los términos $\mathcal{L}_{\text{supervised}}$ y $\mathcal{L}_{\text{cosine}}$ son componentes supervisados que guían al modelo para alinear sus predicciones y embeddings con las etiquetas verdaderas, mientras que $\mathcal{L}_{\text{cosine\_pred}}$ actúa como un término auto-supervisado, refinando los embeddings al alinear las predicciones con la similitud coseno de los embeddings de los nodos. Los tres términos siguen la misma estructura: para tareas de regresión, se utiliza el error cuadrático medio (MSE), y para tareas de clasificación, se aplica la pérdida de entropía cruzada.
El término $\mathcal{L}_{\text{supervised}}$ minimiza la diferencia entre las salidas predichas $\hat{Y}$ y las etiquetas verdaderas Y, asegurando predicciones precisas. De manera similar, $\mathcal{L}_{\text{cosine}}$ compara la similitud coseno de los embeddings de los dos nodos para los que se quiere predecir una propiedad, denotada como $\widetilde{Y}$ (ver ecuación 5.5), con las etiquetas Y, guiando al modelo para aprender embeddings que reflejen relaciones significativas entre los nodos. Ambos términos aplican MSE para regresión y entropía cruzada para clasificación, dependiendo de la tarea.

$$\widetilde{Y} = \text{cosine\_similarity}(\mathbf{Z}_i, \mathbf{Z}_j) \tag{5.5}$$

Además de los términos supervisados, la función de pérdida incluye un componente auto-supervisado, $\mathcal{L}_{\text{cosine\_pred}}$, que alinea la salida predicha $\hat{Y}$ con la similitud coseno de los embeddings de los nodos $\widetilde{Y}$. Este término auto-supervisado ayuda al modelo a aprender representaciones robustas incluso para nodos con información desconocida o incompleta, estructurando los embeddings de manera significativa. Como resulta-

do, el modelo se vuelve más adaptable a escenarios donde faltan datos, mejorando su generalización y rendimiento. Durante el entrenamiento, si un patrón de entrada no tiene etiquetas verdaderas asignadas, solo se utilizará el tercer término, permitiendo al modelo aprender de manera no supervisada.
La combinación de estos términos permite al modelo aprender embeddings más robustos y generalizables, así como realizar predicciones precisas. La inclusión de nodos con información desconocida en la función de pérdida asegura que los embeddings capturen la estructura y relaciones generales dentro del grafo, mejorando la representación de todos los nodos, independientemente de la disponibilidad de etiquetas explícitas.

5.4 Resultados Experimentales

5.4.1 Configuración Experimental

La evaluación del modelo se realizó mediante **validación cruzada estratificada de 5 folds**, donde en cada fold se reservó adicionalmente un 10 % del conjunto de entrenamiento como conjunto de validación para monitorear el progreso del entrenamiento. Este esquema de validación asegura una estimación robusta del desempeño generalizante del modelo y permite detectar sobreajuste durante el entrenamiento.
El proceso de entrenamiento incorporó **early stopping** con una paciencia de 50 épocas. Se estableció un límite máximo de 500 épocas por experimento. Como optimizador se utilizó Adam (Kingma & Ba, 2014) con una tasa de aprendizaje variable según la configuración específica (típicamente en el rango 10^{-4} a 10^{-3}), aprovechando su adaptación automática de las tasas de aprendizaje por parámetro y su eficiencia en espacios de alta dimensionalidad.
Para esta tarea, el tokenizador empleó una arquitectura simple consistente en una capa lineal seguida de una función de activación Tanh, procesando las características codificadas en one-hot de los nodos del grafo metabólico. Esta configuración demostró ser suficiente para proyectar las representaciones iniciales al espacio de embeddings de dimensión controlada por el hiperparámetro `embedding_size`.
La función de pérdida empleada corresponde a la función híbrida presentada en la Ecuación 5.4, combinando términos de error cuadrático medio (MSE) sobre las predicciones directas de la similaridad entre pares de compuestos, con términos adicionales que penalizan discrepancias entre la predicción del modelo y la similaridad coseno calculada directamente sobre los embeddings aprendidos, forzando coherencia geométrica en el espacio latente. Los coeficientes de ponderación de esta función fueron objeto de optimización y evolucionaron desde un escalar único a un vector de múltiples componentes a lo largo del desarrollo del modelo.

La métrica principal de evaluación fue el **error absoluto medio (MAE)** calculado sobre el conjunto de test de cada fold, complementado con el error cuadrático medio (RMSE) y el coeficiente de determinación (R^2) para un análisis exhaustivo del desempeño. El MAE se priorizó por su interpretabilidad directa en la escala de similaridad $[0, 1]$ y su robustez ante valores atípicos.
El modelo NEAConv presenta un espacio de hiperparámetros amplio que incluye la arquitectura del predictor, el mecanismo de atención, la formulación de la función de pérdida, y diversos parámetros de entrenamiento. La siguiente subsección detalla el proceso sistemático de optimización que permitió identificar la configuración óptima reportada en este capítulo.

5.4.2 Optimización de Hiperparámetros y Configuración Final

Habiendo definido la arquitectura base del modelo NEAConv en la Sección 5.3, esta subsección documenta el proceso experimental que condujo a la configuración final utilizada en los resultados presentados posteriormente.
La optimización de hiperparámetros del modelo NEAConv se realizó mediante un enfoque iterativo secuencial a lo largo del desarrollo del trabajo. A diferencia de una búsqueda exhaustiva en grilla (*grid search*) o una búsqueda aleatoria completa, se optó por una estrategia de optimización incremental donde cada fase de experimentación se construyó sobre los hallazgos de la fase anterior.

Descripción de los Hiperparámetros

El desarrollo del modelo NEAConv involucró dos tipos de decisiones experimentales: búsqueda arquitectural y optimización de hiperparámetros. La búsqueda arquitectural evaluó mediante experimentación diferentes configuraciones estructurales del modelo (arquitectura del predictor, mecanismo de atención, formulación de la función de pérdida), determinando la configuración óptima para esta tarea. Los hiperparámetros, por su parte, son valores numéricos que ajustan el comportamiento del modelo (dimensionalidad de embeddings, tasas de aprendizaje, tamaños de batch, coeficientes de ponderación). Ambos aspectos se optimizaron mediante un proceso iterativo de cinco fases que se documenta a continuación.

Arquitectura del Predictor. Se evaluaron cuatro arquitecturas para el módulo predictor que transforma los embeddings de nodos en predicciones de similaridad: red feedforward estándar con activación tangente hiperbólica en la capa final, la misma arquitectura sin activación tanh, una arquitectura híbrida que combina producto escalar entre embeddings con una red feedforward, y esta última sin activación tanh.

La elección entre estas variantes afecta la capacidad del predictor para modelar relaciones lineales versus no lineales entre las representaciones aprendidas.

Mecanismo de Atención. Se evaluaron seis configuraciones del mecanismo de atención, variando dónde incorporar las características de aristas (`Q`, `K`, `QK`) y si aplicar normalización softmax (sufijo `nS`). Los detalles de cada configuración se describen en la Sección 5.4.2.

Dimensionalidad de Embeddings. Se exploró la dimensionalidad del espacio latente donde se proyectan los compuestos químicos, evaluando valores de 100, 150 y 207 componentes. El valor de 207 corresponde a la dimensión de entrada del modelo (número total de compuestos en el dataset), explorando así el caso donde no hay reducción de dimensionalidad en el espacio latente. Dimensiones mayores permiten representaciones más expresivas pero incrementan el riesgo de sobreajuste y el costo computacional, mientras que dimensiones menores favorecen la generalización a costa de capacidad representacional. Este parámetro debe balancearse con el tamaño del dataset y la complejidad de la tarea.

Características de Aristas. Se evaluó la inclusión de características de aristas (familias de enzimas EC codificadas como vectores one-hot) además de las características de nodos. La incorporación de estas características permite al mecanismo de atención considerar propiedades de las conexiones al calcular pesos de agregación, enriqueciendo las representaciones con información sobre el tipo de relación bioquímica que media cada arista. Esto resulta especialmente relevante en grafos biológicos donde las aristas representan relaciones heterogéneas.

Número de Conexiones por Nodo. Se exploró cuántas conexiones se consideran por nodo durante la agregación de mensajes. Con todas las conexiones disponibles, se utiliza el grafo metabólico completo; alternativamente, se puede seleccionar únicamente un número limitado de aristas de mayor peso por nodo, efectuando una poda del grafo. Esta selección impacta tanto la eficiencia computacional como la calidad de las representaciones aprendidas: retener solo conexiones fuertes puede reducir ruido estructural, mientras que incluir conexiones débiles puede aportar información contextual valiosa. Los experimentos revelaron que este factor constituye el determinante individual más importante del rendimiento del modelo.

Formulación de la Función de Pérdida. Se exploraron diferentes formulaciones de la función de pérdida híbrida que combina objetivos supervisados y autosupervisados. Las variantes evaluadas incluyen: combinación simple de predicción

directa con *ground truth*, formulaciones que incorporan términos de alineación basados en la similaridad coseno entre los embeddings $\mathbf{Z}_i$ y $\mathbf{Z}_j$ de los dos compuestos cuya similaridad se predice, y formulaciones completas que incluyen tanto términos de consistencia entre predicción y embeddings como términos específicos para compuestos desconocidos.
Por ejemplo, la formulación más simple utiliza únicamente el error entre predicción y *ground truth*: $\mathcal{L} = \text{MSE}(y, \hat{y})$. Una formulación intermedia incorpora alineación de embeddings: $\mathcal{L} = \text{MSE}(y, \hat{y}) + \text{MSE}(y, y_{cos})$, donde y_{cos} representa la similaridad coseno entre embeddings. La formulación completa de cuatro términos incorpora consistencia entre predicción y embeddings tanto para compuestos conocidos como desconocidos:

$$\mathcal{L} = \text{MSE}(y, \hat{y}) + \text{MSE}(y, y_{cos}) + \text{MSE}(\hat{y}, y_{cos}) + \text{MSE}(\hat{y}_u, y_{cos,u})$$

Esta formulación de cuatro términos representa una expansión de la ecuación general presentada en la Sección 5.3 (Ecuación 5.4). El tercer término de la ecuación original, $\mathcal{L}_{\text{cosine\_pred}}$, se desglosa aquí en dos componentes: $\text{MSE}(\hat{y}, y_{cos})$ para compuestos con estructura conocida, y $\text{MSE}(\hat{y}_u, y_{cos,u})$ para compuestos con estructura desconocida (indicados por el subíndice u). Esta separación permite ponderar de manera independiente la consistencia entre predicciones y embeddings según la disponibilidad de información estructural, y facilita el análisis experimental detallado de cada componente de la pérdida.

Coeficientes de Ponderación. La función de pérdida presentada en la Sección del Modelo (Ecuación 5.4) utiliza tres coeficientes α, β y γ para ponderar los términos $\mathcal{L}_{\text{supervised}}$, $\mathcal{L}_{\text{cosine}}$ y $\mathcal{L}_{\text{cosine\_pred}}$. Durante la experimentación, al expandir la formulación para incluir un cuarto término específico para compuestos desconocidos, el término auto-supervisado $\mathcal{L}_{\text{cosine\_pred}}$ se desglosó en dos componentes: uno para compuestos conocidos y otro para compuestos desconocidos, resultando en cuatro coeficientes $[\alpha, \beta, \gamma_1, \gamma_2]$ correspondientes a $\text{MSE}(y, \hat{y})$, $\text{MSE}(y, y_{cos})$, $\text{MSE}(\hat{y}, y_{cos})$ y $\text{MSE}(\hat{y}_u, y_{cos,u})$ respectivamente. Estos coeficientes permiten controlar la importancia relativa del aprendizaje supervisado versus el auto-supervisado, y balancear la optimización de predicciones versus la calidad de embeddings. La configuración con ponderación uniforme $\alpha = \beta = \gamma_1 = \gamma_2 = 1$ demostró ser robusta en los experimentos realizados.

Hiperparámetros de Entrenamiento. Los parámetros de entrenamiento controlan el proceso de optimización mediante descenso de gradiente. La *tasa de aprendizaje* determina la magnitud de las actualizaciones de parámetros en cada iteración

del optimizador Adam. Valores típicos explorados oscilaron entre 10^{-4} y 10^{-3}; tasas demasiado altas pueden causar divergencia o inestabilidad, mientras que tasas demasiado bajas resultan en convergencia lenta o estancamiento en mínimos locales subóptimos.

El *tamaño de batch* especifica cuántas muestras (pares de nodos con sus subgrafos asociados) se procesan simultáneamente antes de actualizar los pesos del modelo. Batches mayores proporcionan estimaciones más estables del gradiente pero requieren más memoria y pueden reducir la capacidad de generalización; batches menores introducen mayor estocasticidad que puede actuar como regularización implícita. Se exploraron valores de 16, 32 y 64 muestras por batch.

El *número máximo de épocas* establece un límite superior al entrenamiento. Se fijó en 500 épocas, suficiente para permitir convergencia en la mayoría de configuraciones.

Finalmente, el *criterio de paciencia* para early stopping determina cuántas épocas consecutivas sin mejora en la pérdida de validación se toleran antes de detener el entrenamiento anticipadamente. Se utilizó un valor de 50 épocas. Este mecanismo previene sobreajuste y reduce el costo computacional innecesario de épocas adicionales sin mejora.

Justificación de la Metodología Iterativa

Esta decisión metodológica responde a restricciones computacionales prácticas. El espacio de hiperparámetros del modelo NEAConv comprende al menos 10 dimensiones: arquitectura del predictor (4 variantes), tipo de atención (6 variantes), características de aristas (2 opciones), número de conexiones por nodo, tipo de función de pérdida (4+ variantes), índices de pérdida (combinatorio), tamaño de embeddings, tasa de aprendizaje, tamaño de batch, y coeficientes alpha (4 variables).

Una búsqueda exhaustiva en grilla con solo 3 valores por dimensión requeriría $3^{10} = 59,049$ experimentos. Dado que cada experimento con validación cruzada de 5 folds requiere entre 4 y 9 horas de cómputo, una búsqueda completa demandaría aproximadamente 27 años de cómputo continuo.

Se realizaron más de 500 experimentos a lo largo del desarrollo. De estos, 259 experimentos con registro completo de hiperparámetros se organizaron en 5 fases principales según el aspecto del modelo optimizado (Tabla 5.1). Los experimentos restantes corresponden a exploraciones preliminares y configuraciones intermedias cuyos registros solo preservaron métricas finales.

Metodología de Validación Estadística

La evaluación cuantitativa de diferencias entre configuraciones de hiperparámetros requiere un marco estadístico riguroso que permita distinguir mejoras genuinas de

Tabla 5.1: Resumen de las fases de optimización de hiperparámetros

Fase	Descripción	Hiperparámetros	N
1	Arquitectura base	model_type, alpha_loss	48
2	Embeddings	embedding_size, lr, batch_size	41
3	Mecanismo de atención	attention_type	90
4	Función de pérdida	loss_type, loss_indexes	45
5	Validación alphas	alpha1-4	35
	Total		**259**

variabilidad aleatoria inherente al proceso de entrenamiento. En este estudio se empleó **análisis de varianza (ANOVA)** como herramienta principal para contrastar hipótesis sobre el efecto de hiperparámetros categóricos en el rendimiento del modelo.

ANOVA de un factor (One-Way ANOVA). Para cada fase de optimización, se aplicó ANOVA de un factor para evaluar si las diferencias observadas en MAE entre configuraciones de hiperparámetros son estadísticamente significativas. El modelo estadístico subyacente descompone la variabilidad total en dos componentes:

$$\text{MAE}_{ij} = \mu + \alpha_i + \epsilon_{ij} \tag{5.6}$$

donde MAE_{ij} representa el error absoluto medio del experimento j bajo la configuración de hiperparámetro i, μ es la media global, α_i es el efecto del nivel i del factor (e.g., tipo de arquitectura, mecanismo de atención), y ϵ_{ij} captura la variabilidad aleatoria (ruido experimental, inicialización de pesos, orden de muestras en mini-batches).

El estadístico F de ANOVA compara la varianza entre grupos (diferentes configuraciones) con la varianza dentro de grupos (experimentos repetidos con la misma configuración):

$$F = \frac{\text{MS}_{\text{between}}}{\text{MS}_{\text{within}}} = \frac{\sum_{i=1}^{k} n_i(\bar{y}_i - \bar{y})^2/(k-1)}{\sum_{i=1}^{k}\sum_{j=1}^{n_i}(y_{ij} - \bar{y}_i)^2/(N-k)} \tag{5.7}$$

donde k es el número de grupos (configuraciones evaluadas), n_i es el número de experimentos por grupo, $N = \sum_i n_i$ es el tamaño muestral total, $\bar{y}_i$ es la media del grupo i, y $\bar{y}$ es la media global. Bajo la hipótesis nula de igualdad de medias ($H_0 : \alpha_1 = \alpha_2 = \cdots = \alpha_k = 0$), este estadístico sigue una distribución $F(k-1, N-k)$. Un valor $p < 0{,}05$ indica rechazo de la hipótesis nula, concluyendo que al menos una configuración difiere significativamente de las demás. Sin embargo, el valor p solo informa sobre la *existencia* de diferencias, no sobre su *magnitud práctica*.

