La investigación reportada en esta tesis es parte de los programas de investigación del CICESE (Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California).

La investigación fue financiada por el CONAHCYT (Consejo Nacional de Humanidades, Ciencias y Tecnologías).

Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California

Maestría en Ciencias
en Ciencias de la Computación

Algoritmos basados en descriptores a nivel de aminoácidos y estructuras terciarias predichas para predecir péptidos antimicrobianos mediante aprendizaje de grafos

Tesis
para cubrir parcialmente los requisitos necesarios para obtener el grado de Maestro en Ciencias

Presenta:
Greneter Cordoves Delgado

Ensenada, Baja California, México
2024

Tesis defendida por

Greneter Cordoves Delgado

y aprobada por el siguiente Comité

Dr. César Raúl García Jacas
Director de tesis

Dr. Carlos Alberto Brizuela Rodríguez

Dr. Edgar Leonel Chávez González

Dr. Ansel Yoan Rodríguez González

Dr. Pedro Gilberto López Mariscal
Coordinador del Posgrado en Ciencias de la Computación

Dra. Ana Denise Re Araujo
Directora de Estudios de Posgrado

Resumen de la tesis que presenta **Greneter Cordoves Delgado** como requisito parcial para la obtención del grado de Maestro en Ciencias en Ciencias de la Computación.

Algoritmos basados en descriptores a nivel de aminoácidos y estructuras terciarias predichas para predecir péptidos antimicrobianos mediante aprendizaje de grafos

Resumen aprobado por:

Dr. César Raúl García Jacas
Director de tesis

La resistencia a los antimicrobianos constituye una grave amenaza para la salud humana. El descubrimiento de fármacos basados en péptidos antimicrobianos (AMP) es uno de los enfoques estudiados para abordarla. Los modelos basados en aprendizaje superficial y profundo (DL) se han construido principalmente a partir de secuencias de aminoácidos para predecir AMPs. Los avances recientes en la predicción de estructuras terciarias (3D) han abierto nuevas oportunidades en este campo. En este sentido, se han propuesto recientemente modelos basados en grafos derivados de estructuras de péptidos predichas. Sin embargo, estos modelos no se corresponden con los enfoques más avanzados para codificar información evolutiva y son costosos en términos de memoria y tiempo debido a su dependencia de alineamientos múltiples de secuencias (MSA). En este trabajo, se presentó el *framework* esm-AxP-GDL para crear modelos independientes de MSA basados en grafos generados a partir de estructuras de péptidos predichas por ESMFold, cuyos nodos se caracterizan con información evolutiva a nivel de aminoácidos derivada de los modelos Evolutionary Scale Modeling (ESM-2). Se implementó una red de atención de grafos (GAT) para evaluar la utilidad del *framework* en la clasificación de AMPs. Para ello, se utilizó un conjunto compuesto por 67,058 péptidos. Se demostró que la metodología propuesta permitió construir modelos GAT con capacidades de generalización consistentemente superiores a 20 modelos de vanguardia basados en DL y no basados en DL. Los mejores modelos GAT se desarrollaron utilizando información evolutiva derivada de los modelos ESM-2 de 36 y 33 capas. Estudios de similitud mostraron que los mejores modelos GAT construidos codificaban espacios químicos diferentes, por lo que se fusionaron para mejorar significativamente la clasificación. Además, se demostró que los modelos son altamente dependientes de las estructuras predichas por ESMFold y de la información evolutiva derivada de los modelos ESM-2. En general, los resultados sugieren que esm-AxP-GDL es una herramienta prometedora para desarrollar modelos efectivos, dependientes de la estructura y libres de MSA, que pueden aplicarse con éxito en el cribado de grandes conjuntos de datos. Este *framework* debería ser útil no solo para clasificar AMPs, sino también para modelar otras actividades de péptidos y proteínas.

Palabras clave: Evolutionary Scale Modeling, ESM-2, QSAR, péptidos antimicrobianos, aprendizaje profundo geométrico, aprendizaje profundo de grafos, red de atención de grafos, ESMFold

Abstract of the thesis presented **by Greneter Cordoves Delgado** as a partial requirement to obtain the Master of Science degree in Computer Science.

Algorithms based on amino acid-level descriptors and predicted tertiary structures to predict antimicrobial peptides through deep graph learning

Abstract approved by:

Dr. César Raúl García Jacas
Thesis Director

Antimicrobial resistance constitutes a grave threat for human health. Antimicrobial peptide (AMP)-based drug discovery is one of the currently studied approaches to address it. Traditional learning- and deep learning (DL)-based models have been mainly developed from amino acid sequences to predict AMPs. Recent developments in tertiary structure prediction have opened new avenues in this field. So, models based on graphs derived from predicted peptide structures have been recently created. But they are not in accordance with the state-of-the-art (SOTA) methods to codify evolutionary information and, in addition, they are memory- and time-consuming because rely on multiple sequence alignment. Herein, it is presented a framework (termed esm-AxP-GDL) to build alignment-independent models based on graphs created from ESMFold-predicted structures and whose nodes are characterized with evolutionary information derived from the ESM-2 models. For each predicted structure, the geometrical distance between the α carbon atoms of every pair of amino acids is calculated using the Euclidean metric. If that distance is less than or equal to a given threshold, then an edge is established between those amino acids. Thereby, a graph representation per predicted structure is created, where the nodes represent the amino acids, and the edges represent the structural information. A graph attention network (GAT) was implemented in the framework. To assess this framework, several computational studies to classify AMPs were run on a dataset comprised of 67,058 peptides (22,461 AMPs, 44,597 non-AMPs). The implemented methodology allowed to develop models with generalization abilities consistently better than 20 SOTA non-DL based and DL based models. The best models were built using evolutionary information derived from the ESM-2_t36 and ESM-2_t33 models. Moreover, similarity studies showed that the built models codify different chemical spaces, and thus they can be combined to significantly improve the classification. Additionally, the use of distance functions was implemented to build the graph edges. So, it can be studied if topologically different graphs could be created by using distances other than the Euclidean. This framework should be modified for modeling protein activities.

Keywords: Evolutionary Scale Modeling, ESM-2, QSAR, antimicrobial peptides, geometric deep learning, graph deep learning, graph attention network, ESMFold.

Dedicatoria

A mi familia.

A todas las personas que persiguen sus sueños
sin importar los desafíos.

Agradecimientos

Al Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California (CICESE) y al Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCyT) por financiar esta investigación.

A mi director de tesis, el Dr. César Raúl García Jacas, por brindarme la oportunidad de trabajar en este proyecto de investigación e introducirme en el campo de la Bioinformática. Su excelencia como investigador, su guía, sus exigencias, sus enseñanzas y su motivación contribuyeron de manera invaluable a mi formación académica y profesional.

A mi comité de tesis por sus observaciones y recomendaciones que contribuyeron a mejorar mi investigación.

Tabla de contenido

Página

Lista de figuras

Figura Página

Lista de tablas

Tabla Página

Capítulo 1. Introducción

El creciente aumento de agentes patógenos resistentes a fármacos antimicrobianos representa una grave amenaza para la salud humana (Larkin, 2023; T. Li et al., 2023; WHO, 2023). Este fenómeno, conocido como resistencia antimicrobiana (AMR, por sus siglas en inglés), ocurre cuando los patógenos dejan de responder a los medicamentos creados para combatirlos, haciendo que las infecciones sean más difíciles o imposibles de tratar. Esto aumenta el riesgo de propagación de enfermedades graves, discapacidades y muertes (WHO, 2023). En el 2019, la AMR causó al menos 1,27 millones de muertes en todo el mundo y estuvo relacionada con aproximadamente 5 millones de fallecimientos (Centers for Disease Control and Prevention, 2024). La AMR es un proceso natural. Sin embargo, su desarrollo se ha visto principalmente acelerado por el uso inadecuado y excesivo de medicamentos para tratar, prevenir, y controlar infecciones en humanos, animales, y plantas (WHO, 2023). Según datos recopilados de 87 países en el 2020 y de 27 países seguidos desde el 2017, la AMR está causando infecciones del torrente sanguíneo potencialmente mortales (Larkin, 2023). Por ejemplo, en comparación con el 2017, se observó en el 2020 un aumento de al menos un 15% en infecciones del torrente sanguíneo provocadas por especies resistentes de Escherichia coli y Salmonella, así como por cepas de gonorrea (Larkin, 2023). Además, los niveles de resistencia aumentaron por encima del 50% en bacterias que suelen desencadenar infecciones del torrente sanguíneo como Klebsiella pneumoniae y Acinetobacter baumannii (Larkin, 2023).

Actualmente, se estudian alternativas a los fármacos convencionales para hacer frente a la AMR, siendo una de ellas el descubrimiento de fármacos basados en péptidos antimicrobianos (AMPs, por sus siglas en inglés) (Hancock et al., 2016; Usmani et al., 2017; Ahmed et al., 2019). Los AMPs son compuestos químicos formados por la unión de dos o más aminoácidos (hasta 100) que están presentes en todas las formas de vida (Zasloff y Neundorf, 2019). El creciente interés en los AMPs se debe a que son un componente evolutivamente conservado del sistema inmune innato (Radek y Gallo, 2007; Lei et al., 2019; Cai et al., 2020) con capacidades inhibitorias contra un amplio espectro de microorganismos como bacterias (Waghu et al., 2018; Liu et al., 2019), hongos (Devi y Sashidhar, 2019; Fernández de Ullivarri et al., 2020), parásitos (Mor, 2009; Pretzel et al., 2013; Lacerda et al., 2016), y virus (Brice y Diamond, 2019; Vilas Boas et al., 2019). Los AMPs no solo representan una posible alternativa para el tratamiento de infecciones producidas por patógenos, sino también para el tratamiento de enfermedades no transmisibles tales como la diabetes y el cáncer (Chung et al., 2020; Huan et al., 2020; H. Wang et al., 2021; Rodríguez et al., 2021; H. Lee et al., 2023).

La identificación de AMPs mediante experimentos de laboratorio (wet lab) es un proceso largo y costoso. Por lo tanto, existe el interés científico en desarrollar y usar enfoques computacionales para seleccionar potenciales AMPs y acelerar así el descubrimiento de fármacos basados en dichos péptidos. Uno de los métodos computacionales más utilizado para este fin es la modelación QSAR (del inglés - Quantitative Structure Activity Relationships). La modelación QSAR busca establecer relaciones entre las propiedades fisicoquímicas, topológicas y/o geométricas de los compuestos químicos con su actividad biológica mediante el uso de técnicas estadísticas multivariadas o de técnicas de aprendizaje de automático (*machine learning*) (Muratov et al., 2020). Específicamente, los modelos de aprendizaje automático (ML, por sus siglas en inglés) se destacan como herramientas eficientes para examinar grandes conjuntos de datos e identificar potenciales AMPs (Santos-Júnior et al., 2024).

1.1 Antecedentes

En la modelación QSAR para la predicción de AMPs, se han principalmente construido modelos de ML (Agüero-Chapin, Antunes, y Marrero-Ponce, 2023; Aguilera-Puga et al., 2024) basados en secuencias utilizando tanto clasificadores tradicionales (Meher et al., 2017; Bhadra et al., 2018; Pinacho-Castellanos et al., 2021; Ruiz-Blanco et al., 2022) como de aprendizaje profundo (DL, por sus siglas en inglés) (Meher et al., 2017; Su et al., 2019; Fu et al., 2020; J. Li et al., 2020; J. Yan et al., 2020; Sharma et al., 2021a, 2021b; Y. Zhang et al., 2021; Sharma, Shrivastava, Singh, Kumar, Saxena, et al., 2022; Sharma, Shrivastava, Singh, Kumar, Singh, et al., 2022; Du et al., 2023), así como aplicando métodos de redes de similitud (Ayala-Ruano et al., 2022; Agüero-Chapin et al., 2023). El desarrollo casi exclusivo de estos modelos basados en secuencia se debió principalmente a la baja exactitud de los enfoques existentes para predecir la estructura terciaria (3D) de los péptidos.

Sin embargo, el reciente éxito de métodos computacionales como AlphaFold2 (Jumper et al., 2021), Alphafold3 (Abramson et al., 2024), RoseTTAFold (Baek et al., 2021), ESMFold (Z. Lin et al., 2023), ESM-3 (Hayes et al., 2024), y HelixFold-Single (Fang et al., 2023), en la predicción de estructuras 3D de proteínas a partir de sus secuencias de aminoácidos está generando nuevas oportunidades para construir mejores modelos predictivos basados en información estructural. La estructura 3D proporciona información que no se encuentra en la secuencia de aminoácidos, tales como la topología espacial, las interacciones entre residuos, la accesibilidad al solvente, la dinámica de los residuos, entre otros (Durairaj et al., 2023). Ante este escenario, se pueden crear modelos basados en aprendizaje no profundo utilizando descriptores geométricos para péptidos y proteínas (Terán et al., 2019; Marrero-Ponce et al., 2020; Emonts y Buyel,

2023), así como desarrollar modelos basados en redes neuronales que exploten la información de grafos (GNN, por sus siglas en inglés) (Khemani et al., 2024) representando las estructuras peptídicas predichas (K. Yan et al., 2023; Cordoves-Delgado y García-Jacas, 2024).

Con respecto al uso de GNNs, Yan et al (K. Yan et al., 2023) recientemente introdujeron un protocolo denominado sAMPpred-GAT, el cual está basado en una Red de Atención de Grafos (GAT, por sus siglas en inglés) para clasificar AMPs generales, es decir, péptidos relacionados con diferentes actividades antimicrobianas. Hasta la presente tesis, sAMPpred-GAT era el único marco de trabajo (*framework*) reportado en la literatura en usar las estructuras de péptidos predichas para construir representaciones de grafos mediante la aplicación de un umbral de distancia geométrica para definir las relaciones estructurales (aristas) entre cada par de aminoácidos (nodos) de un péptido específico.

1.2 Planteamiento del problema

Sin embargo, sAMPpred-GAT presenta varias desventajas que limitan su uso prospectivo en el cribado de grandes conjuntos de datos y que no permiten obtener mejores modelos predictivos. En primer lugar, los descriptores utilizados para caracterizar los nodos de los grafos no se corresponden con los avances más recientes para transferir información evolutiva. En segundo lugar, tanto el método de predicción de estructuras 3D como los descriptores para caracterizar los nodos dependen de un alineamiento múltiple de secuencias (MSA, por sus siglas en inglés), el cual es bien conocido constituye un problema NP-difícil (Paruchuri et al., 2022). En tercer lugar, solo se aplica la distancia Euclidiana para calcular las distancias inter-aminoácidos y definir las aristas de los grafos. Y, por último, se utiliza una única representación de grafo por estructura 3D de péptido. Esta última desventaja está fuera del alcance de la presente tesis, y, por lo tanto, no se discutirá como parte del problema científico.

Respecto al uso de información evolutiva para caracterizar los nodos, sAMPpred-GAT utiliza información evolutiva a nivel de aminoácido obtenida de una matriz de puntuación específica de posición (PSSM, por sus siglas en inglés) (Mohammadi et al., 2022). Esta matriz se obtiene de un MSA. El uso de PSSMs podría considerarse obsoleto para transferencia de información evolutiva en comparación con la familia de modelos ESM-2 (del inglés - Evolutionary Scale Modeling-2) (Z. Lin et al., 2023). Los modelos ESM-2 son modelos masivos entrenados sobre millones de secuencias de proteínas. La salida de un modelo ESM-2 es una incrustación (*i.e.*, una representación numérica) que caracteriza a cada aminoácido en relación con el espacio compuesto por miles de millones de aminoácidos pertenecientes a proteínas evolutivamente

diversas. Tanto las incrustaciones de los modelos ESM-2 como los de su predecesor ESM-1b (Rives et al., 2021) se han aplicado con éxito en la predicción del efecto de mutaciones (Rives et al., 2021), la estructura secundaria (Rives et al., 2021), y la estructura terciaria (Z. Lin et al., 2023) de proteínas, así como en predecir péptidos antihipertensivos (Du, Ding, et al., 2024), y proteínas y péptidos alergénicos (García-Jacas, García-González, et al., 2022; Du, Xu, et al., 2024). Además, han mejorado características del estado del arte, como las características derivadas de PSSMs, para la predicción de mapas de contactos (Rives et al., 2021). También se han aplicado en la clasificación de AMPs generales y sus tipos funcionales antibacterianos, antifúngicos, antiparasitarios y antivirales (García-Jacas, García-González, et al., 2022; Martínez-Mauricio et al., 2024), demostrando mejores capacidades de modelado que características calculadas (García-Jacas et al., 2022). Por lo tanto, las incrustaciones ESM-2 son una alternativa que supera a características basadas en MSAs (*e.g.*, las derivadas de PSSMs) para transferir información evolutiva para tareas de modelación específicas (*e.g.*, predicción de AMPs).

En cuanto al método de predicción de estructuras 3D, sAMPpred-GAT usa el método trRosetta (Du et al., 2021). trRosetta depende de MSAs para derivar 526 canales de características de uno y dos sitios para crear una red neuronal profunda para predecir las geometrías inter-aminoácidos. El uso de trRosetta y de información evolutiva derivada de PSSMs hacen que sAMPpred-GAT sea un protocolo dependiente de alineamiento, lo que conlleva a un alto consumo de memoria y tiempo computacional (Zielezinski et al., 2017). En consecuencia, estas dos fases deben llevarse a cabo de forma independiente antes de utilizar el modelo sAMPpred-GAT, dificultando así su uso en el cribado de grandes volúmenes de datos. Por otro lado, trRosetta no está en correspondencia con los métodos más recientes para predecir estructuras 3D de proteínas, tales como AlphaFold2 (Jumper et al., 2021), Alphafold3 (Abramson et al., 2024), RoseTTAFold (Baek et al., 2021), ESMFold (Lin et al., 2023), y HelixFold-Single (Fang et al., 2023).

Específicamente, el modelo ESMFold es un predictor de estructura 3D basado en el modelo ESM-2 de 36 capas y 3 mil millones de parámetros (Z. Lin et al., 2023). A diferencia de trRosetta (Du et al., 2021), AlphaFold2 (Jumper et al., 2021), Alphafold3 (Abramson et al., 2024), y RoseTTAFold (Baek et al., 2021), ESMFold no depende de MSAs ni de plantillas de estructuras de proteínas similares para lograr un rendimiento competitivo. Por lo tanto, ESMFold es más eficiente computacionalmente que los enfoques anteriormente mencionados. Según la métrica TM-score (Xu y Zhang, 2010), ESMFold (0.83) alcanzó un rendimiento comparable a AlphaFold (0.88) y RoseTTAFold (0.83) en el conjunto de datos CAMEO, aunque fue (0.68) notablemente inferior a AlphaFold (0.85) en el conjunto de datos CASP14. Estas diferencias de rendimiento entre los modelos ESMFold y AlphaFold se deben a la perplejidad (Huyen, 2019) de las predicciones del modelo ESM-2 de 36 capas. Para proteínas con baja perplejidad, ESMFold y AlphaFold

presentaron un rendimiento comparable. A pesar de esta brecha de rendimiento, ESMFold puede ser valioso para implementar protocolos dependientes de geometría que demanden alto procesamiento de datos, como el cribado de grandes conjuntos de datos para descubrir potenciales AMPs mediante modelos de DL basados en grafos (GDL, por sus siglas en inglés).

Finalmente, es importante resaltar que un paso importante para construir buenos modelos basados en estructuras 3D es una adecuada caracterización o representación de los aspectos geométricos de los péptidos. La representación de los aspectos geométricos (3D) de las estructuras moleculares como una extensión natural de sus aspectos topológicos (2D) se remonta a mediados de los años 1980 (Todeschini y Consonni, 2009). El concepto de matriz de distancia geométrica se ha comúnmente utilizado para representar la geometría molecular calculando la métrica Euclidiana entre pares de átomos en moléculas de pequeño y mediano tamaño, o entre pares de aminoácidos en péptidos/proteínas (Todeschini y Consonni, 2009). La matriz de distancia geométrica es la base para calcular diferentes descriptores moleculares geométricos (Todeschini y Consonni, 2009), así como para crear mapas de contacto de péptidos y proteínas. Esto constituye la razón por la cual las aplicaciones del estado del arte basadas en aprendizaje de grafos (Fasoulis et al., 2021; Gligorijević et al., 2021; Jamasb et al., 2021; Baranwal et al., 2022; K. Yan et al., 2023; Réau et al., 2023; Z. Gao et al., 2023; Cordoves-Delgado y García-Jacas, 2024) solo aplican un umbral basado en la distancia Euclidiana para crear los grafos a partir de las estructuras 3D de péptidos, a pesar de que no existe evidencia, más allá del razonamiento intuitivo, que justifique la distancia Euclidiana como la más adecuada. Inspirado por esto último, Marrero-Ponce et al. (Marrero-Ponce et al., 2015) definieron el concepto de matriz de (dis-)similitud espacial como una generalización de la matriz de distancia geométrica.

La matriz de (dis-)similitud espacial se calcula utilizando varias funciones de distancia (consulte la Tabla S7 en (Todeschini y Consonni, 2009)), incluyendo la distancia Euclidiana. La distancia Euclidiana es un caso específico de la definición de Minkowski (Deza y Deza, 2016). Por lo tanto, cualquier función derivada de la definición de Minkowski (*e.g.*, la distancia de Manhattan) así como cualquier función que cumpla con las propiedades de no negatividad, simetría y reflexividad para todos los pares de puntos en un espacio N-dimensional pueden ser utilizadas como función de distancia. La matriz de (dis-)similitud espacial se ha usado en el cálculo de descriptores moleculares 3D (Marrero-Ponce et al., 2015), donde se demostró que funciones de distancia diferentes a la Euclidiana (*e.g.*, Lance-Williams) permiten obtener descriptores con mejor capacidad de modelación y mayor capacidad para discriminar entre moléculas estructuralmente diferentes (Marrero-Ponce et al., 2015). Por lo tanto, se considera que los algoritmos basados en GDL para predecir AMPs pueden beneficiarse de representaciones de grafos que no se construyan únicamente con

la distancia Euclidiana, teniendo en cuenta que se pueden crear grafos con topologías diferentes que contribuyan a codificar espacios químicos distintos, y, por ende, contribuyan a obtener modelos con mejor capacidad predictiva.

Teniendo en cuenta lo anteriormente expuesto, se plantea el siguiente problema científico, objetivo general y objetivos específicos.

1.3 Problema científico

Los modelos predictivos de AMPs basados en grafos construidos a partir de las estructuras 3D predichas, utilizan descriptores a nivel de aminoácidos y métodos de predicción de estructuras 3D dependientes de alineamiento, usan descriptores a nivel de aminoácidos que no contienen información evolutiva de las proteínas actualmente conocidas, y se limitan a la distancia Euclidiana para calcular las distancias inter-aminoácidos y definir las aristas de los grafos. Como consecuencia, estos modelos no se pueden usar en el cribado de grandes conjuntos de datos, y dejan de codificar y de aprender información evolutiva y geométrica relevante de los péptidos, lo cual dificulta el desarrollo de mejores modelos para identificar AMPs.

1.4 Objetivos

1.4.1 Objetivo general

Proponer algoritmos que usen características a nivel de aminoácido derivadas de los modelos ESM-2, y criterios geométricos aplicando diferentes funciones de distancias para construir grafos topológicamente diferentes desde estructuras predichas por ESMFold, que permitan construir modelos QSAR robustos basados en grafos para identificar AMPs.

1.4.2 Objetivos específicos

- Implementar la generación de estructuras con ESMFold para derivar representaciones basadas en

grafos usando criterios geométricos.

- Implementar el cálculo de características a nivel de aminoácidos con ESM-2 para caracterizar los vértices de los grafos.
- Construir modelos basados en grafos usando una arquitectura basada en Red de Atención de Grafos (GAT).
- Validar el rendimiento de los modelos construidos en conjuntos de datos de la literatura.
- Comparar el rendimiento de los modelos construidos con respecto a modelos propuestos en la literatura.

1.5 Aportes

El presente trabajo tiene como aporte metodológico un *framework* guiado por GDL para construir modelos predictivos de AMPs a partir del uso de estructuras peptídicas predichas por ESMFold, de características a nivel de aminoácido derivadas de los modelos ESM-2, y de diferentes funciones de distancias. Este *framework* puede ser también aplicado para modelar cualquier tarea relacionada con la predicción de actividades (o propiedades) biológicas de péptidos y proteínas. El aporte práctico se muestra con la creación de nuevos modelos para la predicción de AMPs generales, así como del software esm-AxP-GDL para la construcción y uso prospectivo de dichos modelos. Este software y los modelos están disponibles en https://github.com/cicese-biocom/esm-AxP-GDL.git.

1.6 Organización de la tesis

La organización del presente trabajo de tesis es la siguiente. En el Capítulo 2. se abordan los fundamentos teóricos que sustentan el presente trabajo de tesis. En el Capítulo 3. se presenta el *framework* esm-AxP-GDL para construir modelos basados en GDL a partir del uso de estructuras de péptidos predichas por ESMFold y del uso de características a nivel de aminoácidos derivadas de los modelos ESM-2 para la predicción de AMPs. En el Capítulo 4. se evalúa la utilidad del *framework* mediante la construcción y validación de diferentes modelos para predecir AMPs, y se realizan estudios comparativos con respecto a modelos del estado del arte. Por último, en el Capítulo 5. se exponen las conclusiones y el trabajo futuro.

Capítulo 2. Aprendizaje profundo de grafos representando péptidos

En este capítulo se presentan los fundamentos teóricos que sustentan el presente trabajo de tesis. En la sección 2.1 se abordan los conceptos básicos relacionados con los péptidos antimicrobianos, incluyendo sus estructuras y características. En la sección 2.2 se discute los métodos de predicción de estructuras 3D y se presenta el método de predicción ESMFold. Posteriormente, en la sección 2.3 se introducen definiciones básicas de grafos que son esenciales para el desarrollo de esta tesis. En la sección 2.4 se aborda sobre las representaciones de péptidos como grafos, y se detalla el proceso de caracterización de nodos usando modelos ESM-2. Finalmente, en la sección 2.5 se introducen los conceptos relacionados con aprendizaje profundo de grafos y se describe la arquitectura de aprendizaje de grafos GAT.

2.1 Péptidos antimicrobianos

Los péptidos antimicrobianos (AMPs), también conocidos como péptidos de defensa del huésped (Divyashree et al., 2019), son un grupo diverso de moléculas que constituyen una parte esencial de la inmunidad innata para combatir las diferentes infecciones en los organismos vivos (Divyashree et al., 2019). Los AMPs suelen estar codificados genéticamente y pueden expresarse de forma constitutiva para proteger determinados nichos o inducirse en respuesta a patógenos invasores (Simonson, 2022). La mayoría de los AMPs presentan una estructura bastante concisa que suele tener una longitud de entre 2 y 100 aminoácidos (Pinacho-Castellanos et al., 2021; K. Yan et al., 2023; Bucataru y Ciobanasu, 2024). La composición de aminoácidos varia significativamente en función de su estructura, actividad, y origen (Simonson, 2022).

2.1.1 Estructura de los péptidos

Los péptidos son cadenas cortas de aminoácidos unidos por un enlace covalente denominado enlace peptídico (Yan, 2024). Los aminoácidos son pequeñas moléculas orgánicas con una propiedad definitoria: todos poseen un grupo ácido carboxílico ($-COOH$), un grupo amino ($-NH_2$) y un átomo de hidrógeno (H), unidos todos a un átomo central de carbono, conocido como carbono alfa (α). Este carbono α también lleva una cadena lateral específica (o grupo R), cuya identidad distingue a un aminoácido de otro (ver

Figura 1A) (Alberts et al., 2019). Según la naturaleza química de la cadena lateral, los aminoácidos se clasifican en ácidos, básicos, polares sin carga, o apolares (Alberts et al., 2019). Los enlaces formados por el carbono α con el grupo carboxilo, el grupo amino y el átomo de hidrógeno se conocen como la cadena principal (Harris y Korolchuk, 2023).

Los enlaces peptídicos se forman mediante reacciones de condensación que unen un aminoácido con el siguiente, en las cuales el grupo carboxilo de un aminoácido reacciona con el grupo amino del aminoácido siguiente. Durante este proceso se libera una molécula de agua y se crea una cadena lineal de aminoácidos con un grupo amino en un extremo, conocido como N-terminal, y un grupo carboxilo en el otro, conocido como C-terminal (ver Figura 1B) (Alberts et al., 2019; Harris y Korolchuk, 2023). Una vez que los aminoácidos se unen se les denomina residuos de aminoácidos. Un péptido tiene generalmente menos de 100 aminoácidos (cadenas más largas se consideran proteínas) (Pinacho-Castellanos et al., 2021; K. Yan et al., 2023; Bucataru y Ciobanasu, 2024).

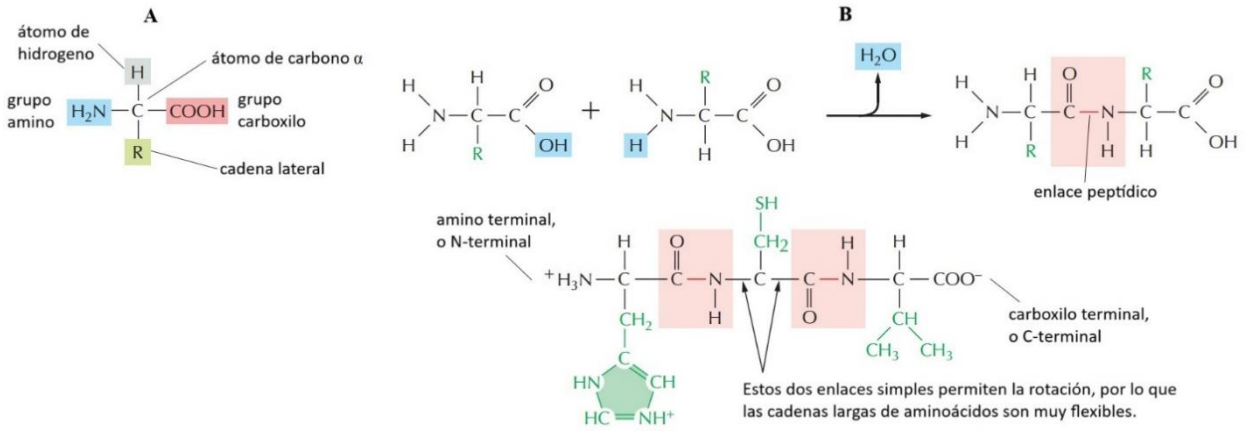

Figura 1. Formación de péptidos y proteínas por la unión de aminoácidos mediante enlaces peptídicos. (A) Estructura de un aminoácido. (B) Enlace peptídico. Adaptado de (Alberts et al., 2019), Panel. 2-6.

La estructura de un péptido incluye varios niveles interdependientes de organización (Alberts et al., 2019). El primer nivel es la estructura primaria, determinada por la secuencia de aminoácidos del péptido. El siguiente nivel de organización es la estructura secundaria, que incluye las hélices α y las hojas β que se forman en determinados segmentos de la cadena polipeptídica. Finalmente, la estructura 3D es una conformación tridimensional completa formada por toda la cadena polipeptídica, incluidas las hélices α, las hojas β, y todos los bucles y pliegues que se forman entre los extremos N-terminal y C-terminal. Aunque existe un nivel adicional de organización que es la estructura cuaternaria, esta no se analiza en las siguientes subsecciones porque no está relacionada con este trabajo.

2.1.1.1 Estructura primaria

La estructura primaria de los péptidos se define como la secuencia lineal de aminoácidos que lo componen unidos por enlaces peptídicos. La secuencia específica de aminoácidos es determinante. Cambiar un solo aminoácido puede alterar la estructura y función global del péptido (Wei y Kumbar, 2020). En la Tabla 1 se presentan los 20 aminoácidos canónicos que se encuentran en péptidos, y sus códigos de tres y una letra, clasificados por la naturaleza química de su cadena lateral.

Tabla 1. Los 20 aminoácidos canónicos clasificados por la naturaleza química de su cadena lateral, junto con sus códigos de tres y una letra. Tabla basada en Panel 2-6 de (Alberts et al., 2019).