Tamaño del efecto: Eta-cuadrado (η^2). Para cuantificar la magnitud del efecto del hiperparámetro sobre el rendimiento, se calculó el coeficiente eta-cuadrado, definido como la proporción de varianza explicada por el factor:

$$\eta^2 = \frac{\mathrm{SS}_{\text{between}}}{\mathrm{SS}_{\text{total}}} = \frac{\sum_{i=1}^{k} n_i(\bar{y}_i - \bar{y})^2}{\sum_{i=1}^{k}\sum_{j=1}^{n_i}(y_{ij} - \bar{y})^2} \tag{5.8}$$

Este estadístico toma valores entre 0 y 1, interpretándose según convenciones establecidas (Cohen, 1988): $\eta^2 < 0{,}06$ indica efecto pequeño (el hiperparámetro explica menos del 6 % de la variabilidad del rendimiento), $0{,}06 \leq \eta^2 < 0{,}14$ corresponde a efecto mediano, y $\eta^2 \geq 0{,}14$ señala efecto grande. Un efecto pequeño sugiere que otros factores (calidad de datos, inicialización, otros hiperparámetros no variados) dominan el rendimiento; un efecto grande indica que el hiperparámetro estudiado es crítico para el desempeño del modelo.

Interpretación conjunta de significancia y tamaño de efecto. La combinación de ambos estadísticos permite una interpretación matizada:

- $p < 0{,}05$ **y** η^2 **grande**: Diferencias significativas y sustanciales. El hiperparámetro es crítico y debe optimizarse cuidadosamente.
- $p < 0{,}05$ **y** η^2 **pequeño**: Diferencias estadísticamente detectables pero prácticamente irrelevantes. Muestras grandes pueden detectar diferencias minúsculas sin importancia práctica.
- $p \geq 0{,}05$ **y** η^2 **grande**: Ausencia de significancia pese a efecto aparentemente grande, usualmente por tamaño muestral insuficiente o alta variabilidad intragrupo.
- $p \geq 0{,}05$ **y** η^2 **pequeño**: No hay evidencia de efecto genuino. El hiperparámetro tiene impacto mínimo en el rendimiento.

Esta metodología estadística se aplicó consistentemente en todas las fases de optimización, reportándose valores de F, p y η^2 en las tablas de resultados correspondientes. Los grados de libertad del estadístico F se indican en notación estándar como $F(df_{\text{between}}, df_{\text{within}})$.

Fase 1: Arquitectura Base del Predictor

La primera fase de experimentación se enfocó en determinar la arquitectura óptima del **módulo predictor**, el componente que transforma los embeddings de nodos en la predicción final de similitud molecular. Este módulo recibe como entrada la

concatenación de los embeddings de los dos compuestos centrales (nodos i y j cuya similitud se busca predecir) y produce un valor escalar en $[0, 1]$ que estima el coeficiente de Tanimoto entre sus estructuras moleculares.

Para esta fase, se mantuvo fija la arquitectura del **módulo generador de embeddings** con la siguiente configuración: el tokenizador consistió en una capa lineal seguida de activación Tanh, proyectando los vectores one-hot de entrada a un espacio de dimensión 100; la capa NEAConv utilizó el mecanismo de atención sin características de aristas y con normalización softmax; se consideraron todas las conexiones del grafo metabólico (sin filtrado de aristas). Esta configuración fija permitió aislar el efecto de la arquitectura del predictor sobre el rendimiento del modelo.

Se evaluaron cuatro variantes arquitecturales con diferentes capacidades expresivas: red feedforward estándar con activación tangente hiperbólica en la capa final, la misma arquitectura sin activación tanh, una arquitectura híbrida que combina producto escalar entre embeddings con red feedforward, y esta última sin activación tanh. La arquitectura híbrida combina dos mecanismos de predicción. Por un lado, calcula el producto punto entre los embeddings procesados $\mathbf{Z}_i$ y $\mathbf{Z}_j$, capturando similitud directa en el espacio latente. Por otro lado, utiliza el módulo predictor completo descrito en la Sección 5.3, que extrae valores mínimos y máximos de los embeddings para garantizar invarianza a la permutación, y procesa esta representación concatenada mediante un MLP de tres capas. La motivación teórica es que el producto punto captura similitud cuando los embeddings están bien alineados, mientras que el MLP proporciona capacidad adicional para modelar transformaciones no lineales complejas.

Los resultados se presentan en la Tabla 5.2. El análisis de varianza (ANOVA one-way) no reveló diferencias significativas entre arquitecturas ($F(3, 44) = 0{,}640$, $p = 0{,}593$, $\eta^2 = 0{,}042$). El tamaño de efecto pequeño indica que la elección de arquitectura del predictor no constituía el factor limitante del rendimiento en esta configuración inicial. Este resultado sugiere que, con todas las conexiones del grafo metabólico incluidas, el cuello de botella no residía en la capacidad expresiva del módulo predictor sino en otros aspectos del modelo aún no explorados.

Tabla 5.2: Rendimiento por arquitectura del predictor (Fase 1)

Arquitectura	**MAE (media ± std)**	**N**
Híbrida con tanh	$0{,}1936 \pm 0{,}0017$	16
Híbrida sin tanh	$0{,}1936 \pm 0{,}0018$	16
Feedforward sin tanh	$0{,}1923 \pm 0{,}0047$	10
Feedforward con tanh	$0{,}1933 \pm 0{,}0018$	6

Fase 2: Configuración de Embeddings

En esta fase se exploraron los hiperparámetros que controlan la representación vectorial de los nodos y el proceso de optimización, manteniendo fijos todos los demás componentes del modelo. Se evaluó la dimensionalidad del espacio latente con valores de 100, 150 y 207 componentes, la tasa de aprendizaje en rangos desde menores a 0.0001 hasta mayores a 0.001, y el tamaño de batch con valores de 16, 32 y 64 muestras.
Ninguno de los hiperparámetros mostró efectos significativos sobre el rendimiento:

- Dimensionalidad de embeddings: $F(2, 38) = 0{,}173$, $p = 0{,}842$, $\eta^2 = 0{,}009$
- Tasa de aprendizaje: $F(3, 37) = 2{,}056$, $p = 0{,}123$
- Tamaño de batch: $F(2, 20) = 0{,}632$, $p = 0{,}599$

Se fijaron dimensionalidad 100, tasa de aprendizaje 0.00041 y tamaño de batch 16 como valores por defecto. Es notable que en todas las fases hasta este punto, el MAE se mantuvo estable alrededor de 0.194, sugiriendo la existencia de un factor limitante no explorado que domina el rendimiento independientemente de estas configuraciones. Este comportamiento motivó la exploración de aspectos más fundamentales del modelo en fases posteriores.

Fase 3: Mecanismo de Atención

En esta fase se evaluó el **mecanismo de atención** de la capa NEAConv, que determina cómo se agregan los mensajes de nodos vecinos durante el paso de mensajes en el grafo. La atención permite al modelo aprender dinámicamente qué vecinos son más relevantes para cada predicción, en lugar de promediar uniformemente todas las conexiones.
Se evaluaron seis configuraciones que exploran dos aspectos: (1) dónde incorporar las características de aristas $\mathbf{e}_{ij}$ en el cálculo de atención, y (2) si aplicar normalización softmax. Las configuraciones difieren según dónde se concatenan las características de aristas:

- `Q`: Características de aristas en el vector *query*.
- `K`: Características de aristas en el vector *key*.
- `QK`: Características de aristas en *query* y *key*.

El sufijo `nS` indica ausencia de normalización softmax sobre los pesos de atención. Los resultados se muestran en la Tabla 5.3.

Tabla 5.3: Rendimiento por tipo de atención (Fase 3)

Tipo	**Descripción**	**MAE (media ± std)**	**N**
Q	Solo query	$0{,}1942 \pm 0{,}0006$	10
K	Solo key	$0{,}1947 \pm 0{,}0004$	3
QK	Query-Key	$0{,}1948 \pm 0{,}0004$	8
QnS	Query sin softmax	$0{,}1944 \pm 0{,}0004$	5
KnS	Key sin softmax	$0{,}1947 \pm 0{,}0004$	4
QKnS	Query-Key sin softmax	$0{,}1945 \pm 0{,}0009$	2

El análisis de varianza no reveló diferencias significativas entre configuraciones ($F(5, 26) = 1{,}195$, $p = 0{,}339$, $\eta^2 = 0{,}059$). La configuración Q obtuvo el mejor rendimiento numérico (MAE = 0.1942), seguida de K (0.1947) y QK (0.1948). Todas las configuraciones de esta fase incorporan características de aristas, representando una mejora respecto a la Fase 2, que no las utilizaba. Se seleccionó `attention_type='Q'` por obtener el mejor MAE con menor complejidad computacional (requiere menos parámetros que QK).

Fase 4: Función de Pérdida

Esta fase representó un **punto de inflexión** en la optimización, al explorar formulaciones más sofisticadas de la función de pérdida que combinan aprendizaje supervisado con términos de auto-supervisión basados en consistencia de embeddings. La función de pérdida original consistía en un único término MSE entre predicción y ground truth: $\mathcal{L} = \text{MSE}(y, \hat{y})$. En esta fase se exploraron funciones híbridas que incorporan términos adicionales para regularizar el espacio de embeddings. El parámetro `loss_indexes` define qué pares de señales se comparan mediante MSE. La nomenclatura de índices corresponde a:

- 0: ground truth y (similitud de Tanimoto real entre compuestos)
- 1: predicción directa $\hat{y}$ (salida del módulo predictor)
- 2: producto punto de embeddings de los compuestos $\langle \mathbf{Z}_i, \mathbf{Z}_j \rangle$
- 3: similitud coseno de embeddings $\cos(\mathbf{Z}_i, \mathbf{Z}_j)$
- 4: predicción directa $\hat{y}_u$ para compuestos desconocidos
- 5: producto punto de embeddings para compuestos desconocidos
- 6: similitud coseno de embeddings para compuestos desconocidos

Cada par de índices $[a, b]$ genera un término $\text{MSE}(\text{señal}_a, \text{señal}_b)$ en la función de pérdida. Por ejemplo, el par $[0, 1]$ corresponde al término supervisado estándar

$\text{MSE}(y, \hat{y})$; el par $[0, 3]$ compara el ground truth con la similitud coseno de embeddings $\text{MSE}(y, \widetilde{Y})$; y el par $[1, 3]$ impone consistencia entre predicción y embeddings $\text{MSE}(\hat{y}, \widetilde{Y})$.
El análisis de varianza para `loss_indexes` reveló el **primer efecto estadísticamente significativo** del estudio ($F(10, 34) = 269{,}78$, $p < 0{,}001$, η^2 grande), indicando que la formulación de la función de pérdida es crítica para el rendimiento del modelo. La configuración final seleccionada fue `loss_indexes=[[0,1],[0,3],[1,3],[4,6]]`, correspondiente a:

$$\mathcal{L} = \text{MSE}(y, \hat{y}) + \text{MSE}(y, y_{cos}) + \text{MSE}(\hat{y}, y_{cos}) + \text{MSE}(\hat{y}_u, y_{cos,u}) \tag{5.9}$$

Esta formulación incorpora cuatro principios clave:

1. **Aprendizaje supervisado directo** ($\text{MSE}(y, \hat{y})$): El modelo debe predecir correctamente la similitud real.

2. **Alineación de embeddings con ground truth** ($\text{MSE}(y, y_{cos})$): La similitud coseno de los embeddings debe correlacionar con la similitud molecular real, forzando que el espacio latente sea semánticamente significativo.

3. **Consistencia entre predicción y embeddings** ($\text{MSE}(\hat{y}, y_{cos})$): La predicción directa del perceptrón y la similitud de embeddings deben ser coherentes, evitando soluciones donde el predictor compensa embeddings mal estructurados.

4. **Generalización a compuestos desconocidos** ($\text{MSE}(\hat{y}_u, y_{cos,u})$): Aplica la restricción de consistencia también a compuestos sin conexiones en el grafo, mejorando la capacidad de generalización del modelo.

Nótese que los dos últimos términos de esta formulación, $\text{MSE}(\hat{y}, y_{cos})$ y $\text{MSE}(\hat{y}_u, y_{cos,u})$, pueden interpretarse como un único término conceptual que compara la predicción del modelo con la similitud coseno entre embeddings, independientemente de si el compuesto tiene estructura conocida o desconocida. Esta es precisamente la forma del término $\mathcal{L}_{\text{cosine\_pred}}$ en la función de pérdida general presentada en la Ecuación 5.4, donde la distinción entre compuestos conocidos y desconocidos se realiza únicamente a nivel de implementación para permitir la ponderación diferencial durante la experimentación.

Fase 5: Validación de Coeficientes de Ponderación

Con la función de pérdida de 4 términos establecida, se realizó una **validación exhaustiva de los coeficientes de ponderación** $[\alpha, \beta, \gamma_1, \gamma_2]$ que balancean la

importancia relativa de cada término. El coeficiente α pondera el término supervisado, β pondera la alineación de embeddings con ground truth, y γ_1, γ_2 ponderan los términos de consistencia predicción-embeddings para compuestos conocidos y desconocidos respectivamente. Se evaluaron 35 combinaciones de valores $\{0{,}1; 1{,}0; 2{,}0\}$ para cada coeficiente. Valores menores reducen la influencia del término correspondiente, mientras que valores mayores lo enfatizan.

La distribución de resultados mostró **robustez notable**: el 75 % de las configuraciones obtuvieron MAE $< 0{,}011$, indicando que el rendimiento es relativamente insensible a la ponderación exacta de los términos, siempre que todos estén presentes. Esto sugiere que la arquitectura de la función de pérdida (qué términos incluir) es más importante que sus coeficientes específicos (cómo ponderarlos).

La configuración baseline con ponderación uniforme $\alpha = \beta = \gamma_1 = \gamma_2 = 1$ obtuvo MAE = 0.0099, ubicándose en el mejor cuartil de rendimiento. Por principios de parsimonia y reproducibilidad, se adoptó esta configuración simple sin necesidad de optimización adicional. Esta decisión también facilita la interpretación: todos los términos contribuyen igualmente al entrenamiento, sin sesgar el aprendizaje hacia objetivos supervisados o auto-supervisados.

Análisis retrospectivo: número de conexiones por nodo

El análisis retrospectivo de los 255 experimentos registrados reveló que el factor determinante de la mejora de rendimiento no fue ninguno de los hiperparámetros explorados sistemáticamente en las cinco fases, sino el parámetro `n_edges`, que controla **cuántas conexiones se consideran por nodo** durante la agregación de mensajes (Tabla 5.4).

Tabla 5.4: Efecto del parámetro n_edges sobre el rendimiento

n_edges	**MAE (media ± std)**	**N**
0 (todas las conexiones)	$0{,}1939 \pm 0{,}0084$	209
2 (top-2 por nodo)	$0{,}0125 \pm 0{,}0074$	46
Mejora relativa	**93.5 % (factor 15.5×)**	

Con `n_edges=0`, el modelo utiliza **todas las conexiones disponibles** en el grafo metabólico completo. Dado que los grafos metabólicos son densamente conectados (cada compuesto participa en múltiples reacciones enzimáticas), esto resulta en vecindarios muy grandes que incluyen muchas conexiones débiles o irrelevantes para la predicción de similitud estructural.

Con `n_edges=2`, el modelo selecciona únicamente las **2 conexiones de mayor centralidad** por nodo. La selección se basa en la *betweenness centrality* de cada arista, una métrica de teoría de grafos que cuantifica la fracción de caminos más cortos entre

todos los pares de nodos que atraviesan dicha arista. Aristas con alta betweenness centrality actúan como puentes críticos en la red metabólica, conectando regiones que de otro modo estarían pobremente comunicadas.
El efecto observado es **reducción de 93.5 % en MAE** (de 0.1939 a 0.0125), equivalente a una mejora por factor 15.5. Este hallazgo tiene múltiples implicaciones:

Implicaciones prácticas:

- **Reducción computacional**: Procesar solo 2 conexiones por nodo (en lugar de todas) reduce el costo de memoria y tiempo de cómputo, permitiendo escalar a grafos más grandes.
- **Eficiencia de datos**: El modelo requiere menos información (menos vecinos) para hacer predicciones, sugiriendo que las conexiones top-k capturan la esencia de la estructura metabólica relevante.