Aminoácido	Código de tres letras	Código de una letra	Tipo
Arginina	Arg	R	Básico
Lisina	Lys	K	
Histidina	His	H	
Ácido aspártico	Asp	D	Ácido
Ácido glutámico	Glu	E	
Asparagina	Asn	N	Polar no cargado
Glutamina	Gln	Q	
Serina	Ser	S	
Treonina	Thr	T	
Tirosina	Tyr	Y	
Alanina	Ala	A	No polar
Valina	Val	V	
Leucina	Leu	L	
Isoleucina	Ile	I	
Prolina	Pro	P	
Fenilalanina	Phe	F	
Metionina	Met	M	
Triptófano	Trp	W	
Glicina	Gly	G	
Cisteína	Cys	C	

2.1.1.2 Estructura secundaria

La estructura secundaria se refiere a las estructuras plegadas localmente que se forman dentro de un polipéptido debido a las interacciones entre los átomos de la cadena principal (Wei y Kumbar, 2020). Los tipos más comunes de estructuras secundarias son la hélice α y la hoja β (Harris y Korolchuk, 2023). La hélice α (ver Figura 2A) se caracteriza por la interacción de enlaces de hidrógeno (H) entre los grupos CO y NH de los aminoácidos situados en las posiciones i e $i+4$ de la cadena principal. En la estructura de hoja β (ver Figura 2B (a)), las cadenas polipeptídicas extendidas, denominadas hebras β, se organizan en un patrón en zigzag mediante una serie de enlaces de hidrógeno inter-hebra (X. Yan, 2024). Las hebras β adyacentes que forman una hoja β pueden ser paralelas (ver Figura 2B (b)) o antiparalelas (ver Figura 2B

(c)). Cuando los segmentos de una cadena polipeptídica están orientados en la misma dirección, se forma una hoja β paralela, en caso contrario, se forma una hoja β antiparalela. Ambos tipos de hoja β producen una estructura muy rígida y acanalada, y forman el núcleo de muchas proteínas (Alberts et al., 2019).

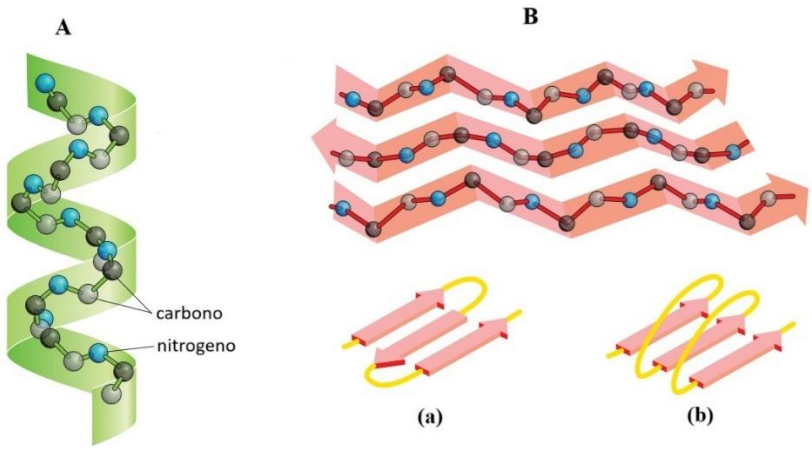

Figura 2. Estructuras de plegamiento (A) hélice $\boldsymbol{\alpha}$ y (B) hojas $\boldsymbol{\beta}$ de una cadena polipeptídica. También, se muestran las hojas $\boldsymbol{\beta}$ en sus dos formas: (a) hoja $\boldsymbol{\beta}$ antiparalela y (b) hoja $\boldsymbol{\beta}$ paralela. Adaptado de (Alberts et al., 2019), Fig. 4-12, 4-13 y 4-17.

2.1.1.3 Estructura terciaria

La estructura 3D de un péptido es la disposición tridimensional de los átomos dentro de una misma cadena polipeptídica que se forma principalmente debido a las interacciones entre los grupos laterales de los aminoácidos que componen el péptido (Wei y Kumbar, 2020). Esta estructura 3D se estabiliza mediante diversas interacciones, tales como enlaces de hidrógeno, enlaces iónicos, interacciones dipolo, y fuerzas hidrofóbicas y disulfídicas (Alberts et al., 2019). Las interacciones hidrofóbicas agrupan los aminoácidos hidrofóbicos en el interior, mientras que los hidrofílicos se localizan en la superficie para interactuar con el agua (Wei y Kumbar, 2020).

La estructura 3D está definida por los ángulos de torsión (o diedros) de la cadena principal y de las cadenas laterales, conocidos como ángulos phi (ϕ), psi (ψ), omega (ω), y chi (χ) (Seetharama, 2022). La Figura 3 muestra estos ángulos diedros y cómo ellos se forman. En la Figura 3A se observa que el ángulo de torsión ω está definido por cuatro átomos ($C_\alpha - C' - N - C_\alpha$), y que los ángulos ϕ y ψ se definen por la rotación alrededor de los enlaces simples del átomo de C_α. Estos dos últimos ángulos se describen por los ángulos entre los planos formados por tres átomos de la cadena principal, como se muestra en la Figura 3B. La conformación de la cadena lateral de los aminoácidos en el péptido está definida por los ángulos de torsión χ, que involucra cuatro átomos (Seetharama, 2022).

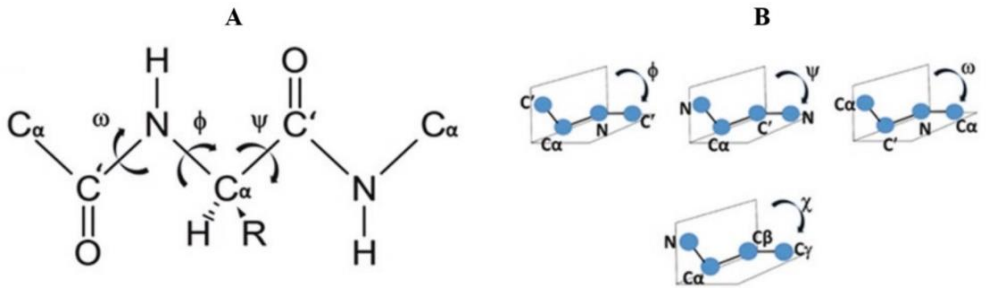

Figura 3. Ángulos de torsión en la estructura 3D de un péptido. Tomado de (Seetharama, 2022).

2.1.2 Caracterización de los péptidos antimicrobianos

Según su origen, los AMPs pueden clasificarse en naturales o sintéticos. Los AMPs naturales se obtienen de una variedad de fuentes biológicas, incluyendo microorganismos, artrópodos marinos, plantas, hongos y mamíferos. Por su parte, los AMPs sintéticos se diseñan en laboratorio a partir de secuencias de aminoácidos modificadas para mejorar su actividad antimicrobiana, selectividad hacia microorganismos específicos, y resistencia a la degradación (Bucataru y Ciobanasu, 2024). La acción de los péptidos en las células objetivo depende de las características generales de los AMPs, como el carácter hidrofóbico, la carga neta, la conformación, y la estructura secundaria. Según Bucataru et al. (2024), los AMPs se agrupan en cuatro categorías principales según su estructura: hélices α, hojas β, mezcla de α y β, y extendidas (o espiral aleatoria). Las hélices α tienen una disposición en espiral que les permite interactuar eficazmente con membranas microbianas. Las hojas β se forman en estructuras plegadas, estabilizadas por enlaces disulfuro, y muestran actividad antimicrobiana. La mezcla de α y β combina elementos de ambas estructuras, proporcionando amplias propiedades antimicrobianas y efectos inmunomoduladores. Por último, los AMPs con una estructura extendidas presentan conformaciones flexibles, lo que les permite interactuar de diversas maneras con las membranas microbianas.

Los AMPs son extremadamente diversos, pero comparten varias propiedades comunes, como la carga positiva, la hidrofobicidad y la anfipaticidad. Aproximadamente el 50% de los aminoácidos en los AMPs son hidrofóbicos, lo que facilita su interacción con las membranas microbianas (Bucataru y Ciobanasu, 2024). Según Kumar et al. (2018), los AMPs muestran una carga neta positiva que varía de +2 a +13, y pueden incluir un dominio catiónico específico atribuible a residuos de lisina y arginina (y a veces histidina). Según su carga neta, se dividen en catiónicos (88%), neutros (6%) y aniónicos (6%). La mayoría de los AMPs son anfipáticos, aunque existen casos raros de AMPs compuestos exclusivamente por aminoácidos hidrofóbicos o hidrofílicos (Bucataru y Ciobanasu, 2024). Esta carga positiva está relacionada con la actividad antimicrobiana, pero un exceso de carga puede aumentar la actividad hemolítica y reducir la

actividad antimicrobiana. Esto sucede porque una interacción extremadamente fuerte entre el péptido y el grupo cabeza de fosfolípido puede dificultar la translocación del péptido hacia la capa interna de la membrana. Los AMPs, generalmente catiónicos, atacan membranas bacterianas aniónicas formando estructuras helicoidales anfipáticas, aunque también pueden dirigirse a otros objetivos como las paredes celulares y ribosomas bacterianos. Esta capacidad para atacar múltiples objetivos reduce la probabilidad de que las bacterias desarrollen resistencia (Simonson, 2022).

La hidrofobicidad de los AMPs, definida por el porcentaje de residuos hidrofóbicos (*e.g.*, valina, leucina, y fenilalanina), suele ser del 50%. La hidrofobicidad determina la medida en que los AMPs solubles en agua pueden integrarse en la bicapa lipídica de la membrana. Esta es esencial para la permeabilización de la membrana. Sin embargo, niveles excesivos de hidrofobicidad pueden conducir a toxicidad en células mamíferas y pérdida de selectividad antimicrobiana (Kumar et al., 2018). Finalmente, la anfipaticidad, definida como la proporción de residuos hidrofílicos e hidrofóbicos en la estructura, permite a los AMPs formar estructuras como la hélice α, que tiene una cara polar y otra no polar, facilitando la interacción con las membranas microbianas y la permeabilización de la membrana.

El mecanismo principal de acción de los AMPs es la interacción y disrupción de la membrana (Simonson, 2022). Según Kumar et al. (2018), los AMPs se pueden clasificar en dos mecanismos principales de acción: eliminación directa y modulación inmune. El mecanismo de eliminación directa se divide en dos subcategorías: acción dirigida a la membrana y acción no dirigida a la membrana. De forma general, los AMPs matan a los microbios al alterar la integridad de la membrana a través de la interacción con las membranas celulares cargadas negativamente, inhibiendo la síntesis de ADN, ARN y proteínas, y a veces interactuando con componentes intracelulares (Divyashree et al., 2019). En cuanto a la membrana, los AMPs pueden formar poros mediante varios modelos: barrel-stave, toroidal-pore, carpet y aggregate. Estos mecanismos aumentan la permeabilidad de la membrana, lo que lleva a la pérdida de componentes celulares y a la muerte celular (Q. Y. Zhang et al., 2021).

2.2 Métodos de predicción de la estructura terciaria

Históricamente, la Biología Estructural (Carugo y Djinović-Carugo, 2023) ha dependido de métodos experimentales para determinar la estructura 3D de las proteínas, tales como la cristalografía de rayos X y la resonancia magnética nuclear (Maveyraud y Mourey, 2020; Hu et al., 2021). Desde 1971, la base de datos Protein Data Bank (PDB) (Berman et al., 2000) almacena las estructuras terciarias (3D) obtenidas

experimentalmente. Esta base de datos ha aumentado de tamaño a lo largo de los años, y hasta el año 2023 contenía 214,121 estructuras, con una tasa de crecimiento promedio entre el 2017 y el 2021 de 13,723 estructuras por año. Sin embargo, esta cifra es insignificante en comparación con el número de secuencias de péptidos y proteínas almacenadas en UniProt (Bairoch et al., 2005), que hasta el 2023 contenía 245,482,527. Este fenómeno se refiere como brecha de conocimiento secuencia-estructura (Schwede, 2013).

La limitada disponibilidad de estructuras 3D de proteínas que pueden ser obtenidas experimentalmente ha impulsado la necesidad de utilizar métodos computacionales para predecirlas. Los avances en este campo han sido promovidos en gran medida por la competencia CASP (del inglés - Critical Assessment of Structure Prediction) (https://predictioncenter.org/). CASP es un concurso a nivel mundial dedicado a la predicción de la estructura 3D de las proteínas a partir de su secuencia de aminoácidos, que ha servido durante mucho tiempo como el estándar de oro para la evaluación de la precisión de la predicción de estructuras (Jumper et al., 2021). En los últimos años, modelos basados en redes neuronales profundas como AlphaFold2 (Jumper et al., 2021), Alphafold3 (Abramson et al., 2024), trRosetta (Du et al., 2021), RoseTTAFold (Baek et al., 2021), ESMFold (Z. Lin et al., 2023), ESM-3 (Hayes et al., 2024) y HelixFold-Single (Fang et al., 2023) han revolucionado la predicción de la estructura 3D de las proteínas a partir de su secuencia de aminoácidos. Estos modelos están cerrando rápidamente la brecha secuencia-estructura y generando nuevas oportunidades para construir mejores modelos predictivos basados en información estructural.

Los modelos AlphaFold, trRosetta y RoseTTAFold dependen de MSA. El MSA es una técnica utilizada para analizar el grado de conservación o variabilidad entre secuencias biológicas, permitiendo observar cambios específicos en los residuos de aminoácidos en posiciones determinadas (Gore y Jagtap, 2024). Esto facilita la identificación de regiones de similitud que pueden indicar relaciones funcionales, estructurales, y/o evolutivas entre secuencias (Paruchuri et al., 2022). Este proceso recibe como entrada tres o más secuencias, y, para generar un análisis de similitud entre ellas, los algoritmos de alineamiento suelen basarse en el concepto de distancias de edición, es decir, cuántos cambios son necesarios para transformar una secuencia en otra (Ranganathan et al., 2019). Dado que el número de posibles alineaciones crece exponencialmente con el aumento de secuencias, el MSA se considera un problema NP-difícil (Zielezinski et al., 2017; Paruchuri et al., 2022), lo cual limita su uso en el análisis de grandes cantidades de datos.

En contraste, los modelos ESMFold, ESM-3 y HelixFold-Single son independientes de MSAs (Fang et al.,

2023; Z. Lin et al., 2023; Hayes et al., 2024). Estos modelos se basan en modelos de lenguaje de proteínas, los cuales han demostrado ser una buena alternativa a los MSAs para extraer conocimiento evolutivo, y además pueden ser aplicados sobre millones secuencias de entrada. La principal ventaja de estos modelos independientes de MSAs es su eficiencia computacional respecto a los que dependen de MSAs. Por ejemplo, en una sola GPU NVIDIA V100, ESMFold predice la estructura de una proteína con 384 residuos en 14.2 segundos, seis veces más rápido que el modelo AlphaFold2 (Z. Lin et al., 2023). Asimismo, HelixFold-Single utilizando una GPU NVIDIA A100 (40G), realiza la predicción de proteínas de menos de 100 aminoácidos en aproximadamente una milésima parte del tiempo requerido por AlphaFold2, incluso para proteínas de más de 800 aminoácidos (Fang et al., 2023).

2.2.1 Modelo ESMFold

El modelo ESMFold (Lin et al., 2023) es un predictor de estructura 3D que utiliza el modelo de lenguaje ESM-2 (ver sección 2.4.1.1) de 36 capas y 3 mil millones de parámetros para generar predicciones precisas de estructura directamente desde la secuencia de aminoácidos. Para utilizar el modelo ESMFold, se dispone de la biblioteca fair-esm (https://pypi.org/project/fair-esm/) desarrollada en Python. A continuación, se presenta un script de ejemplo que ilustra cómo predecir la estructura 3D de la secuencia peptídica IFSAIAGLLSNLL usando ESMFold. Una vez que se ejecuta el script, la variable pdb_str contendrá la estructura PDB de la proteína en formato texto.

```
import torch
import esm

# Cargar el modelo ESMFold (en GPU).
model = esm.pretrained.esmfold_v1()
model = model.eval().cuda() # Desactiva el dropout para obtener resultados determinísticos.

# Secuencia peptídica.
sequence = 'IFSAIAGLLSNLL'

# Genera la estructura 3D a partir de la secuencia.
with torch.no_grad():
        pdb_str = model.infer_pdb(sequence)
```

2.3 Definiciones sobre grafos

Un grafo es una estructura de datos compuesta por nodos (o vértices) y aristas que conectan a los nodos.

Matemáticamente, se puede denotar como $G = (V, E)$, donde $V = \{1, \dots, N\}$ es el conjunto de nodos, y $E \subseteq V \times V$ es el conjunto de aristas, donde $(i, j) \in E$ denota una arista que conecta al nodo i con el nodo j, y, por lo tanto, se dice que el nodo j es vecino (o adyacente) al nodo i. Las aristas de los grafos pueden ser dirigidas o no dependiendo de si existen dependencias direccionales entre los nodos. En un grafo no dirigido las aristas se representan como conexiones bidireccionales, tal que cada arista (i, j) es equivalente a la arista (j, i). El conjunto de vecinos (o vecindario) de un nodo se define como $\mathcal{N}_i = \{j \in V \mid (i, j) \in E\}$. Tanto los nodos como las aristas pueden tener atributos asociados que proporcionan información adicional sobre los nodos y las relaciones entre estos. Comúnmente, se usa una matriz para las características de los nodos y otra para las características de las aristas. Cuando las aristas tienen atributos, el grafo se considera ponderado. Por último, es importante mencionar que un grafo G es simple si no tiene auto ciclos (*i.e.*, aristas de la forma (i, i)), aristas múltiples, o redundantes. En las siguientes subsecciones se abordan las representaciones de grafos adoptadas en esta tesis.

2.3.1 Matriz de adyacencia

Para un grafo $G = (V, E)$, donde V es el conjunto de nodos y E el conjunto de aristas, la representación mediante una matriz de adyacencia supone que los nodos están numerados como $1, 2, \dots, |V|$ de alguna manera arbitraria. Entonces, la representación de la matriz de adyacencia consiste en una matriz A de dimensión $|V| \times |V|$, tal que (Cormen et al., 2022) (ver ecuación (1)):

$$A_{i,j} = \begin{cases} 1, & si\ (i,j) \in E \\ 0, & en\ otro\ caso \end{cases} \tag{1}$$

Si G es un grafo ponderado, se utiliza una matriz adicional W de dimensión $|V| \times |V|$, definida según la ecuación (2), donde $\omega_{i,j}$ es un vector de características que representa los atributos de la arista (i, j).

$$W_{i,j} = \begin{cases} \omega_{i,j}, & si\ (i,j) \in E \\ 0, & en\ otro\ caso \end{cases} \tag{2}$$

2.3.2 Lista de adyacencia

La representación mediante lista de adyacencia de un grafo $G = (V, E)$ consiste en un arreglo A de $|V|$

listas, una para cada nodo en V. Para cada $i \in V$, la lista de adyacencia $A[i]$ contiene todos los vértices j, tal que existe una arista $(i, j) \in E$. Es decir, $A[i]$ consta de todos los vértices adyacentes a i en G (Cormen et al., 2022). Si G es ponderado, donde $\omega_{i,j} \in \mathbb{R}^{F_e}$ representa el vector de características de la arista $(i, j) \in E$, y F_e es el número de características de cada arista, cada entrada en la lista de adyacencia $A[i]$ no solo contendrá el vértice j, sino también el vector de características $\omega_{i,j}$ asociado a la arista (i, j).

2.4 Representación de péptidos como grafos

La representación de péptidos como grafos es un enfoque utilizado para modelar las interacciones y relaciones estructurales entre aminoácidos. Este enfoque utiliza grafos a nivel de residuo, donde los nodos representan los aminoácidos (residuos) y las aristas las relaciones entre ellos. Las definiciones de las aristas se basan en interacciones intramoleculares o en límites basados en la distancia Euclidiana (Jamasb et al., 2021) que permiten representar diferentes aspectos geométricos de los péptidos. En este contexto, Graphein es una biblioteca de Python que proporciona múltiples métodos para la generación de grafos basados en la estructura 3D de péptidos y proteínas (Jamasb et al., 2021). Esta herramienta permite crear representaciones de grafos usando criterios fisicoquímicos (*e.g.*, interacciones aromáticas, iónicas, enlaces de hidrógeno, entre otros), criterios basados en k-vecinos más cercanos, basados en triangulación de Delunay (Jamasb et al., 2021), y basado en la distancia Euclidiana. Estas relaciones inter-aminoácidos pueden incluirse en el mismo grafo para capturar diversos aspectos de la información estructural.

El criterio basado en distancia Euclidiana es el más utilizado para crear aristas. Este consiste en crear una arista si la distancia Euclidiana entre los átomos de carbono α (o carbono β) de cada par de aminoácidos (nodo) es menor o igual a un umbral específico (Gligorijević et al., 2021; Abdin et al., 2022; K. Yan et al., 2023; Ge et al., 2024). Comúnmente, el concepto de matriz de distancia geométrica se ha utilizado para representar la geometría molecular calculando la distancia Euclidiana entre pares de átomos en moléculas de pequeño y mediano tamaño, o entre pares de aminoácidos en péptidos y proteínas (Todeschini y Consonni, 2009). La matriz de distancia geométrica constituye la base para crear mapas de contacto (Dubitzky et al., 2013) de péptidos. Esta es la razón por la cual aplicaciones del estado del arte basadas en aprendizaje de grafos (Fasoulis et al., 2021; Gligorijević et al., 2021; Jamasb et al., 2021; Baranwal et al., 2022; K. Yan et al., 2023; Réau et al., 2023; Z. Gao et al., 2023; Cordoves-Delgado y García-Jacas, 2024) solo aplican un umbral basado en la distancia Euclidiana para construir los grafos a partir de las estructuras 3D de péptidos y proteínas, a pesar de que no existe evidencia, más allá del razonamiento intuitivo, que justifique la distancia Euclidiana como la más adecuada. Inspirado en esto último, Marrero-Ponce et al.

(2015) definieron el concepto de matriz de (dis-)similitud espacial como una generalización de la matriz de distancia geométrica.

La matriz de (dis-)similitud espacial se calcula utilizando varias funciones de distancia (ver sección 2.4.2). Esta matriz se ha usado en el cálculo de descriptores moleculares 3D (Marrero-Ponce et al., 2015), donde se demostró que funciones de distancia distintas de la Euclidiana (*e.g.*, Lance-Williams) (ver Figura 4) permiten calcular descriptores con una mejor capacidad de modelado y mayor capacidad para discriminar entre moléculas estructuralmente diferentes (Marrero-Ponce et al., 2015). Por lo tanto, se considera que los algoritmos basados en aprendizaje de grafo de péptidos pueden beneficiarse de representaciones que no se limiten a la distancia Euclidiana. Estas representaciones pudieran generar grafos con topologías diversas que contribuyan a codificar espacios químicos distintos, para potencialmente mejorar el rendimiento de los modelos de ML.

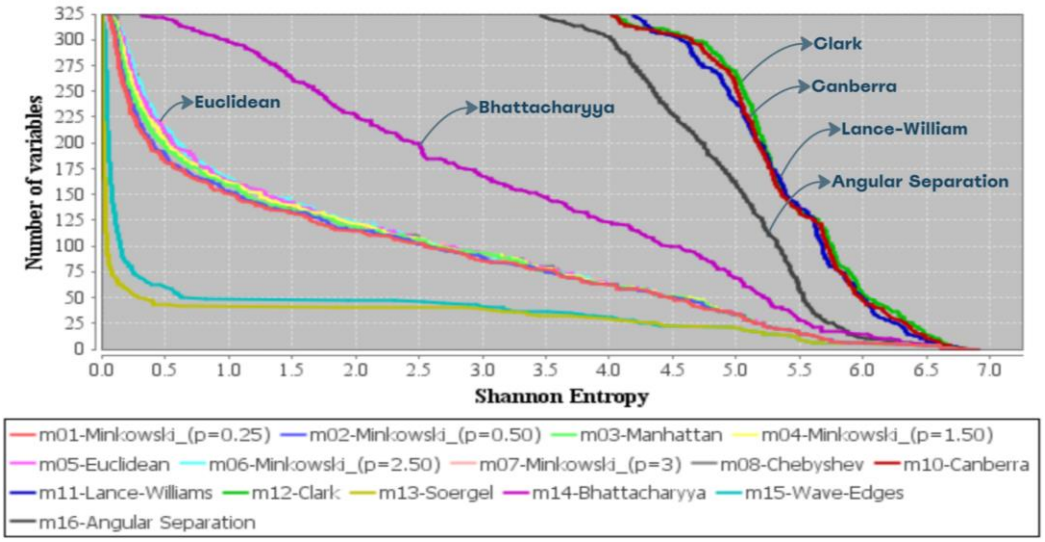

Figura 4. Contenido de información basado en Entropía de Shannon de descriptores moleculares geométricos basados en formas algebraicas lineales, bilineales y cuadráticas cuando se calculan a partir de diferentes funciones de distancias. Adaptado de (Marrero-Ponce et al., 2015), Fig. 1(C).

2.4.1 Caracterización de nodos representando aminoácidos

En los grafos a nivel de aminoácidos (residuo), los nodos (aminoácidos) pueden ser representados mediante vectores numéricos derivados de un esquema de codificación diseñado para capturar las características distintivas de cada aminoácido y su relación con otros en el espacio de representación. Estos esquemas se basan principalmente en la información de la secuencia (*e.g.*, one-hot encoding y position encoding) (Abdin et al., 2022; K. Yan et al., 2023) o en la representación de diversas propiedades

fisicoquímicas (*e.g.*, Z-scores o vectores de propiedades hidrofóbicas, estéricas y electrónicas) (Todeschini y Consonni, 2009; Abdin et al., 2022). Sin embargo, un número creciente de estudios han demostrado que la información evolutiva extraída de PSSMs es mucho más informativa que aquella basada únicamente en las características de la secuencia (Mohammadi et al., 2022). Una PSSM es una representación que describe patrones de secuencias biológicas, donde cada fila corresponde a un posible carácter (*e.g.*, un aminoácido) y cada columna representa una posición en la secuencia. Los valores en la PSSM pueden ser de varios tipos, siendo las más comunes los conteos simples, probabilidades o puntuaciones logarítmicas, que indican la importancia o la frecuencia de cada carácter en una posición específica de la secuencia (Ranganathan et al., 2019). Sin embargo, estas matrices se derivan de MSA, que es considerado un problema NP-difícil (Paruchuri et al., 2022). Además, estas no incorporan información sobre las correlaciones entre posiciones, lo que es crucial para capturar la coevolución de aminoácidos en las secuencias. La información contenida en ellas está limitada al conjunto de secuencias a partir de la cual se obtienen (Ranganathan et al., 2019), por lo que se consideran obsoletas para transferir información evolutiva en comparación con modelos de última generación como la familia de modelos ESM-2 (Lin et al., 2023).

2.4.1.1 Modelos ESM-2

Los modelos ESM-2 son modelos de lenguaje de proteínas (PLMs, por siglas en inglés) que tienen el potencial de identificar patrones en las secuencias de proteínas a lo largo de la evolución, lo que elimina la necesidad de recurrir a bases de datos evolutivas externas, a realizar MSA, o a usar plantillas para aprender conocimientos coevolutivos (Z. Lin et al., 2023). Un PLM tiene como objetivo predecir la probabilidad de observar una secuencia de proteína específica, aprovechando la información recopilada de secuencias de proteínas hasta el momento (Ruffolo y Madani, 2024). Para lograr esto, estos modelos aprenden patrones y relaciones entre aminoácidos sobre secuencias de cualquier longitud mediante el uso de arquitecturas de procesamiento de lenguaje natural, tales como los Transformadores (Rives et al., 2021; Z. Lin et al., 2023). Los modelos ESM-2 se entrenaron con aproximadamente 65 millones de secuencias de proteínas únicas de la base de datos UniRef (Bateman, 2019). Utilizan una arquitectura BERT (del inglés - Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018) que se basa en el modelado de lenguaje enmascarado (Rives et al., 2021; Ruffolo y Madani, 2024) para aprender información contextual global de las secuencias de entrada. Durante este proceso, se ocultan aleatoriamente algunos *tokens*, y el modelo debe predecir el identificador original de estos *tokens* basándose únicamente en el contexto. En el entrenamiento de ESM-2, se enmascaró el 15% de los

aminoácidos (Z. Lin et al., 2023).

La familia de modelos ESM-2 consta de seis modelos que varían en número de capas (6, 12, 30, 33, 36 y 48), número de parámetros (8 millones, 35 millones, 150 millones, 650 millones, 3 miles de millones y 15 miles de millones), y dimensión de las incrustaciones (320, 480, 640, 1280, 2560 y 5120), respectivamente. La salida de los modelos ESM-2 es una incrustación (*embedding*) que se representa como una matriz de dimensión $N \times M$ por secuencia de péptido, donde N es el número de aminoácidos y M es el tamaño de la incrustación. Una incrustación es una representación numérica de baja dimensionalidad que caracteriza datos pertenecientes a un espacio de alta dimensionalidad, tal que datos contextualmente similares son representados por incrustaciones cercanas en el espacio (Joel Barnard, 2023). La información contenida en cada fila de la incrustación se puede utilizar para caracterizar evolutivamente cada aminoácido de la secuencia. Así, es posible extraer incrustaciones con dimensiones de 320, 480, 640, 1,280, 2,560 y 5,120 para caracterizar los nodos (aminoácidos) en los grafos representando estructuras de péptidos predichas (Cordoves-Delgado y García-Jacas, 2024). Para utilizar los modelos ESM-2, se dispone de la biblioteca fair-esm (https://pypi.org/project/fair-esm/) desarrollada en Python. A continuación, se presenta un *script* de ejemplo que ilustra cómo extraer la incrustación correspondiente para la secuencia peptídica IFSAIAGLLSNLL usando el modelo ESM-2 de 6 capas (esm2_t6_8M_UR50D). La incrustación de salida correspondiente será una matriz de 13×320, donde 13 es el número de aminoácidos de la secuencia y 320 es el tamaño de la incrustación:

```
import torch
import esm

# Cargar el modelo ESM-2.
model, alphabet = esm.pretrained.esm2_t33_650M_UR50D()
batch_converter = alphabet.get_batch_converter()
model.eval() # Desactiva el dropout para obtener resultados determinísticos

# Secuencias peptídicas.
data = [('seq0', 'IFSAIAGLLSNLL')]

# Convertir las secuencias a tokens.
batch_labels, batch_strs, batch_tokens = batch_converter(data)

# Extraer representaciones por residuo (en CPU).
with torch.no_grad():
          results = model(batch_tokens, repr_layers=[33], return_contacts=True)
token_representations = results["representations"][33]

# Generar representaciones por secuencia.
# El token 0 es un token de inicio de secuencia, por lo que el primer residuo es el token 1.
sequence_representations = []
for i, (_, seq) in enumerate(data):
          sequence_representations.append(token_representations[i, 1: len(seq) + 1])
```

2.4.2 Funciones para el cálculo de las distancias inter-aminoácidos

Como se mencionó anteriormente, el cálculo de distancias inter-aminoácidos es un enfoque ampliamente utilizado para establecer relaciones entre aminoácidos (Gligorijević et al., 2021; Abdin et al., 2022; K. Yan et al., 2023; Ge et al., 2024). Este enfoque consiste en utilizar las coordenadas Cartesianas (x, y, z) del átomo de carbono α (o carbono β) de cada aminoácido para calcular la distancia entre pares de aminoácidos utilizando alguna función de distancia, comúnmente la Euclidiana. Una función $d: X \times X \rightarrow \mathbb{R}$ se considera una distancia en X si, $\forall\, x, y \in X$, se cumplen las siguientes propiedades (Deza y Deza, 2016):

1. No negatividad: $d(x, y) \geq 0$
2. Simetría: $d(x, y) = d(y, x)$
3. Reflexividad: $d(x, x) = 0$.

La distancia Euclidiana se considera un caso específico de la definición de Minkowski (Deza y Deza, 2016). Por lo tanto, cualquier función derivada de la definición de Minkowski (*e.g.*, la distancia de Manhattan) y cualquier función que cumpla con las propiedades de no negatividad, simetría y reflexividad para todos los pares de puntos en un espacio $N-$dimensional pueden ser utilizadas como función de distancia. En la Tabla 2 se presentan las funciones de distancias usadas en esta tesis (ver Tabla 2), las cuales se han aplicado con éxito en estudios de similitud molecular y estudios QSAR (Todeschini y Consonni, 2009).

Tabla 2. Funciones de distancia utilizadas para el cálculo de las distancias inter-aminoácidos.