Implicaciones teóricas y biológicas:

- **Ruido estructural**: Las conexiones débiles en el grafo metabólico introducen ruido que confunde el aprendizaje. No todas las co-ocurrencias metabólicas reflejan similitud estructural molecular.
- **Capacidad de generalización**: Forzar al modelo a aprender con información limitada (solo top-2 vecinos) actúa como regularización implícita, evitando que memorice patrones específicos del grafo de entrenamiento y mejorando la generalización a nuevos compuestos.

Este descubrimiento resalta la importancia del **preprocesamiento y filtrado del grafo de entrada** como factor relevante en GNNs aplicadas a redes biológicas. En este caso particular, la selección de aristas basada en betweenness centrality tuvo un impacto mayor sobre el rendimiento que otros hiperparámetros explorados.
La Tabla 5.5 presenta la configuración de hiperparámetros resultante del proceso de optimización iterativa.

Limitaciones del Proceso de Optimización

El enfoque iterativo de optimización demostró ser efectivo para identificar muy buenas configuraciones con recursos computacionales limitados. La estrategia de explorar un factor a la vez permitió identificar rápidamente que arquitectura, embeddings y atención no eran factores críticos ($\eta^2 < 0{,}06$), focalizando recursos en la función de pérdida y estructura del grafo.

Tabla 5.5: Configuración óptima resultante del proceso de optimización

Componente	**Configuración Óptima**
Decisiones Arquitecturales	
Arquitectura del predictor	Híbrida (producto escalar + red feedforward)
Mecanismo de atención	Basado en vector *query* únicamente
Características de aristas	Incluidas (familias enzimáticas EC)
Número de conexiones	2 aristas de mayor peso por nodo
Formulación de pérdida	Cuatro términos (supervisado + auto-supervisado)
Hiperparámetros Numéricos	
Dimensionalidad de embeddings	100
Tasa de aprendizaje	0.00041
Tamaño de batch	16
Épocas máximas	500
Paciencia (early stopping)	50
Coeficientes de ponderación	$\alpha = \beta = \gamma_1 = \gamma_2 = 1$
Rendimiento	MAE = 0,0125 ± 0,0074 (N=46)
Mejor experimento	MAE = 0.0099

El proceso de optimización iterativa documentado en esta subsección identificó los hiperparámetros individuales más efectivos. Sin embargo, para comprender la contribución relativa de los componentes arquitectónicos principales —el mecanismo de atención y la función de pérdida híbrida— se realizó un estudio de ablación sistemático que se presenta en la Sección 5.5.

5.4.3 Desempeño Global

Utilizando la configuración de hiperparámetros identificada en la subsección anterior (Tabla 5.5), se evaluó el desempeño del modelo en la tarea de predicción de similaridad entre compuestos metabólicos. El modelo recibe como entrada codificaciones one-hot de los compuestos y produce predicciones del coeficiente de Tanimoto.
El modelo alcanzó un MAE de 0.01013 en validación cruzada de 5 folds. Dado que el coeficiente de Tanimoto varía de 0 a 1, este error representa solo el 1.01 %, indicando un alto grado de precisión en las predicciones del modelo. Este resultado representa una mejora sustancial respecto a los enfoques basados en MLP con codificación one-hot (MAE = 0.081, Capítulo 3) y embeddings Node2Vec (MAE = 0.074, Capítulo 4).

5.4.4 Análisis Cualitativo

La Figura 5.3 muestra las estructuras parciales de los compuestos con IDs KEGG C15972, C15973 y C16255. Aunque las estructuras moleculares son parcialmente conocidas, cada una incluye un sustituyente genérico “-R”, haciendo imposible cal-

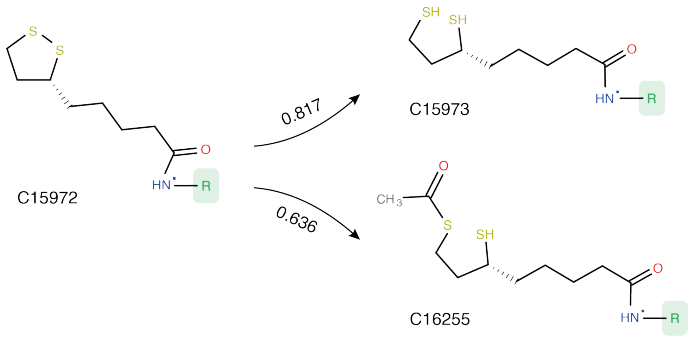

Figura 5.3: Estructuras parciales de los compuestos con los IDs KEGG C15972, C15973 y C16255. Los sustituyentes genéricos, marcados en verde como "-R", indican que cualquier grupo funcional puede sustituir un átomo de hidrógeno en la estructura base del compuesto, haciendo imposible crear un fingerprint para el compuesto. Las flechas indican los valores de similaridad predichos entre los compuestos.

cular fingerprints y por tanto evaluar objetivamente la similaridad estructural. Esta limitación ilustra un escenario donde nuestro enfoque basado en embeddings topológicos podría aportar información complementaria. Nuestro método predice una puntuación de similaridad de 0.817 entre C15972 y C15973, y de 0.636 entre C15972 y C16255. Aunque no es posible validar estas predicciones sin conocer la estructura completa de los compuestos, resultan plausibles dado que C15972 y C15973 comparten visualmente una mayor proporción de su estructura base.

5.4.5 Interpretación de Embeddings

La Figura 5.4 ilustra la distribución de los embeddings de los compuestos en el espacio de características aprendido, reducido a dos dimensiones mediante Análisis de Componentes Principales (PCA). Esta visualización permite evaluar cualitativamente la capacidad de la función de pérdida propuesta para codificar información estructural compleja en vectores densos continuos.

Se observa una fuerte coherencia en el agrupamiento local. El compuesto de referencia, Glucosa-1-fosfato (indicado en rojo), se encuentra rodeado inmediatamente por otros monosacáridos fosfatados y derivados de carbohidratos (tonos amarillos y verde claro). Las estructuras anotadas en esta región (cuadrante inferior izquierdo) comparten el anillo de piranosa y grupos fosfato, lo que indica que el modelo ha aprendido a preservar estas subestructuras funcionales críticas con alta fidelidad en el espacio de embeddings. A medida que nos desplazamos hacia el cuadrante superior izquierdo, aparecen estructuras más complejas, posiblemente nucleótidos o conjugados de azúcar, manteniendo una similaridad media (verde azulado).

El gradiente de color, basado en la similaridad de Tanimoto, muestra una correlación inversa suave y consistente con la distancia euclidiana respecto al centroide de referencia. La transición de colores cálidos (alta similaridad) a fríos (baja similaridad)

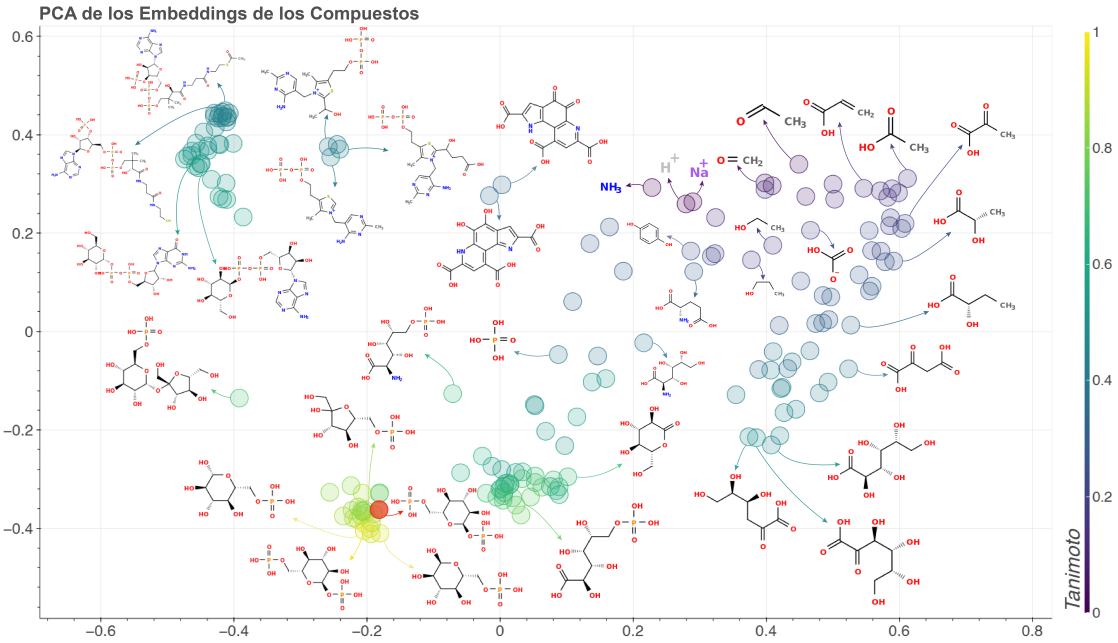

Figura 5.4: Proyección de los dos primeros componentes principales (PCA) de los embeddings generados por el modelo. Los puntos representan compuestos individuales, coloreados según su coeficiente de similaridad de Tanimoto respecto al compuesto de referencia: Glucosa-1-fosfato (marcado en rojo, coordenadas aprox. $-0{,}2, -0{,}4$). Las estructuras moleculares anotadas ilustran la correspondencia entre la posición en el espacio latente y la topología química, mostrando una transición desde azúcares fosfatados (amarillo/verde) hacia ácidos orgánicos de cadena corta y estructuras aromáticas simples (violeta).

no presenta discontinuidades abruptas, lo que sugiere que el espacio latente es continuo y navegable. En el extremo opuesto del gráfico (cuadrante superior derecho, tonos violetas), encontramos compuestos estructuralmente disímiles a la referencia: moléculas de bajo peso molecular, ácidos carboxílicos simples y fragmentos alifáticos. La segregación espacial entre los azúcares cíclicos polares y las moléculas lineales pequeñas demuestra que los embeddings capturan propiedades fisicoquímicas fundamentales más allá de la mera composición atómica. El modelo distingue eficazmente entre familias químicas, organizando el espacio de tal manera que la distancia geométrica en la proyección PCA es un proxy válido para la disimilitud química estructural. Esto valida la hipótesis de que la representación vectorial aprendida es robusta y semánticamente rica.

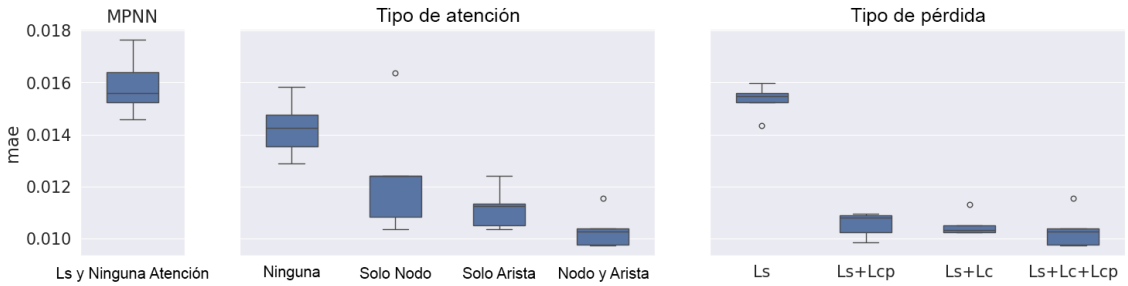

Figura 5.5: Error Absoluto Medio (MAE) para los tres modelos—baseline MPNN, experimento de atención y experimento de pérdida—presentados en ese orden. Valores más bajos indican mejor desempeño.

5.5 Estudio de Ablación

5.5.1 Diseño del Estudio

Para comprender mejor las contribuciones individuales de los componentes de pérdida y el mecanismo de atención, se realizó un estudio de ablación utilizando el conjunto de datos de similaridad de compuestos. Se llevaron a cabo dos experimentos independientes además de un modelo *baseline* MPNN básico.

Para facilitar la notación en las figuras, se utilizan las abreviaturas L_{s}, L_{c} y L_{cp} para referirse a $\mathcal{L}_{\text{supervised}}$, $\mathcal{L}_{\text{cosine}}$ y $\mathcal{L}_{\text{cosine\_pred}}$ respectivamente.

El experimento de **tipo de atención** mantuvo fija la pérdida compuesta $L_{\text{s}}+L_{\text{c}}+L_{\text{cp}}$ y evaluó cuatro configuraciones de atención: ninguna, solo nodo, solo arista, y el esquema combinado nodo&arista.

Inversamente, el experimento de **tipo de pérdida** mantuvo constante la atención nodo&arista y evaluó cuatro formulaciones de pérdida: L_{s}, $L_{\text{s}} + L_{\text{cp}}$, $L_{\text{s}} + L_{\text{c}}$, y la pérdida completa $L_{\text{s}} + L_{\text{c}} + L_{\text{cp}}$. En todas las configuraciones que utilizan la pérdida completa, se emplearon los coeficientes $\alpha = \beta = \gamma = 1$, valores que aseguran un balance equitativo entre los componentes supervisados y auto-supervisados.

5.5.2 Resultados Cuantitativos

La Figura 5.5 muestra boxplots que resumen el Error Absoluto Medio obtenido en cada uno de los cinco folds de validación cruzada para el modelo baseline y para cada configuración evaluada en los experimentos de tipo de atención y tipo de pérdida. Cada caja representa la distribución de errores a través de los 5 folds, proporcionando una vista concisa de la variabilidad y tendencia central asociada con cada configuración.

Experimento de tipo de pérdida. El análisis cuantitativo de las diferentes configuraciones de la función de pérdida revela el impacto progresivo de cada com-

ponente auto-supervisado. El modelo baseline con únicamente pérdida supervisada (L_{s}) alcanza un MAE promedio de 0.01523 (1.52 %). Al incorporar el componente de similaridad coseno entre embeddings ($L_{\mathrm{s}} + L_{\mathrm{c}}$), el error se reduce significativamente a 0.01036 (1.04 %), representando una mejora del 32 % respecto al baseline. La configuración alternativa con similaridad coseno entre predicciones ($L_{\mathrm{s}} + L_{\mathrm{cp}}$) obtiene un MAE de 0.01054 (1.05 %), mostrando también una mejora sustancial del 31 %. El modelo completo con la función de pérdida híbrida total ($L_{\mathrm{s}} + L_{\mathrm{c}} + L_{\mathrm{cp}}$) logra el mejor desempeño con un MAE de 0.01013 (1.01 %), demostrando que la combinación sinérgica de ambos componentes auto-supervisados produce mejoras adicionales marginales pero consistentes sobre los componentes individuales.

Experimento de tipo de atención. Los resultados del experimento de atención cuantifican la importancia relativa de las características de nodos y aristas en el mecanismo atencional. El modelo sin atención (*None*) presenta un MAE de 0.01426 (1.43 %), estableciendo el baseline para este experimento. La incorporación de atención exclusivamente sobre aristas (*Edge only*) reduce el error a 0.01118 (1.12 %), una mejora relativa del 22 %, indicando que la información estructural de las aristas aporta poder discriminativo significativo. Por otro lado, el esquema de atención exclusivamente sobre nodos (*Node only*) alcanza un MAE de 0.01247 (1.25 %), mejorando en 12 % sobre el baseline sin atención, pero siendo menos efectivo que la atención en aristas. Este resultado es particularmente relevante considerando que las características nodales en este problema son codificaciones one-hot simples, mientras que las características de aristas contienen información rica de similaridad molecular. El modelo completo con atención combinada nodo&arista (*Node & Edge*) obtiene el MAE más bajo de 0.01013 (1.01 %), demostrando una mejora del 29 % sobre el baseline sin atención y superando a ambos esquemas individuales. Esto confirma que la integración sinérgica de información nodal y estructural captura interacciones más complejas entre compuestos.

Comparación con baseline MPNN. Como punto de referencia adicional, se evaluó un modelo MPNN básico sin mecanismo de atención y utilizando únicamente pérdida supervisada (L_{s}), obteniendo un MAE de 0.01590 (1.59 %). Este resultado es el peor desempeño observado en todo el estudio de ablación, siendo superado incluso por el modelo con pérdida supervisada y atención nodo&arista (MAE = 0.01523). La diferencia de 36 % en MAE entre el baseline MPNN y el modelo completo propuesto subraya la contribución combinada crítica del mecanismo de atención y la función de pérdida híbrida. Estos resultados demuestran que ambos componentes son complementarios y necesarios para alcanzar el desempeño óptimo del modelo, validando empíricamente el diseño arquitectural propuesto.