Distancia	Función[a]	Distancia	Función[a]
Euclidiana	$d_{XY} = \sqrt{\sum_{j=1}^{h} \left(x_j - y_j\right)^2}$	Soergel	$d_{XY} = \frac{\sum_{j=1}^{h} \lvert x_j - y_j \rvert}{\sum_{j=1}^{h} max\{x_j,\ y_j\}}$
Canberra	$d_{XY} = \sum_{j=1}^{h} \frac{\lvert x_j - y_j \rvert}{\lvert x_j \rvert + \lvert y_j \rvert}$	Bhattacharyya	$d_{XY} = \sqrt{\sum_{j=1}^{h} \left(\sqrt{x_j} - \sqrt{y_j}\right)^2}$
Lance-Williams	$d_{XY} = \sum_{j=1}^{h} \frac{\sum_{j=1}^{h} \lvert x_j - y_j \rvert}{\sum_{j=1}^{h} \left(\lvert x_j \rvert + \lvert y_j \rvert\right)}$	Separación Angular	$d_{XY} = 1 - \frac{\sum_{j=1}^{h} x_j . y_j}{\sqrt{\sum_{j=1}^{h} {x_j}^2 . \sum_{j=1}^{h} {y_j}^2}}$
Clark	$d_{XY} = \sqrt{\sum_{j=1}^{h} \left(\frac{\lvert x_j - y_j \rvert}{\lvert x_j \rvert + \lvert y_j \rvert}\right)^2}$		

[a]: Las variables x_j y y_j indican el valor de la coordenada j del átomo de carbono $\propto$ correspondiente a los aminoácidos X y Y, respectivamente. El valor de h es igual a 3.

2.5 Aprendizaje profundo de grafos

2.5.1 Redes Neuronales de Grafos

Las Redes Neuronales de Grafos (GNNs, por sus siglas en inglés) son un tipo de modelo de DL que se utiliza para aprender de datos estructurados en forma de grafos (Khemani et al., 2024). Utilizan un mecanismo conocido como paso de mensajes que actualiza iterativamente el vector de características de los nodos agregando los vectores de características de sus vecinos (ver ecuación (3)). La entrada de una capa es el conjunto de características de los nodos $\{x_i \in \mathbb{R}^F \mid i \in V\}$ y el conjunto de aristas E. La capa produce como salida un nuevo conjunto de características para los nodos $\{x_i' \in \mathbb{R}^F \mid i \in V\}$, acorde a su vecindad $\mathcal{N}_i = \{j \in V \mid (i,j) \in E\}$:

$$x_i' = ACTUALIZAR\left(x_i, AGREGAR\left(\{x_j \mid j \in \mathcal{N}_i\}\right)\right) \tag{3}$$

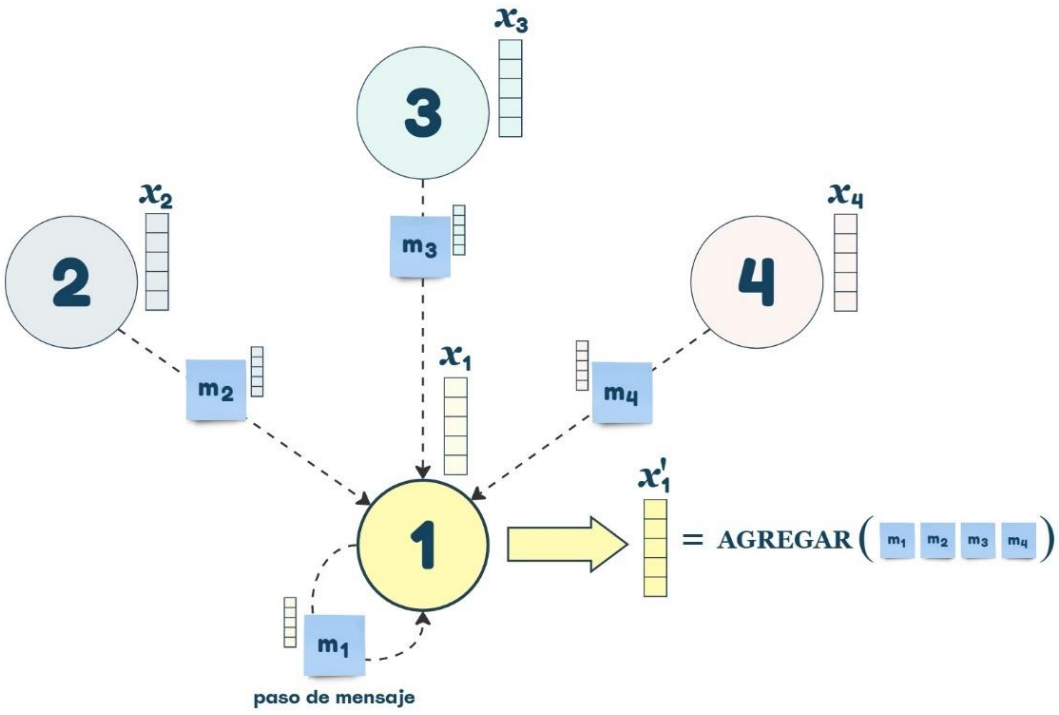

Figura 5. Mecanismo de paso de mensajes en una GNN. El vector de características del nodo 1 se actualiza mediante la agregación de los vectores de características de sus nodos vecinos (*i.e.*, nodo 2, nodo 3 y nodo 4).

En la Figura 5 se muestra un ejemplo del mecanismo de paso de mensajes de un nodo en una GNN. En la

primera iteración, el nodo 1 actualiza su vector de características al agregar la información de sus vecinos inmediatos (nodo 2, nodo 3, y nodo 4). Este proceso ocurre de manera paralela para cada nodo de la red. Así, en la segunda iteración, el nodo 1 incorpora la información que sus vecinos han recibido de sus propios vecinos, lo que permite capturar sucesivamente la información de un vecindario más amplio. Notar que, cuando el número de iteraciones alcanza el diámetro del grafo (*i.e.*, la distancia máxima más corta entre cualquier par de nodos), la información de todos los nodos y sus respectivos vecindarios está capturada en el vector de características final de cada nodo.

Existen diversas variantes de GNN, tales como las Redes Convolucionales de Grafos (GCN, por sus siglas del inglés) (X. M. Zhang et al., 2021), GraphSAGE (acrónimo de Graph SAmple and aggreGatE) (Hamilton et al., 2017), las Redes de Atención de Grafos (GAT) (Veličković et al., 2017; Brody et al., 2021), entre otras (Nikolentzos et al., 2019; Wu et al., 2021; Khemani et al., 2024). Estas se diferencian principalmente en las funciones de actualización y agregación.

2.5.2 Redes de Atención de Grafos

En muchas arquitecturas de GNN, tales como las GCN (Zhang et al., 2021) y GraphSAGE (Hamilton et al., 2017), los vectores de características de todos los vecinos $j \in \mathcal{N}_i$ se agregan con igual importancia, sin considerar la relevancia específica de cada vecino para la actualización de las características del nodo. Para abordar esta limitación, las GATs (Veličković et al., 2017; Brody et al., 2021) incorporan un mecanismo de atención que asigna un coeficiente de atención (o peso) a cada vecino, que indica la importancia de ese vecino j para el nodo i. Para implementar este mecanismo, las GAT (Veličković et al., 2017) calculan un promedio ponderado de las características de $\mathcal{N}_i$. Es decir, se define una puntuación $e_{i,j}$ para cada arista (i, j) que refleja la importancia de las características del vecino j para el nodo i (ver ecuación (4)):

$$e_{i,j} = LeakyReLU\left(a^T\left[Wx_i || Wx_j\right]\right) \tag{4}$$

donde $a \in \mathbb{R}^{2F'}$ es un mecanismo de atención basado en una red neuronal feedforward de una sola capa, ^{T} indica transposición, $W^{F' \times F}$ es una matriz de transformación lineal compartida, y || denota la concatenación de vectores. Leaky ReLU es una función de activación ReLU modificada. Una función ReLU genera una salida igual a cero para entradas negativas y una salida igual a la entrada cuando esta es positiva (Nair y Hinton, 2010). Por su parte, una función Leaky ReLU introduce una pequeña pendiente

para las entradas negativas, permitiendo salidas ligeramente negativas en lugar de cero.

Estas puntuaciones se normalizan entre todos los vecinos $j \in \mathcal{N}_i$ usando la función Softmax (ver ecuación (5))

$$\alpha_{i,j} = sofmax_j(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} exp(e_{ij})} \tag{5}$$

Posteriormente, se calcula un promedio ponderado de las características transformadas de los nodos vecinos (seguido de una función de activación no lineal σ) como la nueva representación de i, utilizando los coeficientes de atención normalizados (ver ecuación (6)):

$$x_i' = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \cdot W x_j\right) \tag{6}$$

Las GAT utilizan atención multi cabeza (*multi-head*) para estabilizar el proceso de aprendizaje (Veličković et al., 2017). Esto permite al modelo captar distintos tipos de relaciones en los datos. Este mecanismo utiliza K mecanismos de atención independientes, cada uno transformando las características de los nodos según la ecuación (6). Cada cabeza de atención aprende una representación de cada nodo, y estas representaciones se concatenan para formar una representación de salida con KF' características por nodo. En la capa final, en lugar de concatenar, se promedian las características, y la aplicación de la función de activación no lineal σ se pospone hasta esta etapa. La agregación en la capa final se define como sigue (ver ecuación (7)):

$$x_i' = \sigma\left(\frac{1}{k}\sum_{k=1}^{K}\sum_{j \in \mathcal{N}_i} {\alpha_{ij}}^{(k)} W^{(k)} x_j\right) \tag{7}$$

En (Brody et al., 2021) se propone una modificación en el orden de las operaciones internas de la arquitectura GAT, dando lugar a una nueva variante denominada GATv2, que introduce un mecanismo de atención más expresivo. En GATv2, se aplica la capa a después de la de no linealidad (LeakyReLU), y la capa W después de la concatenación para calcular la puntuación $e_{i,j}$. La nueva forma de calcular la puntuación

se presenta en la siguiente ecuación (ver ecuación (8)):

$$e_{i,j} = a^T LeakyReLU\left(W\left[x_i || x_j\right]\right) \quad (8)$$

Si el grafo tiene vectores de características asociados a las aristas $\omega_{i,j}$, donde $\omega_{i,j}$ es el vector de características de la arista (i, j), las ecuaciones (4) y (8) correspondientes a la puntuación de atención e_{ij} de GAT y GATv2 respectivamente, se redefinen como (9) y (10):

$$e_{i,j} = LeakyReLU\left(a^T\left[Wx_i \left|\left| Wx_j \right|\right| W\omega_{i,j}\right]\right) \quad (9)$$

$$e_{i,j} = a^T LeakyReLU\left(W\left[x_i \left|\left| x_j \right|\right| \omega_{i,j}\right]\right) \quad (10)$$

2.6 Conclusiones parciales

Los AMPs son cadenas cortas de aminoácidos cuya composición varia significativamente en función de su estructura, actividad, y origen. Para predecir las estructuras 3D de los péptidos, se han propuesto métodos predictivos que pueden clasificarse en dependientes y libres de alineamiento. Los métodos dependientes de alineamiento no son prácticos para protocolos que requieren un alto procesamiento de datos debido a su complejidad computacional. En contraste, los modelos de lenguaje de proteínas, como ESMFold y HelixFold-Single, son una alternativa más eficiente para realizar tales predicciones porque no dependen de MSAs.

La representación de péptidos como grafos es un enfoque usado para modelar las interacciones y relaciones estructurales entre aminoácidos. En estos grafos los nodos representan los aminoácidos y las aristas las relaciones entre ellos. Las aristas se construyen generalmente aplicando un umbral de distancia Euclidiana entre pares de aminoácidos, pero investigaciones en el campo de descriptores moleculares han demostrado que el uso de funciones de distancia distintas de la Euclidiana puede mejorar la capacidad de modelado y discriminación entre moléculas estructuralmente diferentes. Por lo tanto, los algoritmos de aprendizaje de grafo de péptidos pueden beneficiarse de representaciones que no se limiten a la distancia Euclidiana, lo cual puede generar grafos con topologías diversas que codifiquen espacios químicos distintos, potencialmente mejorando los modelos de aprendizaje automático (ML). Asimismo, no se han explorado enfoques basados en otros criterios, tales como fisicoquímicos, geométricos, y topológicos para

establecer las relaciones inter-aminoácidos.

Un aspecto por mejorar en la construcción de modelos basados en grafos es la caracterización de los nodos. Hasta la fecha, los vectores de características de cada nodo se han obtenido utilizando propiedades fisicoquímicas de cada aminoácido, o descriptores a nivel de aminoácido dependientes de MSAs, como las matrices PSSM. Estas matrices no incorporan información sobre las correlaciones entre posiciones, lo que es crucial para capturar la coevolución de aminoácidos en las secuencias. Además, su información está limitada al conjunto de secuencias de las que se derivan. Por lo tanto, estas se consideran obsoletas para transferir información evolutiva en comparación con modelos de última generación como la familia de modelos ESM-2, los cuales serán los usados en este trabajo para caracterizar los nodos.

Para el aprendizaje de representaciones de grafos se han desarrollado arquitecturas GNN que utilizan un mecanismo de paso de mensaje para aprender de datos estructurados en forma de grafos al agregar información de los nodos vecinos. Las GAT, en particular, mejoran este proceso al asignar diferentes pesos a la información de cada vecino, lo que les permite capturar de manera más efectiva las relaciones complejas entre aminoácidos en tareas como la predicción de AMPs.

Capítulo 3. Desarrollo del *framework* esm-AxP-GDL para modelar y predecir péptidos antimicrobianos

En el presente capítulo se presenta el *framework* esm-AxP-GDL para la modelación y predicción de AMPs. En la sección 3.1 se ofrece una descripción general del *framework*. En la sección 3.1 se explica el enfoque de construcción de grafos basado en distancia. En la sección 3.3 se explica el enfoque de construcción de grafos basado en la secuencia de aminoácidos. En la sección 3.4 se aborda la arquitectura de aprendizaje utilizada en el *framework*. En la sección 3.5 se describen varios métodos para validar los modelos. En la sección 3.6 se explica el diseño del *framework*. Finalmente, en la sección 3.7 se detalla cómo instalar el *framework* y utilizarlo en diferentes entornos.

3.1 Descripción general del *framework*

El *framework* esm-AxP-GDL (ver Figura 6) recibe como entrada un archivo de valores separados por comas (CSV) que contiene el identificador, la secuencia de aminoácidos, la actividad (0 y 1 para actividades negativas y positivas, respectivamente), y la partición de datos a la que pertenece cada secuencia de péptido. Se utilizan los números 1, 2 y 3 para representar las particiones de entrenamiento, validación y prueba, respectivamente. La estructura 3D de cada secuencia de péptido en el archivo de entrada se predice con el modelo ESMFold (Z. Lin et al., 2023). Todas las estructuras predichas se guardan en archivos PDB (del inglés - Protein Data Bank). Estos archivos PDB pueden reutilizarse para evitar la fase de ESMFold si se usa el mismo conjunto de datos para otra tarea de modelado. Del mismo modo, se pueden especificar archivos PDB que contengan estructuras 3D predichas por otros métodos (Baek et al., 2021; Du et al., 2021; Jumper et al., 2021; Abramson et al., 2024; Hayes et al., 2024), con el objetivo de aprovechar las otras características del *framework*. Para cada estructura predicha, se calcula la distancia geométrica entre los átomos de carbono α de cada par de aminoácidos utilizando una función de distancia. Si esa distancia es menor o igual a un umbral dado, entonces se establece una relación (arista) entre esos aminoácidos. Las aristas definidas se pueden ponderar con el valor de distancia. Luego de aplicar estos criterios se crea una representación de grafo para cada estructura predicha, donde los nodos representan los aminoácidos y las aristas representan la información estructural (ver sección 3.1). Los nodos de los grafos construidos se caracterizan con información evolutiva utilizando la familia de modelos ESM-2 (ver sección 2.4.1.1). El umbral de distancia y el modelo ESM-2 a usar se especifican como parámetros de entrada. En el Anexo A se describe la lista completa de parámetros del *framework*.

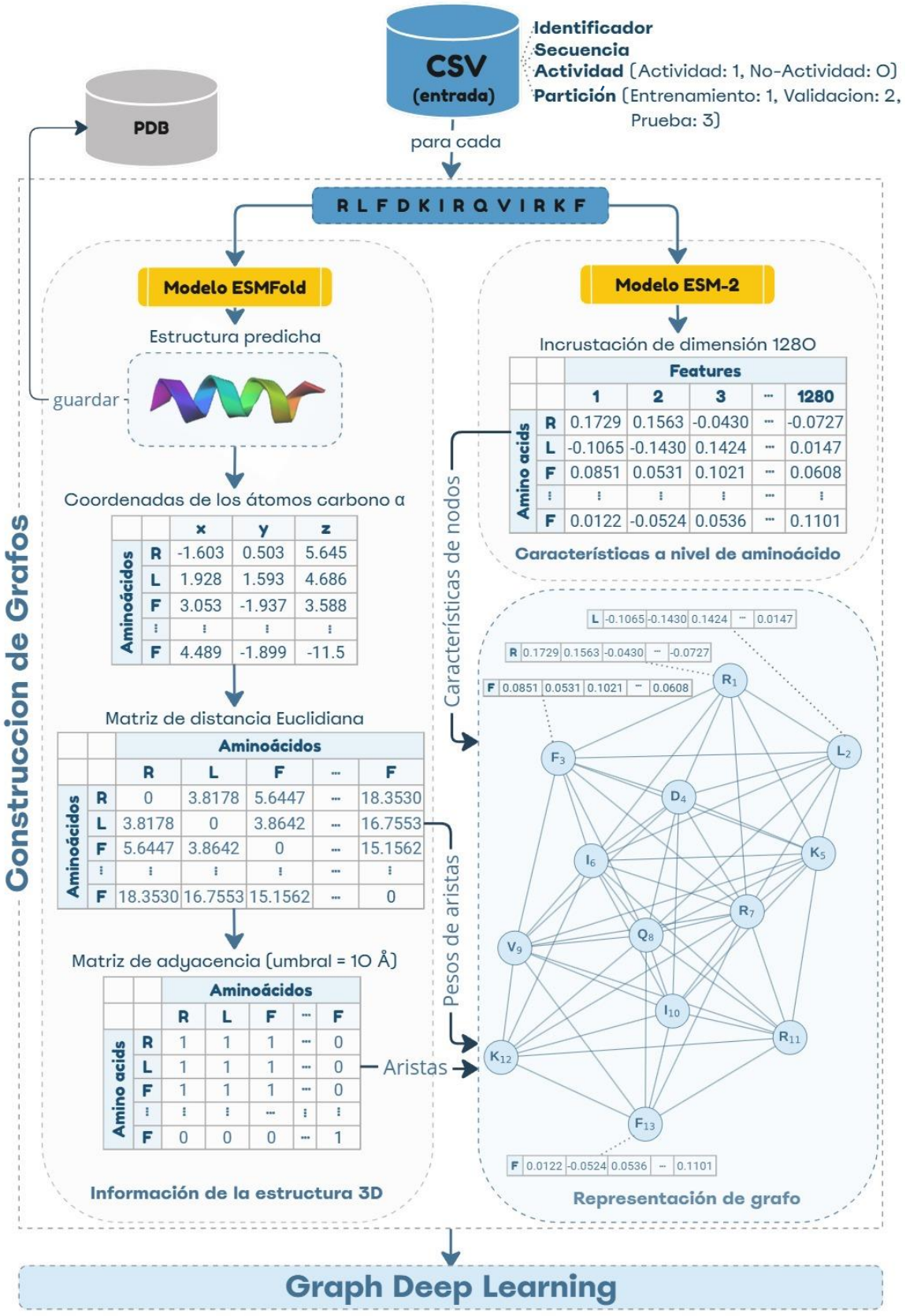

Figura 6. Descripción general del flujo de trabajo implementado en el *framework* esm-AxP-GDL. Se ejemplifica usando la secuencia peptídica 'RLFDKIRQVIRKF', la función de distancia Euclidiana con un umbral igual a 10 Å, y el modelo ESM-2_t33.

Una vez que se crean todos los grafos se seleccionan aquellos que pertenecen a las particiones de entrenamiento y validación para entrenar un modelo basado en GDL. Si no se proporcionan datos de validación en el archivo de entrada, estos se extraen de los datos de entrenamiento mediante una división aleatoria (80% entrenamiento, 20% validación). Los datos de validación se utilizan para evaluar cada modelo entrenado por cada época. Se utilizó la arquitectura GAT (ver sección 3.4) implementada en el protocolo sAMPpred-GAT (ver sección 2.5 en (K. Yan et al., 2023)) porque este trabajo no tiene como propósito introducir una nueva arquitectura de GDL para clasificar AMPs y sus tipos funcionales (u otras actividades de péptidos/proteínas), sino presentar un *framework* para aprovechar tanto la información evolutiva a nivel de aminoácidos obtenida desde los modelos ESM-2, así como el uso del modelo ESMFold para predecir la estructura 3D de AMPs. Por lo tanto, se puede implementar cualquier otra arquitectura de GDL en lugar de utilizar siempre la arquitectura GAT proporcionada. Los péptidos del conjunto de prueba se usan únicamente al ejecutar el paso de predicción.

El proceso de entrenamiento se realiza durante un número específico de épocas. Para cada época, se calcula el valor de pérdida basado en la entropía cruzada en los conjuntos de entrenamiento y validación, respectivamente. La exactitud (ACC), el coeficiente de correlación de Matthew (MCC), el área bajo la curva (AUC), y la métrica de *recall* para cada clase se calculan en los conjuntos de validación y prueba, respectivamente (ver Anexo B). El *framework* esm-AxP-GDL está disponible en el repositorio https://github.com/cicese-biocom/esm-AxP-GDL.

3.2 Construcción de grafos basados en distancia

Como se mencionó anteriormente, para cada estructura predicha se genera un grafo simple que puede ser ponderado o no ponderado, donde los nodos representan los aminoácidos y las aristas se definen de acuerdo con un umbral de distancia entre los átomos de carbono α de cada par de aminoácidos. Los grafos se denotan como $G = (V, X_v, A, \omega)$. Aquí V es el conjunto de nodos que representa los aminoácidos de la secuencia, donde $N = |V|$. X_v es el conjunto de vectores de características de los nodos (ver sección 2.4.1). A es una matriz de adyacencia de NxN (Ver sección 2.3.1), donde cada elemento $A_{i,j}$ se define según la ecuación (11). ω es una matriz de pesos de NxN (Ver sección 2.3.1), cuyos coeficientes $\omega_{i,j}$ se definen según la ecuación (12). En el caso de grafos no ponderados, ω es una matriz vacía ($\omega = \emptyset$).

$$A_{i,j} = \begin{cases} 1, & si\ d_{i,j} \leq d_{th} \\ 0, & en\ otro\ caso \end{cases} \tag{11}$$

$$\omega_{i,j} = \begin{cases} d_{i,j}, & si\ \ A_{ij} = 1 \\ 0, & en\ otro\ caso \end{cases} \tag{12}$$

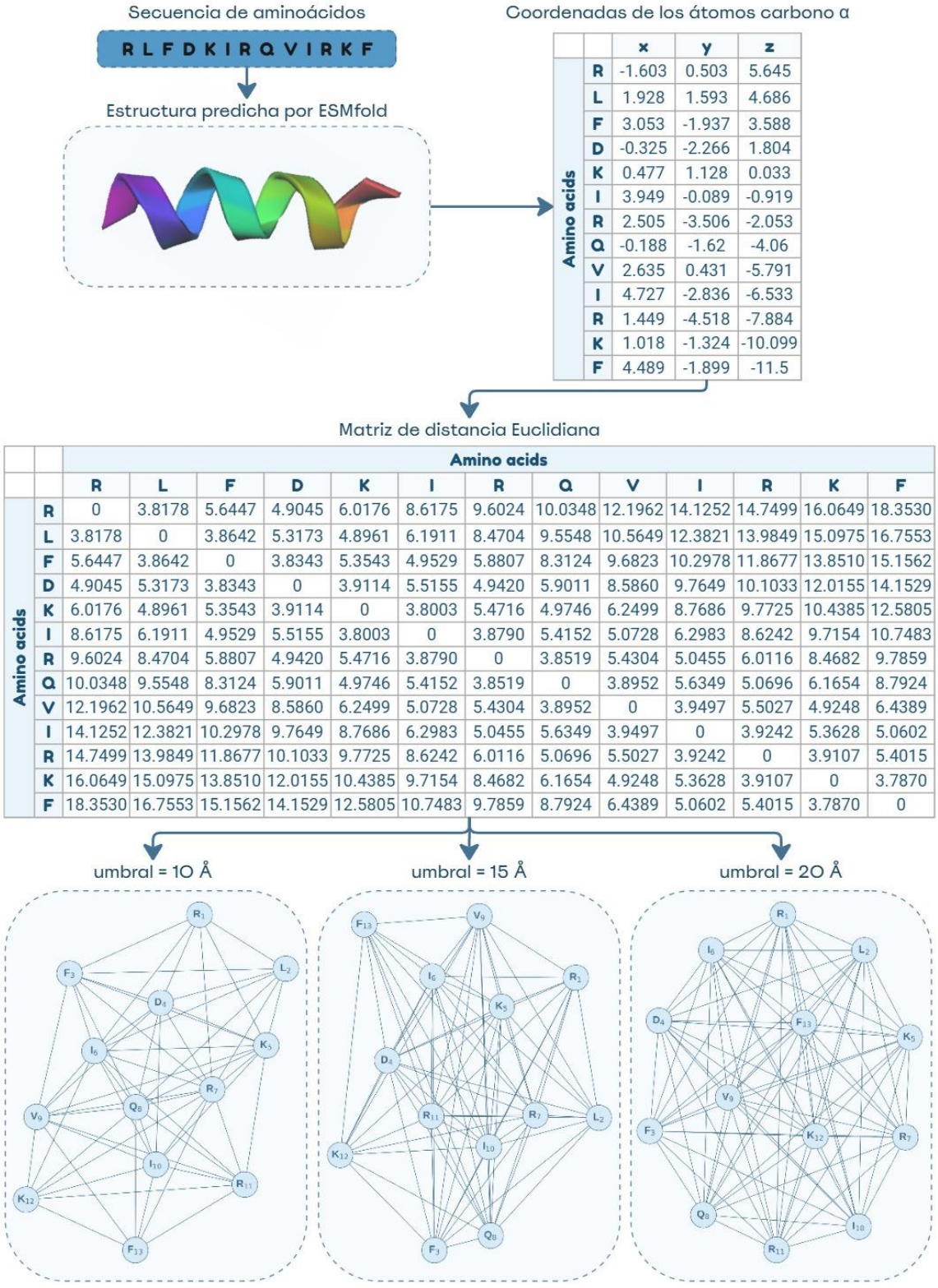

Figura 7. Ejemplos de grafos no dirigidos y no ponderados construidos al aplicar tres umbrales de distancia Euclidiana.

En las ecuaciones (11) y (12), i y j representan dos aminoácidos de la secuencia, $d_{i,j}$ es la distancia (ver sección 2.4.2) entre los aminoácidos i y j, y d_{th} es un umbral de distancia predefinido. Para construir las representaciones de grafos basadas en distancia se debe especificar el valor 'distance_based_threshold' en el parámetro 'edge_construction_functions'. La Figura 7 muestra ejemplos de grafos creados utilizando la distancia Euclidiana con tres umbrales de distancia diferentes.

3.3 Construcción de grafos basados en la secuencia de aminoácidos

Los grafos basados en la secuencia de aminoácidos solo codifican relaciones entre aminoácidos adyacentes en la secuencia. A pesar de esto, estos grafos pueden utilizarse para construir modelos de GDL que no dependen de la estructura 3D. En la construcción de estos grafos, para cada secuencia de aminoácidos se genera un grafo simple dirigido, donde las aristas se establecen entre cada par de aminoácidos adyacente en la secuencia. Los grafos se denotan como $G = (V, X_v, A, \omega)$. Aquí, V es el conjunto de nodos que representa los aminoácidos de la secuencia, donde $N = |V|$. X_v es el conjunto de vectores de características de los nodos (ver sección 2.4.1). A es una matriz de adyacencia de NxN (Ver sección 2.3.1), donde cada elemento $A_{i,j}$ se define según la ecuación (13). ω es una matriz de pesos de NxN (Ver sección 2.3.1), cuyos coeficientes $\omega_{i,j}$ se definen según la ecuación (14). En el caso de grafos no ponderados, ω es una matriz vacía ($\omega = \emptyset$). En las ecuaciones (13) y (14), i y j representan aminoácidos de la secuencia, $d_{i,j}$ es la distancia entre los aminoácidos i y j (ver sección 2.4.2), y d_{th} es un umbral de distancia predefinido. En esta tesis no se experimentó con representaciones de grafos basadas en la secuencia de aminoácidos. Para usar las representaciones de grafos basadas en secuencia se especifica el valor 'sequence_based' en el parámetro 'edge_construction_functions'.

$$A_{i,j} = \begin{cases} 1, & si\ i\ es\ adyacente\ a\ j \\ 0, & en\ otro\ caso \end{cases} \tag{13}$$

$$\omega_{i,j} = \begin{cases} d_{i,j}, & si\ A_{i,j} = 1 \\ 0, & en\ otro\ caso \end{cases} \tag{14}$$

3.4 Descripción de la arquitectura de aprendizaje del *framework*

El *framework* utiliza la arquitectura de aprendizaje implementada en el protocolo sAMPpred-GAT (ver sección 2.5 en (Yan et al., 2023)), la cual está basada en GAT (ver sección 2.5.2). Esta arquitectura recibe

como entrada las representaciones de grafos correspondientes a cada péptido, donde cada grafo se denota como $G = (X_v, E, X_e)$. Aquí $X_v = \{x_1, x_2, \ldots, x_N\}$ es el conjunto de vectores de características de todos los nodos, donde $x_i \in \mathbb{R}^F$. E es el conjunto de aristas (ver sección 2.3.2). $X_e = \{\omega_{i,j} \in \mathbb{R}^{F_e} \mid (i,j) \in E\}$ es el conjunto de vectores de características de todas las aristas, donde $\omega_{i,j}$ denota el vector de características de la arista (i,j). F y F_e son el número de características de cada nodo y de cada arista, respectivamente. En el caso de grafos no ponderados, X_e es un conjunto vacío ($X_e = |\emptyset|$).

La arquitectura consta de tres módulos principales: (i) las capas de convolución de grafos (ver sección 3.4.1), (ii) la capa de agrupamiento de los k mejores (*top k pooling*) (ver sección 3.4.2), y (iii) las capas lineales (ver sección 3.4.3). Las capas de convolución de grafos se utilizan para aprender información discriminativa de las relaciones inter-aminoácidos de cada grafo representando una estructura peptídica. La capa de agrupamiento de los k mejores se utiliza para seleccionar de forma adaptativa los k nodos más importantes de un grafo y generar así una representación compacta. Por último, las capas lineales usan esta representación más compacta para predecir si un péptido es AMP o no. Para mejorar el aprendizaje y la generalización en los diferentes módulos, se aplica una función de activación ReLU. ReLU genera una salida igual a cero para entradas negativas y una salida igual a la entrada cuando esta es positiva (Nair y Hinton, 2010). Además, se usa la técnica de *dropout* (Srivastava et al., 2014) para aleatoriamente ignorar algunas salidas y así evitar el sobreajuste. Se utiliza la función de pérdida negative log-likelihood para medir el error entre las etiquetas predichas y las etiquetas verdaderas. El algoritmo Adam (Kingma y Ba, 2014) se usa para optimizar el modelo durante el proceso de entrenamiento. En la Figura 8 se muestra la arquitectura de la red neuronal usada. Esta se implementó utilizando PyTorch (https://pytorch.org/) y PyTorch Geometric (Fey y Lenssen, 2019).