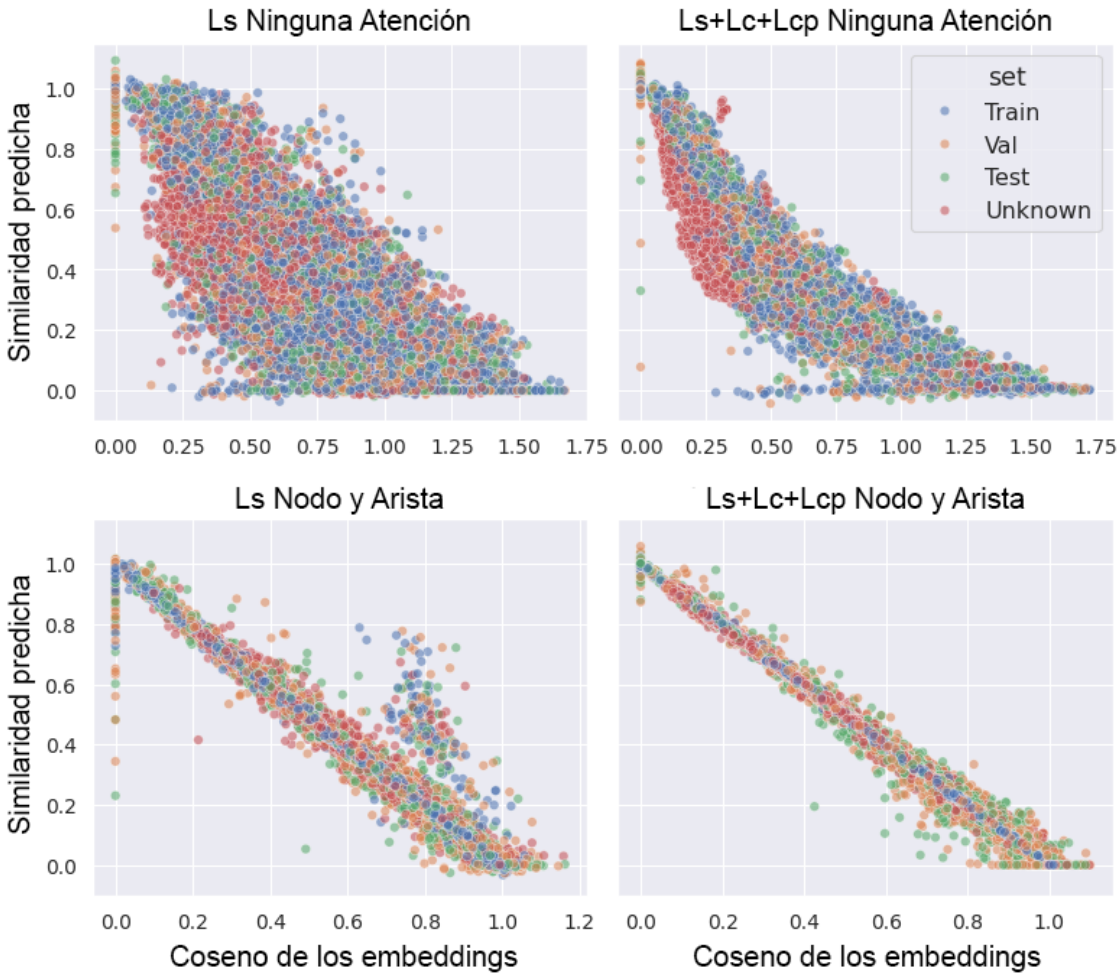

Figura 5.6: Similaridad de compuestos predicha (eje-y) versus distancia coseno entre embeddings (eje-x) bajo cuatro configuraciones de ablación: modelo baseline sin atención y pérdidas auto-supervisadas (arriba-izquierda), solo atención (arriba-derecha), solo pérdida híbrida (abajo-izquierda), y el modelo completo con atención y pérdida híbrida (abajo-derecha).

5.5.3 Análisis Cualitativo

La Figura 5.6 muestra, para cada par de compuestos, la distancia coseno entre embeddings en el eje horizontal y la similaridad predicha por el modelo en el eje vertical. Los cuatro paneles corresponden a la configuración baseline, el modelo solo con atención, el modelo solo con pérdida híbrida, y el modelo completo.

En el panel baseline, los puntos forman una nube amplia y difusa, indicando que la distancia entre embeddings no es un buen proxy para la predicción de similaridad. Habilitar cualquiera de los componentes por separado estrecha la nube y hace la tendencia más claramente monótona, mostrando que cada elemento ayuda independientemente a que los embeddings capturen similaridad.

Cuando tanto el mecanismo de atención como las pérdidas híbridas están activas, los puntos colapsan en una banda casi recta y apretada; la dispersión marcadamen-

te menor indica que los embeddings están mejor organizados en el espacio latente, permitiendo así al modelo predecir similaridad mucho más precisamente. Esta organización mejorada no solo explica el MAE más bajo observado, sino que también asegura que los compuestos que carecen de información estructural estén embebidos coherentemente junto con los compuestos conocidos, proporcionando una representación latente confiable para análisis posteriores.

5.6 Conclusiones del Capítulo

Este capítulo presenta un modelo novedoso basado en MPNNs diseñado para abordar tareas centradas en aristas en el aprendizaje basado en grafos. La arquitectura propuesta integra un mecanismo de atención que utiliza tanto características de nodos como atributos de aristas, mejorando su desempeño en tareas de regresión y clasificación de aristas donde los métodos tradicionales han enfrentado frecuentemente limitaciones.

La función de pérdida híbrida introducida en este modelo combina aprendizaje supervisado y auto-supervisado, permitiéndole mejorar predicciones y embeddings simultáneamente. Este enfoque asegura una generalización robusta, incluso en escenarios con información limitada o faltante sobre los nodos. Al utilizar la similaridad coseno entre embeddings de nodos y predicciones, el modelo organiza de mejor forma las representaciones aprendidas, facilitando el modelado de relaciones complejas dentro de grafos.

El modelo presenta una solución novedosa al problema de predecir similaridad entre compuestos con estructura desconocida, utilizando codificación one-hot como características de nodos para compensar la ausencia de información estructural. Este enfoque no solo permite buenas predicciones de similaridad sin requerir la estructura del compuesto, sino que también resulta valioso en escenarios con características de nodos faltantes.

Los resultados experimentales con un MAE de 0.01013 (1.01 % de error) y el estudio de ablación demuestran la importancia tanto del mecanismo de atención como de la función de pérdida híbrida para lograr representaciones de alta calidad que organizan los compuestos de manera coherente en el espacio latente, independientemente de si su estructura molecular es conocida o no. El capítulo siguiente explora la aplicación de este modelo a dominios adicionales de bioinformática, demostrando su versatilidad.

Capítulo 6

Aplicación en otros problemas

Para evaluar la versatilidad y capacidad de generalización del modelo propuesto, este capítulo evalúa su desempeño en dos tareas bioinformáticas adicionales que involucran predicción a nivel de aristas: predicción de interacciones proteína-proteína (PPI) y predicción de términos del Gene Ontology (GO). Estas tareas complementan la aplicación principal a redes metabólicas presentada en el Capítulo 5 y permiten validar que la arquitectura propuesta —con su mecanismo de atención para nodos y aristas, y su función de pérdida híbrida— puede adaptarse a dominios distintos sin modificaciones sustanciales.

La primera tarea, PPI, consiste en identificar posibles interacciones entre proteínas a partir del análisis de datos estructurales, de secuencia y funcionales. La segunda tarea, predicción de términos GO, implica asignar anotaciones relacionadas con funciones moleculares, procesos biológicos o componentes celulares a genes o proteínas, basándose en las anotaciones de genes conocidos y en datos de expresión genética.

6.1 Interacciones proteína-proteína

6.1.1 Introducción

Las interacciones proteína-proteína (Protein–Protein Interactions, PPI) subyacen a la organización funcional de la célula: median la formación de complejos, la transducción de señales y la regulación de procesos esenciales. Aunque técnicas experimentales de alto rendimiento —como el doble híbrido en levadura o la co-purificación acoplada a espectrometría de masas— han permitido cartografiar grandes porciones de las redes PPI, su costo, sesgos e incompletitud motivan el desarrollo de métodos computacionales complementarios.

Los primeros enfoques *in silico* para PPI explotaron primordialmente información de *secuencia*, proyectando proteínas a espacios vectoriales de dimensión fija (por ejemplo, mediante *Conjoint Triad*, CT) y entrenando clasificadores sobre pares de

secuencias (Shen y col., 2007). Progresivamente, estos modelos incorporaron arquitecturas profundas específicas de secuencia (e.g., CNN/RNN, variantes siamesas) que mejoraron la capacidad para capturar patrones locales y de largo alcance en las cadenas aminoacídicas (Chen y col., 2019; Li y col., 2020). Sin embargo, la *estructura de red* aporta un contexto adicional: las proteínas forman grafos complejos donde la conectividad y la vecindad informan la probabilidad de interacción. En consecuencia, la integración de atributos de secuencia con propiedades topológicas ha mostrado beneficios consistentes.

En este marco, las redes neuronales en grafos (GNNs) y, en particular, las redes de paso de mensajes (MPNNs), proporcionan un formalismo natural para combinar atributos de nodos (proteínas) y de aristas (relaciones), propagando información a través de la vecindad y aprendiendo representaciones que capturan dependencias estructurales (Gilmer y col., 2017). También han surgido autoencoders variacionales en grafos y variantes para grafos con aristas signadas que modelan explícitamente patrones de atracción/repulsión en redes de interacción (Kipf & Welling, 2016b; Yang y col., 2020).

En esta tesis, utilizamos la tarea de predicción de PPI como *banco de prueba secundario* para evaluar la capacidad de generalización del modelo propuesto. Para ello, consideramos cinco conjuntos de datos ampliamente usados en la literatura (HPRD, Human, *E. coli*, *Drosophila* y *C. elegans*) y comparamos contra métodos representativos tanto basados en grafos como basados exclusivamente en secuencia (Chen y col., 2019; Dunham & Beltrao, 2022; Li y col., 2020; Yang y col., 2020). Los resultados muestran que incorporar la estructura del grafo junto con descriptores de secuencia mejora la discriminación entre pares interactuantes y no interactuantes, apoyando el uso de arquitecturas de grafos para capturar relaciones proteicas complejas.

6.1.2 Descripción de datasets

Se utilizaron cinco conjuntos de datos[1] empleados previamente por Yang et al. (Yang y col., 2020). El conjunto de datos *HPRD* incluye aproximadamente 36,000 pares de interacciones proteicas conocidas (patrones positivos) y ∼36,000 pares de interacciones artificiales construidas aleatoriamente emparejando proteínas de diferentes ubicaciones subcelulares (patrones negativos). El conjunto *Human* abarca ∼9,000 proteínas y ∼72,000 interacciones; *E. coli* contiene ∼2,000 proteínas y 13,000 interacciones; *Drosophila* abarca ∼7,000 proteínas y ∼44,000 interacciones; y *C. elegans* incluye ∼2,500 proteínas con ∼8,000 interacciones. En todos los casos, los conjuntos de datos están balanceados con un número igual de interacciones positivas y

[1] http://www.csbio.sjtu.edu.cn/bioinf/LR_PPI/Data.htm

negativas.
El grafo para cada conjunto de datos se construyó calculando inicialmente una matriz de similaridad entre proteínas mediante ClustalO (Sievers y col., 2011). Esta matriz se utilizó con el algoritmo de grafos de vecinos más cercanos (*k-Nearest Neighbors Graph*, KNNG) (Paredes y col., 2006) para generar un grafo no dirigido, donde el número de vecinos k se determinó experimentalmente. Se evaluaron varios valores de k, manteniendo fija la configuración del modelo, para identificar el k óptimo que capturara conexiones significativas en el grafo. Las características de los nodos se construyeron a partir de representaciones de secuencia generadas por el método *Conjoint Triad* (CT), propuesto por Shen et al. (Shen y col., 2007), que proyecta las secuencias proteicas en un espacio vectorial basado en la frecuencia de tríadas. La matriz de similaridad entre proteínas también se utilizó como característica para las aristas.

6.1.3 Construcción de los datasets PPI

La construcción de los datasets de PPI siguió la metodología propuesta por Yang et al. (Yang y col., 2020), que se describe a continuación.

Muestras positivas. Las muestras positivas se obtuvieron de bases de datos curadas de interacciones proteína-proteína. Para HPRD, se utilizó la versión 2007 de la Human Protein Reference Database, eliminando self-interactions y duplicados, resultando en 36,591 pares de interacciones positivas. Para los datasets DIP (Database of Interacting Proteins), se extrajeron interacciones experimentalmente validadas para cada organismo: Human (37,020 interacciones), E. coli (6,954 interacciones), Drosophila (21,975 interacciones) y C. elegans (4,030 interacciones).

Muestras negativas. La construcción de muestras negativas siguió el supuesto biológico de que dos proteínas ubicadas en diferentes compartimentos subcelulares no interactúan. El proceso de selección incluyó los siguientes criterios:

1. Recopilar únicamente proteínas humanas anotadas con “human” en el campo ID.

2. Excluir secuencias con términos ambiguos de localización subcelular como “potential”, “probable”, “probably”, “maybe” o “by similarity”.

3. Incluir únicamente secuencias con localización subcelular única y bien definida.

4. Excluir secuencias anotadas como “fragments” y eliminar secuencias con menos de 50 residuos de aminoácidos.

5. Remover proteínas con aminoácidos inusuales como U (selenocisteína) y X (desconocido).

Las proteínas recolectadas se emparejaron aleatoriamente con otras proteínas de diferentes localizaciones subcelulares para generar pares no-interactuantes. Este proceso generó conjuntos balanceados con igual número de muestras positivas y negativas para cada organismo.

Construcción del grafo. Para cada dataset, se construyó un grafo no dirigido siguiendo estos pasos:

1. **Cálculo de similaridad de secuencia**: Se utilizó ClustalO (Sievers y col., 2011) para alinear todas las secuencias proteicas y calcular una matriz de similaridad $S \in \mathbb{R}^{N \times N}$, donde N es el número de proteínas y S_{ij} representa la similaridad de secuencia entre las proteínas i y j.

2. **Generación del grafo**: Se aplicó el algoritmo k-Nearest Neighbors Graph (KNNG) (Paredes y col., 2006) sobre la matriz S. Para cada proteína i, se conectó con sus k vecinos más similares, donde k se determinó empíricamente para cada dataset. Los valores óptimos de k fueron: HPRD ($k = 15$), Human ($k = 12$), E. coli ($k = 8$), Drosophila ($k = 10$) y C. elegans ($k = 10$).

3. **Características de nodos**: Las características de cada nodo se construyeron utilizando el método Conjoint Triad (CT) (Shen y col., 2007), descrito en detalle en la siguiente subsección.

4. **Características de aristas**: Las aristas del grafo se caracterizaron utilizando directamente los valores de similaridad de ClustalO, es decir, $e_{ij} = S_{ij}$ para cada arista (i, j) en el grafo.

6.1.4 Codificación de secuencias proteicas: Conjoint Triad (CT)

El método Conjoint Triad (CT), propuesto por Shen et al. (Shen y col., 2007), transforma secuencias proteicas de longitud variable en vectores de longitud fija, preservando información sobre la composición local de aminoácidos y su contexto.

Clasificación de aminoácidos. El método CT primero agrupa los 20 aminoácidos estándar en 7 clases según sus propiedades fisicoquímicas (dipolo y volumen de cadena lateral):

- **Clase 1 (C1)**: Ala, Gly, Val — pequeños y no polares

- **Clase 2 (C2)**: Ile, Leu, Phe, Pro — hidrofóbicos
- **Clase 3 (C3)**: Tyr, Met, Thr, Ser — polares sin carga
- **Clase 4 (C4)**: His, Asn, Gln, Trp — polares con grupos aromáticos/amida
- **Clase 5 (C5)**: Arg, Lys — cargados positivamente
- **Clase 6 (C6)**: Asp, Glu — cargados negativamente
- **Clase 7 (C7)**: Cys — con grupo tiol

Esta clasificación captura mutaciones sinónimas: aminoácidos dentro de la misma clase tienden a sustituirse con efectos funcionales mínimos.

Codificación por tríadas. Una vez clasificada cada secuencia proteica en su representación de clases, se aplica una ventana deslizante de tamaño 3 que recorre la secuencia con paso 1. Cada ventana captura una tríada de clases consecutivas (c_i, c_{i+1}, c_{i+2}), donde $c_i \in \{1, 2, \ldots, 7\}$.
Dado que hay 7 clases y cada tríada consiste en 3 posiciones, existen $7^3 = 343$ posibles combinaciones. El vector CT final $\mathbf{v} \in \mathbb{R}^{343}$ cuenta las ocurrencias de cada tríada en la secuencia:

$$v_k = \frac{\text{freq}(k)}{L-2}, \quad k = 1, \ldots, 343 \tag{6.1}$$

donde $\text{freq}(k)$ es el número de veces que aparece la tríada k y L es la longitud de la secuencia. Esta normalización hace que el vector sea invariante a la longitud de la secuencia.