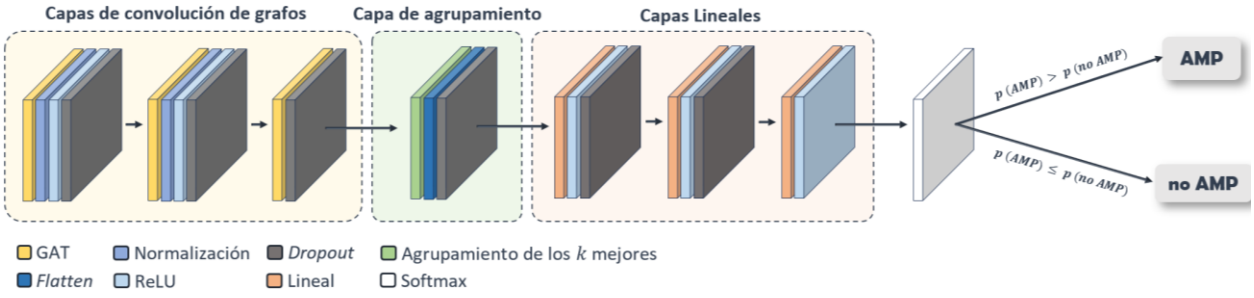

Figura 8. Arquitectura de la red profunda de grafos usada en el *framework*. Se utilizan tres capas GAT con atención multi-cabeza para aprender las relaciones inter-aminoácidos en cada péptido representando como un grafo. Luego, se utiliza una capa de agrupamiento para generar una representación compacta de cada grafo. Posteriormente, las capas lineales permiten aprender una representación más discriminativa. Finalmente, después de la última capa lineal, se aplica una función Softmax para realizar la predicción final. $\boldsymbol{p}(.)$ denota la probabilidad asignada a cada clase.

3.4.1 Capas de convolución de grafos

En este módulo se utiliza una arquitectura GAT (Veličković et al., 2017) de tres capas utilizando atención multi-cabeza. Cada capa GAT actualiza las características de los nodos mediante un mecanismo de atención que pondera la importancia de los nodos vecinos (ver sección 2.5.2). Se usan una función de activación ReLU (Nair y Hinton, 2010) y una función de *dropout* (Srivastava et al., 2014) después de cada capa GAT, excepto en la última capa, para mitigar el problema de descenso del gradiente y como mecanismo de regularización, respectivamente. La formulación de las L capas GAT se presenta a continuación (K. Yan et al., 2023):

Sea $x_i^{(l)} \in \mathbb{R}^{F_l}$ la representación del nodo i con una dimensión F_l en la capa l $(1 \leq l \leq L)$. $x_i^{(0)} = x_i \in \mathbb{R}^F$ es el vector de características del nodo i (ver sección 2.4.1). x_i^L es la característica del nodo i después de la última capa GAT. GAT actualiza la representación del nodo de forma iterativa mediante la ecuación (15).

$$x_i^{(l)} = \sum_{u \in N_i \cup \{i\}} \propto_{ij}^{(l)} W^{(l)} x_i^{(l-1)} \tag{15}$$

donde $W^{(l)} \in \mathbb{R}^{F_l \times F_{l-1}}$ es la matriz de transformación de proyección en la capa l, y $\mathcal{N}_i$ es el vecindario del nodo i (ver sección 2.3). Los coeficientes de atención $\propto_{ij}^{(l)}$ se calculan mediante la ecuación siguiente (ver ecuación (16)):

$$\propto_{ij}^{(l)} = \frac{\exp\left(g\left(a^{(l)\,T}\left[W^{(l)} x_i^{(l-1)} || W^{(l)} x_j^{(l-1)}\right]\right)\right)}{\sum_{k \in N_i \cup \{i\}} \exp\left(g\left(a^{(l)\,T}\left[W^{(l)} x_i^{(t-1)} || W^{(l)} x_j^{(l-1)}\right]\right)\right)} \tag{16}$$

donde $g(\cdot)$ es una función de activación LeakyReLU, $a^{(l)} \in R^{2F_l}$ es un vector de parámetros aprendibles, y || representa la concatenación. $\propto_{ij}^{(l)}$ puede ser redefinido según las ecuaciones (8), (9) y (10) para grafos no ponderados con GATv2, grafos ponderados con GAT y grafos ponderados con GATv2, respectivamente.

3.4.2 Capa de agrupamiento de los k mejores nodos

La capa de agrupamiento de los k mejores selecciona de forma adaptativa un subconjunto de nodos para

generar una representación de grafo reducida (con menos nodos) que conserva la mayor cantidad posible de información del grafo original (H. Gao y Ji, 2019). Para lograr esto, se calcula para cada nodo i una puntuación de proyección escalar s_i, que mide cuánta información del nodo i se conserva al proyectar su vector de características x_i sobre la dirección de un vector aprendido p. Posteriormente, se calcula la nueva representación x_i' basándose en los k nodos con las puntuaciones más altas, donde $k \in (0, 1]$. Dado que la selección se basa en la proyección unidimensional (1D) del vector de características de cada nodo, la conectividad en el nuevo grafo se mantiene consistente entre los nodos seleccionados (Gao y Ji, 2019). A continuación, se detalla el proceso de cálculo (ver ecuaciones (17), (18) y (22)):

$$s_i = \frac{x_i^T p}{||p||_2}, i \in V \tag{17}$$

$$idx = top_k(s) \tag{18}$$

$$x_i' = x_t \odot tanh(s_t), \qquad t \in idx \tag{19}$$

donde, $s = \{s_i \in \mathbb{R} \mid i \in V\}$ es el conjunto de puntuaciones de proyección. $top_k(\cdot)$ se utiliza para seleccionar los k nodos más informativos y devuelve su conjunto de índices idx. $tanh(s_t)$ es la función no lineal tangente hiperbólica, $\odot$ es el producto elemento por elemento, y $||\cdot||_2$ es la norma L_2 (o Euclidiana) del vector p.

Finalmente, la representación del grafo z con los k mejores nodos, se obtiene linealmente concatenando las nuevas representaciones x_i' de los nodos seleccionados (ver ecuación (20)):

$$z = ||_{t \in idx} x_t' \tag{20}$$

3.4.3 Capas lineales

Varias capas de salida se utilizan para predecir si la secuencia de entrada es un AMP o no AMP. Para ello, se usan capas lineales (Goodfellow et al., 2016) que permiten aprender una representación más discriminativa. La operación realizada por cada capa lineal se describe mediante la ecuación siguiente (ver ecuación (21)):

$$x^{(l)} = W_o^{(l)} x^{(l-1)} + b^{(l)} \tag{21}$$

donde, $x^{(l-1)}$ y $x^{(l)}$ son el vector de entrada y el vector de salida de la l -ésima capa lineal, respectivamente. $x^{(0)} = flatten(z)$. $W_o^{(l)}$ es la matriz de pesos, y $b^{(t)}$ es el sesgo (*bias*) de la l-ésima capa lineal. Se aplica la función ReLU y *dropout* después de cada capa lineal, excepto en la última.

Después de la última capa, se utiliza una función Softmax (Goodfellow et al., 2016) para realizar la predicción final (ver ecuación (22)).

$$y_i = \frac{exp\left(x_i{}^{(M)}\right)}{\sum_{j \in c} exp\left(x_j{}^{(M)}\right)} \tag{22}$$

donde $c = \{0, 1\}$ denota etiqueta binaria, con 1 para la clase AMP y 0 para la clase no AMP. $y_i \in \{y_0, y_1\}$ denota la probabilidad asignada a cada clase. M es el número total de capas lineales.

3.5 Métodos para validar los modelos

Para comprobar el rendimiento de los modelos basados en distancia (ver sección 3.1), y analizar cuánto depende este rendimiento de la información derivada de la estructura 3D y de las incrustaciones ESM-2, se implementaron varios métodos de validación que se describen a continuación.

3.5.1 Perturbación aleatoria en las coordenadas geométricas

Este método consiste en introducir una perturbación aleatoria en las coordenadas geométricas de los átomos de carbono α utilizadas para construir los grafos basados en distancia (ver sección 3.1). Con este propósito, se reemplaza aleatoriamente un porcentaje determinado de las coordenadas geométricas (x, y, z) de los átomos de carbono α originales por coordenadas aleatorias. Estas coordenadas aleatorias se generan dentro del intervalo de las coordenadas geométricas predichas para las secuencias de entrada siguiendo una distribución uniforme. Es decir, para las coordenadas x se generan números aleatorios dentro del intervalo de las coordenadas x de todas las estructuras predichas. Esta misma lógica se aplica para las coordenadas y y z. Al perturbar la matriz de coordenadas geométricas se altera la relación original

entre la información de la estructura 3D de las secuencias de péptidos y la actividad biológica correspondiente. Esto debe conllevar a obtener modelos con un pobre rendimiento, especialmente cuando se introduce un alto porcentaje de aleatoriedad. A continuación, se presenta el pseudocódigo que describe este proceso.

```
RANDOM-COORDINATE-MATRIX (matrix, randomness_percentage, min, max)
num_rows = número de filas en matrix
num_to_scramble = int (num_rows * (randomness_percentage / 100))
indices = seleccionar num_to_scramble índices aleatorios únicos en el intervalo [0,
   num_rows-1]
atom_coordinates = matrix
coord_0 = generar num_to_scramble valores aleatorios en el intervalo [min[0], max[0]]
coord_1 = generar num_to_scramble valores aleatorios en el intervalo [min[1], max[1]]
coord_2 = generar num_to_scramble valores aleatorios en el intervalo [min[2], max[2]]
para cada (coord_idx, matrix_idx) en enumerate(indices)
    atom_coordinates[matrix_idx][0] = coord_0[coord_idx]
    atom_coordinates[matrix_idx][1] = coord_1[coord_idx]
    atom_coordinates[matrix_idx][2] = coord_2[coord_idx]
return atom_coordinates
```

Para utilizar este método de validación se debe especificar el valor 'random_coordinates' en el parámetro 'validation_mode'. Además, se debe indicar en el parámetro 'randomness_percentage' el porcentaje de filas que se generarán aleatoriamente.

3.5.2 Perturbación aleatoria en las incrustaciones ESM-2

Este método aplica una perturbación aleatoria en las incrustaciones ESM-2 utilizadas para caracterizar los nodos de los grafos (ver sección 3.1). Estas incrustaciones ESM-2 codifican información evolutiva para cada aminoácido (filas de la matriz) en las secuencias peptídicas (ver sección 2.4.1.1). Para introducir la aleatoriedad, se reemplaza un porcentaje de los valores originales en cada fila por los valores generados aleatoriamente en el intervalo de valores de las incrustaciones originales. Los valores aleatorios siguen una distribución uniforme. Al perturbar las incrustaciones originales, se modifica la relación original entre la información evolutiva de las secuencias de péptidos y la actividad biológica correspondiente. En cuanto mayor sea el porcentaje de perturbación de las incrustaciones, peor debe ser el rendimiento de los modelos que se construyen. A continuación, se presenta el pseudocódigo que describe este proceso.

```
RANDOM-NODE-FEATURES (feature_matrix, randomness_percentage, min, max)
1 total_coefficients = número de columnas en feature_matrix
2 total_coefficients_to_build = int (total_coefficients * (randomness_percentage / 100))
3 idxs_to_replace = seleccionar total_coefficients_to_build índices aleatorios únicos en el
intervalo [0, total_coefficients-1]
4 para cada feature_vector en feature_matrix
5     vals_to_replace = generar total_coefficients_to_build valores aleatorios en el
      intervalo [min, max]
6     reemplazar valores en feature_vector[idxs_to_replace] con vals_to_replace
7 return feature_matrix
```

Para utilizar este método de validación se debe especificar el valor 'random_embeddings' en el parámetro 'validation_mode'. Además, se debe indicar en el parámetro 'randomness_percentage' el porcentaje de filas que se generarán aleatoriamente.

3.5.3 Construcción de grafos basados en mapas de contacto ESM-2

Este método consiste en generar modelos utilizando grafos basados mapas de contacto ESM-2. Un mapa de contacto ESM-2 es una matriz simétrica cuyos coeficientes indican la probabilidad (P) de que dos aminoácidos (i, j) estén en contacto en la estructura 3D (ver sección A2.1 del Material Suplementario de (Z. Lin et al., 2023)). Estos mapas son cruciales para la predicción de estructuras 3D (Lin et al., 2023), y constituyen una buena representación para generar modelos para predecir AMPs. Sin embargo, se espera que los modelos derivados de representaciones de grafos basadas en distancias tengan un rendimiento superior porque aprovechan información directa derivada de la estructura 3D de los péptidos. A diferencia de los dos métodos de validación antes explicados, los grafos basados en mapas de contacto ESM-2 se pueden utilizar para crear modelos independientes de la estructura 3D de los péptidos.

Con este propósito, para cada secuencia de aminoácidos se genera un grafo simple, donde las aristas se establecen de acuerdo con un umbral de probabilidad entre cada par de aminoácidos. Los grafos se denotan como $G = (V, X_v, A, \omega)$. Aquí V es el conjunto de nodos que representa los aminoácidos de la secuencia, donde $N = |V|$. X_v es el conjunto de vectores de características de los nodos (ver sección 2.4.1.1). A es una matriz de adyacencia de NxN (ver sección 2.3.1), donde cada elemento $A_{i,j}$ se define según la ecuación (23). ω es una matriz de pesos de NxN (Ver sección 2.3.1), cuyos elementos $\omega_{i,j}$ se definen según la ecuación (24). En el caso de grafos no ponderados, ω es una matriz vacía ($\omega = \emptyset$).

$$A\_\{i,j\} = \begin{cases} 1, & si\ p_{i,j} \leq p_{th} \\ 0, & en\ otro\ caso \end{cases} \quad (23)$$

$$\omega_{i,j} = \begin{cases} p_{i,j}, & si\ A_{i,j} = 1 \\ 0, & en\ otro\ caso \end{cases} \quad (24)$$

donde, i y j representan a dos aminoácidos de la secuencia, $p_{i,j}$ es la probabilidad de que los aminoácidos i y j estén en contacto, y p_{th} es un umbral de probabilidad predefinido. Para utilizar representaciones de grafos basadas en mapas de contacto ESM-2 se debe especificar el valor 'esm2_contact_map_X' en el parámetro 'edge_construction_functions', donde X puede tomar los valores 50, 60, 70, 80 y 90. Estos valores corresponden a umbrales de probabilidad de 0.5, 0.6, 0.7, 0.8 y 0.9, respectivamente.

3.6 Diseño del *framework* esm-AxP-GDL

El flujo de trabajo del *framework* esm-AxP-GDL se implementó utilizando el patrón de diseño Template Method (Freeman y Robson, 2021). Este patrón define el esqueleto de un algoritmo en una clase abstracta y delega a las subclases la implementación concreta de todos o algunos pasos del algoritmo. La Figura 9 muestra el diagrama UML de las clases responsables de implementar el flujo de trabajo con el patrón Template Method. La clase *GDLWorkflow* es la clase base que contiene el método run_workflow, el cual actúa como plantilla del algoritmo encargado de orquestar la ejecución de los modos de entrenamiento, prueba e inferencia. Las subclases *TrainingWorkflow*, *TestWorkflow* e *InferenceWorkflow* implementan los pasos específicos del mismo. El resto de los métodos definidos en *GDLWorkflow* se ejecutan en ese mismo orden en el método plantilla. En la Tabla 3, se describe brevemente cada paso del método plantilla.

La creación de directorios se implementó utilizando el patrón Factory Method (Freeman y Robson, 2021). Se creó la clase abstracta *PathCreator* con el método create_path. Las subclases *TrainingModePathCreator*, *TestModePathCreator* e *InferenceModePathCreator* implementan este método para crear los directorios específicos para los modos de entrenamiento, prueba e inferencia, respectivamente. La carga de datos se implementó también usando el patrón de diseño Factory Method (Freeman y Robson, 2021). En este caso, se definió la clase abstracta *DataLoader* con el método read_file. La subclase *CSVLoader* implementa este método para leer archivos CSV de entrada. Este diseño facilita la extensibilidad del *framework* para soportar otros tipos de archivos en el futuro, por ejemplo, archivos FASTA.

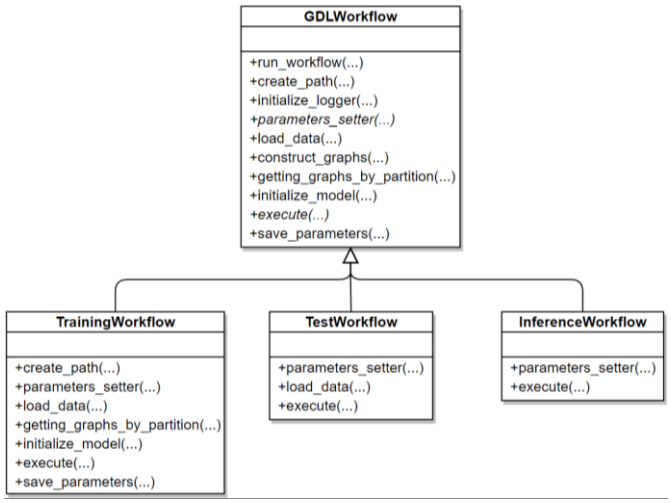

Figura 9. Diagrama UML de las clases responsables de implementar el flujo de trabajo del *framework* con el patrón Template Method.

Tabla 3. Descripción de los pasos del método plantilla run_workflow.

Método	Descripción
create_path	Crea el directorio donde se salvarán los archivos generados durante la ejecución del *framework*.
initialize_logger	Inicializa el archivo de registro (log), en el cual se registran los eventos ocurridos durante la ejecución del *framework*.
parameters_setter	Valida los argumentos de entrada y ajusta los parámetros necesarios para la ejecución del *framework*.
load_data	Carga y valida los conjuntos de datos de entrada.
construct_graphs	Genera las representaciones de grafos de las secuencias del conjunto de datos de entrada.
getting_graphs_by_partition	Recupera las representaciones de datos según su partición.
initialize_Modelo	Inicializa el modelo GDL según la tarea especifica a realizar.
execute	Ejecuta el proceso de entrenamiento, prueba o inferencia del modelo.
save_parameters	Salva los parámetros utilizados en la ejecución del *framework*.

La validación de los conjuntos de datos se implementó utilizando el patrón de diseño Template Method (Freeman y Robson, 2021). En este caso, la clase *DatasetValidator* es la clase base que contiene el método processing_dataset, que actúa como plantilla del algoritmo encargado de validar los conjuntos de datos etiquetados y sin etiquetar. La subclase *LabeledDatasetValidator* implementa la validación cuando se usa datos etiquetados. Las validaciones incluyen la detección de secuencias con aminoácidos no naturales, identificadores de secuencias o secuencias duplicadas, actividad o partición incorrecta, y secuencias sin su correspondiente estructura 3D en formato PDB (si se requiere su lectura).

La construcción de aristas de los grafos se implementó utilizando el patrón de diseño Decorator (Freeman y Robson, 2021) Este patrón permite dinámicamente envolver un objeto dentro de otro para obtener una única salida a partir de la combinación de la salida de cada uno de los objetos. De esta forma se evita crear subclases para cada combinación de funcionalidades. De este modo, se pueden establecer las aristas de los grafos a partir de diferentes criterios (*e.g.*, geométricos, fisicoquímicos) sin necesidad de definir una clase para cada posible combinación. Esto mejora la extensibilidad del *framework* para usar criterios distintos a los basados en distancias. En la Figura 10 se muestra un ejemplo de construcción de aristas utilizando el patrón de diseño Decorator y aplicando tres criterios diferentes: basado de distancia (ver sección 3.1), basado en mapas de contacto ESM-2 (ver sección 3.5.2), y basado en la secuencia de aminoácidos (ver sección 3.3). Como se puede observar, el patrón Decorator envuelve cada criterio dentro del siguiente, permitiendo que las aristas generadas (*i.e.*, la matriz de adyacencia) por un criterio se adicionen a las aristas creadas por el criterio anterior. De esta forma, cada decorador añade su propio conjunto de aristas, construyendo gradualmente un grafo que incorpora todos los criterios especificados.

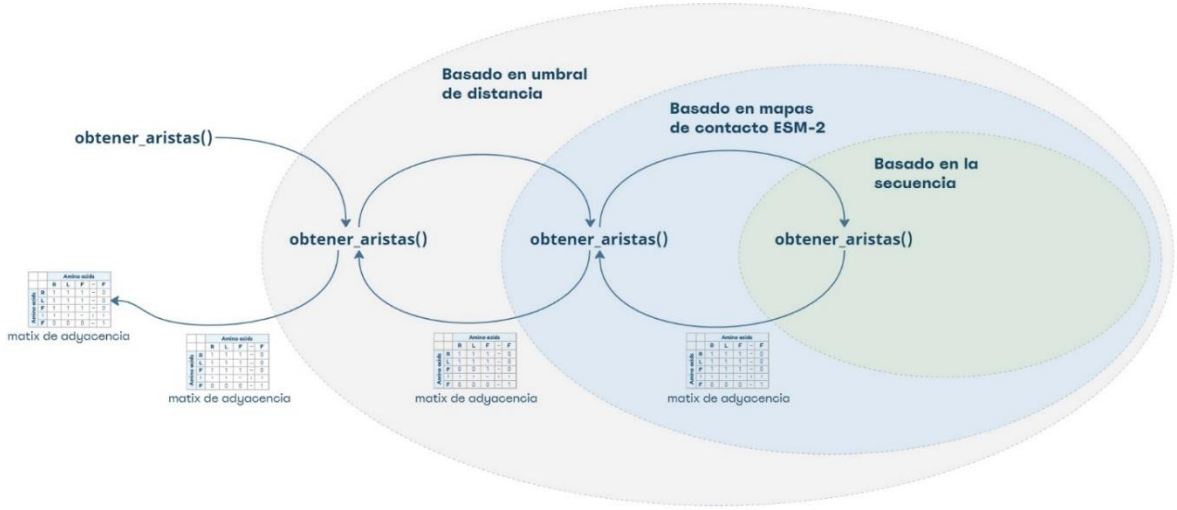

Figura 10. Representación del proceso de construcción de aristas usando el patrón de diseño Decorator con tres criterios diferentes: umbral de distancia, mapas de contacto ESM-2 y secuencia de aminoácidos.

La Figura 11 muestra el diagrama UML de las clases responsables para la construcción de aristas. La clase abstracta *EdgesDecorator* modela un criterio de construcción de arista genérico, mientras que las clases concretas *DistanceBasedThresholdDecorator, ESM2ContactMapDecorator,* y *SequenceBasedDecorator* representan los criterios de construcción de aristas a partir de distancias (ver sección 3.1), mapas de contacto ESM-2 (ver sección 3.5.2), y la secuencia de aminoácidos (ver sección 3.3), respectivamente.

Para manejar las dependencias de las clases se utilizó el patrón de diseño Dependency Injection. Este

patrón facilita el desacoplamiento de las clases al permitir que sus dependencias sean inyectadas desde el exterior, en lugar de ser instanciadas (creadas) internamente por las propias clases. El acoplamiento se refiere al grado de dependencia entre los módulos de un software. Un bajo acoplamiento es deseable porque permite que los componentes del sistema sean modificados o reemplazados sin significativamente afectar a otros componentes. Para lograr esto, se definió la clase *ApplicationContext,* la cual se encarga de inyectar las dependencias necesarias según el modo de ejecución del *framework* (entrenamiento, validación o inferencia).

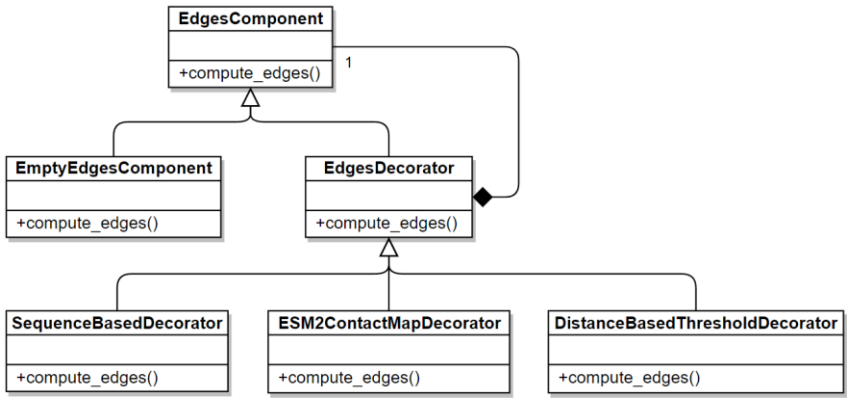

Figura 11. Diagrama UML de las clases responsables para la construcción de aristas.

Además del desacoplamiento, se busca mantener una alta cohesión en el diseño. La cohesión mide qué tan bien un módulo o clase se ajusta a un único propósito o responsabilidad. Una alta cohesión indica que una clase o módulo está diseñado en torno a un conjunto relacionado de funciones, lo cual facilita su mantenimiento y comprensión. En este sentido, el diseño de las clases del *framework* asegura que cada componente tenga una función claramente definida y estrechamente alineada con sus responsabilidades específicas, contribuyendo a la mantenibilidad y escalabilidad de este.

3.7 Instalación y uso del *framework* esm-AxP-GDL

Para la instalación del *framework* esm-AxP-GDL se requiere primeramente descargar (o clonar) el mismo desde el repositorio https://github.com/cicese-biocom/esm-AxP-GDL. Posteriormente, se deben instalar todas sus dependencias. El Anexo C muestra la estructura de directorios del *framework*, así como una breve descripción de los archivos. El *framework* es compatible exclusivamente con sistemas operativos Linux y tiene las siguientes dependencias de softwares: CUDA Toolkit (versión 11.3), Compilador de C++ (versión 9.2), y Python (versión 3.7 a 3.9), así como las bibliotecas de Python especificadas en el archivo

environment.yml. Este archivo se encuentra disponible en el directorio de archivos del *framework*.

El *framework* se ejecuta desde línea de comandos utilizando alguno de los siguientes scripts de Python: train.py, test.py, o inference.py. Estos scripts permiten entrenar, evaluar, o usar para inferencia un modelo, respectivamente. Los detalles sobre los argumentos de entrada se encuentran especificados en el Anexo A. En el directorio del *framework* se proporcionan los scripts de ejemplo train.sh, test.sh, e inference.sh para ejecutar los modos entrenamiento, prueba, e inferencia, respectivamente. En las subsecciones 3.7.1, 3.7.2, y 3.7.3 se dan detalles específicos sobre cómo usar el *framework* en un entorno local, con contenedores Docker, o en arquitecturas de supercómputo.

3.7.1 Uso del *framework* en un entorno local

Para utilizar el *framework* en un entorno local se recomienda utilizar un entorno Conda para gestionar las dependencias de Python. Para este fin, es necesario instalar previamente Ananconda o Miniconda. A continuación, se detallan los pasos necesarios para instalar las dependencias utilizando un entorno Conda a partir del archivo environment.yml:

1. Instalar las herramientas: CUDA Toolkit 11.3, Compilador de C++ 9.2 y Anaconda o Miniconda.
2. Crear el entorno Conda en la línea de comandos (terminal) con el siguiente comando: `conda env create -f environment.yml`.
3. Activar el entorno Conda en la línea de comandos (terminal) con el siguiente comando: `conda env create -f environment.yml`.
4. Verificar que el entorno Conda se activó correctamente en la línea de comandos (terminal) con el siguiente comando: `conda env list`.

3.7.2 Uso del *framework* con contenedores Docker

Se diseñó una imagen Docker para el desarrollo y uso del *framework*. Esta imagen empaqueta todas las dependencias necesarias. En el directorio del *framework* se encuentran disponibles el archivo Dockerfile con las instrucciones para crear la imagen, así como el archivo docker-compose.yml con la configuración del contenedor. La Figura 12 muestra la arquitectura del *framework* usando contenerización basada en

Docker. A continuación, se detallan los pasos para instalar las dependencias utilizando un entorno Docker:

1. Instalar Docker según su guía de instalación, disponible en: https://docs.docker.com/engine/installation/.
2. Instalar CUDA Toolkit 11.3.
3. Construir la imagen Docker localmente en la línea de comandos (terminal) con el siguiente comando: `docker-compose build`. La línea de comandos debe estar en el directorio que contiene el archivo `docker-compose.yml`.

Para ejecutar el *framework* utilizando Docker se debe configurar los scripts (sh) correspondientes a los modos de entrenamiento, prueba, e inferencia. Para ejecutar estos scripts se usa el comando `docker-compose run --rm nombre-del-servicio sh nombre-del-script.sh` en la línea de comandos. Este comando inicializa un contenedor Docker basado en la configuración del servicio especificado en el archivo docker-compose.yml. La opción `--rm` asegura que el contenedor se elimine automáticamente después de la ejecución, manteniendo el entorno limpio. Por ejemplo, para entrenar un modelo se debe ejecutar el comando `docker-compose run --rm esm-axp-gdl-env sh train.sh`, donde train.sh contiene las instrucciones necesarias para el proceso de entrenamiento.

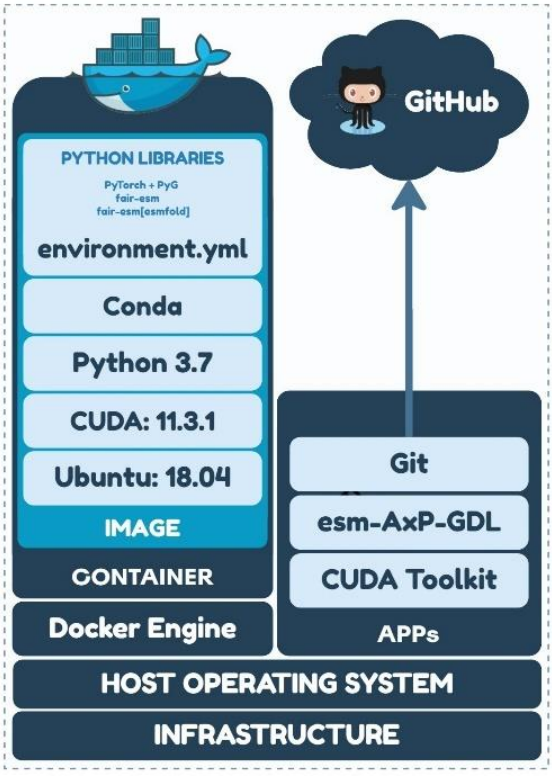

Figura 12. Arquitectura del *framework* usando contenedores Docker

3.7.3 Uso del *framework* en arquitecturas de supercómputo

Para utilizar el *framework* en arquitecturas de supercómputo es necesario configurar el entorno de trabajo mediante el uso de módulos específicos del sistema. A continuación, se detallan los pasos necesarios:

1. Cargar el entorno Python con el comando: `module load python/ondemand-jupyter-python3.7`.
2. Cargar el Compilador de C++ con el comando: `module load gcc/9.2.0.`
3. Cargar el CUDA Toolkit con el comando: module load cuda/11.3.0.
4. Crear el entorno Conda a partir del archivo environment.yml con el comando: `conda env create -f environment.yml`.

En el directorio del *framework* se proporcionan los scripts plantilla train_SLURM.sh, test_SLURM.sh, e inference_SLURM.sh para ejecutar trabajos tipo Slurm (del inglés − Simple Linux Utility for Resource Management) para el entrenamiento, prueba, o inferencia, respectivamente. Slurm es un sistema de gestión de recursos y programación de trabajos utilizado en supercomputadoras, que facilita la asignación de recursos computacionales y la ejecución de trabajos.

3.8 Conclusiones parciales

En este capítulo se presentó el *framework* esm-AxP-GDL desarrollado para modelar y predecir AMPs. Este permite la construcción de modelos independientes de alineamiento a partir del uso de representaciones de grafos basados en distancia. Estos grafos se generan a partir de estructuras de péptidos predichas por ESMFold, cuyos nodos son caracterizados con información evolutiva a nivel de aminoácidos derivada de los modelos ESM-2. A diferencia de los enfoques de la literatura, el enfoque propuesto no depende de MSA. Por lo tanto, permite desarrollar modelos computacionalmente más eficientes. El *framework* se diseñó para ser aplicado a la modelación de cualquier tarea relacionada con la predicción de actividades biológicas o propiedades de péptidos y proteínas.

Capítulo 4. Predicción de Péptidos Antimicrobianos usando estructuras predichas por ESMFold y características ESM-2 con Aprendizaje Profundo de Grafos

En este capítulo se evalúa la utilidad del *framework* esm-AxP-GDL para construir modelos basados en GDL aprovechando estructuras de péptidos predichas por ESMFold y características de aminoácidos derivadas de los modelos ESM-2 para la predicción de AMPs. En la sección 4.1 se describe el conjunto de datos AMPDiscover usado en este trabajo. En la sección 4.2 se estudia la importancia de las incrustaciones ESM-2 para desarrollar modelos basados en GDL. También se comparan los modelos construidos con respecto a modelos reportados en la literatura para la predicción de AMPs, se evalúa la capacidad de generalización en un conjunto de prueba externo, y se analiza la no correlación respecto a coordenadas geométricas y vectores de características aleatorios. Finalmente, en la sección 4.3 se presenta un análisis de distancias inter-aminoácidos.