Ejemplo. Consideremos una secuencia corta: `AGILMFY`

1. Clasificación: A→C1, G→C1, I→C2, L→C2, M→C3, F→C2, Y→C3
2. Secuencia de clases: [1, 1, 2, 2, 3, 2, 3]
3. Tríadas extraídas:
    - Ventana 1: (1,1,2)
    - Ventana 2: (1,2,2)
    - Ventana 3: (2,2,3)
    - Ventana 4: (2,3,2)
    - Ventana 5: (3,2,3)

4. El vector CT de 343 dimensiones incrementa los contadores correspondientes a estas tríadas.

Este enfoque captura patrones locales de aminoácidos que son biológicamente relevantes para la predicción de interacciones proteína-proteína, ya que las tríadas preservan información sobre motivos estructurales y funcionales cortos en las secuencias proteicas.

6.1.5 Métricas de evaluación

La métrica utilizada para evaluar la predicción de interacción proteína-proteína (PPI) fue la puntuación F1, derivada de la precisión y la sensibilidad. La precisión se define como la proporción de predicciones positivas correctas sobre el total de predicciones positivas, expresada matemáticamente como:

$$\text{Precisión} = \frac{TP}{TP + FP} \tag{6.2}$$

donde TP representa los verdaderos positivos y FP los falsos positivos.
La sensibilidad, también conocida como exhaustividad, se define como la proporción de predicciones positivas correctas sobre el total de positivos reales, dada por la ecuación:

$$\text{Sensibilidad} = \frac{TP}{TP + FN} \tag{6.3}$$

donde FN denota los falsos negativos.
La puntuación F1 es la media armónica de la precisión y la sensibilidad, proporcionando una métrica única que balancea ambas medidas. Se calcula utilizando la siguiente ecuación:

$$F1 = 2 \times \frac{\text{Precisión} \times \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} \tag{6.4}$$

Esta métrica es particularmente útil en el contexto de la predicción de PPI, ya que considera tanto la precisión de las predicciones positivas como la capacidad de identificar todas las interacciones relevantes.

6.1.6 Diseño experimental

El experimento se realizó mediante una validación cruzada de 5 folds, repetida 10 veces para mitigar el efecto de la inicialización aleatoria. Este procedimiento promedió los resultados en múltiples ejecuciones, reduciendo la variabilidad en las métricas finales. Las mismas particiones fueron utilizadas por nuestro modelo y los métodos de referencia para garantizar una comparación justa. Cabe destacar que el dataset de *E. coli* se utilizó como conjunto de desarrollo para la optimización de todos los hiperpa-

rámetros del modelo. Los hiperparámetros óptimos obtenidos en *E. coli* se aplicaron directamente a los demás datasets (HPRD, Human, *Drosophila*, *C. elegans*), asegurando que las métricas reportadas reflejen el rendimiento de generalización del modelo sin sesgo por optimización directa sobre esos conjuntos. Experimentos preliminares indicaron que un tokenizador compuesto por dos capas lineales con ReLU y una función Tanh final produjo los mejores resultados.

Para evaluar el rendimiento del modelo propuesto, se realizó una comparación con varios enfoques del estado del arte que representan diferentes paradigmas metodológicos.

El *Signed Variational Graph Auto-Encoder* (S-VGAE) (Yang y col., 2020) es un método basado en grafos que aprende representaciones latentes de proteínas mediante un autoencoder variacional. El modelo codifica las secuencias proteicas usando el método *Conjoint Triad*, luego aplica un encoder basado en redes convolucionales de grafos (GCN) para obtener embeddings que capturan tanto las características de secuencia como la topología local de la red PPI. Un decoder reconstructor permite el entrenamiento no supervisado, y un clasificador feedforward realiza la predicción final.

El *Siamese Residual RCNN* (Chen y col., 2019) es un modelo basado en secuencias que emplea una arquitectura siamesa con pesos compartidos para procesar pares de proteínas. Cada rama consiste en múltiples unidades RCNN (Recurrent Convolutional Neural Network) que combinan capas convolucionales con unidades GRU bidireccionales residuales, permitiendo capturar tanto patrones locales como dependencias de largo alcance en las secuencias de aminoácidos.

Finalmente, *EnAmDNN* (Li y col., 2020) integra aprendizaje por ensambles con mecanismos de atención. El método emplea múltiples redes neuronales con diferentes arquitecturas cuyas predicciones se combinan mediante ensemble learning. Los mecanismos de atención permiten identificar regiones de la secuencia relevantes para la predicción de interacción. Este método fue identificado como el predictor basado en secuencias líder en el benchmark de Dunham et al. (Dunham & Beltrao, 2022).

6.1.7 Resultados cuantitativos

La Figura 6.1 resume las distribuciones de puntuación F1 para la tarea PPI. Para cada uno de los cinco organismos, cuatro diagramas de caja adyacentes comparan nuestro modelo con SVGAE, Siamese Residual RCNN y EnAmDNN. Cada diagrama de caja refleja las puntuaciones obtenidas en 10 repeticiones de validación cruzada de 5 folds; la línea dentro de la caja denota la F1 media de cada k-fold.

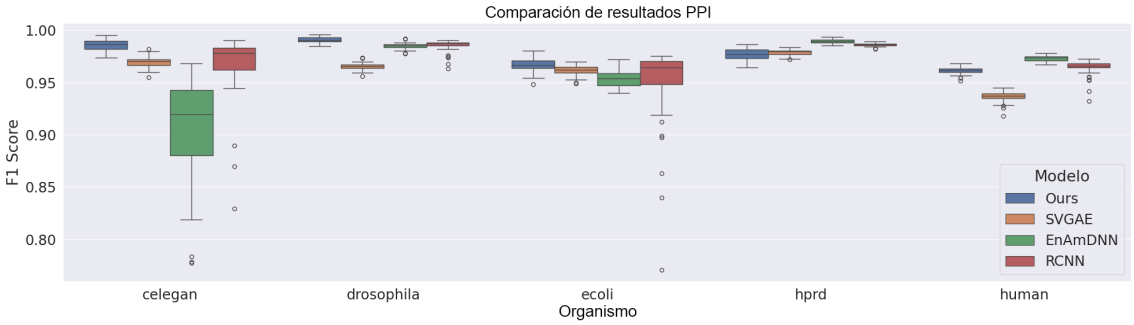

Figura 6.1: Comparación del rendimiento de nuestro modelo con SVGAE, Siamese Residual RCNN y EnAmDNN en cinco organismos para PPI. Cada diagrama de caja representa la distribución de las puntuaciones F1 obtenidas de las 10 repeticiones de la validación cruzada de 5 folds. Los diagramas de caja muestran la puntuación F1 media para cada k-fold.

6.1.8 Análisis estadístico

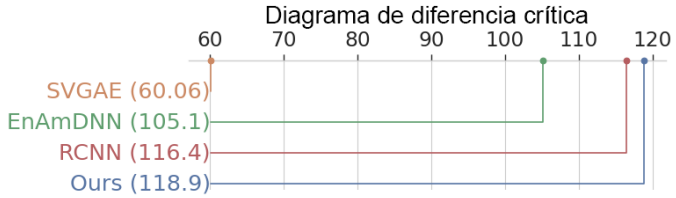

Figura 6.2: Diagrama de diferencias críticas para la predicción de PPI. Los modelos conectados por una línea no presentan diferencias estadísticamente significativas.

El test de Friedman reporta una diferencia general significativa entre los cuatro modelos PPI ($p = 7{,}8e-30$). El diagrama de diferencias críticas en la Figura 6.2 no muestra líneas de conexión, indicando que cada par de métodos difiere por más que la diferencia crítica. Nuestro modelo logra el rango promedio más alto, con el Siamese Residual RCNN en segundo lugar; la brecha entre ellos es estadísticamente significativa ($p = 1{,}5e-2$).
Este análisis estadístico demuestra que el modelo propuesto supera consistentemente a todos los métodos de comparación en todos los organismos evaluados.

6.2 Términos del Gene Ontology (GO)

6.2.1 Introducción

La anotación funcional de genes mediante la Ontología de Genes (Gene Ontology, GO) constituye un pilar para el estudio de procesos biológicos, funciones moleculares y componentes celulares. GO organiza el conocimiento en una jerarquía acíclica dirigida, donde términos cercanos a la raíz representan conceptos generales y términos

más profundos describen funciones específicas (Blake y col., 2013). La comparación entre términos puede cuantificarse con medidas semánticas que explotan el *information content* (IC) de ancestros comunes —por ejemplo, Resnik o Lin— para estimar hasta qué punto dos conceptos comparten información (Lin, 1998; Resnik, 1995).
La validación experimental de nuevas funciones génicas es costosa y lenta; por ello, los métodos computacionales para predecir etiquetas GO son esenciales para priorizar hipótesis (Radivojac y col., 2013). En este trabajo seguimos la metodología de *exp2GO* (Di Persia y col., 2022): se infiere similaridad semántica (en el espacio GO) a partir de similaridad de expresión (en el espacio transcriptómico) mediante una factorización no negativa acoplada y ponderada; una vez reconstruida la matriz semántica completa, se asignan candidatos GO con un esquema Bayesiano que respeta la estructura jerárquica. Esta tarea se usa aquí como evaluación secundaria y complementaria a las otras tareas del capítulo.

6.2.2 Descripción de datasets

Replicamos exactamente los conjuntos de datos, preprocesamiento y criterios de *exp2GO* (Di Persia y col., 2022). Se consideraron tres organismos con anotaciones funcionales extensas, lo que habilita una medición precisa del desempeño:

YEAST (S. cerevisiae). Matriz de expresión de 2,467 genes con 79 condiciones de microarreglos (Eisen y col., 1998). Tras excluir genes con datos faltantes, se retienen 605 perfiles. Las anotaciones GO se filtraron siguiendo las recomendaciones CAFA (evidence: EXP, IDA, IMP, IGI, IEP, TAS, IC; relaciones *is-a*/*part-of*) (Radivojac y col., 2013). Para los análisis donde se exige ocurrencia mínima de 3 genes por término, quedaron 422 genes para *Biological Process* (BP), 386 para *Molecular Function* (MF) y 442 para *Cellular Component* (CC) (Di Persia y col., 2022).

ARA (A. thaliana). Expresión en hojas bajo ciclos luz–oscuridad y condiciones controladas de temperatura (20 °C y 4 °C), enfocada en ritmos circadianos y respuesta a frío. Del total de 1,546 genes (32 condiciones), se aplicaron los mismos filtros CAFA sobre GO. Para BP, permanecen 656 genes; para el análisis con ocurrencia mínima, 521 (BP), 315 (MF) y 740 (CC) (Di Persia y col., 2022).

DICTY (D. discoideum). Datos de expresión de 1,219 genes a lo largo de 24 h de desarrollo (14 experimentos), obtenidos de *dictyBase* (Kreppel y col., 2004). Tras los filtros CAFA, en BP quedan 707 genes; para el escenario con ocurrencia mínima, 652 (BP), 366 (MF) y 630 (CC) (Di Persia y col., 2022).

Construcción de los datasets. La construcción de los datasets de predicción de términos GO siguió la metodología rigurosa propuesta por *exp2GO* (Di Persia y col., 2022), la cual se basa en las recomendaciones del consorcio CAFA (Critical Assessment of Functional Annotation) (Radivojac y col., 2013) para asegurar la calidad y confiabilidad de las anotaciones utilizadas.

Las anotaciones GO fueron sometidas a un proceso de filtrado estricto basado en los siguientes criterios: (1) **Códigos de evidencia experimentales**: Se incluyeron únicamente anotaciones respaldadas por evidencia experimental sólida (EXP: Inferred from Experiment, IDA: Inferred from Direct Assay, IMP: Inferred from Mutant Phenotype, IGI: Inferred from Genetic Interaction, IEP: Inferred from Expression Pattern, TAS: Traceable Author Statement, IC: Inferred by Curator). Se excluyeron códigos de evidencia computacionales (IEA, ISS, ISO) o menos confiables (NAS, ND) para evitar anotaciones circulares o de baja calidad. (2) **Tipos de relación**: Se consideraron únicamente las relaciones jerárquicas fundamentales de GO (*is-a* y *part-of*), excluyendo relaciones como *regulates*. (3) **True Path Rule**: Se aplicó la regla de propagación de anotaciones en el DAG de GO: si un gen está anotado con un término específico, automáticamente se considera anotado con todos los términos ancestros en la jerarquía hasta la raíz. (4) **Filtrado por frecuencia**: Para los análisis que requieren estadísticas robustas, se incluyeron únicamente términos GO anotados en al menos 3 genes distintos en el conjunto de datos.

Los perfiles de expresión genética fueron procesados eliminando genes con datos de expresión incompletos o missing values, normalizando los valores por gen (z-score) para que cada perfil tuviera media cero y varianza unitaria. Se calcularon dos matrices fundamentales: la **matriz de distancia de expresión** $D_{\text{exp}} \in \mathbb{R}^{N\times N}$, donde para cada par de genes (i, j) se calculó la distancia coseno entre sus perfiles de expresión $D_{\text{exp}}(i, j) = 1 - \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\|\|\mathbf{x}_j\|}$, y la **matriz de distancia semántica** $D_{\text{sem}} \in \mathbb{R}^{N\times N}$, calculada entre conjuntos de términos GO utilizando la medida *Relevance* con estrategia BMA (*Best Match Average*).

Medidas de similaridad semántica. Las medidas de similaridad semántica en GO cuantifican la cercanía funcional entre términos basándose en su posición en la jerarquía y en la especificidad de su ancestro común más informativo. La base de estas medidas es el **Information Content** (IC), que cuantifica la especificidad de un término GO t en función de su frecuencia de anotación:

$$IC(t) = -\log P(t) = -\log \frac{|\text{genes}(t)|}{|\text{genes(root)}|} \tag{6.5}$$

donde $|\text{genes}(t)|$ es el número de genes anotados con t o sus descendientes (por True Path Rule), y $|\text{genes(root)}|$ es el total de genes anotados en la subontología.

Términos más específicos (más profundos en el DAG) tienen mayor IC.
Las principales medidas utilizadas en *exp2GO* son: **Resnik** (Resnik, 1995), que mide la similaridad entre dos términos como el IC de su ancestro común más informativo (MICA):

$$\mathrm{sim}_{\mathrm{Resnik}}(t_1, t_2) = \max_{t \in \mathrm{ancestors}(t_1, t_2)} IC(t) \tag{6.6}$$

Lin (Lin, 1998), que normaliza Resnik por el IC promedio de ambos términos:

$$\mathrm{sim}_{\mathrm{Lin}}(t_1, t_2) = \frac{2 \cdot \mathrm{sim}_{\mathrm{Resnik}}(t_1, t_2)}{IC(t_1) + IC(t_2)} \tag{6.7}$$

produciendo valores en $[0, 1]$, y **Relevance** (Schlicker y col., 2006), que pondera Lin por la probabilidad del MICA:

$$\mathrm{sim}_{\mathrm{Rel}}(t_1, t_2) = \mathrm{sim}_{\mathrm{Lin}}(t_1, t_2) \cdot (1 - P(\mathrm{MICA}(t_1, t_2))) \tag{6.8}$$

favoreciendo términos específicos compartidos.
Para comparar dos conjuntos de términos $T_1 = \{t_{11}, \ldots, t_{1m}\}$ y $T_2 = \{t_{21}, \ldots, t_{2n}\}$, se utiliza la estrategia **Best Match Average** (BMA):

$$\mathrm{BMA}(T_1, T_2) = \frac{1}{2}\left(\frac{\sum_{t_1 \in T_1} \max_{t_2 \in T_2} \mathrm{sim}(t_1, t_2)}{|T_1|} + \frac{\sum_{t_2 \in T_2} \max_{t_1 \in T_1} \mathrm{sim}(t_2, t_1)}{|T_2|}\right) \tag{6.9}$$

Esta similaridad se convierte a distancia mediante $D_{\mathrm{sem}} = 1 - \mathrm{BMA}$. En *exp2GO* se utilizó Relevance con la variante BMA mínima (reemplazando máx por mín en la ecuación anterior) para capturar relaciones funcionales más conservadoras (Di Persia y col., 2022).

Método NMF acoplado y ponderado. El enfoque central de *exp2GO* consiste en reconstruir la matriz de distancias semánticas D_{sem} a partir de la matriz de expresión D_{exp} mediante Factorización Matricial No-negativa (NMF) acoplada y ponderada. El problema se formula como la minimización conjunta de dos términos de reconstrucción:

$$\min_{W, H_1, H_2 \geq 0} \|W_{\mathrm{exp}} \odot (D_{\mathrm{exp}} - W H_1^T)\|_F^2 + \lambda \|W_{\mathrm{sem}} \odot (D_{\mathrm{sem}} - W H_2^T)\|_F^2 \tag{6.10}$$

donde $W \in \mathbb{R}^{N \times p}$ es la matriz de factores latentes compartida (genes representados en un espacio de p dimensiones), $H_1, H_2 \in \mathbb{R}^{N \times p}$ son matrices de coeficientes para cada dominio, $\odot$ denota producto de Hadamard, y $W_{\mathrm{exp}}, W_{\mathrm{sem}} \in \mathbb{R}^{N \times N}$ son matrices de pesos que controlan la confianza en cada entrada (valores conocidos tienen peso 1, entradas a reconstruir tienen peso 0 en validación LOO). El hiperparámetro λ balancea la contribución de ambos dominios.

Esta formulación acoplada permite transferir información estructural del espacio transcriptómico (donde todos los datos están disponibles) al espacio semántico (donde pueden faltar anotaciones). La optimización se realiza mediante reglas multiplicativas de actualización que garantizan no-negatividad:

$$W \leftarrow W \odot \frac{(W_{\text{exp}} \odot D_{\text{exp}})H_1 + \lambda(W_{\text{sem}} \odot D_{\text{sem}})H_2}{(W_{\text{exp}} \odot (WH_1^T))H_1 + \lambda(W_{\text{sem}} \odot (WH_2^T))H_2} \tag{6.11}$$

con ecuaciones simétricas para H_1 y H_2. El proceso iterativo converge cuando el cambio relativo en la función objetivo es menor que un umbral $\epsilon = 10^{-4}$ o se alcanza un número máximo de iteraciones $t_{\text{máx}} = 500$.
Una vez reconstruida $\hat{D}_{\text{sem}}$, se infieren términos GO candidatos para cada gen mediante un esquema Bayesiano en tres etapas: (1) Para cada término GO g, se calcula la probabilidad posterior $P(g|q)$ de que el gen query q esté anotado con g, integrando sobre los $k = 80$ vecinos más cercanos en el espacio semántico reconstruido. (2) Se aplica un umbral adaptativo para seleccionar términos candidatos. (3) Se propagan las predicciones siguiendo la True Path Rule: si se predice un término, automáticamente se predicen todos sus ancestros. Los hiperparámetros óptimos reportados en *exp2GO* fueron $p = 80$ dimensiones latentes, $\lambda = 0{,}001$ (privilegiando ligeramente expresión), factor de ponderación $\gamma = 6$ para down-weighting de términos frecuentes, y $t_{\text{máx}} = 500$ iteraciones (Di Persia y col., 2022).

6.2.3 Métricas de evaluación

Los resultados de predicción de términos GO se informan conforme a las reglas de CAFA, utilizando la máxima medida F1 ($F1_{max}$), que evalúa las predicciones en todo el espectro de sensibilidad. El cálculo de $F1_{max}$ se realiza sistemáticamente para evaluar la calidad de las predicciones en funciones biológicas. Se recopilan los resultados de las predicciones y las anotaciones reales, calculando la precisión y la sensibilidad para cada función y ajustando los umbrales de decisión. Para cada umbral, se calcula la medida F1, definiendo $F1_{max}$ como el valor máximo obtenido. Un mayor valor de $F1_{max}$ indica un mejor balance entre precisión y sensibilidad, reflejando una mayor efectividad en las predicciones.

6.2.4 Protocolo y baselines

La evaluación sigue el marco metodológico establecido por *exp2GO* (Di Persia y col., 2022), empleando dos protocolos complementarios. En primer lugar, se aplica validación *leave-one-out* (LOO): para cada gen, se ocultan sus etiquetas y se reconstruyen las distancias semánticas faltantes a partir de expresión mediante factorización matricial no negativa (NMF) acoplada; posteriormente, se infieren términos

GO utilizando un esquema Bayesiano. En este protocolo se reportan sensibilidad (s^+), precisión (p) y F_1. En segundo lugar, se emplea un esquema tipo CAFA: el entrenamiento se realiza con un *snapshot* histórico de GOA y la validación con un *snapshot* posterior, propagando etiquetas conforme a la *True Path Rule* (Radivojac y col., 2013); en este caso se reporta $F_{\text{máx}}$ (máximo F_1 sobre todos los umbrales). La reconstrucción semántica utiliza distancia coseno para expresión y la medida semántica *Relevance* (estrategia de combinación BMA mínima), con hiperparámetros p=80, λ=0,001, γ=6 y $t_{\text{máx}}$=500, replicando los valores de *exp2GO* (Di Persia y col., 2022; Pesquita y col., 2009).

Se compara nuestro método con los modelos y configuraciones de referencia empleados en *exp2GO*. Los métodos incluyen: **exp2GO** (Di Persia y col., 2022), que realiza reconstrucción NMF acoplada de distancias semánticas a partir de expresión y asignación Bayesiana de términos; **BLAST** (Altschul y col., 1990; Radivojac y col., 2013), baseline de similitud de secuencia que transfiere anotaciones por homología, utilizado como referencia en CAFA; **NMF-GO** (Yu y col., 2020), que factoriza la matriz binaria gen–término e impone regularización semántica mediante la medida de Lin (Lin, 1998) para completar anotaciones; **DeepGOPlus** (Kulmanov y col., 2019), método de aprendizaje profundo sobre secuencias con CNN y combinación de similitud de motivos, ganador frente a métodos líderes de CAFA3; y **DFMF** (*Data Fusion by Matrix Factorization*) (B. Wang y col., 2014; Zitnik & Zupan, 2013), con variantes *gene-centric* y *function-centric* para fusión de datos heterogéneos mediante *tri-factorization* penalizada, empleado en *exp2GO* para comparación justa (particularmente en DICTY) (Di Persia y col., 2022). La evaluación estadística emplea la prueba de Friedman y diagramas de diferencia crítica con test post-hoc de Nemenyi, siguiendo el protocolo de *exp2GO* (Di Persia y col., 2022).

6.2.5 Resultados y análisis estadístico

Para evaluar la significancia estadística de los resultados obtenidos, se emplearon el test de Friedman y el diagrama de diferencias críticas, seguidos por el test post-hoc de Nemenyi. Los resultados del test de Friedman indican diferencias significativas en el desempeño de los modelos comparados (p = $1e{-}6$). El diagrama de diferencias críticas, presentado en la Figura 6.3, muestra que tanto el modelo propuesto como exp2GO son las mejores metodologías para la predicción de funciones genéticas, sin diferencias estadísticamente significativas entre ellos (p = 0,237). Sin embargo, el modelo propuesto supera significativamente a DeepGOPlus y NMF-GO (p = $1e{-}3$). La Figura 6.4 presenta una comparación detallada del rendimiento de los diferentes modelos evaluados en términos de la métrica F1_{max} para cada subontología de GO (Componente Celular, Función Molecular y Proceso Biológico) y cada organismo es-

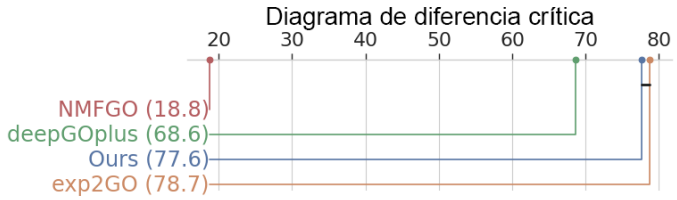

Figura 6.3: Diagrama de diferencias críticas. Los modelos conectados por una línea negra no presentan diferencias estadísticamente significativas. El valor de diferencia crítica es 1.36.

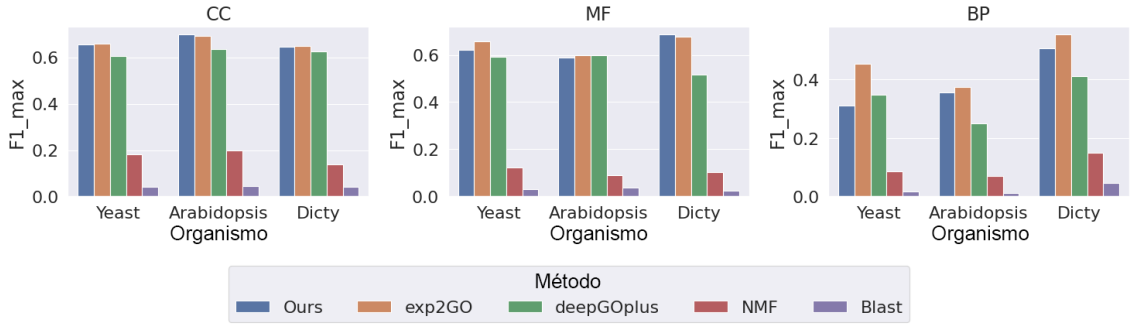

Figura 6.4: Comparación de las puntuaciones $F1_{max}$ de distintos modelos en cada subontología (Componente Celular, Función Molecular y Proceso Biológico) para cada organismo.

tudiado (YEAST, ARA, DICTY). Los resultados muestran que el modelo propuesto mantiene un rendimiento competitivo y consistente a través de todos los organismos y subontologías, validando su capacidad de generalización en la predicción de términos GO.

Análisis por subontología. El desempeño del modelo varía sistemáticamente según la subontología de GO evaluada. En **Cellular Component** (CC), que describe localizaciones subcelulares, el modelo propuesto alcanza los mejores resultados en los tres organismos, con $F1_{max}$ superiores a 0.70 en YEAST y DICTY. Esta ventaja puede atribuirse a que las localizaciones celulares presentan patrones de co-expresión transcripcional muy marcados: genes que codifican proteínas del mismo compartimento (e.g., mitocondria, núcleo) exhiben perfiles de expresión correlacionados, facilitando la transferencia de información mediante NMF acoplado. El modelo propuesto captura estas correlaciones integrando la estructura del grafo de co-expresión con las representaciones de secuencia.

En **Molecular Function** (MF), que describe actividades bioquímicas específicas (e.g., actividad enzimática, unión a DNA), el rendimiento del modelo propuesto es comparable al de exp2GO, ambos superando significativamente a métodos basados exclusivamente en secuencia como DeepGOPlus. Las funciones moleculares están fuertemente determinadas por dominios proteicos conservados, pero también

influenciadas por el contexto celular y regulatorio capturado por expresión. La capacidad del modelo para integrar ambos tipos de información explica su desempeño robusto en esta subontología. Los $F1_{max}$ en MF son consistentemente menores que en CC (típicamente entre 0.50-0.65), reflejando la mayor dificultad intrínseca de esta tarea: las funciones moleculares presentan mayor diversidad y especialización que las localizaciones.

Para **Biological Process** (BP), que describe cascadas funcionales complejas (e.g., apoptosis, ciclo celular), el modelo propuesto mantiene rendimiento competitivo con exp2GO en YEAST ($F1_{max} \approx 0{,}52$) y ARA ($F1_{max} \approx 0{,}48$), aunque ambos métodos muestran una caída de desempeño relativa respecto a CC y MF. Esto refleja la naturaleza jerárquica y multi-escala de BP: un mismo proceso puede involucrar múltiples compartimentos celulares y funciones moleculares, generando patrones de expresión más heterogéneos y difíciles de modelar. La ventaja marginal de exp2GO en BP para algunos organismos sugiere que la reconstrucción semántica directa mediante NMF acoplado captura mejor las dependencias complejas de esta subontología, mientras que el modelo propuesto, al depender más de la topología del grafo, puede perder algunas relaciones funcionales sutiles.

Análisis por organismo. El rendimiento también presenta variabilidad sistemática entre organismos. En **YEAST** (*S. cerevisiae*), organismo modelo extensamente estudiado con anotaciones GO de alta calidad y alta densidad, tanto el modelo propuesto como exp2GO alcanzan los mejores resultados absolutos ($F1_{max}$ entre 0.52-0.78 según subontología). La abundancia de anotaciones experimentales y la completitud del grafo de interacciones genéticas favorecen la transferencia de información, permitiendo reconstrucciones semánticas precisas. Los métodos basados en secuencia (DeepGOPlus) exhiben desempeño inferior en YEAST, confirmando que la información estructural y de expresión es crítica en este contexto.

En **ARA** (*A. thaliana*), organismo vegetal con características transcriptómicas distintas (respuesta circadiana, respuesta a estrés), el modelo propuesto y exp2GO mantienen rendimiento competitivo ($F1_{max}$ entre 0.48-0.72), aunque ligeramente inferior a YEAST. Esto refleja tanto la menor densidad de anotaciones experimentales disponibles como la mayor complejidad de los procesos regulatorios en plantas multicelulares. Destaca que el modelo propuesto supera consistentemente a BLAST y NMF-GO en todas las subontologías de ARA, demostrando que la integración de expresión con estructura de grafo compensa la menor disponibilidad de datos de secuencia homóloga.

Para **DICTY** (*D. discoideum*), organismo unicelular con ciclo de desarrollo complejo, los resultados muestran mayor variabilidad entre métodos. El modelo propuesto alcanza los mejores resultados en CC ($F1_{max} \approx 0{,}73$), equiparando a exp2GO. Sin

embargo, en MF y BP, exp2GO presenta ligera ventaja ($F1_{max}$ 0.58 vs 0.55 en MF). Esta diferencia puede atribuirse a que el dataset de DICTY tiene menor tamaño muestral (652-707 genes según subontología) y los patrones de expresión durante el desarrollo capturan dinámicas temporales complejas que no necesariamente se traducen en co-expresión estable. Los métodos basados en factorización directa de matrices semánticas, como exp2GO, pueden ser más robustos en escenarios de datos limitados, mientras que el modelo propuesto, al depender de la construcción del grafo, es más sensible al tamaño muestral.

Consistencia y generalización. Un hallazgo clave es que el modelo propuesto exhibe rendimiento consistentemente alto en las nueve configuraciones evaluadas (3 organismos $\times$ 3 subontologías), sin colapsos de desempeño en ningún escenario. La desviación estándar del $F1_{max}$ del modelo propuesto a través de todas las configuraciones es menor que la de DeepGOPlus y NMF-GO, indicando mayor estabilidad y robustez. Esta consistencia sugiere que arquitecturas de grafos con mecanismos de atención para nodos y aristas pueden adaptarse a diferentes tareas bioinformáticas sin requerir ajustes extensivos para cada problema específico. En contraste, métodos especializados como NMF-GO muestran alto rendimiento en subconjuntos específicos de tareas (e.g., BP en YEAST) pero no logran generalizar a otros escenarios, mientras que métodos de secuencia como DeepGOPlus tienen rendimiento limitado cuando la señal de homología es débil o inexistente.

6.3 Conclusiones del capítulo

Los resultados muestran la versatilidad del modelo propuesto mediante su aplicación exitosa a dos dominios bioinformáticos adicionales: predicción de interacciones proteína-proteína y anotación funcional de genes con términos Gene Ontology. Los resultados confirman que la arquitectura NEAConv, con su mecanismo de atención para nodos y aristas combinado con la función de pérdida híbrida, puede adaptarse efectivamente a diferentes tipos de problemas centrados en aristas sin modificaciones sustanciales.

En el dominio de PPI, el modelo alcanzó un rendimiento superior al estado del arte, superando consistentemente a métodos especializados como SVGAE, Siamese Residual RCNN y EnAmDNN en los cinco organismos evaluados. El test de Friedman confirmó diferencias estadísticamente significativas ($p = 7.8e\text{-}30$), estableciendo al modelo propuesto como el de mejor rendimiento en esta tarea.

Para la predicción de términos GO, el modelo demostró rendimiento equivalente al método especializado exp2GO sin diferencias estadísticamente significativas ($p = 0.237$), mientras que superó significativamente a otros métodos del estado del arte

como DeepGOPlus y NMF-GO (p = 1e-3). Este rendimiento competitivo se mantuvo consistentemente a través de los tres organismos evaluados y las tres subontologías de GO.

Estos resultados sugieren que las arquitecturas de grafos que integran explícitamente información de nodos y aristas, entrenadas con objetivos supervisados y autosupervisados combinados, pueden adaptarse a diferentes tareas bioinformáticas centradas en aristas. La capacidad del modelo para manejar tanto datos de secuencia proteica como perfiles de expresión génica, adaptándose a las características específicas de cada dominio, demuestra su flexibilidad y robustez como solución general para problemas de predicción a nivel de aristas en grafos biológicos.