4.1 Conjunto de datos de referencia de péptidos antimicrobianos

Se utilizó el conjunto de referencia AMPDiscover propuesto por Pinacho-Castellanos et al. (Pinacho-Castellanos et al., 2021) para clasificar AMPs generales. El conjunto AMPDiscover está compuesto por 19,548 secuencias de péptidos de entrenamiento, 5,125 de validación, y 15,685 de prueba (ver Tabla 4 para una descripción). Las secuencias positivas del conjunto AMPDiscover se obtuvieron de la base de datos starPepDB (Aguilera-Mendoza et al., 2019, 2023), que es el repositorio más grande relacionado con AMPs desarrollado hasta la fecha. Este contiene un total de 22,642 secuencias de AMPs no redundantes compiladas desde 40 bases de datos públicas diferentes (ver Tabla 1 en (Aguilera-Mendoza et al., 2019)). Las secuencias negativas para los conjuntos de entrenamiento y validación fueron creadas siguiendo varios criterios aplicados en la literatura (Torrent et al., 2011; Xiao et al., 2013; Gabere y Noble, 2017; Veltri et al., 2018). Estos autores obtuvieron las secuencias negativas para el conjunto de prueba desde (Gabere y Noble, 2017). A partir del conjunto de prueba, Pinacho-Castellanos et al. (2021) también crearon dos conjuntos reducidos para realizar comparaciones justas con respecto a varios modelos del estado del arte. Los conjuntos de prueba reducidos se construyeron eliminando duplicados con los conjuntos de entrenamiento de la literatura. Uno de los conjuntos reducidos está compuesto por secuencias de hasta 100 aminoácidos, mientras que el otro solo contiene secuencias de hasta 30 aminoácidos. El conjunto

compuesto por secuencias de hasta 30 aminoácidos se utilizó para evaluar el rendimiento de los modelos en la clasificación de AMPs de longitud corta, principalmente porque son más fáciles y menos costosos de sintetizar, modificar y optimizar que AMPs de mayor tamaño.

Tabla 4. Conjuntos de entrenamiento, validación y prueba usados para la clasificación de AMPs generales.

Conjunto de Datos	Total de Secuencias	Secuencias Positivas	Secuencias Negativas
Entrenamiento	19,548	9,781	9,767
Validación	5,125	2,564	2,561
Prueba	15,685	4,914	10,771
Prueba (reducido-100) [a]	13,888	3,117	10,771
Prueba (reducido-30) [a, b]	7,801	2,133	5,668
Externo	26,700	5,202	21,498

[a]: Contiene secuencias que no están duplicadas con los conjuntos de entrenamiento utilizados por los modelos reportados en la literatura.

[b]: Contiene secuencias de hasta 30 aminoácidos.

Además, en este trabajo se creó un conjunto de prueba externo uniendo los conjuntos ABPDiscover (Pinacho-Castellanos et al., 2021), AFPDiscover (Pinacho-Castellanos et al., 2021), AVPDiscover (Pinacho-Castellanos et al., 2021), AniAMPpred (Sharma et al., 2021a) , Deep-ABPpred (Sharma et al., 2021b), Deep-AFPpred (Sharma, Shrivastava, Singh, Kumar, Saxena, et al., 2022) y Deep-AVPpred (Sharma, Shrivastava, Singh, Kumar, Singh, et al., 2022). Los conjuntos de datos ABPDiscover, AFPDiscover y AVPDiscover (Pinacho-Castellanos et al., 2021) se crearon de manera similar a AMPDiscover (ver sección 2.1 en (Pinacho-Castellanos et al., 2021)). En cuanto al conjunto de datos AniAMPpred (Sharma et al., 2021a), sus autores obtuvieron las secuencias de AMPs generales de las bases de datos Protein (Schoch et al., 2020) y starPepDB (Aguilera-Mendoza et al., 2023). Las secuencias de AMPs del conjunto Deep-ABPpred (Sharma et al., 2021b) se recopilaron de la Base de Datos de Péptidos Antimicrobianos (APD) (G. Wang et al., 2016), del Repositorio de Datos de Péptidos Antimicrobianos (DRAMP) (Kang et al., 2019) y de la Base de Datos de Péptidos Antimicrobianos de la Leche (Théolier et al., 2014). Las secuencias antifúngicas del conjunto Deep-AFPpred (Sharma, Shrivastava, Singh, Kumar, Saxena, et al., 2022) se obtuvieron de las bases de datos CAMP (Waghu et al., 2016), DRAMP (Kang et al., 2019) y StarPepDB (Aguilera-Mendoza et al., 2023), mientras que las secuencias de péptidos antivirales del conjunto Deep-AVPpred (Sharma, Shrivastava, Singh, Kumar, Singh, et al., 2022) se obtuvieron de las bases de datos AVPpred (Thakur et al., 2012), DBAASP (Pirtskhalava et al., 2016), DRAMP (Kang et al., 2019), SATPdb (Singh et al., 2016) y starPepDB (Aguilera-Mendoza et al., 2023). Las secuencias negativas de estos conjuntos se crearon a partir de secuencias de proteínas revisadas y manualmente anotadas disponibles en la base de datos UniProt (Bateman, 2019). Una vez que se unieron todos estos conjuntos, se eliminaron los duplicados respecto al conjunto AMPDiscover, y se obtuvo un total de 26,700 secuencias únicas. El Anexo D contienen un enlace

a los archivos FASTA de todos los conjuntos.

4.2 Marco metodológico para evaluar la utilidad del *framework* esm-AxP-GDL

Se evaluó la utilidad del *framework* esm-AxP-GDL construyendo un total de 45 modelos basados en GAT utilizando tres dimensiones de capas ocultas (*i.e.*, 64, 128 y 256), aplicando tres umbrales de distancia Euclidiana (*i.e.*, 10, 15 y 20) para construir las aristas del grafo, y utilizando 5 de los 6 modelos ESM-2 para caracterizar cada nodo con información evolutiva. Debido a limitaciones de hardware (NVIDIA RTX A5500 24 GB), no se pudo usar el modelo ESM-2 de 48 capas porque requiere 28 GB de memoria GPU dedicada, la cual es superior a la disponible (24 GB). La elección de aplicar tres umbrales de distancia Euclidiana para construir las aristas del grafo se fundamenta en estudios previos que han demostrado que estas distancias representan, de manera aproximada, interacciones interatómicas cortas y medias (Maljković, 2019; Rao et al., 2020). Estos umbrales permiten generar representaciones de grafos que varían en densidad, desde configuraciones más dispersas hasta más densas. El umbral de 10 Å se ha asociado comúnmente con interacciones cortas, mientras que los umbrales de 15 y 20 Å corresponden a interacciones medias, con una separación de 5 Å. Los grafos creados son no ponderados y no dirigidos. Se consideraron auto ciclos en las capas de atención, se iteró un total de 200 épocas, la tasa de aprendizaje se estableció en 0.0001, el valor de *dropout* se fijó en 0.25, y el tamaño del lote se estableció en 512. El algoritmo Adam (Kingma y Ba, 2014) se usó para fines de optimización. Los modelos obtenidos en la última época se utilizaron para realizar los diferentes análisis.

4.2.1 Impacto de las incrustaciones ESM-2, los umbrales de distancia, y las dimensiones de la capa oculta en el rendimiento de los modelos construidos

Se examinó el impacto de las incrustaciones ESM-2, los umbrales de distancia, y las dimensiones de las capas ocultas con respecto a la capacidad de generalización lograda en el conjunto de prueba por los modelos GAT creados (ver Anexo E). La Figura 13 muestra el diagrama de caja y bigote correspondiente a los valores de MCC_{prueba} obtenidos al usar cada modelo ESM-2 para caracterizar los nodos de los grafos. Se puede notar que los modelos GAT basados en las incrustaciones ESM-2_t36 (de dimensión 2,560) produjeron los valores más altos de MCC_{prueba}. Específicamente, estos modelos lograron el mejor rendimiento (MCC_{prueba} = 0.9521), el segundo mejor (MCC_{prueba} = 0.9505), y el tercer mejor (MCC_{prueba} = 0.9499) de todos los modelos construidos. También se puede observar que la distribución de los valores

de MCC_{prueba} correspondientes a las incrustaciones ESM-2_t36 presenta la mediana por encima de la mediana de las otras distribuciones. Esto indica que las incrustaciones ESM-2_t36 contribuyeron a derivar modelos con mejor rendimiento que la mayoría de los modelos desarrollados con otras incrustaciones ESM-2. Obsérvese además que la distribución de las incrustaciones ESM-2_t33 (de dimensión 1,280) presenta el rango intercuartílico más pequeño. Esto indica que, cuando se utilizan, siempre se pueden construir modelos con buenas capacidades de generalización independientemente de la topología de los grafos de entrada y la dimensión de la capa oculta considerada.

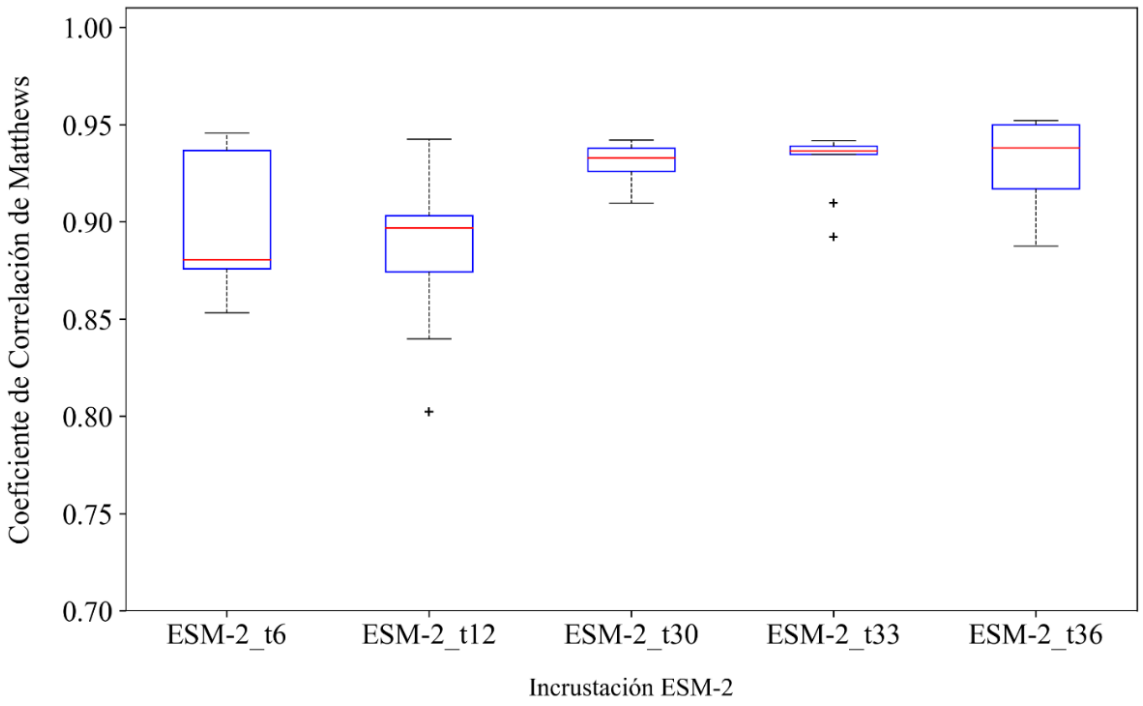

Figura 13. Gráficos de caja y bigote correspondientes a los valores de MCC_{prueba} obtenidos por los modelos GAT construidos al usar diferentes incrustaciones ESM-2.

Además, la Figura 14 muestra el diagrama de caja y bigote correspondiente a los valores de MCC_{prueba} obtenidos al aplicar diferentes umbrales de distancia para construir las representaciones de los grafos a partir de las estructuras de péptidos predichas. La mejor distribución de valores de MCC_{prueba} corresponde al umbral de distancia de 15 angstroms (Å) según el rango intercuartílico, mientras que el uso del umbral de distancia de 10 Å permitió derivar los modelos con las mejores capacidades de generalización. De hecho, al analizar los 10 mejores modelos según sus valores de MCC_{prueba}, 5 de los 10 mejores modelos (incluidos los dos mejores) se crearon utilizando un umbral de distancia de 10 Å; 3 de los 10 mejores modelos se crearon utilizando un umbral de distancia de 15 Å; y solo 2 de los 10 mejores modelos (que fueron los peores) se crearon utilizando un umbral de distancia de 20 Å. En este sentido, cabe destacar

que la mediana correspondiente a la distribución de 20 Å es inferior a la mediana de las otras dos distribuciones. Esto implica que usar grafos densos no necesariamente permitirá obtener modelos con mejores capacidades de generalización que cuando se usan grafos dispersos. El umbral de distancia de 20 Å se aplicó en sAMPpred-GAT (Yan et al., 2023) sin examinar si era el más adecuado. Por lo tanto, según los resultados de este análisis, es importante estudiar cuál es el umbral de distancia más adecuado para construir representaciones basadas en grafos a partir de estructuras predichas.

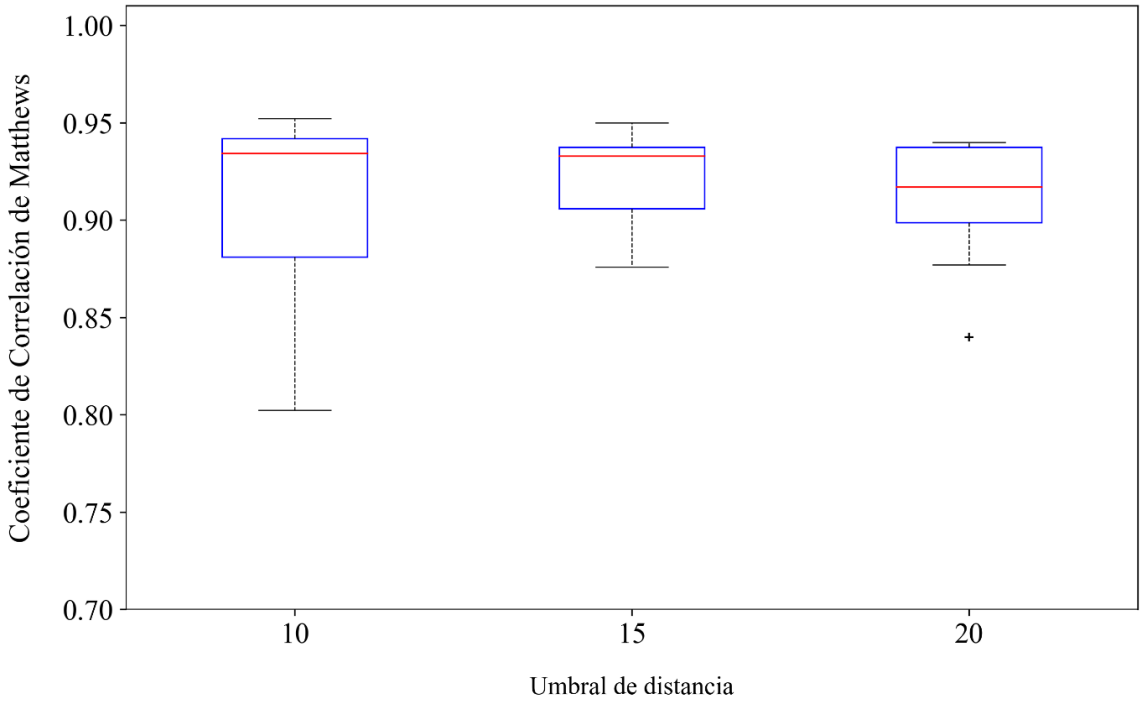

Figura 14. Gráficos de caja y bigote correspondientes a los valores de MCC_{prueba} obtenidos por los modelos GAT construidos al utilizar diferentes umbrales de distancia geométrica.

Finalmente, la Figura 15 muestra el diagrama de caja y bigote correspondiente a los valores de MCC_{prueba} alcanzados por los modelos construidos al variar la dimensión de la capa oculta (HLD, por sus siglas en inglés). Como un resultado, se puede observar que una HLD igual a 128 permitió obtener los mejores modelos. No obstante, el valor más alto de MCC_{prueba} fue producido por un modelo creado con una HLD igual a 256, pero la pequeña diferencia de rendimiento (ΔMCC_{prueba} = 0.0016) entre el mejor modelo y el segundo mejor modelo (que utilizó una HLD igual a 128) indica que usar una HLD igual a 256 no contribuye a representar mejor los datos de entrada y, en consecuencia, no se construirán modelos con capacidades de generalización notablemente mejores. En este sentido, se puede analizar que 4 de los 5 mejores modelos según sus valores de MCC_{prueba} se crearon utilizando una HLD igual a 128. Estos hallazgos también implican un proceso de entrenamiento más eficiente, tomando en cuenta que cuanto menor sea la HLD,

menor será el tiempo de entrenamiento de los modelos.

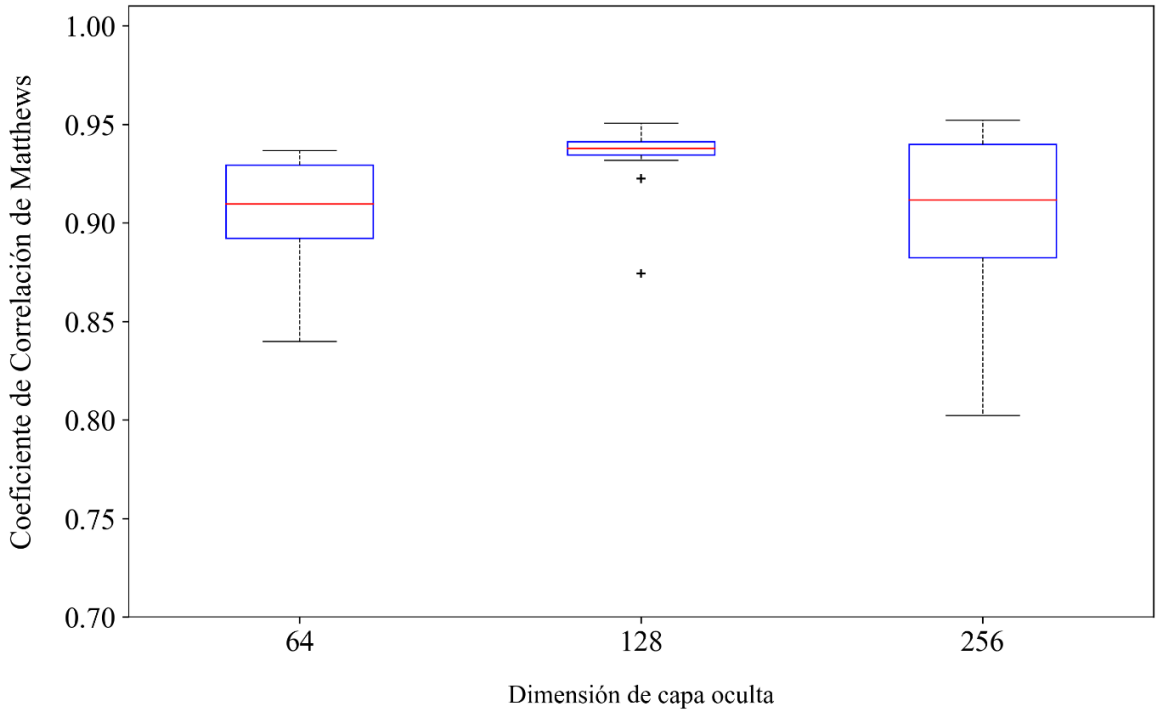

Figura 15. Gráfico de caja y bigote correspondientes a los valores de MCC_{prueba} obtenidos por los modelos GAT construidos al utilizar diferentes tamaños de capa oculta.

4.2.2 Análisis comparativo con respecto a modelos construidos sobre el conjunto de datos de entrenamiento utilizado en este trabajo.

Se compararon los modelos GAT construidos con los modelos reportados en la literatura (ver Tabla 5) que fueron entrenados y evaluados en el conjunto de referencia utilizado en este trabajo. De esta manera, se aseguran comparaciones justas. En cuanto a los modelos de la literatura, AMPDiscover fue el mejor modelo creado donde se propuso el conjunto de referencia (Pinacho-Castellanos et al., 2021). El modelo Random Forest (García-Jacas, García-González, et al., 2022) basado en las características aprendidas ESM-1b fue el mejor modelo desarrollado en un estudio realizado para evaluar la complementariedad entre características calculadas y características aprendidas en la clasificación de AMPs. En ese mismo estudio (García-Jacas, García-González, et al., 2022), las arquitecturas profundas AMPScanner (Veltri et al., 2018), APIN (Su et al., 2019) y APIN-Fusion (Su et al., 2019) fueron reentrenadas 30 veces, y cada uno de esos modelos reentrenados se evaluó en el conjunto de prueba.

Por un lado, se puede observar en la Tabla 5A que el mejor modelo GAT (MCC_{prueba} = 0.9521) obtuvo valores

de SN_{prueba}, SP_{prueba}, ACC_{prueba} y MCC_{prueba} mucho mejores que los reportados hasta la fecha. El mejor modelo de la literatura (MCC_{prueba} = 0.8550) se basa en la arquitectura APIN-Fusion, que es una red convolucional multi-escala fusionada con descriptores de composición de aminoácidos y composición de dipéptidos. Según la métrica MCC_{prueba}, el mejor modelo basado en la arquitectura APIN-Fusion fue superado en términos absolutos por el mejor modelo GAT en 0.0971, lo que representa una mejora del 11.34%. Además, se puede analizar que los valores de sensibilidad (SN_{prueba} = 0.9961) y especificidad (SP_{prueba} = 0.9707) obtenidos por el mejor modelo GAT fueron mejores que los más altos reportados en la literatura por 8.83% (SN_{prueba} = 0.9153) y 1.73% (SP_{prueba} = 0.9542), respectivamente.

Tabla 5. Comparación de la capacidad de generalización alcanzada en el conjunto de prueba (ver Tabla 4) por los modelos construidos con el conjunto de entrenamiento utilizado en este trabajo.

Modelo	SN_{prueba}	SP_{prueba}	ACC_{prueba}	MCC_{prueba}	AUC_{prueba}
A) Rendimiento de los mejores modelos					
Este trabajo	**0.9961**	**0.9707**	**0.9786**	**0.9521**	**0.9942**
AMPDiscover – ver Tabla 2 en (Pinacho-Castellanos et al., 2021)	0.9110	0.9090	0.9098	0.7990	0.9630
Modelo Random Forest basado en ESM-1b – ver Tabla 2 en (García-Jacas, García-González, et al., 2022)	0.9040	0.9450	0.9319	0.8430	-
AMPScanner (reentrenado) – ver Tabla S1 en (García-Jacas, Pinacho-Castellanos, et al., 2022)	0.8757	0.9407	0.9203	0.8151	0.9597
APIN (reentrenado) – ver Tabla S12.1 en (García-Jacas, García-González, et al., 2022)	0.9153	0.9306	0.9259	0.8317	-
APIN-Fusion (reentrenado) – ver Tabla 13.1 en (García-Jacas, García-González, et al., 2022)	0.9011	0.9542	0.9376	0.8550	-
B) Rendimiento medio					
Este trabajo	**0.9929** (.005)	**0.9458** (.026)	**0.9605** (.017)	**0.9149** (.033)	**0.9927** (.001)
AMPScanner (reentrenado) – ver Tabla S1 en (García-Jacas, Pinacho-Castellanos, et al., 2022)	0.9207 (.017)	0.8666 (.039)	0.8835 (.022)	0.7544 (.033)	0.9611 (.002)
APIN (reentrenado) – ver Tabla S12.1 en (García-Jacas, García-González, et al., 2022)	0.9301 (.006)	0.9015 (.015)	0.9105 (.008)	0.8045 (.015)	-
APIN-Fusion (reentrenado) – ver Tabla 13.1 en (García-Jacas, García-González, et al., 2022)	0.9269 (.013)	0.8932 (.050)	0.9038 (.030)	0.7942 (0.49)	-
C) Rendimiento de los peores modelos					
Este trabajo	**0.9992**	0.8528	**0.8986**	**0.8022**	**0.9921**
AMPScanner (reentrenado) – ver Tabla S1 en (García-Jacas, Pinacho-Castellanos, et al., 2022)	0.9528	0.7731	0.8294	0.6762	0.9624
APIN (reentrenado) – ver Tabla S12.1 en (García-Jacas, García-González, et al., 2022)	0.9359	**0.8734**	0.8930	0.7733	-
APIN-Fusion (reentrenado) – ver Tabla 13.1 en (García-Jacas, García-González, et al., 2022)	0.9683	0.7225	0.7995	0.6408	-

En general, estos resultados sugieren que tanto el uso de estructuras predichas por ESMFold como el uso de información evolutiva a nivel de aminoácidos derivada de los modelos ESM-2 permiten generar valiosas

representaciones estructurales basadas en grafos, lo que lleva a construir modelos GDL con rendimientos consistentemente buenos, incluso mejores que varias arquitecturas profundas del estado del arte (ver también Figura 16). En este sentido, estos resultados también parecen contradecir las conclusiones de García-Jacas et al. (García-Jacas, Pinacho-Castellanos, et al., 2022) en un estudio entre modelos basados en DL y no basados en DL en la clasificación de AMPs. En ese estudio, varios modelos no basados en DL se construyeron con conjuntos de datos propuestos para construir modelos basados en DL (Veltri et al., 2018; Su et al., 2019; Fu et al., 2020; J. Li et al., 2020; J. Yan et al., 2020; Sharma et al., 2021b, 2021a; Y. Zhang et al., 2021; Sharma, Shrivastava, Singh, Kumar, Singh, et al., 2022). Se demostró que los modelos basados en DL no superan a los modelos no basados en DL, y que ambos tipos de modelos realizan predicciones bastante similares. Sin embargo, los modelos analizados en ese estudio se crearon con información derivada solo desde secuencias de aminoácidos. En consecuencia, se considera que los buenos resultados logrados con los modelos construidos aquí se deben al uso de datos de entrada derivados de la información 3D de las estructuras predichas, lo cual también se ha discutido en otros trabajos (Durairaj et al., 2023).

4.2.3 Análisis comparativo con respecto a modelos construidos sobre otros conjuntos de datos de entrenamiento

En primer lugar, se realizó comparaciones con respecto a modelos profundos basados en LSTM (del inglés - Long Sgura 15Aguhort-Term Memory), BERT, y arquitecturas de Atención. Estos modelos fueron introducidos por Ma et al. (2022) para identificar AMPs generales del microbioma intestinal humano. Los resultados consensuados de estos modelos también se incluyeron en las comparaciones (*i.e.*, AMP si los tres modelos clasifican un péptido como AMP, de lo contrario, no AMP). Este estudio se llevó a cabo en el conjunto de prueba y los resultados se muestran en la Figura 16 (y en la Tabla S15 en (García-Jacas, García-González, et al., 2022)). En segundo lugar, se hicieron comparaciones con respecto a los modelos AmPEP (Bhadra et al., 2018) , Deep-AmPEP30 (J. Yan et al., 2020), RF-AmPEP30 (J. Yan et al., 2020), iAMP-2L (Xiao et al., 2013), Adam (H. T. Lee et al., 2015), MLAMP (W. Lin y Xu, 2016), AMPfun (Chung et al., 2020), CAMPR3-ANN (Waghu et al., 2016), CAMPR3-RF (Waghu et al., 2016), CAMPR3-DA (Waghu et al., 2016), y CAMPR3-SVM (Waghu et al., 2016). En este caso, se utilizaron los conjuntos de prueba reducidos. Los resultados se muestran en la Figura 17 (y en la Tabla 5 en (Zielezinski et al., 2017), y en el Anexo F y el Anexo G).

Primero, se puede observar en la Figura 16 que los modelos GAT construidos presentaron capacidad de generalización (métrica MCC) comparables o superiores a los enfoques desarrollados por Ma et al. (2022).

Notar que el peor modelo GAT fue superior al modelo LSTM en un 18.28%, mientras que fue inferior a los modelos basados en *Attention*, BERT y Consensus por 0.01%, 0.89% y 6.33%, respectivamente, pero el rendimiento del mejor modelo GAT y el rendimiento promedio de todos los modelos GAT siempre fueron mejores que los rendimientos reportados por Ma et al. (2022). El mejor modelo GAT superó a los modelos basados en LSTM, *Attention*, BERT y Consensus en un 40.39%, 18.67%, 17.64% y 11.62%, respectivamente, mientras que el rendimiento promedio de los modelos GAT fue mejor que los resultados obtenidos por los modelos basados en LSTM, *Attention*, BERT y Consensus en un 34.9%, 14.03%, 13.05% y 7.26%, respectivamente. Las comparaciones entre los modelos GAT y los modelos de Ma et al. (2022) se realizaron sin eliminar duplicados entre el conjunto de entrenamiento de Ma et al. (2022) y el conjunto de prueba utilizado en este trabajo. A pesar de esto, los modelos GAT fueron significativamente mejores.

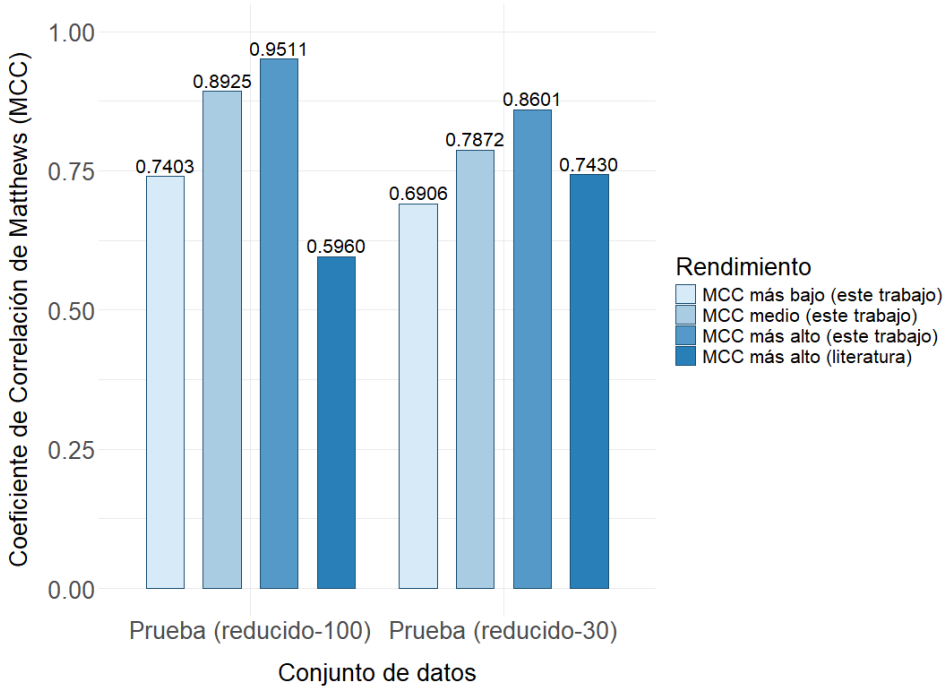

Figura 16. Comparación de los valores más bajos, promedio y más altos de MCC_{prueba} alcanzado por los modelos GAT creados en este trabajo en relación con el valor más alto de MCC_{prueba} obtenido en los dos conjuntos de prueba reducidos por 11 modelos de la literatura.

Conclusiones similares se pueden extraer de los resultados mostrados en la Figura 17. En cuanto al conjunto reducido compuesto por secuencias de hasta 100 aminoácidos, se puede analizar que los rendimientos más bajos, promedio y más altos alcanzados en este trabajo fueron mejores que el rendimiento más alto de la literatura en un 24.21%, 49.75% y 59.58%, respectivamente. En relación con el conjunto de prueba reducido compuesto por AMPs cortos, el mejor rendimiento reportado en la literatura se obtuvo por el modelo RF-AmPEP30 (J. Yan et al., 2020) ($MCC_{reduced\text{-}30}$ = 0.743), que es inferior a los

rendimientos promedio y más altos obtenidos en este trabajo en un 5.61% y 13.61%, respectivamente. El rendimiento más bajo obtenido aquí para predecir AMPs cortos fue inferior al rendimiento de RF-AmPEP30 en un 7.05%. Todos estos resultados confirman que se pueden desarrollar buenos modelos basados en grafos aprovechando las estructuras predichas por ESMFold y la información evolutiva derivada de los modelos ESM-2. A continuación, se muestran estudios adicionales para determinar cómo se desempeñan los mejores modelos GAT en conjuntos de secuencias diferentes del conjunto AMPDiscover.