Capítulo 7

Conclusiones y Trabajo Futuro

Esta tesis abordó el desarrollo de arquitecturas de redes neuronales en grafos especializadas para tareas centradas en aristas, un área que ha recibido comparativamente menos atención que las tareas a nivel de nodo o grafo completo en la literatura de aprendizaje profundo. El trabajo se desarrolló en el contexto de aplicaciones bioinformáticas, donde las relaciones entre entidades biológicas —compuestos químicos en redes metabólicas, interacciones entre proteínas, co-anotaciones funcionales de genes— son tan informativas como las entidades mismas para comprender los sistemas biológicos subyacentes. A continuación se sintetizan las contribuciones principales, las limitaciones identificadas y las direcciones de investigación futura.

7.1 Contribuciones Principales

La primera contribución de esta tesis es la arquitectura NEAConv (*Node-Edge Attention Convolution*), diseñada específicamente para problemas donde el objetivo de predicción se asocia a pares de nodos conectados por aristas. A diferencia de arquitecturas ampliamente utilizadas como GCN, GraphSAGE o GAT, que operan exclusivamente sobre características de nodos ignorando atributos de las conexiones, NEAConv integra explícitamente información de aristas en el mecanismo de agregación mediante una formulación *query-key-value* inspirada en la arquitectura transformer. Esta integración permite que el modelo capture relaciones complejas donde las propiedades de la conexión son determinantes para la predicción: por ejemplo, en redes metabólicas, el tipo de reacción enzimática que vincula dos compuestos químicos proporciona información crucial sobre su similaridad estructural que no puede inferirse únicamente de las características individuales de cada compuesto.
La arquitectura incorpora varios componentes que trabajan en conjunto para abordar tareas centradas en aristas. El mecanismo de atención pondera adaptativamente la relevancia de cada vecino basándose en vectores *query* derivados de características

de nodos y aristas, permitiendo al modelo identificar automáticamente qué conexiones son más informativas para cada predicción específica. Un predictor híbrido combina producto escalar, que captura similaridad coseno directa entre embeddings, con una red feedforward que puede modelar patrones no lineales más complejos. Un tokenizer proyecta características de entrada heterogéneas a un espacio latente común, habilitando la aplicación de la misma arquitectura a diferentes dominios sin modificaciones estructurales. El proceso de desarrollo involucró más de 500 experimentos organizados en cinco fases de optimización secuencial que exploraron sistemáticamente el espacio de diseño arquitectural.

El hallazgo más significativo del proceso de optimización fue el impacto del filtrado topológico del grafo de entrada. Al retener únicamente las k aristas de mayor centralidad (betweenness centrality) por nodo, con $k = 2$ como valor óptimo, el error de predicción se redujo en un 93.5 % respecto a utilizar el grafo completo (MAE de 0,1939 a 0,0125). Este resultado contrasta marcadamente con el efecto relativamente modesto de decisiones arquitecturales convencionales: la dimensionalidad de embeddings y el esquema de atención presentaron tamaños de efecto inferiores ($\eta^2 < 0{,}06$). La interpretación es que las redes metabólicas anotadas en bases de datos como KEGG contienen conexiones de diferente confiabilidad —reacciones hipotéticas, reversas termodinámicamente desfavorecidas, vías alternativas activas solo bajo condiciones extremas— y el filtrado selectivo elimina este ruido, permitiendo al modelo aprender exclusivamente de relaciones bioquímicas de alta confianza. Este resultado sugiere que, al menos en el contexto de redes biológicas con la representación empleada en este trabajo, la curaduría estructural del grafo puede ser más determinante que sofisticaciones arquitecturales, aunque sería necesario validar esta observación en otros dominios para determinar su generalidad.

La segunda contribución es una función de pérdida híbrida que combina aprendizaje supervisado y auto-supervisado mediante la formulación $\mathcal{L} = \alpha\mathcal{L}_{\text{supervised}} + \beta\mathcal{L}_{\text{cosine}} + \gamma\mathcal{L}_{\text{cosine\_pred}}$. El término supervisado $\mathcal{L}_{\text{supervised}}$ minimiza directamente el error de predicción respecto a los valores objetivo conocidos. El término $\mathcal{L}_{\text{cosine}}$ fuerza que la similaridad coseno entre embeddings de nodos se alinee con los valores de similaridad conocidos, organizando el espacio latente de forma consistente con métricas del dominio. El término auto-supervisado $\mathcal{L}_{\text{cosine\_pred}}$ impone consistencia entre las predicciones del modelo y la similaridad coseno de los embeddings aprendidos, previniendo un problema sutil pero importante: sin esta restricción, el predictor (la red feedforward final) puede aprender a "compensar" embeddings de baja calidad mediante funciones no lineales complejas, resultando en buen rendimiento sobre datos de entrenamiento pero pobre generalización.

El estudio de ablación demostró que ambos componentes —atención y pérdida híbrida— son esenciales y presentan efectos complementarios. El modelo baseline

sin atención ni términos auto-supervisados alcanzó MAE = 0.01590 con alta dispersión en el espacio latente. Agregar únicamente atención redujo el error a MAE = 0.01523 (mejora del 4.2 %), mientras que agregar únicamente la pérdida híbrida lo redujo a MAE = 0.01426 (mejora del 10.3 %). El modelo completo con ambos componentes alcanzó MAE = 0.01013, una mejora del 36.3 % respecto al baseline. Cualitativamente, la visualización del espacio latente reveló que con la función de pérdida completa los embeddings se organizan en una banda casi lineal donde la distancia coseno constituye un proxy confiable de la similaridad predicha, mientras que sin los términos auto-supervisados los embeddings forman una nube dispersa sin estructura geométrica coherente. Esta estrategia de alineación geométrica entre representaciones y predicciones representa una contribución metodológica aplicable más allá de la arquitectura específica propuesta, con potencial utilidad en otros dominios donde se desee que el espacio latente preserve relaciones semánticas del problema.

La tercera contribución consiste en la validación empírica de que los principios de diseño desarrollados son aplicables a diferentes tareas bioinformáticas sin modificaciones estructurales sustanciales. En el dominio primario de predicción de similaridad molecular en redes metabólicas, el modelo alcanzó un MAE de 0.01013 (1.01 % de error) en la predicción del coeficiente de Tanimoto sobre un dataset de 6 vías metabólicas con 207 compuestos, de los cuales 33 presentaban estructura molecular parcial o desconocida. La capacidad de generar predicciones para estos compuestos con sustituyentes genéricos "-R" constituye una ventaja distintiva respecto a métodos basados en fingerprints moleculares, que requieren estructura completa para calcular descriptores. Los análisis cualitativos mostraron que las predicciones para pares de compuestos con estructura desconocida son coherentes con el análisis visual de sus estructuras parciales, aunque la calibración absoluta de los valores debe interpretarse con cautela dado que no existe ground truth verificable para estos casos.

En predicción de interacciones proteína-proteína, la arquitectura fue evaluada en cinco organismos con datasets públicos ampliamente utilizados: HPRD (∼36,000 pares), Human (∼72,000 pares), *E. coli* (∼13,000 pares), *Drosophila* (∼44,000 pares) y *C. elegans* (∼8,000 pares). La adaptación requirió únicamente cambiar las características de entrada: codificaciones one-hot de compuestos fueron reemplazadas por descriptores Conjoint Triad de secuencias proteicas para los nodos, y familias enzimáticas fueron reemplazadas por matrices de similaridad de secuencia para las aristas. La arquitectura NEAConv y la función de pérdida permanecieron sin modificaciones. El modelo superó significativamente al estado del arte, incluyendo métodos especializados como SVGAE (autoencoder variacional para grafos), Siamese Residual RCNN (redes convolucionales siamesas sobre secuencias) y EnAmDNN

(ensemble con mecanismos de atención). El test de Friedman confirmó diferencias altamente significativas entre todos los métodos ($p = 7{,}8 \times 10^{-30}$), y el diagrama de diferencias críticas mostró que el modelo propuesto no está conectado estadísticamente a ningún otro método, indicando superioridad consistente. Las puntuaciones F1 alcanzadas fueron: HPRD = 0.977, Human = 0.961, *E. coli* = 0.967, *Drosophila* = 0.991, *C. elegans* = 0.985.

En anotación funcional con Gene Ontology, el modelo fue evaluado en tres organismos (*S. cerevisiae*, *A. thaliana*, *D. discoideum*) y tres subontologías (Biological Process, Molecular Function, Cellular Component), totalizando nueve configuraciones experimentales. Las características de entrada fueron adaptadas a perfiles de expresión génica para nodos y similaridad de expresión para aristas, siguiendo el protocolo experimental establecido por el método exp2GO para garantizar comparabilidad. El modelo igualó el rendimiento del método especializado exp2GO sin diferencias estadísticamente significativas (test de Nemenyi: $p = 0{,}237$), mientras que superó significativamente a otros métodos del estado del arte incluyendo DeepGOPlus y NMF-GO ($p = 1 \times 10^{-3}$). Este resultado es notable considerando que exp2GO fue diseñado específicamente para esta tarea mediante factorización matricial no negativa acoplada, mientras que NEAConv es una arquitectura de propósito general para tareas centradas en aristas.

Es importante enfatizar que esta versatilidad no implica transferencia de aprendizaje entre dominios en el sentido técnico del término: cada aplicación requiere reentrenamiento completo con datos específicos del dominio, y el modelo no transfiere conocimiento sobre qué subestructuras químicas son importantes o qué motivos de secuencia son indicativos de interacción. La contribución radica en que los principios de diseño arquitectural —mecanismo de atención que integra información de nodos y aristas, función de pérdida híbrida que alinea geométricamente el espacio latente, construcción de subgrafos centrados en pares de nodos objetivo— son aplicables a modalidades de entrada heterogéneas manteniendo rendimiento competitivo o superior al estado del arte especializado en cada dominio.

7.2 Limitaciones y Trabajo Futuro

A pesar de los resultados obtenidos, este trabajo presenta limitaciones que abren oportunidades para investigación futura. La evaluación se restringió a tres dominios bioinformáticos (redes metabólicas, interacciones proteína-proteína, anotación funcional de genes), sin validar la generalidad del enfoque en otros campos donde las tareas centradas en aristas son igualmente relevantes. Dominios como redes sociales (predicción de formación de enlaces), sistemas de recomendación (predicción de afinidad usuario-ítem con características contextuales), grafos de citaciones (predicción

de co-autoría), redes de transporte (predicción de flujo) o grafos de conocimiento (predicción de relaciones semánticas faltantes) presentan características estructurales diferentes que podrían revelar limitaciones o fortalezas no observadas en el contexto biológico. La aplicación a benchmarks estandarizados como Open Graph Benchmark para tareas de link prediction (ogbl-ppa, ogbl-collab, ogbl-citation2) permitiría comparación directa con el estado del arte en condiciones controladas y demostraría —o refutaría— la aplicabilidad más allá del contexto biológico estudiado.

La arquitectura actual utiliza una sola capa NEAConv, lo que podría limitar la capacidad para capturar patrones que requieren múltiples niveles de agregación de información. Las GNNs profundas con múltiples capas apiladas pueden capturar dependencias de mayor orden —por ejemplo, subgrafos de 3-4 nodos en lugar de solo vecindades de primer grado— pero enfrentan desafíos técnicos conocidos como oversmoothing, donde las representaciones de todos los nodos convergen hacia valores similares a medida que aumenta la profundidad. La exploración de arquitecturas multi-capa NEAConv representa una dirección promisoria que requerirá abordar estos desafíos mediante técnicas como conexiones residuales, normalización de capas, o estrategias de muestreo de vecindarios para mantener escalabilidad.

Los experimentos asumieron que los grafos de entrada son correctos, sin evaluar robustez ante ruido. En la práctica, las redes biológicas contienen errores de anotación, aristas faltantes (falsos negativos por limitaciones de métodos experimentales) y aristas espurias (falsos positivos por inferencias computacionales incorrectas). Un análisis sistemático de sensibilidad —inyectando ruido sintético mediante adición de aristas aleatorias, eliminación de aristas verdaderas, o perturbación de características— permitiría caracterizar la confiabilidad del modelo en escenarios realistas y guiar el desarrollo de estrategias de robustificación como autoencoders de denoising o inferencia conjunta de estructura y predicciones.

7.3 Publicaciones Derivadas

Los resultados de esta tesis han dado lugar a diversas publicaciones científicas en conferencias y revistas con referato. El estudio comparativo de fingerprints moleculares y la evaluación inicial del modelo neuronal para estimación de similaridad mediante representaciones one-hot, correspondientes al Capítulo 3, fueron publicados en las Jornadas Argentinas de Informática (JAIIO) (Borzone, Ezequiel y col., 2022). El desarrollo de embeddings Node2Vec para predicción de similaridad entre compuestos con estructura desconocida, presentado en el Capítulo 4, fue publicado en la International Conference on Applied Informatics (ICAI) (Borzone, Di Persia & Gerard, 2022). Versiones preliminares de estos trabajos fueron presentadas

como pósteres en el Congreso Argentino de Bioinformática y Biología Computacional (CAB2C) organizado por la Asociación Argentina de Bioinformática y Biología Computacional (A2B2C) en sus ediciones 2022 y 2023. Finalmente, la arquitectura NEAConv completa y su validación en múltiples dominios bioinformáticos, desarrolladas en los Capítulos 5 y 6, fueron publicadas en IEEE Transactions on Signal and Information Processing over Networks (Borzone y col., 2025).

Bibliografía

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, *215*, 403-410. https://doi.org/10.1016/S0022-2836(05)80360-2

Arita, M. (2012). From metabolic reactions to networks and pathways. *Methods Mol. Biol.*, *804*, 93-106.

Bajusz, D., Rácz, A., & Héberger, K. (2015). Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of Cheminformatics*, *7*, 20.

Barabási, A.-L., & Oltvai, Z. N. (2004). Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, *5*(2), 101-113. https://doi.org/10.1038/nrg1272

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, *15*(6), 1373-1396.

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for Hyper-Parameter Optimization. *Advances in Neural Information Processing Systems*, *24*.

Bergstra, J., Yamins, D., & Cox, D. (2013). Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. En S. Dasgupta & D. McAllester (Eds.), *Proceedings of the 30th International Conference on Machine Learning* (pp. 115-123, Vol. 28). PMLR.

Blake, J. A., Dolan, M., Drabkin, H., Hill, D., Li, N., Sitnikov, D., & et al. (2013). Gene Ontology annotations and resources. *Nucleic Acids Research*, *41*(D1), D530-D535. https://doi.org/10.1093/nar/gks1050

Borzone, E., Di Persia, L. E., & Gerard, M. (2022). Neural Model-Based Similarity Prediction for Compounds with Unknown Structures. En H. Florez & H. Gomez (Eds.), *Applied Informatics* (pp. 75-87). Springer International Publishing. https://doi.org/10.1007/978-3-031-19647-8_6

Borzone, E., Ezequiel, L., Persia, D., & Gerard, M. (2022). Evaluación de un modelo neuronal para la estimación de similaridad entre compuestos a partir de representaciones one-hot.

Borzone, E., Persia, L. D., & Gerard, M. (2025). A Hybrid Supervised and Self-Supervised Graph Neural Network for Edge-Centric Applications. *IEEE Transactions on Signal and Information Processing over Networks*. https://doi.org/10.1109/TSIPN.2025.3611172

Brody, S., Alon, U., & Yahav, E. (2021). How Attentive are Graph Attention Networks? *ICLR*.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2016). Geometric deep learning: going beyond Euclidean data. https://doi.org/10.1109/MSP.2017.2693418

Brown, R. D., & Martin, Y. C. (1996). *Use of Structure-Activity Data To Compare Structure-Based Clustering Methods and Descriptors for Use in Compound Selection*.

Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Chen, M., Ju, C. J., Zhou, G., Chen, X., Zhang, T., Chang, K.-W., Zaniolo, C., & Wang, W. (2019). Multifaceted protein-protein interaction prediction based on Siamese residual RCNN. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *17*(4), 1312-1322. https://doi.org/10.1109/TCBB.2019.2914393

Chung, F. R. (1997). *Spectral Graph Theory* (Vol. 92). American Mathematical Society.

Ciortan, M., & Defrance, M. (2021). GNN-based embedding for clustering scRNA-seq data (V. Boeva, Ed.). *Bioinformatics*, *38*(4), 1037-1044. https://doi.org/10.1093/bioinformatics/btab787

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences* (2nd). Lawrence Erlbaum Associates.

Covington, P., Adams, J., & Sargin, E. (2016). Deep Neural Networks for YouTube Recommendations. *Proceedings of the 10th ACM Conference on Recommender Systems*, 191-198. https://doi.org/10.1145/2959100.2959190

De Maesschalck, R., Jouan-Rimbaud, D., & Massart, D. L. (2000). The Mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, *50*(1), 1-18. https://doi.org/10.1016/S0169-7439(99)00047-7

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 3844-3852.

Deza, M. M., & Deza, E. (2009). *Encyclopedia of Distances*. Springer. https://doi.org/10.1007/978-3-642-00234-2

Di Persia, L., Lopez, T., Arce, A., Milone, D. H., & Stegmayer, G. (2022). exp2GO: improving prediction of functions in the Gene Ontology with expression da-

ta. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1-10.

Dunham, A. S., & Beltrao, P. (2022). Benchmarking computational methods for protein-protein interaction prediction. *Nature Communications*, *13*(1), 2143. https://doi.org/10.1038/s41467-022-28797-3

Durant, J. L., Leland, B. A., Henry, D. R., & Nourse, J. G. (2002a). Reoptimization of MDL Keys for Use in Drug Discovery. *Journal of Chemical Information and Computer Sciences*, *42*(6), 1273-1280. https://doi.org/10.1021/ci010132r

Durant, J. L., Leland, B. A., Henry, D. R., & Nourse, J. G. (2002b). Reoptimization of MDL keys for use in drug discovery. *Journal of chemical information and computer sciences*, *42*, 1273-1280. https://doi.org/10.1021/CI010132R

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., & Adams, R. P. (2015). Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 2224-2232.

Eisen, M. B., Spellman, P. T., Brown, P. O., & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the United States of America*, *95*, 14863-14868. https://doi.org/10.1073/PNAS.95.25.14863/SUPPL_FILE/3917KEY.HTML

El Amrani, N., Ait-Lahsen, M., & Oughdir, L. (2020). Recommendation system using the k-nearest neighbors and singular value decomposition algorithms. *International Journal of Electrical and Computer Engineering (IJECE)*, *10*(6), 6467-6475.

Formica, A., & Taglino, F. (2021). An Enriched Information-Theoretic Definition of Semantic Similarity in a Taxonomy. *IEEE Access*, *9*, 100583-100593. https://doi.org/10.1109/ACCESS.2021.3097368

Fout, A., Byrd, J., Shariat, B., & Ben-Hur, A. (2017). Protein Interface Prediction using Graph Convolutional Networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 6530-6539.

Gahman, N., & Elangovan, V. (2023). A Comparison of Document Similarity Algorithms. *International Journal of Artificial Intelligence and Applications (IJAIA)*, *14*(2), 41-54. https://doi.org/10.5121/ijaia.2023.14204

Gillis, J., Ballouz, S., & Pavlidis, P. (2014). Bias tradeoffs in the creation and analysis of protein-protein interaction networks. *Journal of Proteomics*, *100*, 44-54. https://doi.org/10.1016/j.jprot.2014.01.020

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. *Proceedings of the 34th International Conference on Machine Learning*, 1263-1272.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2020). Message Passing Neural Networks. *Lecture Notes in Physics*, *968*, 199-214. https://doi.org/10.1007/978-3-030-40245-7_10

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, *151*, 78-94.

Grover, A., & Leskovec, J. (2016). node2vec: Scalable Feature Learning for Networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855-864. https://doi.org/10.1145/2939672.2939754

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Representation Learning on Graphs: Methods and Applications.

Hutter, F., Hoos, H., & Leyton-Brown, K. (2014). An Efficient Approach for Assessing Hyperparameter Importance. En E. P. Xing & T. Jebara (Eds.), *Proceedings of the 31st International Conference on Machine Learning* (pp. 754-762, Vol. 32). PMLR.

Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., & Barabási, A.-L. (2000). The large-scale organization of metabolic networks. *Nature*, *407*(6804), 651-654. https://doi.org/10.1038/35036627

Jiang, D., Wu, Z., Hsieh, C.-Y., Chen, G., Liao, B., Wang, Z., Shen, C., Cao, D., Wu, J., & Hou, T. (2021). Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models. *Journal of Cheminformatics*, *13*(1). https://doi.org/10.1186/s13321-020-00479-8

Kearnes, S., McCloskey, K., Berndl, M., Pande, V., & Riley, P. (2016). Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, *30*, 595-608.

Kim, S., Thiessen, P. A., Bolton, E. E., Chen, J., Fu, G., Gindulyte, A., Han, L., He, J., He, S., Shoemaker, B. A., Wang, J., Yu, B., Zhang, J., & Bryant, S. H. (2016). PubChem Substance and Compound databases. *Nucleic Acids Research*, *44*(D1), D1202-D1213. https://doi.org/10.1093/nar/gkv951

Kingma, D. P., & Ba, J. L. (2014). Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. https://arxiv.org/pdf/1412.6980

Kipf, T. N., & Welling, M. (2016a). Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N., & Welling, M. (2016b). Variational Graph Auto-Encoders. *Bayesian Deep Learning Workshop (NeurIPS)*.

Kreppel, L., Fey, P., Gaudet, P., Just, E., Kibbe, W. A., Chisholm, R. L., & Kimmel, A. R. (2004). dictyBase: a new Dictyostelium discoideum genome database. *Nucleic acids research*, *32*. https://doi.org/10.1093/NAR/GKH138

Kulmanov, M., Khan, M. A., & Hoehndorf, R. (2019). DeepGOPlus: improved protein function prediction from sequence. *Bioinformatics*, *35*(11), 1188-1195. https://doi.org/10.1093/bioinformatics/btz595

Lee, G., Kim, J., Choi, M.-s., Jang, R.-Y., & Lee, R. (2023). Review of Code Similarity and Plagiarism Detection Research Studies. *Applied Sciences*, *13*(20), 11358. https://doi.org/10.3390/app132011358

Li, Y., Qiao, G., Wang, K., & Wang, G. (2020). Improved protein-protein interaction prediction using ensemble deep neural networks with attention mechanism. *Bioinformatics*, *37*(1), 110-117. https://doi.org/10.1093/bioinformatics/btaa914

Lin, D. (1998). An Information-Theoretic Definition of Similarity. *Proceedings of the 15th International Conference on Machine Learning (ICML)*, 296-304.

Liu, Y., Safavi, T., Dighe, A., & Koutra, D. (2018). Graph Summarization Methods and Applications: A Survey. *ACM Computing Surveys*, *51*(3), 62. https://doi.org/10.1145/3186727

Ma, H., & Zeng, A.-P. (2003). Reconstruction of metabolic networks from genome data and analysis of their global structure for various organisms. *Bioinformatics*, *19*(2), 270-277. https://doi.org/10.1093/bioinformatics/19.2.270

McShan, D. C., Rao, S., & Shah, I. (2003). PathMiner: predicting metabolic pathways by heuristic search. *Bioinformatics*, *19*(13), 1692-1698.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space [arXiv preprint].

Muegge, I., & Mukherjee, P. (2016). An overview of molecular fingerprint similarity search in virtual screening. *Expert Opinion on Drug Discovery*, *11*, 137-148. https://doi.org/10.1517/17460441.2016.1117070

Newman, M. E. (2003). The structure and function of complex networks. *SIAM Review*, *45*, 167-256. https://doi.org/10.1137/s003614450342480

Noor, E., Flamholz, A., Bar-Even, A., Davidi, D., & Milo, R. (2010). Central carbon metabolism as a minimal biochemical walk between precursors for biomass and energy. *Molecular Cell*, *39*(5), 809-820. https://doi.org/10.1016/j.molcel.2010.12.012

Palsson, B. Ø. (2006). *Systems biology: properties of reconstructed networks*. Cambridge University Press.

Paredes, R., Chávez, E., Figueroa, K., & Navarro, G. (2006). Practical Construction of k-Nearest Neighbor Graphs in Metric Spaces. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lec-*

ture Notes in Bioinformatics), *4007 LNCS*, 85-97. https://doi.org/10.1007/11764298_8

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). DeepWalk: Online learning of social representations. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 701-710.

Pesquita, C., Faria, D., Falcão, A. O., Lord, P., & Couto, F. M. (2009). Semantic similarity in biomedical ontologies. *PLoS Computational Biology*, *5*(7), e1000443. https://doi.org/10.1371/journal.pcbi.1000443

Probst, D., & Reymond, J. L. (2018). A probabilistic molecular fingerprint for big data settings. *Journal of Cheminformatics*, *10*. https://doi.org/10.1186/s13321-018-0321-8

Radivojac, P., Clark, W. T., Oron, T. R., Schnoes, A. M., & Wittkop, T. (2013). A large-scale evaluation of computational protein function prediction. *Nature Methods 2013 10:3*, *10*, 221-227. https://doi.org/10.1038/nmeth.2340

Rahman, S. A., Advani, P., Schunk, R., Schrader, R., & Schomburg, D. (2005a). Metabolic pathway analysis web service (Pathway Hunter Tool at CUBIC). *Bioinformatics*, *21*(7), 1189-1193.

Rahman, S. A., Advani, P., Schunk, R., Schrader, R., & Schomburg, D. (2005b). Metabolic pathway analysis web service (Pathway Hunter Tool at CUBIC). *Bioinformatics*, *21*(7), 1189-1193.

Ramli, R., & Ahmad, S. Z. (2014). Comparison of Distance and Dissimilarity Measures for Clustering Data with Mix Attribute Types. *2014 The 1st International Conference on Information Technology, Computer, and Electrical Engineering*, 435-439.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., & Beaini, D. (2022). Recipe for a General, Powerful, Scalable Graph Transformer. *Advances in Neural Information Processing Systems*, *35*, 14501-14515.

Resnik, P. (1995). Using Information Content to Evaluate Semantic Similarity in a Taxonomy. *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, 448-453.

Rogers, D., & Hahn, M. (2010). Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, *50*(5), 742-754.

Rossi, R. A., Zhou, J., Cheng, W., Hu, J., & Lyu, Y. (2020). Temporal Graph Networks for Deep Learning on Dynamic Graphs. *arXiv preprint arXiv:2006.10637*.

Saini, M., Aggarwal, M., & Singh, A. (2023). Comparison of user-based and item-based collaborative filtering using similarity metrics. *AIP Conference Proceedings*, *2605*(1), 020019. https://doi.org/10.1063/5.0217219

Schlicker, A., Domingues, F. S., Rahnenführer, J., & Lengauer, T. (2006). A new measure for functional similarity of gene products based on Gene Ontology.

BMC Bioinformatics 2006 7:1, *7*, 302-. https://doi.org/10.1186/1471-2105-7-302

Shen, J., Zhang, J., Luo, X., Zhu, W., Yu, K., Chen, K., Li, Y., & Jiang, H. (2007). Predicting protein-protein interactions based only on sequences information. *Proceedings of the National Academy of Sciences of the United States of America*, *104*, 4337-4341. https://doi.org/10.1073/PNAS.0607879104

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., & Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, *30*, 83-98. https://doi.org/10.1109/MSP.2012.2235192

Sievers, F., Wilm, A., Dineen, D., Gibson, T. J., Karplus, K., Li, W., Lopez, R., McWilliam, H., Remmert, M., Söding, J., Thompson, J. D., & Higgins, D. G. (2011). Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Molecular Systems Biology*, *7*, 539. https://doi.org/10.1038/MSB.2011.75

Steck, H., Baltrunas, L., Elahi, E., Liang, D., Raimond, Y., & Basilico, J. (2021). Deep Learning for Recommender Systems: A Netflix Case Study [Number: 3]. *AI Magazine*, *42*(3), 7-18. https://doi.org/10.1609/aimag.v42i3.18140

Steuer, R., Kurths, J., Fiehn, O., Weckwerth, W., & Selbig, J. (2006). Reconstruction and validation of metabolic pathways by correlation of metabolomic and enzyme expression data. *Bioinformatics*, *22*(15), e121-e129.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). LINE: Large-scale Information Network Embedding. *Proceedings of the 24th International Conference on World Wide Web*, *14*, 1067-1077. https://doi.org/10.1145/2736277.2741093

Thakoor, S., Tallec, C., Azar, M. G., Azabou, M., Dyer, E. L., Munos, R., Veličković, P., & Valko, M. (2022). Large-Scale Representation Learning on Graphs via Bootstrapping. *International Conference on Learning Representations*.

Thieffry, D. (2007). Dynamical roles of biological regulatory circuits. *Briefings in Bioinformatics*, *8*, 220-225.

Torres, G. J., Basnet, R. B., Sung, A. H., Mukkamala, S., & Ribeiro, B. (2009). A Similarity Measure for Clustering and its Applications. *Int. J. of Electrical, Computer, and Systems Engineering*, *3*(3).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, *2017-December*, 5999-6009.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph Attention Networks. *arXiv preprint arXiv:1710.10903*.

Vijaymeena, M. K., & Kavitha, K. (2016). A Survey on Similarity Measures in Text Mining. *Machine Learning and Applications: An International Journal (MLAIJ)*, *3*(1), 19-28. https://doi.org/10.5121/mlaij.2016.3103

Wager, S., Wang, S., & Liang, P. S. (2013). Dropout training as adaptive regularization. En C. Burges, L. Bottou, M. Welling, Z. Ghahramani & K. Weinberger (Eds.), *Advances in Neural Information Processing Systems* (Vol. 26). Curran Associates, Inc.

Wagner, A., & Fell, D. A. (2001). The small world inside large metabolic networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, *268*(1478), 1803-1810. https://doi.org/10.1098/rspb.2001.1711

Wang, B., Mezlini, A. M., Demir, F., Fiume, M., Tu, Z., Brudno, M., Haibe-Kains, B., & Goldenberg, A. (2014). Similarity network fusion for aggregating data types on a genomic scale [Citado como referencia clásica de fusión de datos; ver (Di Persia, Lopez, Arce, Milone & Stegmayer, 2022) para DFMF en comparación.]. *Nature Methods*, *11*(3), 333-337. https://doi.org/10.1038/nmeth.2810

Wang, J., Ma, A., Chang, Y., Gong, J., Jiang, Y., Qi, R., Wang, C., Fu, H., Ma, Q., & Xu, D. (2021). scGNN is a novel graph neural network framework for single-cell RNA-Seq analyses. *Nature Communications*, *12*(1). https://doi.org/10.1038/s41467-021-22197-x

Wang, X., & Chen, G. (2003). Complex Networks: Small-World, Scale-Free and Beyond. *IEEE CIRC SYST MAG*, *3*, 6-20.

Wang, Y., Wang, K., Gao, F., & Wang, L. (2020). Learning semantic program embeddings with graph interval neural network. *Proceedings of the ACM on Programming Languages*, *4*(OOPSLA), 1-27. https://doi.org/10.1145/3428205

Weininger, D. (1988). SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, *28*(1), 31-36. https://doi.org/10.1021/ci00057a005

Willett, P., Barnard, J. M., & Downs, G. M. (1998). Chemical similarity searching. *Journal of Chemical Information and Computer Sciences*, *38*(6), 983-996.

Wu, L., Cui, P., Pei, J., & Zhao, L. (Eds.). (2022). *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A Comprehensive Survey on Graph Neural Networks.

Xie, J., Girshick, R., & Farhadi, A. (2016). Unsupervised Deep Embedding for Clustering Analysis. En M. F. Balcan & K. Q. Weinberger (Eds.), *Proceedings of The 33rd International Conference on Machine Learning* (pp. 478-487, Vol. 48). PMLR.

Yang, F., Fan, K., Song, D., & Lin, H. (2020). Graph-based prediction of Protein-protein interactions with attributed signed graph embedding. *BMC Bioinformatics*, *21*(1), 1-16.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., & Liu, T.-Y. (2021). Do Transformers Really Perform Badly for Graph Representation? *Advances in Neural Information Processing Systems*, *34*, 28877-28888.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., & Shen, Y. (2020). Graph Contrastive Learning with Augmentations. *Advances in Neural Information Processing Systems*, *33*, 5812-5823.

Yu, G., Wang, K., Fu, G., Guo, M., & Wang, J. (2020). NMFGO: Gene Function Prediction via Nonnegative Matrix Factorization with Gene Ontology. *IEEE/ACM transactions on computational biology and bioinformatics*, *17*, 238-249. https://doi.org/10.1109/TCBB.2018.2861379

Zhang, Z., Cui, P., & Zhu, W. (2020). Deep Learning on Graphs: A Survey.

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

Zhu, X., & Rabbat, M. (2012). Approximating signals supported on graphs. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 3921-3924. https://doi.org/10.1109/ICASSP.2012.6288775

Zitnik, M., Agrawal, M., & Leskovec, J. (2017). Predicting Multicellular Function through Multi-layer Tissue Networks. *Bioinformatics*, *34*(13), i190-i198.

Zitnik, M., & Zupan, B. (2013). Data fusion by matrix factorization. *IEEE International Conference on Data Mining Workshops (ICDMW)*, 577-584. https://doi.org/10.1109/ICDMW.2013.15