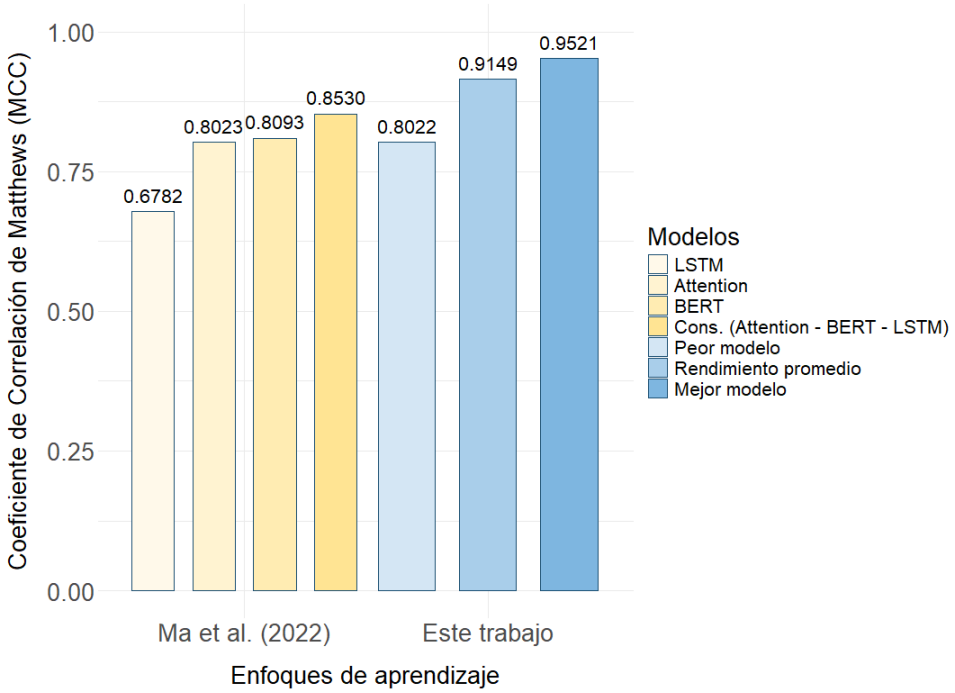

Figura 17. Comparación de los valores más bajos, promedio y más altos de MCC_{prueba} alcanzado por los modelos GAT creados en este trabajo en relación con los valores de MCC_{prueba} obtenidos en el conjunto de prueba por los modelos profundos propuestos por Ma et al (2022).

4.2.4 Validación de los mejores modelos construidos en un conjunto de prueba externo

A partir de los resultados explicados anteriormente, se seleccionaron los tres mejores modelos GAT construidos con umbrales de distancia de 10 Å y 15 Å, respectivamente, en función de sus valores de MCC_{prueba}. Estos modelos se evaluaron en el conjunto de prueba externo (ver sección 4.1), y los resultados se muestran en la Tabla 6. En primer lugar, se puede observar que 4 de los 6 mejores modelos presentaron una capacidad de generalización moderada porque lograron valores de $MCC_{externo}$ entre 0.45 y 0.5. Solo dos modelos obtuvieron un pobre rendimiento, presentando valores de $MCC_{externo}$ inferiores a 0.37. La

métrica MCC tiene en cuenta los verdaderos y falsos positivos y negativos. Así que al profundizar en la métrica $SN_{externo}$, se puede analizar que todos los modelos tuvieron una tasa de verdaderos positivos alta. Principalmente obtuvieron valores de $SN_{externo}$ superiores a 0.93, excepto un modelo que obtuvo un valor de $SN_{externo}$ igual a 0.895. Estos son resultados esperados, teniendo en cuenta que el conjunto de entrenamiento se construyó a partir de la base de datos StarPepDB, que es el repositorio relacionado con AMPs más grande hasta la fecha. Por lo tanto, los modelos GAT aprendieron características discriminativas de AMPs que representan casi todo el universo de tales péptidos.

Sin embargo, al analizar la métrica $SP_{externo}$, se puede observar que los modelos GAT principalmente presentaron una tasa de verdaderos negativos de moderada a baja. En este sentido, cabe destacar que 4 de los 6 mejores modelos solo pudieron recuperar entre el 62% y el 70.4% de los no-AMPs. Los otros dos modelos alcanzaron valores de $SP_{externo}$ cercanos a 0.5. Estos resultados indican que las representaciones de grafos construidas no fueron suficientes para aprender buenas características discriminativas para identificar no-AMPs. Por un lado, esto puede deberse al hecho de que el conjunto de datos negativos utilizado para el entrenamiento no es representativo del universo de no-AMPs, lo cual se ha discutido en (Sidorczuk et al., 2022) como un problema a tener en cuenta al construir modelos predictivos para esta tarea. Por otro lado, los resultados relacionados con la métrica $SP_{externo}$ también sugieren mejorar los grafos que representan no-AMPs, con el propósito de obtener mejores características aprendidas. Un enfoque podría ser el uso de representaciones de grafos topológicamente diferentes por estructura de péptido predicha para realizar aprendizaje multi-instancia (Zankov et al., 2024). Esto se debe a que un único grafo no debería contener toda la información estructural de una estructura de péptido, lo que podría estar afectando negativamente la construcción de modelos con mejores tasas de verdaderos negativos. Por ejemplo, se pueden usar diferentes criterios basados en la distancia y/o criterios fisicoquímicos para construir diferentes grafos, como se ha detallado en otros estudios (Jamasb et al., 2021).

Tabla 6. Métricas de generalización alcanzadas en el conjunto de pruebas externo por los mejores modelos.

Modelos								
ID	**HLD**	**Umbral**	**Incrustación**	**$SN_{externo}$**	**$SP_{externo}$**	**$ACC_{externo}$**	**$MCC_{externo}$**	**$AUC_{externo}$**
Modelo 1	128	10	ESM-2_t6	0.9485	0.6226	0.6861	0.4525	0.9099
Modelo 2	128	10	ESM-2_t33	0.9348	0.6842	0.7331	0.4944	0.9346
Modelo 3	128	10	ESM-2_t36	0.8950	0.7039	0.7412	0.4819	0.8972
Modelo 4	128	15	ESM-2_t12	0.9321	0.5275	0.6063	0.3669	0.8866
Modelo 5	128	15	ESM-2_t30	0.9404	0.6789	0.7298	0.4939	0.9286
Modelo 6	128	15	ESM-2_t36	0.9312	0.4976	0.5821	0.3448	0.8663

Debido a las tasas de verdaderos negativos de moderadas a bajas, se analizó cuán diferentes son las

predicciones de no-AMPs entre los modelos 2, 3 y 5 mostrados en la Tabla 6. Estos modelos fueron los tres mejores en cuanto a sus valores de $MCC_{externo}$. Para llevar a cabo este estudio, se calculó la medida de desacuerdo (Kuncheva y Whitaker, 2003) para cada par de modelos. Esta medida es la razón entre el número de predicciones en las que un modelo es correcto y el otro es incorrecto respecto al número total de instancias (ver Anexo H). Cuanto mayor es el desacuerdo, mayor es la posibilidad de obtener mejores predicciones combinando modelos. Como resultado, hay un alto desacuerdo (mayor al 20%) entre los modelos 2 y 3 (0.2605), los modelos 2 y 5 (0.277), y los modelos 3 y 5 (0.2755). En consecuencia, primero se combinaron los modelos 2 y 5, que presentan el mayor desacuerdo, utilizando un criterio basado en un *AND lógico*. Luego, se calculó la métrica de desacuerdo con respecto al modelo 3, obteniendo un valor de 0.2314, que indica que los tres modelos pueden combinarse juntos. La Tabla 7 muestra las métricas de desempeño al combinar los modelos. El Anexo I muestra las predicciones de consenso para todas las instancias negativas del conjunto de prueba externo.

Como era de esperar, las predicciones por consenso son mucho mejores que las predicciones generadas por el mejor modelo individual, mejorando entre un 19.13% y un 29.79% con respecto a la métrica $MCC_{externo}$. Cabe destacar que el valor más alto de $SP_{externo}$ obtenido por un modelo individual ($SP_{externo}$ = 0.7039) se mejoró significativamente entre un 16.49% y un 24.69%. Además, se puede observar en la Tabla 7 que la combinación de los tres modelos individuales mejora el rendimiento en comparación con la combinación de cualquier par de ellos, entre un 5.86% y un 8.95% en relación con la métrica $MCC_{externo}$, lo cual está respaldado por el análisis basado en la métrica de desacuerdo explicada anteriormente.

Tabla 7. Métricas de generalización alcanzadas en el conjunto de prueba externo por los modelos de consenso creados mediante la fusión de los tres mejores modelos individuales.

Modelos Combinados	$SN_{externo}$	$SP_{externo}$	$ACC_{externo}$	$MCC_{externo}$
Modelo 2 – Modelo 3	0.8631	0.8244	0.8319	0.5890
Modelo 2 – Modelo 5	0.8933	0.8200	0.8343	0.6062
Modelo 3 – Modelo 5	0.8595	0.8291	0.8351	0.5924
Modelo 2 – Modelo 3 – Modelo 5	0.8339	0.8777	0.8691	0.6417

Los modelos 2 y 3 se construyeron con grafos derivados a partir de un umbral de distancia de 10 Å, y sus nodos se caracterizaron con las incrustaciones ESM-2_t33 y ESM-2_t36, respectivamente. El modelo 5 se construyó con grafos derivados a partir de un umbral de distancia de 15 Å, y sus nodos se caracterizaron con las incrustaciones ESM-2_t30. Por lo tanto, los resultados de combinar estos modelos demuestran que el uso de diferentes incrustaciones ESM-2 y umbrales de distancia para obtener representaciones basadas en grafos permite desarrollar modelos que codifican diferentes espacios químicos, los cuales pueden ser

exitosamente utilizados en tareas de cribado prospectivas.

Finalmente, la Figura 18 muestra el rendimiento obtenido tanto por los tres mejores modelos individuales como por el modelo de consenso de ellos en conjuntos de secuencias de no-AMP cuyas similitudes están por debajo de un umbral. Se usó umbrales de similitud que van desde 0.1 hasta 0.9 con un paso de 0.1. Los conjuntos se crearon eliminando pares de no-AMP similares entre los conjuntos de entrenamiento, validación, prueba y externo. En este estudio sólo se utilizaron los pares no similares pertenecientes al conjunto externo. Además, en la Figura 18 se muestra el número total de no-AMP no redundantes por conjunto construido (ver Anexo J para archivos FASTA). El Anexo K muestra los valores de SP logrados por cada modelo. En general, se puede observar que todos los modelos lograron un rendimiento tan bueno como el obtenido en el conjunto de prueba externo (valor de referencia). Para los conjuntos construidos con umbrales de similitud entre 0.4 y 0.9, los valores de SP obtenidos por los modelos fueron comparables con los valores de $SP_{externo}$ (ver Tabla 6 y Tabla 7), mientras que para los conjuntos construidos con umbrales de similitud entre 0.1 y 0.3, los modelos lograron valores de SP superiores a 0.8. Estos resultados sugieren que la metodología implementada permite construir modelos complementarios que, al combinarse, tienen también un buen rendimiento en conjuntos de secuencias diferentes a los utilizados para entrenarlos, lo cual es deseable en aplicaciones prospectivas.

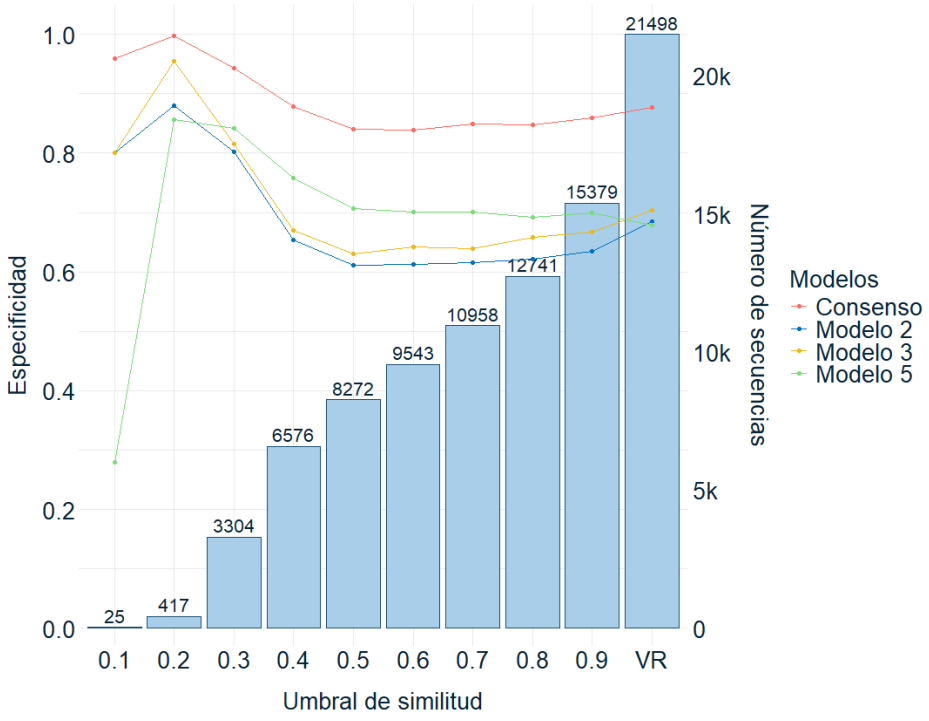

Figura 18. Tasas de verdaderos negativos obtenidas por los tres mejores modelos individuales y el modelo de consenso de ellos en conjuntos que contienen sólo pares no AMP cuyas similitudes están por debajo de un umbral. VR (valor de referencia) representa los valores de especificidad en el conjunto externo.

4.2.5 Análisis de la importancia de la estructura 3D y las incrustaciones ESM-2 en la construcción de los modelos

En esta sección se analiza cuánto depende el rendimiento de los modelos GAT de las estructuras 3D predichas por ESMFold y de las incrustaciones ESM-2. Para ello, se utilizaron los métodos de validación perturbación aleatoria en las coordenadas geométricas, perturbación aleatoria en las incrustaciones ESM-2, y construcción de grafos basados en mapas de contacto ESM-2 (ver sección 3.5). Con cada método se generaron nuevos modelos utilizando el conjunto de datos AMPDiscover como datos de entrada, el modelo ESM-2 de 33 capas para caracterizar evolutivamente los nodos del grafo, un umbral de distancia Euclidiana igual a 10 Å para construir las aristas de los grafos basados en distancia, y un tamaño de capa oculta igual a 128. Los grafos creados son no ponderados y no dirigidos. Se iteraron un total de 200 épocas, la tasa de aprendizaje se estableció en 0.0001, el valor de *dropout* se fijó en 0.25, y el tamaño del lote se estableció en 512. El algoritmo Adam (Kingma y Ba, 2014) fue el utilizado para fines de optimización. Esta configuración de parámetros corresponde al segundo mejor modelo de todos los modelos analizados en la sección 4.2.1. En la construcción de los nuevos modelos no se consideraron auto ciclos en las capas de atención. Para la validación con el método de perturbación aleatoria en las coordenadas geométricas, se establecieron porcentajes de aleatoriedad de 20, 40, 60, 80 y 100. En el caso de la perturbación aleatoria en las incrustaciones ESM-2, los porcentajes de aleatoriedad fueron 5, 10, 15, 20, 25 y 30. Para la construcción de grafos basados en mapas de contacto ESM-2, se usaron los umbrales de probabilidad de contacto 70, 80 y 90.

Para evaluar cuanto depende el rendimiento de los modelos del uso de coordenadas geométricas predichas por ESMFold, se usó el método de validación de coordenadas aleatorias y se crearon 30 modelos para cada porcentaje de aleatoriedad. Se crearon también 30 modelos sin introducir aleatoriedad (valores de referencia). Estos modelos se compararon con respecto a la capacidad de generalización lograda en el conjunto de validación (ver sección 4.1). La Figura 19 muestra el diagrama de caja y bigote correspondiente a los valores de MCC alcanzados por los modelos anteriores. Por un lado, se puede observar que la mejor distribución de valores de MCC corresponde a los modelos generados a partir del uso de coordenadas 3D predichas por ESMFold, los cuales también mostraron las mejores capacidades de generalización. Esto confirma que la metodología propuesta permite crear modelos estables acorde con su capacidad de generalización a pesar de que el proceso de entrenamiento no es determinista. Por otra parte, se observa que a medida que aumenta el porcentaje de coordenadas aleatorias, los modelos construidos exhiben una mayor dispersión en sus predicciones en el conjunto de validación. Notar que con un 100% de coordenadas aleatorias, los modelos generados muestran una capacidad de generalización por debajo de 0.3, y la

mediana de la distribución se acerca a cero, lo que indica que las predicciones de los modelos tienden a ser aleatorias. Esto indica que los modelos construidos en este trabajo a partir de grafos representando estructuras 3D de péptidos son altamente dependiente del uso de coordenadas predichas por ESMFold para lograr buenas capacidades de generalización.

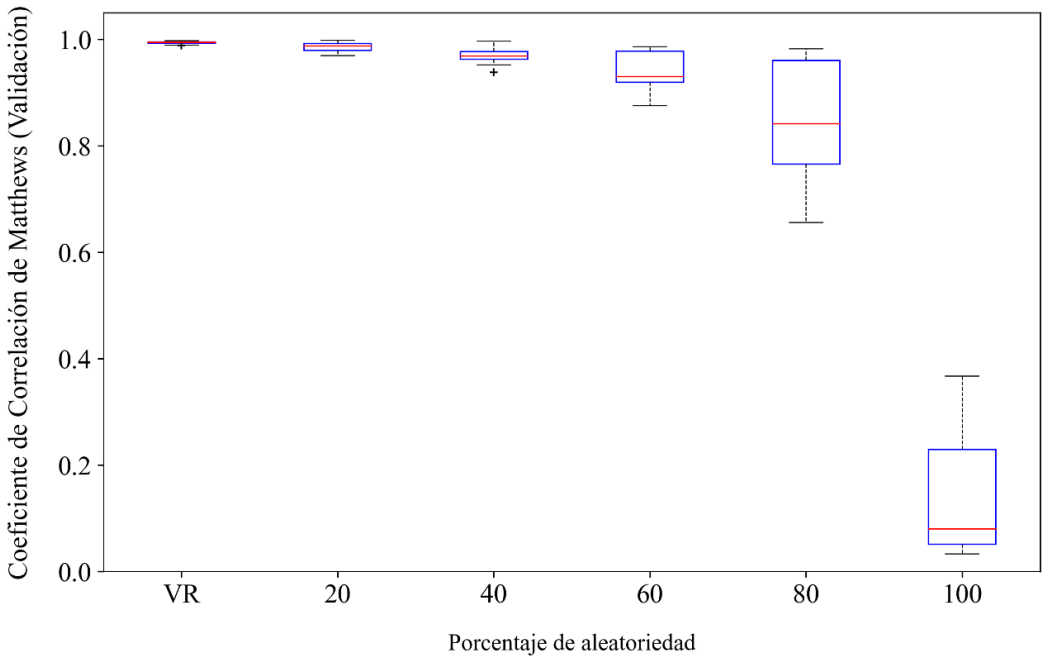

Figura 19. Gráfico de caja y bigote correspondiente a los valores de MCC obtenidos por los modelos construidos al usar grafos creados desde coordenadas geométricas aleatorias con diferentes porcentajes de aleatoriedad. VR (valores de referencia) corresponde a los valores de MCC obtenidos por los modelos construidos con las coordenadas predichas por ESMFold.

Posteriormente, se crearon nuevos modelos utilizando grafos construidos a partir de mapas de contacto ESM-2. Se crearon 30 modelos para cada umbral de probabilidad, y 30 modelos a partir de grafos basados en distancia (valores de referencia). A diferencia del análisis anterior, en este estudio no se usan coordenadas 3D aleatorias. Estos modelos se compararon con respecto a la capacidad de generalización lograda en el conjunto de validación (ver sección 4.1). La Figura 20 muestra el diagrama de caja y bigote correspondiente a los valores de MCC obtenidos por los modelos construidos. En este caso la mejor distribución de valores de MCC también corresponde a los modelos generados a partir de grafos representando estructuras de péptidos predichas por ESMFold, los cuales también mostraron las mejores capacidades de generalización, todas por encima de 0.99. Por su parte, los modelos construidos a partir de mapas de contacto ESM-2 muestran capacidades de generalización más dispersas. No obstante, se puede observar que la mediana de todas las distribuciones alcanza valores superiores a 0.94, y todos los

modelos muestran una capacidad de generalización superior a 0.88, indicando que estos modelos tienen un buen rendimiento. Los buenos resultados de los modelos basados en mapas de contacto ESM-2 son esperados, dado que estos mapas proporcionan una buena representación de la estructura geométrica de los péptidos y son utilizados por el modelo ESMFold para la predicción de la estructura 3D. Sin embargo, es importante tener en cuenta que los modelos derivados de estos mapas pueden verse afectados por probabilidades de contacto altas que, en algunos casos, no corresponde a un contacto físico real debido a la estructura que adoptan las cadenas laterales de cada aminoácido (ver sección 2.2).

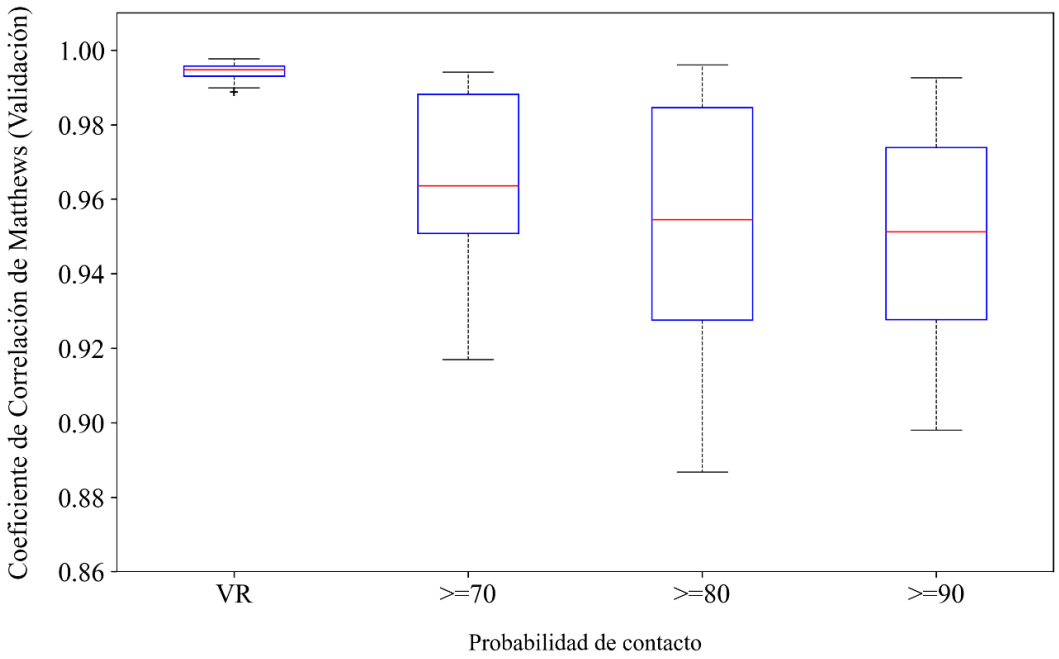

Figura 20. Gráfico de caja y bigote correspondiente a los valores de MCC obtenidos por los modelos construidos utilizando grafos creados a partir de mapas de contacto ESM-2 con diferentes umbrales de probabilidad. VR (valores de referencia) corresponde a los valores de MCC obtenidos por los modelos construidos a partir de grafos basados en distancia.

Teniendo en cuenta que tanto los modelos basados en estructuras 3D como los basados en mapas de contacto ESM-2 mostraron un rendimiento similar, se llevó a cabo un análisis estadístico para determinar si los modelos construidos utilizando las estructuras 3D son superiores a los modelos construidos utilizando los mapas de contacto ESM-2. Este análisis tiene como objetivo evaluar si la mejora en la capacidad de generalización de los modelos que proporcionan las estructuras 3D es estadísticamente significativa como para justificar el paso de complejidad adicional de utilizar estructuras 3D predichas en lugar de los mapas de contacto ESM-2 para construir los grafos. Primeramente, se aplicó la prueba de normalidad de Kolmogorov-Smirnov (Mishra et al., 2019), la cual indicó que los datos no siguen una distribución normal. Por tal motivo, se procedió a aplicar la prueba no paramétrica de Friedman (Derrac

et al., 2011) utilizando un nivel de significancia igual a 0.05. De acuerdo con los rankings calculados por la prueba de Friedman, los mejores modelos fueron aquellos generados a partir de coordenadas 3D, seguidos por los modelos construidos a partir de mapas de contacto ESM-2 con umbrales de 90, 80, y 70, respectivamente (ver Anexo L). El estadístico de Friedman indica que existen diferencias significativas globales entre los enfoques comparados. Por lo tanto, para determinar las diferencias específicas, se aplicó el método post hoc de Holm (Holm, 1979) (ver Anexo M). En este caso, se determinó que los modelos generados a partir de coordenadas 3D son estadísticamente superiores a los otros enfoques basados en mapas de contacto ESM-2. Esto justifica el uso de estructuras 3D predichas para construir modelos basados en aprendizaje de grafos (GDL). Sin embargo, esto no descarta la utilidad de los mapas de contacto ESM-2, a partir de los cuales también se pueden generar modelos con un buen rendimiento.

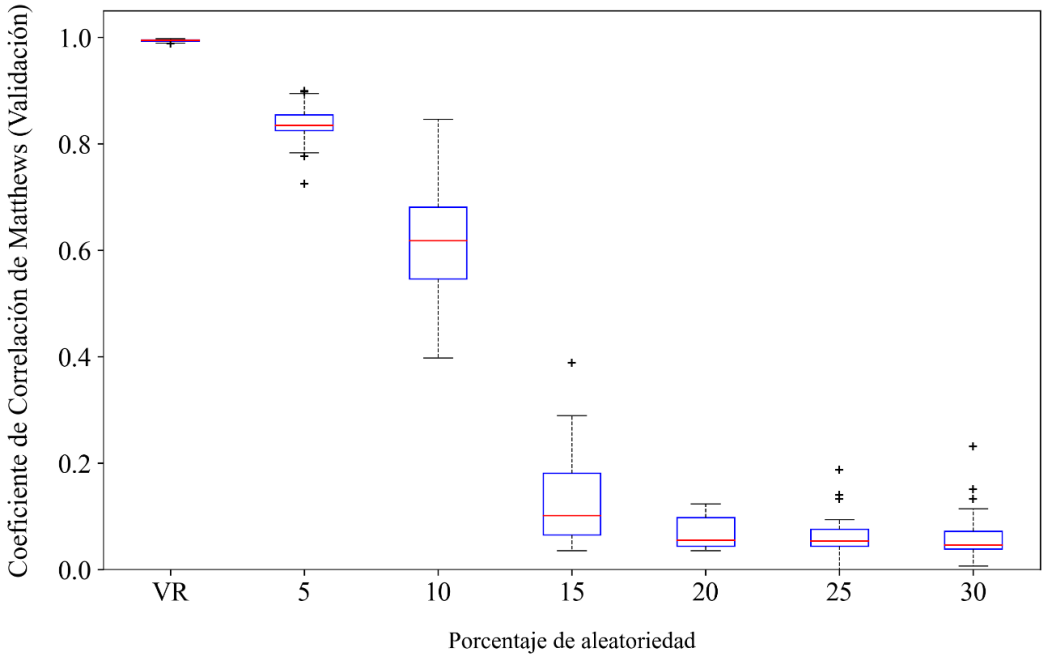

Figura 21. Gráfico de caja y bigote correspondiente a los valores de MCC obtenidos por los modelos construidos utilizando grafos creados a partir de vectores de características derivados del modelo ESM-2 de 33 capas con diferentes porcentajes de valores aleatorios. VR (valores de referencia) corresponde a los valores de MCC obtenidos por los modelos construidos con las incrustaciones originales.

Finalmente, se evaluó cuanto impacta en el rendimiento de los modelos el uso de información evolutiva derivada de los modelos ESM-2 para caracterizar los nodos de los grafos. En este caso se utilizó el modelo ESM-2 de 33 capas que genera incrustaciones de dimensión 1280. Las filas de estas incrustaciones son los vectores de características de cada nodo en los grafos. En cada vector de características se reemplazó un porcentaje de los valores originales por valores generados aleatoriamente con una distribución uniforme dentro del intervalo de valores de las incrustaciones originales. Por ejemplo, si se utiliza un porcentaje de

aleatoriedad del 15%, 192 valores originales de los 1280 de cada vector son reemplazados por valores aleatorios. Se crearon 30 modelos para cada porcentaje de aleatoriedad, y 30 modelos sin introducir aleatoriedad (valores de referencia). Estos modelos se compararon con respecto a la capacidad de generalización alcanzada en el conjunto de validación (ver sección 4.1). La Figura 21 muestra el diagrama de caja y bigote correspondiente a los valores de MCC obtenidos por los modelos creados. Como se observa, a medida que se incrementa el porcentaje de aleatoriedad, la capacidad de generalización de los modelos disminuye notablemente. Específicamente, a partir del 15% de aleatoriedad, todos los modelos muestran valores de MCC por debajo de 0.3, y la mediana de las distribuciones se acerca a cero, indicando que las predicciones de los modelos tienden a ser aleatorias. Esto indica que los modelos son altamente dependientes de las características derivadas de los modelos ESM-2, y cualquier pérdida de esta información significativamente afecta su capacidad de generalización.

4.3 Análisis de distancias inter-aminoácidos

Como se discutió en la sección 2.4, el uso de las distancias inter-aminoácidos es un enfoque ampliamente utilizado para establecer relaciones entre aminoácidos en enfoques basados en grafos. Aunque la distancia Euclidiana ha sido la única utilizada para este propósito, investigaciones en el campo de los descriptores moleculares han demostrado que otras funciones de distancia pueden generar descriptores moleculares con una mayor capacidad de modelación y una mejor capacidad para discriminar entre moléculas estructuralmente diferentes (Marrero-Ponce et al., 2015). En este sentido, se realizó un análisis de las distancias geométricas entre aminoácidos utilizando las funciones de distancia Euclidiana, Separación Angular, Bhattachryya, Canberra, Lance William, Clark, y Soergel (ver sección 2.4.2).

Este estudio sienta las bases para futuras investigaciones que examinen si el uso de distintas funciones de distancias puede generar grafos con topologías diversas a partir de estructuras de péptidos predichas, lo que podría contribuir a codificar espacios químicos diferentes y potencialmente mejorar el rendimiento de los modelos basados grafos representando estructuras de péptidos predichas. Para este fin, se utilizó la base de datos starPepDB, que contiene un total de 45,120 secuencias de péptidos no redundantes compiladas desde 40 bases de datos públicas diferentes (ver Tabla 1 en (Aguilera-Mendoza et al., 2019)). Se seleccionaron secuencias que contienen exclusivamente los 20 aminoácidos canónicos (ver Tabla 1) y cuya longitud está entre 10 y 100 aminoácidos, resultando en un subconjunto de 34,636 secuencias que representan un total de 25,688,946 pares de aminoácidos. Se calcularon las distancias entre pares de aminoácidos utilizando el átomo de carbono α y las funciones de distancia consideradas en este trabajo.

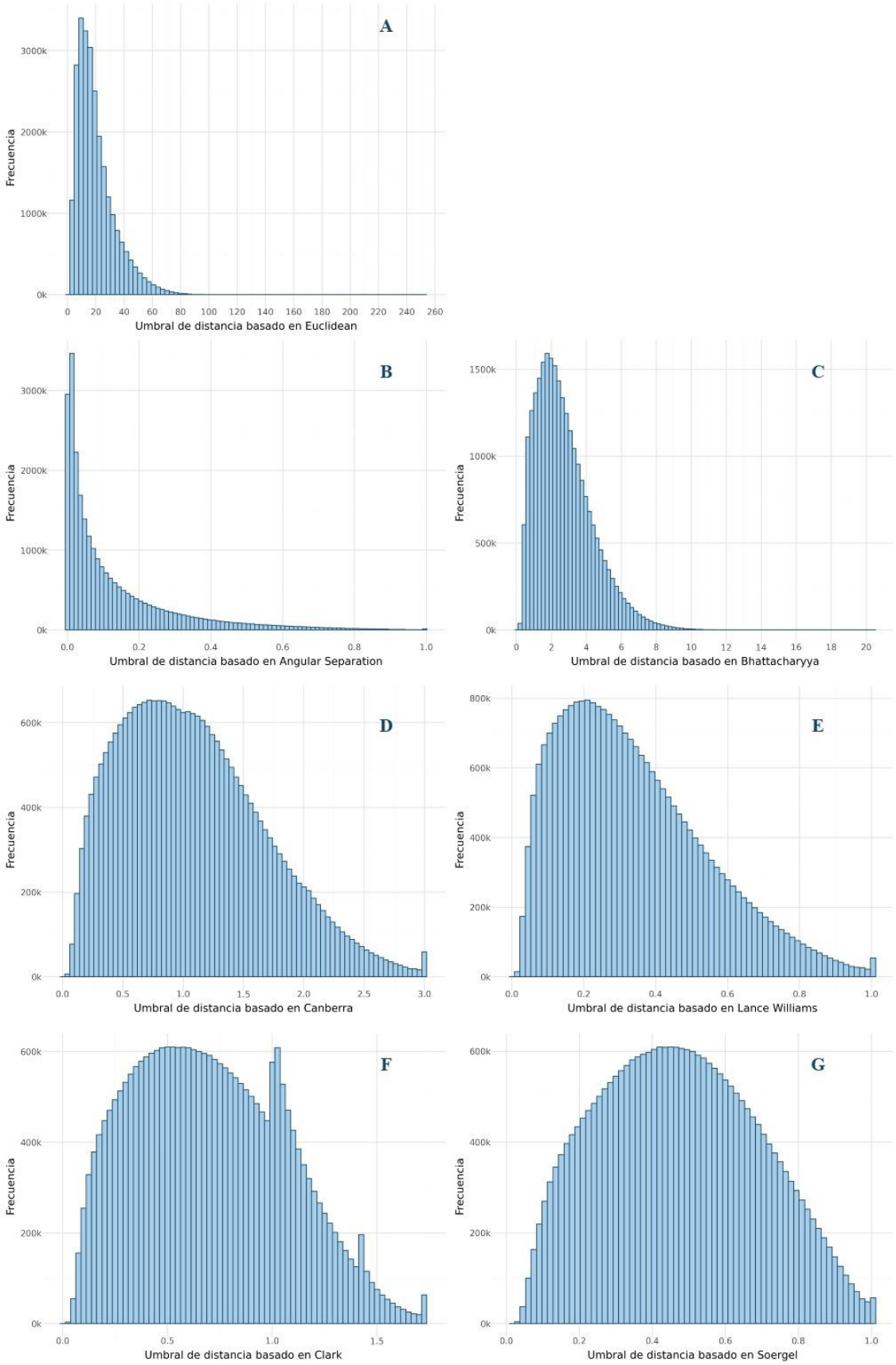

Figura 22. Distancias inter-aminoácidos utilizando diferentes funciones de distancias. (A) Euclidiana. (B) Separación Angular. (C) Bhattachryya. (D) Canberra. (E) Lance-Williams. (F) Clark. (G) Soergel.

La Tabla 8 presenta los percentiles 25, 50 y 75 de estas distancias para cada función de distancia. Estos percentiles se podrán utilizar como umbrales de distancias para definir las aristas en la construcción de grafos. Por ejemplo, el percentil 25 define un umbral que incluye solo las distancias más cortas, permitiendo aristas únicamente entre los pares de aminoácidos más cercanos, lo que resulta en un grafo disperso (con menos aristas). En contraste, los percentiles 50 y 75, abarcan distancias mayores, lo que aumenta el número de aristas y produce grafos más densos. La Figura 22 muestra la distribución de las distancias inter-aminoácidos para cada función de distancia, evidenciando que se obtienen distribuciones diferentes. Esta variabilidad sugiere que los umbrales definidos por los percentiles pueden dar lugar a grafos con topologías distintas, lo que podría generar una diversidad en las representaciones de los péptidos y potencialmente conllevar a codificar diferentes espacios químicos.

Tabla 8. Percentiles de las distancias entre pares de aminoácidos para diferentes funciones de distancia.

	Percentil		
Función de distancia	**25%**	**50%**	**75%**
Euclidiana	10.2836	16.6132	26.2420
Canberra	0.6155	1.0142	1.4712
Lance-Williams	0.1789	0.3041	0.4660
Clark	0.4161	0.6813	0.9830
Soergel	0.3035	0.4664	0.6357
Bhattacharyya	1.5158	2.4304	3.6452
Separación Angular	0.0180	0.0654	0.1799

4.4 Conclusiones parciales

Se demostró qué los modelos que utilizan información evolutiva a nivel de aminoácidos a partir de las incrustaciones ESM-2_t36 logran el mejor rendimiento, seguidos por aquellos que utilizan las incrustaciones ESM-2_t33 y ESM-2_t30. Esto coincide con resultados reportados en la literatura, que indican que los modelos ESM-2 de mayor capacidad tienden a ofrecer un rendimiento superior. Además, todos los modelos desarrollados en este estudio superaron consistentemente a más de 20 modelos del estado del arte basados únicamente en secuencias y construidos tanto con técnicas de DL como con clasificadores tradicionales. También se demostró que la metodología propuesta permite crear modelos codificando espacios químicos distintos, lo cual permitirá realizar predicciones basada en el consenso de diferentes modelos, particularmente para mejorar la recuperación de verdaderos negativos. Se demostró que los modelos son altamente dependientes de las estructuras predichas por ESMFold y de la información evolutiva de los modelos ESM-2 y, por lo tanto, cualquier pérdida de esta información significativamente

afecta su capacidad de generalización. Los modelos basados en mapas de contacto ESM-2 también mostraron un buen rendimiento, aunque los modelos generados a partir de coordenadas 3D demostraron ser estadísticamente superiores. En general, los resultados destacan el *framework* esm-AxP-GDL como una herramienta robusta para la construcción de modelos discriminativos dependientes de la estructura de péptidos e independientes de MSA. El análisis de las distancias inter-aminoácidos utilizando diversas funciones de distancia puede servir como base para estudios futuros que exploren la creación de grafos con topologías diversas.

Capítulo 5. Conclusiones

A partir de los resultados obtenidos en esta tesis se llegó a las siguientes conclusiones y recomendaciones para trabajo futuro:

5.1 Conclusiones

El *framework* esm-AxP-GDL permite construir modelos independientes de alineamiento utilizando grafos basados en distancia, los cuales se construyen a partir de estructuras de péptidos predichas por ESMFold. Los nodos en estos grafos están caracterizados con información evolutiva a nivel de aminoácidos derivada de los modelos ESM-2. La arquitectura del *framework* se basa en Redes de Atención a Grafos (GAT – Graph Attention Networks). Los modelos desarrollados en este estudio superaron consistentemente a más de 20 modelos reportados en la literatura que fueron solo entrenados con secuencias de aminoácidos, así como con técnicas de DL o clasificadores tradicionales. Esto remarca la importancia de aprovechar la información estructural de las estructuras de péptidos predichas. También se demostró que la metodología propuesta permite crear modelos codificando espacios químicos distintos, lo cual permitirá realizar predicciones basada en el consenso de diferentes modelos, particularmente para mejorar la recuperación de verdaderos negativos. Se demostró que los modelos son altamente dependientes de las estructuras predichas por ESMFold y de la información evolutiva de los modelos ESM-2. Cualquier pérdida de esta información afecta significativamente su capacidad de generalización. En general, los resultados destacan que la metodología propuesta en este trabajo es valiosa para la construcción de modelos discriminativos robustos, dependientes de la estructura, e independientes de alineamiento. El *framework* desarrollado puede ser aplicado no solo en la clasificación de AMPs, sino en la modelación y predicción de otras actividades biológicas de péptidos y proteínas.

Sin embargo, algunas limitaciones o aspectos a mejorar pueden ser resaltados. En este sentido, se puede mencionar que el *framework* presenta la limitación de utilizar una única representación de grafo por estructura 3D de péptido, lo que resulta insuficiente para capturar toda la información relacionada con las interacciones inter-aminoácidos. Como alternativa, se podrían desarrollar o utilizar enfoques basados en criterios fisicoquímicos, geométricos, y topológicos para construir grafos topológicamente diferentes por estructura predicha. De esta forma, se pudieran implementar arquitecturas de aprendizaje multi-instancia en lugar de usar una arquitectura de una única instancia representando cada estructura

peptídica predicha. Además, el enfoque propuesto, al ser una arquitectura de aprendizaje profundo, requiere suficiente cantidad de datos para construir los modelos y evitar crear modelos sobre entrenados. Finalmente, el *framework* carece de métodos de Inteligencia Artificial Explicable (XAI, por sus siglas en inglés), lo que limita la interpretabilidad de las predicciones.

5.2 Trabajo futuro

Como trabajo futuro, se recomienda realizar nuevos estudios que exploren el uso de diferentes funciones de distancia para crear representaciones de péptidos como grafos. Esto podría generar grafos con topologías variadas que codifiquen espacios químicos distintos, para de esta manera, potencialmente mejorar el rendimiento de los modelos basados en aprendizaje de grafos para péptidos. Asimismo, se sugiere implementar múltiples criterios para construir grafos topológicamente diferentes por estructura predicha e integrarlos en arquitecturas de aprendizaje multi-instancia. Además, se propone incorporar funcionalidades para determinar dominio de aplicabilidad de los modelos a partir del uso de la métrica de perplejidad y de métricas de centralidad de grafos. Además, se recomienda implementar o proponer métodos de XAI para la interpretación de las predicciones.

Literatura citada

Abdin, O., Nim, S., Wen, H., & Kim, P. M. (2022). PepNN: a deep attention model for the identification of peptide binding sites. *Communications Biology*, *5*(1), 503. https://doi.org/10.1038/s42003-022-03445-2

Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A. J., Bambrick, J., Bodenstein, S. W., Evans, D. A., Hung, C. C., O'Neill, M., Reiman, D., Tunyasuvunakool, K., Wu, Z., Žemgulytė, A., Arvaniti, E., … Jumper, J. M. (2024). Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, *630*(8016), 493-500. https://doi.org/10.1038/s41586-024-07487-w

Agüero-Chapin, G., Antunes, A., & Marrero-Ponce, Y. (2023). A 2022 Update on Computational Approaches to the Discovery and Design of Antimicrobial Peptides. *Antibiotics*, *12*(6), 1011. https://doi.org/10.3390/antibiotics12061011

Agüero-Chapin, G., Antunes, A., Mora, J. R., Pérez, N., Contreras-Torres, E., Valdes-Martini, J. R., Martinez-Rios, F., Zambrano, C. H., & Marrero-Ponce, Y. (2023). Complex Networks Analyses of Antibiofilm Peptides: An Emerging Tool for Next-Generation Antimicrobials' Discovery. *Antibiotics*, *12*(4), 747. https://doi.org/10.3390/antibiotics12040747

Aguilera-Mendoza, L., Ayala-Ruano, S., Martinez-Rios, F., Chavez, E., García-Jacas, C. R., Brizuela, C. A., & Marrero-Ponce, Y. (2023). StarPep Toolbox: an open-source software to assist chemical space analysis of bioactive peptides and their functions using complex networks. *Bioinformatics*, *39*(8), btad506. https://doi.org/10.1093/bioinformatics/btad506

Aguilera-Mendoza, L., Marrero-Ponce, Y., Beltran, J. A., Tellez Ibarra, R., Guillen-Ramirez, H. A., & Brizuela, C. A. (2019). Graph-based data integration from bioactive peptide databases of pharmaceutical interest: toward an organized collection enabling visual network analysis. *Bioinformatics*, *35*(22), 4739-4747. https://doi.org/10.1093/bioinformatics/btz260

Aguilera-Puga, M. d. C., Cancelarich, N. L., Marani, M. M., de la Fuente-Nunez, C., & Plisson, F. (2024). Accelerating the Discovery and Design of Antimicrobial Peptides with Artificial Intelligence. In M. Gore & U. B. Jagtap (Eds.), *Computational Drug Discovery and Design* (pp. 329-352). Humana New York. https://doi.org/10.1007/978-1-0716-3441-7_18

Ahmed, A., Siman-Tov, G., Hall, G., Bhalla, N., & Narayanan, A. (2019). Human Antimicrobial Peptides as Therapeutics for Viral Infections. *Viruses*, *11*(8), 704. https://doi.org/10.3390/v11080704

Alberts, Bruce., Hopkin, K., Johnson, A., Morgan, D., Raff, M., Roberts, K., & Walter, P. (2019). *Essential Cell Biology* (5th ed.). W. W. Norton & Company. https://dokumen.pub/essential-cell-biology-5th-edition-9780393691092.html

Ayala-Ruano, S., Marrero-Ponce, Y., Aguilera-Mendoza, L., Pérez, N., Agüero-Chapin, G., Antunes, A., & Aguilar, A. C. (2022). Network Science and Group Fusion Similarity-Based Searching to Explore the Chemical Space of Antiparasitic Peptides. *ACS Omega*, *7*(50), 46012-46036. https://doi.org/10.1021/acsomega.2c03398

Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G. R., Wang, J., Cong, Q., Kinch, L. N., Dustin Schaeffer, R., Millán, C., Park, H., Adams, C., Glassman, C. R., DeGiovanni, A., Pereira, J. H., Rodrigues, A. V., Van Dijk, A. A., Ebrecht, A. C., … Baker, D. (2021). Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, *373*(6557), 871-876. https://doi.org/10.1126/science.abj8754

Bairoch, A., Apweiler, R., Wu, C. H., Barker, W. C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Martin, M. J., Natale, D. A., O'Donovan, C., Redaschi, N., & Yeh, L. S. L. (2005). The Universal Protein Resource (UniProt). *Nucleic Acids Research*, *33*(suppl_1), D154-D159. https://doi.org/10.1093/nar/gki070

Baranwal, M., Magner, A., Saldinger, J., Turali-Emre, E. S., Elvati, P., Kozarekar, S., VanEpps, J. S., Kotov, N. A., Violi, A., & Hero, A. O. (2022). Struct2Graph: a graph attention network for structure based predictions of protein–protein interactions. *BMC Bioinformatics*, *23*(1), 370. https://doi.org/10.1186/s12859-022-04910-9

Barnard, J. (2023, diciembre 22). *IBM*. What is Embedding? https://www.ibm.com/topics/embedding

Bateman, A. (2019). UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Research*, *47*(D1), D506-D515. https://doi.org/10.1093/nar/gky1049

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., & Bourne, P. E. (2000). The Protein Data Bank. *Nucleic Acids Research*, *28*(1), 235-242. https://doi.org/10.1093/nar/28.1.235

Bhadra, P., Yan, J., Li, J., Fong, S., & Siu, S. W. I. (2018). AmPEP: Sequence-based prediction of antimicrobial peptides using distribution patterns of amino acid properties and random forest. *Scientific Reports*, *8*(1), 1697. https://doi.org/10.1038/s41598-018-19752-w

Brice, D. C., & Diamond, G. (2019). Antiviral Activities of Human Host Defense Peptides. *Current Medicinal Chemistry*, *27*(9), 1420-1443. https://doi.org/10.2174/0929867326666190805151654

Brody, S., Alon, U., & Yahav, E. (2021). How Attentive are Graph Attention Networks? *Proceedings of the 10th International Conference on Learning Representations (ICLR 2022)*, 1-26. https://doi.org/10.48550/arXiv.2105.14491

Bucataru, C., & Ciobanasu, C. (2024). Antimicrobial peptides: Opportunities and challenges in overcoming resistance. *Microbiological Research*, *286*, 127822. https://doi.org/10.1016/J.MICRES.2024.127822

Cai, J., Li, X., Du, H., Jiang, C., Xu, S., & Cao, Y. (2020). Immunomodulatory significance of natural peptides in mammalians: Promising agents for medical application. *Immunobiology*, *225*(3), 151936. https://doi.org/10.1016/j.imbio.2020.151936

Carugo, O., & Djinović-Carugo, K. (2023). Structural biology: A golden era. *PLOS Biology*, *21*(6), e3002187. https://doi.org/10.1371/journal.pbio.3002187

Centers for Disease Control and Prevention (2024, abril 22). *Antimicrobial Resistance*. https://www.cdc.gov/antimicrobial-resistance/about/index.html

Chung, C. R., Kuo, T. R., Wu, L. C., Lee, T. Y., & Horng, J. T. (2020). Characterization and identification of antimicrobial peptides with different functional activities. *Briefings in Bioinformatics*, *21*(3), 1098-1114. https://doi.org/10.1093/bib/bbz043

Cordoves-Delgado, G., & García-Jacas, C. R. (2024). Predicting Antimicrobial Peptides Using ESMFold-Predicted Structures and ESM-2-Based Amino Acid Features with Graph Deep Learning. *Journal of Chemical Information and Modeling*, *64*(10), 4310-4321. https://doi.org/10.1021/acs.jcim.3c02061

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). *Introduction to algorithms* (4th ed.). MIT Press. https://mitpress.mit.edu/9780262046305/introduction-to-algorithms/

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, *1*(1), 3-18. https://doi.org/10.1016/j.swevo.2011.02.002

Devi, M. S., & Sashidhar, R. B. (2019). Antiaflatoxigenic effects of selected antifungal peptides. *Peptides*, *115*, 15-26. https://doi.org/10.1016/j.peptides.2019.02.006

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *North American Chapter of the Association for Computational Linguistics*, *1*, 4171-4186. https://doi.org/10.48550/arXiv.1810.04805

Deza, M. M., & Deza, E. (2016). *Encyclopedia of Distances* (4th ed.). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-52844-0

Divyashree, M., Mani, M. K., Reddy, D., Kumavath, R., Ghosh, P., Azevedo, V., & Barh, D. (2019). Clinical Applications of Antimicrobial Peptides (AMPs): Where do we Stand Now? *Protein & Peptide Letters*, *27*(2), 120-134. https://doi.org/10.2174/0929866526666190925152957

Du, Z., Ding, X., Hsu, W., Munir, A., Xu, Y., & Li, Y. (2024). pLM4ACE: A protein language model based predictor for antihypertensive peptide screening. *Food Chemistry*, *431*, 137162. https://doi.org/10.1016/j.foodchem.2023.137162

Du, Z., Ding, X., Xu, Y., & Li, Y. (2023). UniDL4BioPep: a universal deep learning architecture for binary classification in peptide bioactivity. *Briefings in Bioinformatics*, *24*(3), bbad135. https://doi.org/10.1093/bib/bbad135

Du, Z., Su, H., Wang, W., Ye, L., Wei, H., Peng, Z., Anishchenko, I., Baker, D., & Yang, J. (2021). The trRosetta server for fast and accurate protein structure prediction. *Nature Protocols*, *16*(12), 5634-5651. https://doi.org/10.1038/s41596-021-00628-9

Du, Z., Xu, Y., Liu, C., & Li, Y. (2024). pLM4Alg: Protein Language Model-Based Predictors for Allergenic Proteins and Peptides. *Journal of Agricultural and Food Chemistry*, *72*(1), 752-760. https://doi.org/10.1021/acs.jafc.3c07143

Dubitzky, W., Wolkenhauer, O., Cho, K.-H., & Yokota, H. (2013). *Encyclopedia of Systems Biology*. Springer New York. https://doi.org/10.1007/978-1-4419-9863-7

Durairaj, J., de Ridder, D., & van Dijk, A. D. J. (2023). Beyond sequence: Structure-based machine learning. *Computational and Structural Biotechnology Journal*, *21*, 630-643. https://doi.org/10.1016/j.csbj.2022.12.039

Emonts, J., & Buyel, J. F. (2023). An overview of descriptors to capture protein properties – Tools and perspectives in the context of QSAR modeling. *Computational and Structural Biotechnology Journal*, *21*, 3234-3247. https://doi.org/10.1016/j.csbj.2023.05.022

Fang, X., Wang, F., Liu, L., He, J., Lin, D., Xiang, Y., Zhu, K., Zhang, X., Wu, H., Li, H., & Song, L. (2023). A method for multiple-sequence-alignment-free protein structure prediction using a protein language model. *Nature Machine Intelligence*, *5*(10), 1087-1096. https://doi.org/10.1038/s42256-023-00721-6

Fasoulis, R., Paliouras, G., & Kavraki, L. E. (2021). Graph representation learning for structural proteomics. *Emerging Topics in Life Sciences*, *5*(6), 789-802. https://doi.org/10.1042/etls20210225

Fernández de Ullivarri, M., Arbulu, S., Garcia-Gutierrez, E., & Cotter, P. D. (2020). Antifungal Peptides as Therapeutic Agents. *Frontiers in Cellular and Infection Microbiology*, *10*(105). https://doi.org/10.3389/fcimb.2020.00105

Fey, M., & Lenssen, J. E. (2019). Fast Graph Representation Learning with PyTorch Geometric. *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*, 1-9. https://doi.org/10.48550/arXiv.1903.02428

Freeman, E., & Robson, E. (2021). *Head First Design Patterns: Building Extensible and Maintainable Object-Oriented Software* (2nd ed.). O'Reilly Media. https://www.oreilly.com/library/view/head-first-design/9781492077992/

Fu, H., Cao, Z., Li, M., & Wang, S. (2020). ACEP: Improving antimicrobial peptides recognition through automatic feature fusion and amino acid embedding. *BMC Genomics*, *21*(1), 597. https://doi.org/10.1186/s12864-020-06978-0

Gabere, M. N., & Noble, W. S. (2017). Empirical comparison of web-based antimicrobial peptide prediction tools. *Bioinformatics*, *33*(13), 1921-1929. https://doi.org/10.1093/bioinformatics/btx081

Gao, H., & Ji, S. (2019). Graph U-Nets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(9), 4948-4960. https://doi.org/10.1109/tpami.2021.3081010

Gao, Z., Jiang, C., Zhang, J., Jiang, X., Li, L., Zhao, P., Yang, H., Huang, Y., & Li, J. (2023). Hierarchical graph learning for protein–protein interaction. *Nature Communications*, *14*(1), 1093. https://doi.org/10.1038/s41467-023-36736-1

García-Jacas, C. R., García-González, L. A., Martinez-Rios, F., Tapia-Contreras, I. P., & Brizuela, C. A. (2022). Handcrafted versus non-handcrafted (self-supervised) features for the classification of antimicrobial peptides: complementary or redundant? *Briefings in Bioinformatics*, *23*(6), bbac428. https://doi.org/10.1093/bib/bbac428

García-Jacas, C. R., Pinacho-Castellanos, S. A., García-González, L. A., & Brizuela, C. A. (2022). Do deep learning models make a difference in the identification of antimicrobial peptides? *Briefings in Bioinformatics*, *23*(3), bbac094. https://doi.org/10.1093/bib/bbac094

Ge, J., Jiang, D., Sun, H., Kang, Y., Pan, P., Deng, Y., Hsieh, C. Y., & Hou, T. (2024). Deep-learning-based prediction framework for protein-peptide interactions with structure generation pipeline. *Cell Reports Physical Science*, *5*(6), 101980. https://doi.org/10.1016/j.xcrp.2024.101980

Gligorijević, V., Renfrew, P. D., Kosciolek, T., Leman, J. K., Berenberg, D., Vatanen, T., Chandler, C., Taylor, B. C., Fisk, I. M., Vlamakis, H., Xavier, R. J., Knight, R., Cho, K., & Bonneau, R. (2021). Structure-based protein function prediction using graph convolutional networks. *Nature Communications*, *12*(1), 3168. https://doi.org/10.1038/s41467-021-23303-9

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. https://mitpress.mit.edu/9780262035613/deep-learning/

Gore, M., & Jagtap, U. B. (2024). *Computational Drug Discovery and Design* (2nd ed.). Humana Press. https://doi.org/10.1007/978-1-0716-3441-7

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive Representation Learning on Large Graphs. *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS 2017)*, 1024-1034. https://doi.org/10.48550/arXiv.1706.02216

Hancock, R. E. W., Haney, E. F., & Gill, E. E. (2016). The immunology of host defence peptides: beyond antimicrobial activity. *Nature Reviews Immunology*, *16*(5), 321-334. https://doi.org/10.1038/nri.2016.29

Harris, J. R., & Korolchuk, V. I. (2023). *Biochemistry and Cell Biology of Ageing: Part IV, Clinical Science*. Springer Cham. https://doi.org/10.1007/978-3-031-26576-1

Hayes, T., Rao, R., Akin, H., Sofroniew, N. J., Oktay, D., Lin, Z., Verkuil, R., Tran, V. Q., Deaton, J., Wiggert, M., Badkundri, R., Shafkat, I., Gong, J., Derry, A., Molina, R. S., Thomas, N., Khan, Y., Mishra, C., Kim, C., … Rives, A. (2024). Simulating 500 million years of evolution with a language model. *bioRxiv*. https://doi.org/10.1101/2024.07.01.600583

Holm, S. (1979). A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics*, *6*(2), 65-70. https://doi.org/10.2307/4615733

Hu, Y., Cheng, K., He, L., Zhang, X., Jiang, B., Jiang, L., Li, C., Wang, G., Yang, Y., & Liu, M. (2021). NMR-Based Methods for Protein Analysis. *Analytical Chemistry*, *93*(4), 1866-1879. https://doi.org/10.1021/acs.analchem.0c03830

Huan, Y., Kong, Q., Mou, H., & Yi, H. (2020). Antimicrobial Peptides: Classification, Design, Application and Research Progress in Multiple Fields. *Frontiers in Microbiology*, *11*, 582779. https://doi.org/10.3389/fmicb.2020.582779

Huyen, C. (2019, octubre 18). *Evaluation Metrics for Language Modeling*. The Gradient. https://thegradient.pub/understanding-evaluation-metrics-for-language-models/

Jamasb, A. R., Viñas, R., Ma, E. J., Harris, C., Huang, K., Hall, D., Lió, P., & Blundell, T. L. (2021). Graphein - a Python Library for Geometric Deep Learning and Network Analysis on Protein Structures and Interaction Networks. *Advances in Neural Information Processing Systems*, *35*, 27153-27167. https://doi.org/10.1101/2020.07.15.204701

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., … Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, *596*(7873), 583-589. https://doi.org/10.1038/s41586-021-03819-2

Kang, X., Dong, F., Shi, C., Liu, S., Sun, J., Chen, J., Li, H., Xu, H., Lao, X., & Zheng, H. (2019). DRAMP 2.0, an updated data repository of antimicrobial peptides. *Scientific Data*, *6*(1), 1-10. https://doi.org/10.1038/s41597-019-0154-y

Khemani, B., Patil, S., Kotecha, K., & Tanwar, S. (2024). A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, *11*(1), 1-43. https://doi.org/10.1186/s40537-023-00876-4/tables/13

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*, 1-15. https://doi.org/10.48550/arXiv.1412.6980

Kumar, P., Kizhakkedathu, J. N., & Straus, S. K. (2018). Antimicrobial Peptides: Diversity, Mechanism of Action and Strategies to Improve the Activity and Biocompatibility In Vivo. *Biomolecules 2018*. *8*(1), 4. https://doi.org/10.3390/biom8010004

Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, *51*(2), 181-207. https://doi.org/10.1023/a:1022859003006

Lacerda, A. F., Pelegrini, P. B., De Oliveira, D. M., Vasconcelos, É. A. R., & Grossi-de-Sá, M. F. (2016). Anti-parasitic peptides from arthropods and their application in drug therapy. *Frontiers in Microbiology*, *7*(91). https://doi.org/10.3389/fmicb.2016.00091

Larkin, H. (2023). Increasing Antimicrobial Resistance Poses Global Threat, WHO Says. *JAMA*, *329*(3), 200-200. https://doi.org/10.1001/jama.2022.23552

Lee, H., Lee, S., Lee, I., & Nam, H. (2023). AMP-BERT: Prediction of antimicrobial peptide function based on a BERT model. *Protein Science*, *32*(1), e4529. https://doi.org/10.1002/pro.4529

Lee, H. T., Lee, C. C., Yang, J. R., Lai, J. Z. C., Chang, K. Y., & Ray, O. (2015). A Large-Scale Structural Classification of Antimicrobial Peptides. *BioMed Research International*, *2015*(1), 475062. https://doi.org/10.1155/2015/475062

Lei, J., Sun, L. C., Huang, S., Zhu, C., Li, P., He, J., Mackey, V., Coy, D. H., & He, Q. Y. (2019). The antimicrobial peptides and their potential clinical applications. *American Journal of Translational Research*, *11*(7), 3919-3931. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6684887/

Li, J., Pu, Y., Tang, J., Zou, Q., & Guo, F. (2020). DeepAVP: A Dual-Channel Deep Neural Network for Identifying Variable-Length Antiviral Peptides. *IEEE Journal of Biomedical and Health Informatics*, *24*(10), 3012-3019. https://doi.org/10.1109/jbhi.2020.2977091

Li, T., Wang, Z., Guo, J., de la Fuente-Nunez, C., Wang, J., Han, B., Tao, H., Liu, J., & Wang, X. (2023). Bacterial resistance to antibacterial agents: Mechanisms, control strategies, and implications for global health. *Science of The Total Environment*, *860*, 160461. https://doi.org/10.1016/j.scitotenv.2022.160461

Lin, W., & Xu, D. (2016). Imbalanced multi-label learning for identifying antimicrobial peptides and their functional types. *Bioinformatics*, *32*(24), 3745-3752. https://doi.org/10.1093/bioinformatics/btw560

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., dos Santos Costa, A., Fazel-Zarandi, M., Sercu, T., Candido, S., & Rives, A. (2023). Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, *379*(6637), 1123-1130. https://doi.org/10.1126/science.ade2574

Liu, Y., Ding, S., Shen, J., & Zhu, K. (2019). Nonribosomal antibacterial peptides that target multidrug-resistant bacteria. *Natural Product Reports*, *36*(4), 573-592. https://doi.org/10.1039/c8np00031j

Ma, Y., Guo, Z., Xia, B., Zhang, Y., Liu, X., Yu, Y., Tang, N., Tong, X., Wang, M., Ye, X., Feng, J., Chen, Y., & Wang, J. (2022). Identification of antimicrobial peptides from the human gut microbiome using deep learning. *Nature Biotechnology*, *40*(6), 921-931. https://doi.org/10.1038/s41587-022-01226-0

Maljković, M. M. (2019). DistAA: Database of amino acid distances in proteins and web application for statistical review of distances. *Computational Biology and Chemistry*, *83*, 107130. https://doi.org/10.1016/j.compbiolchem.2019.107130

Marrero-Ponce, Y., García-Jacas, C. R., Barigye, S. J., Valdés-Martiní, J. R., Rivera-Borroto, O. M., Pino-Urias, R. W., Cubillán, N., Alvarado, Y. J., & Le-Thi-Thu, H. (2015). Optimum Search Strategies or Novel 3D Molecular Descriptors: is there a Stalemate? *Current Bioinformatics*, *10*(5), 533-564. https://doi.org/10.2174/1574893610666151008011457

Marrero-Ponce, Y., Teran, J. E., Contreras-Torres, E., García-Jacas, C. R., Perez-Castillo, Y., Cubillan, N., Peréz-Giménez, F., & Valdés-Martini, J. R. (2020). LEGO-based generalized set of two linear algebraic 3D bio-macro-molecular descriptors: Theory and validation by QSARs. *Journal of Theoretical Biology*, *485*, 110039. https://doi.org/10.1016/j.jtbi.2019.110039

Martínez-Mauricio, K. L., García-Jacas, C. R., & Cordoves-Delgado, G. (2024). Examining evolutionary scale modeling-derived different-dimensional embeddings in the antimicrobial peptide classification through a KNIME workflow. *Protein Science*, *33*(4), e4928. https://doi.org/10.1002/pro.4928

Maveyraud, L., & Mourey, L. (2020). Protein X-ray Crystallography and Drug Discovery. *Molecules*, *25*(5), 1030. https://doi.org/10.3390/molecules25051030

Meher, P. K., Sahu, T. K., Saini, V., & Rao, A. R. (2017). Predicting antimicrobial peptides with improved accuracy by incorporating the compositional, physico-chemical and structural features into Chou's general PseAAC. *Scientific Reports*, *7*(1), 42362. https://doi.org/10.1038/srep42362

Mishra, P., Pandey, C. M., Singh, U., Gupta, A., Sahu, C., & Keshri, A. (2019). Descriptive Statistics and Normality Tests for Statistical Data. *Annals of Cardiac Anaesthesia*, *22*(1), 67-72. https://doi.org/10.4103/aca.aca_157_18

Mohammadi, A., Zahiri, J., Mohammadi, S., Khodarahmi, M., & Arab, S. S. (2022). PSSMCOOL: a comprehensive R package for generating evolutionary-based descriptors of protein sequences from PSSM profiles. *Biology Methods and Protocols*, *7*(1), bpac008. https://doi.org/10.1093/biomethods/bpac008

Mor, A. (2009). Multifunctional host defense peptides: antiparasitic activities. *FEBS Journal*, *276*(22), 6474-6482. https://doi.org/10.1111/j.1742-4658.2009.07358.x

Muratov, E. N., Bajorath, J., Sheridan, R. P., Tetko, I. V., Filimonov, D., Poroikov, V., Oprea, T. I., Baskin, I. I., Varnek, A., Roitberg, A., Isayev, O., Curtalolo, S., Fourches, D., Cohen, Y., Aspuru-Guzik, A., Winkler, D. A., Agrafiotis, D., Cherkasov, A., & Tropsha, A. (2020). QSAR without borders. *Chemical Society Reviews*, *49*(11), 3525-3564. https://doi.org/10.1039/d0cs00098a

Nair, V., & Hinton, G. E. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines. *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML 10)*, 807-814. https://doi.org/10.5555/3104322.3104425

Nikolentzos, G., Dasoulas, G., & Vazirgiannis, M. (2019). k-hop Graph Neural Networks. *Neural Networks*, *130*, 195-205. https://doi.org/10.1016/j.neunet.2020.07.008

Paruchuri, T., Kancharla, G. R., Dara, S., Yadav, R. K., Jadav, S. S., Dhamercherla, S., & Vidyarthi, A. (2022). Nature Inspired Algorithms for Solving Multiple Sequence Alignment Problem: A Review. *Archives of Computational Methods in Engineering*, *29*(7), 5237-5258. https://doi.org/10.1007/s11831-022-09769-w

Pinacho-Castellanos, S. A., García-Jacas, C. R., Gilson, M. K., & Brizuela, C. A. (2021). Alignment-Free Antimicrobial Peptide Predictors: Improving Performance by a Thorough Analysis of the Largest Available Data Set. *Journal of Chemical Information and Modeling*, *61*(6), 3141-3157. https://doi.org/10.1021/acs.jcim.1c00251

Pirtskhalava, M., Gabrielian, A., Cruz, P., Griggs, H. L., Squires, R. B., Hurt, D. E., Grigolava, M., Chubinidze, M., Gogoladze, G., Vishnepolsky, B., Alekseev, V., Rosenthal, A., & Tartakovsky, M. (2016). DBAASP v.2: an enhanced database of structure and antimicrobial/cytotoxic activity of natural and synthetic peptides. *Nucleic Acids Research*, *44*(D1), D1104-D1112. https://doi.org/10.1093/nar/gkv1174

Pretzel, J., Mohring, F., Rahlfs, S., & Becker, K. (2013). Antiparasitic peptides. In A. Vilcinskas (Ed.), *Yellow Biotechnology I: Insect Biotechnologie in Drug Discovery and Preclinical Research* (pp. 157-192). Springer Berlin Heidelberg. https://doi.org/10.1007/10_2013_191

Radek, K., & Gallo, R. (2007). Antimicrobial peptides: Natural effectors of the innate immune system. *Seminars in Immunopathology*, *29*(1), 27-43. https://doi.org/10.1007/s00281-007-0064-5

Ranganathan, S., Gribskov, M., Nakai, K., & Schönbach, C. (2019). *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*. Academic Press. https://doi.org/10.1016/b978-0-12-809633-8.20318-x

Rao, R., Meier, J., Sercu, T., Ovchinnikov, S., & Rives, A. (2020). Transformer protein language models are unsupervised structure learners. *Proceedings of the 9th International Conference on Learning Representations (ICLR 2021)*, 1-24. https://doi.org/10.1101/2020.12.15.422761

Réau, M., Renaud, N., Xue, L. C., & Bonvin, A. M. J. J. (2023). DeepRank-GNN: a graph neural network framework to learn patterns in protein–protein interfaces. *Bioinformatics*, *39*(1), btac759. https://doi.org/10.1093/bioinformatics/btac759

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., & Fergus, R. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, *118*(15), e2016239118. https://doi.org/10.1073/pnas.2016239118

Rodríguez, A. A., Otero-González, A., Ghattas, M., & Ständker, L. (2021). Discovery, Optimization, and Clinical Application of Natural Antimicrobial Peptides. *Biomedicines*, *9*(10), 1381. https://doi.org/10.3390/biomedicines9101381

Ruffolo, J. A., & Madani, A. (2024). Designing proteins with language models. *Nature Biotechnology 2024 42:2*, *42*(2), 200-202. https://doi.org/10.1038/s41587-024-02123-4

Ruiz-Blanco, Y. B., Agüero-Chapin, G., Romero-Molina, S., Antunes, A., Olari, L. R., Spellerberg, B., Münch, J., & Sanchez-Garcia, E. (2022). ABP-Finder: A Tool to Identify Antibacterial Peptides and the Gram-Staining Type of Targeted Bacteria. *Antibiotics*, *11*(12), 1708. https://doi.org/10.3390/antibiotics11121708/s1

Santos-Júnior, C. D., Torres, M. D. T., Duan, Y., Rodríguez Del Río, Á., Schmidt, T. S. B., Chong, H., Fullam, A., Kuhn, M., Zhu, C., Houseman, A., Somborski, J., Vines, A., Zhao, X.-M., Bork, P., Huerta-Cepas, J., de la Fuente-Nunez, C., & Coelho, L. P. (2024). Discovery of antimicrobial peptides in the global microbiome with machine learning. *Cell*, *187*(14), 3761-3778. https://doi.org/10.1016/j.cell.2024.05.013

Schoch, C. L., Ciufo, S., Domrachev, M., Hotton, C. L., Kannan, S., Khovanskaya, R., Leipe, D., McVeigh, R., O'Neill, K., Robbertse, B., Sharma, S., Soussov, V., Sullivan, J. P., Sun, L., Turner, S., & Karsch-Mizrachi, I. (2020). NCBI Taxonomy: a comprehensive update on curation, resources and tools. *Database*, *2020*, baaa062. https://doi.org/10.1093/database/baaa062

Schwede, T. (2013). Protein Modeling: What Happened to the "Protein Structure Gap"? *Structure*, *21*(9), 1531-1540. https://doi.org/10.1016/j.str.2013.08.007

Seetharama, D. J. (2022). *Peptide Therapeutics: Fundamentals of Design, Development, and Delivery*. Springer Cham. https://doi.org/https://doi.org/10.1007/978-3-031-04544-8

Sharma, R., Shrivastava, S., Kumar Singh, S., Kumar, A., Saxena, S., & Kumar Singh, R. (2021a). AniAMPpred: artificial intelligence guided discovery of novel antimicrobial peptides in animal kingdom. *Briefings in Bioinformatics*, *22*(6), bbab242. https://doi.org/10.1093/bib/bbab242

Sharma, R., Shrivastava, S., Kumar Singh, S., Kumar, A., Saxena, S., & Kumar Singh, R. (2021b). Deep-ABPpred: identifying antibacterial peptides in protein sequences using bidirectional LSTM with word2vec. *Briefings in Bioinformatics*, *22*(5), bbab065. https://doi.org/10.1093/bib/bbab065

Sharma, R., Shrivastava, S., Singh, S. K., Kumar, A., Saxena, S., & Singh, R. K. (2022). Deep-AFPpred: identifying novel antifungal peptides using pretrained embeddings from seq2vec with 1DCNN-BiLSTM. *Briefings in Bioinformatics*, *23*(1), bbab422. https://doi.org/10.1093/bib/bbab422

Sharma, R., Shrivastava, S., Singh, S. K., Kumar, A., Singh, A. K., & Saxena, S. (2022). Deep-AVPpred: Artificial Intelligence Driven Discovery of Peptide Drugs for Viral Infections. *IEEE Journal of Biomedical and Health Informatics*, *26*(10), 5067-5074. https://doi.org/10.1109/jbhi.2021.3130825

Sidorczuk, K., Gagat, P., Pietluch, F., Kała, J., Rafacz, D., Bąkała, L., Słowik, J., Kolenda, R., Rödiger, S., Fingerhut, L. C. H. W., Cooke, I. R., MacKiewicz, P., & Burdukiewicz, M. (2022). Benchmarks in antimicrobial peptide prediction are biased due to the selection of negative data. *Briefings in Bioinformatics*, *23*(5), bbac343. https://doi.org/10.1093/bib/bbac343

Simonson, T. (2022). *Computational Peptide Science: Methods and Protocols*. Humana Press. https://doi.org/10.1007/978-1-0716-1855-4

Singh, S., Chaudhary, K., Dhanda, S. K., Bhalla, S., Usmani, S. S., Gautam, A., Tuknait, A., Agrawal, P., Mathur, D., & Raghava, G. P. S. (2016). SATPdb: a database of structurally annotated therapeutic peptides. *Nucleic Acids Research*, *44*(D1), D1119-D1126. https://doi.org/10.1093/nar/gkv1114

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929-1958. https://doi.org/10.5555/2627435.2670313

Su, X., Xu, J., Yin, Y., Quan, X., & Zhang, H. (2019). Antimicrobial peptide identification using multi-scale convolutional network. *BMC Bioinformatics*, *20*(1), 730. https://doi.org/10.1186/s12859-019-3327-y

Terán, J. E., Marrero-Ponce, Y., Contreras-Torres, E., García-Jacas, C. R., Vivas-Reyes, R., Terán, E., & Torres, F. J. (2019). Tensor Algebra-based Geometrical (3D) Biomacro-Molecular Descriptors for Protein Research: Theory, Applications and Comparison with other Methods. *Scientific Reports*, *9*(1), 11391. https://doi.org/10.1038/s41598-019-47858-2

Thakur, N., Qureshi, A., & Kumar, M. (2012). AVPpred: collection and prediction of highly effective antiviral peptides. *Nucleic Acids Research*, *40*(W1), W199-W204. https://doi.org/10.1093/nar/gks450

Théolier, J., Fliss, I., Jean, J., & Hammami, R. (2014). MilkAMP: A comprehensive database of antimicrobial peptides of dairy origin. *Dairy Science and Technology*, *94*(2), 181-193. https://doi.org/10.1007/s13594-013-0153-2

Todeschini, R., & Consonni, V. (2009). *Molecular Descriptors for Chemoinformatics* (2nd ed.). Wiley-VCH. https://doi.org/10.1002/9783527628766

Torrent, M., Andreu, D., Nogués, V. M., & Boix, E. (2011). Connecting Peptide Physicochemical and Antimicrobial Properties by a Rational Prediction Model. *PLOS ONE*, *6*(2), e16968. https://doi.org/10.1371/journal.pone.0016968

Usmani, S. S., Bedi, G., Samuel, J. S., Singh, S., Kalra, S., Kumar, P., Ahuja, A. A., Sharma, M., Gautam, A., & Raghava, G. P. S. (2017). THPdb: Database of FDA-approved peptide and protein therapeutics. *PLOS ONE*, *12*(7), e0181748. https://doi.org/10.1371/journal.pone.0181748

Veličković, P., Casanova, A., Liò, P., Cucurull, G., Romero, A., & Bengio, Y. (2017). Graph Attention Networks. *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*, 1-24. https://doi.org/10.48550/arXiv.1710.10903

Veltri, D., Kamath, U., & Shehu, A. (2018). Deep learning improves antimicrobial peptide recognition. *Bioinformatics*, *34*(16), 2740-2747. https://doi.org/10.1093/bioinformatics/bty179

Vilas Boas, L. C. P., Campos, M. L., Berlanda, R. L. A., de Carvalho Neves, N., & Franco, O. L. (2019). Antiviral peptides as promising therapeutic drugs. *Cellular and Molecular Life Sciences 2019 76:18*, *76*(18), 3525-3542. https://doi.org/10.1007/s00018-019-03138-w

Waghu, F. H., Barai, R. S., Gurung, P., & Idicula-Thomas, S. (2016). CAMPR3: a database on sequences, structures and signatures of antimicrobial peptides. *Nucleic Acids Research*, *44*(D1), D1094-D1097. https://doi.org/10.1093/nar/gkv1051

Waghu, F. H., Joseph, S., Ghawali, S., Martis, E. A., Madan, T., Venkatesh, K. V., & Idicula-Thomas, S. (2018). Designing antibacterial peptides with enhanced killing kinetics. *Frontiers in Microbiology*, *9*, 320734. https://doi.org/10.3389/fmicb.2018.00325

Wang, G., Li, X., & Wang, Z. (2016). APD3: the antimicrobial peptide database as a tool for research and education. *Nucleic Acids Research*, *44*(D1), D1087-D1093. https://doi.org/10.1093/nar/gkv1278

Wang, H., Zhao, J., Zhao, H., Li, H., & Wang, J. (2021). CL-ACP: a parallel combination of CNN and LSTM anticancer peptide recognition model. *BMC Bioinformatics*, *22*(1), 512. https://doi.org/10.1186/s12859-021-04433-9

Wei, Gang., & Kumbar, S. G. (2020). *Artificial Protein and Peptide Nanofibers: Design, Fabrication, Characterization, and Applications*. Woodhead Publishing. https://dokumen.pub/artificial-protein-and-peptide-nanofibers-design-fabrication-characterization-and-applications-woodhead-publishing-series-in-biomaterials-1nbsped-0081028504-9780081028506.html

WHO. (2023, noviembre 21). *Antimicrobial resistance*. https://www.who.int/news-room/fact-sheets/detail/antimicrobial-resistance

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4-24. https://doi.org/10.1109/tnnls.2020.2978386

Xiao, X., Wang, P., Lin, W. Z., Jia, J. H., & Chou, K. C. (2013). iAMP-2L: A two-level multi-label classifier for identifying antimicrobial peptides and their functional types. *Analytical Biochemistry*, *436*(2), 168-177. https://doi.org/10.1016/j.ab.2013.01.019

Xu, J., & Zhang, Y. (2010). How significant is a protein structure similarity with TM-score = 0.5? *Bioinformatics*, *26*(7), 889-895. https://doi.org/10.1093/bioinformatics/btq066

Yan, J., Bhadra, P., Li, A., Sethiya, P., Qin, L., Tai, H. K., Wong, K. H., & Siu, S. W. I. (2020). Deep-AmPEP30: Improve Short Antimicrobial Peptides Prediction with Deep Learning. *Molecular Therapy Nucleic Acids*, *20*, 882-894. https://doi.org/10.1016/j.omtn.2020.05.006

Yan, K., Lv, H., Guo, Y., Peng, W., & Liu, B. (2023). sAMPpred-GAT: prediction of antimicrobial peptide by graph attention network and predicted peptide structure. *Bioinformatics*, *39*(1), btac715. https://doi.org/10.1093/bioinformatics/btac715

Yan, X. (2024). *Peptide Self-Assembly and Engineering: Fundamentals, Structures, and Applications: Volume 1-2*. Wiley-VCH. https://doi.org/10.1002/9783527841264

Zankov, D., Madzhidov, T., Varnek, A., & Polishchuk, P. (2024). Chemical complexity challenge: Is multi-instance machine learning a solution? *Wiley Interdisciplinary Reviews: Computational Molecular Science*, *14*(1), e1698. https://doi.org/10.1002/wcms.1698

Zasloff, M., & Neundorf, I. (2019). Antimicrobial Peptides of Multicellular Organisms: My Perspective. En K. Matsuzaki (Ed.), *Antimicrobial Peptides: Basics for Clinical Application* (pp. 3-6). Springer, Singapore. https://doi.org/10.1007/978-981-13-3588-4_1

Zhang, Q. Y., Yan, Z. Bin, Meng, Y. M., Hong, X. Y., Shao, G., Ma, J. J., Cheng, X. R., Liu, J., Kang, J., & Fu, C. Y. (2021). Antimicrobial peptides: mechanism of action, activity and clinical potential. *Military Medical Research 2021 8:1*, *8*(1), 1-25. https://doi.org/10.1186/S40779-021-00343-2

Zhang, X. M., Liang, L., Liu, L., & Tang, M. J. (2021). Graph Neural Networks and Their Current Applications in Bioinformatics. *Frontiers in Genetics*, *12*, 690049. https://doi.org/10.3389/fgene.2021.690049

Zhang, Y., Lin, J., Zhao, L., Zeng, X., & Liu, X. (2021). A novel antibacterial peptide recognition algorithm based on BERT. *Briefings in Bioinformatics*, *22*(6), bbab200. https://doi.org/10.1093/bib/bbab200

Zielezinski, A., Vinga, S., Almeida, J., & Karlowski, W. M. (2017). Alignment-free sequence comparison: benefits, applications, and tools. *Genome Biology*, *18*(1), 186. https://doi.org/10.1186/S13059-017-1319-7

Anexos

Anexo A. Lista de parámetros del *framework* esm-AxP-GDL

Tabla 9. Lista de parámetros necesarios para ejecutar los pasos de entrenamiento (E), prueba (P) e inferencia (I) con el *framework* esm-AxP-GDL.

Parámetro (tipo de dato): Valor por defecto	Descripción	Restricción de valores	Modo	Requerido
--dataset (FilePath)	Conjunto de datos de entrada en formato CSV (*e.g.*, Ruta/al/archivo/DatasetExample.csv.		(E) (P) (I)	✓
--gdl_Modelo_path (Path)	Ruta donde se guardan/cargan los modelos.		(E) (P) (I)	✓
--output_path (Path): None	La ruta para cargar los modelos entrenados y guardar los archivos de salida.		(P) (I)	✓
--esm2_representation (str): esm2_t33	Modelo (incrustración) ESM-2 a usar.	esm2_t6, esm2_t12, esm2_t30, esm2_t33, esm2_t36, esm2_t48	(E)	
--tertiary_structure_method (str): None	Método de predicción de estructura 3D. "None" indica cargar estructuras terciarias existentes de archivos PDB, de lo contrario, se predicen las secuencias del archivo CSV.	esmfold	(E)	
--pdb_path (DirectoyPath): None	Ruta para salvar/cargar las estructuras 3D desde archivos PDBs.		(E)	
--edge_construction_functions (set): None	Funciones para construir aristas.	distance_based_threshol, sequence_based, esm2_contact_map_50, esm2_contact_map_60, esm2_contact_map_70, esm2_contact_map_80. esm2_contact_map_90	(E)	

--distance_function (str): None	Función de distancia para calcular las distancias inter-aminoácidos.	euclidean, canberra, lance_williams, clark, soergel, bhattacharyya, angular_separation	(E)	
--threshold (float): None	Umbral de distancia para establecer las aristas de los grafos.		(E)	
--amino_acid_representation (str): CA	Representación de aminoácido a partir del cual se calcularán las distancias inter-aminoácidos.		(E)	
--number_of_heads (int): 8	Número de cabezas.		(E)	
--hidden_layer_dimension (int): 128	Dimensión de la capa oculta.		(E)	
--add_self_loops: True	"True" si se especifica, de lo contrario, "False". "True" indica usar *auto loops* en la capa de atención.		(E)	
--use_edge_attr: True	"True" si se especifica, de lo contrario, "False". "True" indica usar aristas ponderadas en el aprendizaje de grafos.		(E)	
--learning_rate (float): 1e-4	Tasa de aprendizaje.		(E)	
--dropout_rate (float): 0.25 (E) / 0.5 (V) / 0.5 (P)	Tasa de *dropout*.		(E) (P) (I)	
--batch_size (int): 512	Tamaño del lote.		(E) (P) (I)	
--number_of_epoch (int): 200	Número máximo de épocas.		(E)	
--save_ckpt_per_epoch: True	"True" si se especifica, de lo contrario, "False". "True" indica que se guardará el modelo de cada época de entrenamiento. De lo contrario, solo se guardarán el modelo de la última época y el modelo con el valor de MCC más alto.		(E)	
--validation_mode (str): None	Método de construcción de grafos para la validación del enfoque.	random_coordinates, random_embeddings	(E)	
--randomness_percentage (float): None	Porcentaje de filas que se generarán aleatoriamente.		(E)	
--seed (int): None	Semilla para ejecutar el modo de prueba / inferencia.		(P) (I)	

Anexo B. Métricas para evaluar el rendimiento de modelos

Tabla 10. Definiciones matemáticas para el cálculo de las métricas precisión, sensibilidad, especificidad, coeficiente de correlación de Matthews, y área bajo la curva.

Métrica	Fórmula	Descripción
Precisión (ACC, acrónimo accuracy)	$ACC = \frac{TP + TN}{TP + TN + FP + FN}$	Proporción de predicciones correctas respecto al total de predicciones realizadas.
Sensibilidad (SE o SN, acrónimos de sensitivity)	$SE = \frac{TP}{TP + FN}$	Razón de recuperación de verdaderos positivos, *i.e.*, la proporción de predicciones positivas correctas respecto al total de verdaderos positivos.
Especificidad (SP, acrónimo de specificity)	$SP = \frac{TN}{TN + FP}$	Razon de recuperación de verdaderos negativos, *i.e.*, la proporción de predicciones negativas correctas respecto al total de verdaderos negativos.
Coeficiente de correlación de Matthews (MCC, por sus siglas en inglés)	$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FN) \times (TN + FP)}}$	Mide la capacidad de generalización de un modelo, con un rango que va de -1 a 1. Un valor de 1 indica una correlación o capacidad de generalización perfecta, 0 indica una correlación aleatoria, y -1 indica una correlación inversa. Cuanto más cercano a 1 sea el valor, mayor es la capacidad predictiva del modelo. Es una métrica que tiene en cuenta el desequilibrio entre las clases.
Área bajo la curva ROC (AUC, acrónimo de area under the curve)	$AUC = \frac{\sum_i^{n_{pos}} \left(rank_i - \frac{n_{pos}(n_{pos}+1)}{2} \right)}{n_{pos} \times n_{neg}}$	Mide la capacidad de un modelo para distinguir entre clases positivas y negativas. La curva ROC (del inglés - Receiver Operating Characteristic) es un gráfico que muestra la relación entre la tasa de verdaderos positivos y la tasa de falsos positivos a diferentes umbrales de decisión. El AUC varía entre 0 y 1, donde 1 indica un modelo perfecto que separa completamente las clases, mientras que 0.5 indica predicciones aleatorias. Cuanto más cercano a uno sea el valor, mejor es el modelo para distinguir entre las clases.

TP: Verdaderos Positivos (TP, por sus siglas en inglés)
TN: Verdaderos Negativos (TN, por sus siglas en inglés)
FP: Falsos Positivos (FP, por sus siglas en inglés)
FN: Falsos Negativos (FN, por sus siglas en inglés)
n_{pos}: Número total de ejemplos positivos.
n_{neg}: Número total de ejemplos negativos.

Anexo C. Estructura del directorio de archivos del *framework* esm_AxP_GDL

Figura 23. Estructura del directorio de archivos del *framework* esm_AxP_GDL

Anexo D. Ficheros FASTA correspondientes a los conjuntos de entrenamiento, validación y prueba utilizados

https://drive.google.com/drive/folders/1jtXcxtZRDzjPiZDKJP0TvP8I255GBWj-?usp=sharing

Anexo E. Métricas de rendimiento obtenidas en el conjunto de validación y en el conjunto de prueba por los 45 modelos GAT construidos

Tabla 11. Métricas de rendimiento obtenidas en el conjunto de validación y en el conjunto de prueba por los 45 modelos GAT construidos.

Parámetros			Conjunto de Validación	Conjunto de Prueba				
HDD	**Umbral**	**ESM-2**	**MCC**	**MCC**	**ACC**	**AUC**	**SP**	**SN**
64	10	esm2_t6	0.9736	0.8534	0.9299	0.9893	0.9037	0.9872
64	10	esm2_t12	0.9820	0.9031	0.9551	0.9923	0.9375	0.9937
64	10	esm2_t30	0.9832	0.9259	0.9666	0.9927	0.9559	0.9900
64	10	esm2_t33	0.9891	0.9367	0.9725	0.9908	0.9733	0.9707
64	10	esm2_t36	0.9688	0.8874	0.9483	0.9903	0.9332	0.9813
64	15	esm2_t6	0.9844	0.9366	0.9718	0.9925	0.9647	0.9874
64	15	esm2_t12	0.9747	0.8968	0.9519	0.9925	0.9328	0.9939
64	15	esm2_t30	0.9926	0.9329	0.9698	0.9932	0.9598	0.9919
64	15	esm2_t33	0.9840	0.9364	0.9718	0.9918	0.9659	0.9847
64	15	esm2_t36	0.9673	0.9116	0.9596	0.9920	0.9458	0.9900
64	20	esm2_t6	0.9751	0.8805	0.9436	0.9912	0.9210	0.9931
64	20	esm2_t12	0.9677	0.8398	0.9215	0.9916	0.8885	0.9937
64	20	esm2_t30	0.9883	0.9097	0.9586	0.9928	0.9439	0.9908
64	20	esm2_t33	0.9918	0.9096	0.9588	0.9904	0.9452	0.9884
64	20	esm2_t36	0.9906	0.9170	0.9627	0.9915	0.9532	0.9835
128	10	esm2_t6	0.9922	0.9456	0.9758	0.9941	0.9679	0.9931
128	10	esm2_t12	0.9922	0.8744	0.9401	0.9932	0.9150	0.9953
128	10	esm2_t30	0.9922	0.9343	0.9703	0.9937	0.9588	0.9955
128	10	esm2_t33	0.9953	0.9389	0.9726	0.9930	0.9629	0.9939
128	10	esm2_t36	0.9918	0.9505	0.9782	0.9930	0.9735	0.9884
128	15	esm2_t6	0.9938	0.9223	0.9646	0.9931	0.9513	0.9939
128	15	esm2_t12	0.9957	0.9425	0.9743	0.9941	0.9657	0.9931
128	15	esm2_t30	0.9957	0.9379	0.9722	0.9936	0.9632	0.9919
128	15	esm2_t33	0.9934	0.9354	0.9710	0.9941	0.9610	0.9929
128	15	esm2_t36	0.9961	0.9499	0.9778	0.9937	0.9713	0.9921
128	20	esm2_t6	0.9848	0.9368	0.9714	0.9935	0.9594	0.9976
128	20	esm2_t12	0.9926	0.9399	0.9730	0.9940	0.9629	0.9951
128	20	esm2_t30	0.9934	0.9319	0.9693	0.9933	0.9583	0.9933
128	20	esm2_t33	0.9899	0.9347	0.9707	0.9927	0.9617	0.9906
128	20	esm2_t36	0.9802	0.9385	0.9723	0.9938	0.9612	0.9965
256	10	esm2_t6	0.9809	0.8536	0.9287	0.9911	0.8975	0.9969
256	10	esm2_t12	0.9642	0.8022	0.8986	0.9921	0.8528	0.9992
256	10	esm2_t30	0.9973	0.9420	0.9741	0.9933	0.9659	0.9921
256	10	esm2_t33	0.9906	0.9418	0.9739	0.9933	0.9641	0.9953

256	10	esm2_t36	0.9868	0.9521	0.9786	0.9942	0.9707	0.9961
256	15	esm2_t6	0.9837	0.8759	0.9406	0.9923	0.9143	0.9982
256	15	esm2_t12	0.9840	0.9003	0.9535	0.9926	0.9342	0.9959
256	15	esm2_t30	0.9965	0.9405	0.9732	0.9933	0.9629	0.9959
256	15	esm2_t33	0.9813	0.8921	0.9490	0.9921	0.9261	0.9992
256	15	esm2_t36	0.9902	0.9255	0.9659	0.9923	0.9513	0.9980
256	20	esm2_t6	0.9825	0.8769	0.9412	0.9921	0.9153	0.9978
256	20	esm2_t12	0.9868	0.8877	0.9468	0.9927	0.9230	0.9988
256	20	esm2_t30	0.9938	0.9117	0.9591	0.9924	0.9413	0.9980
256	20	esm2_t33	0.9872	0.9391	0.9725	0.9939	0.9608	0.9980
256	20	esm2_t36	0.9953	0.9380	0.9719	0.9937	0.9603	0.9976

Anexo F. Métricas de rendimiento obtenidas en el conjunto de pruebas reducido compuesto por secuencias de hasta 100 aminoácidos por los 45 modelos GAT construidos

Tabla 12. Métricas de rendimiento obtenidas en el conjunto de pruebas reducido compuesto por secuencias de hasta 100 aminoácidos por los 45 modelos GAT construidos.

Parámetros			Conjunto de prueba reducido (hasta 100 aminoácidos)				
HDD	**Umbral**	**ESM-2**	**MCC**	**ACC**	**AUC**	**SP**	**SN**
64	10	esm2_t6	0.8190	0.9253	0.9926	0.9082	0.9846
64	10	esm2_t12	0.8732	0.9505	0.9938	0.9395	0.9885
64	10	esm2_t30	0.9088	0.9659	0.9946	0.9604	0.9852
64	10	esm2_t33	0.9491	0.9819	0.9923	0.9827	0.9791
64	10	esm2_t36	0.8645	0.9471	0.9916	0.9368	0.9827
64	15	esm2_t6	0.9308	0.9747	0.9939	0.9715	0.9859
64	15	esm2_t12	0.8568	0.9430	0.9948	0.9301	0.9878
64	15	esm2_t30	0.9221	0.9712	0.9948	0.9664	0.9878
64	15	esm2_t33	0.9258	0.9729	0.9933	0.9704	0.9817
64	15	esm2_t36	0.8900	0.9577	0.9949	0.9486	0.9894
64	20	esm2_t6	0.8561	0.9425	0.9944	0.9286	0.9904
64	20	esm2_t12	0.8019	0.9158	0.9941	0.8945	0.9894
64	20	esm2_t30	0.8994	0.9618	0.9949	0.9539	0.9891
64	20	esm2_t33	0.8935	0.9596	0.9943	0.9524	0.9846
64	20	esm2_t36	0.8967	0.9610	0.9933	0.9543	0.9840
128	10	esm2_t6	0.9350	0.9762	0.9958	0.9727	0.9885
128	10	esm2_t12	0.8318	0.9309	0.9942	0.9139	0.9897
128	10	esm2_t30	0.9187	0.9698	0.9954	0.9642	0.9891
128	10	esm2_t33	0.9228	0.9714	0.9951	0.9664	0.9888
128	10	esm2_t36	0.9511	0.9824	0.9946	0.9808	0.9881
128	15	esm2_t6	0.9019	0.9628	0.9950	0.9550	0.9897
128	15	esm2_t12	0.9356	0.9765	0.9944	0.9734	0.9872
128	15	esm2_t30	0.9337	0.9757	0.9961	0.9715	0.9901
128	15	esm2_t33	0.9330	0.9754	0.9956	0.9716	0.9888
128	15	esm2_t36	0.9377	0.9774	0.9943	0.9747	0.9865
128	20	esm2_t6	0.9159	0.9686	0.9948	0.9628	0.9888
128	20	esm2_t12	0.9279	0.9735	0.9945	0.9694	0.9878
128	20	esm2_t30	0.9223	0.9711	0.9956	0.9655	0.9907
128	20	esm2_t33	0.9243	0.9720	0.9941	0.9670	0.9891
128	20	esm2_t36	0.9133	0.9675	0.9967	0.9608	0.9904
256	10	esm2_t6	0.8024	0.9155	0.9953	0.8930	0.9933
256	10	esm2_t12	0.7403	0.8798	0.9957	0.8462	0.9962

256	10	esm2_t30	0.9400	0.9782	0.9954	0.9750	0.9891
256	10	esm2_t33	0.9266	0.9729	0.9963	0.9683	0.9888
256	10	esm2_t36	0.9381	0.9773	0.9975	0.9732	0.9917
256	15	esm2_t6	0.8231	0.9258	0.9968	0.9058	0.9952
256	15	esm2_t12	0.8639	0.9457	0.9964	0.9319	0.9936
256	15	esm2_t30	0.9329	0.9754	0.9959	0.9711	0.9901
256	15	esm2_t33	0.8353	0.9322	0.9950	0.9148	0.9926
256	15	esm2_t36	0.8915	0.9579	0.9958	0.9474	0.9945
256	20	esm2_t6	0.8412	0.9350	0.9958	0.9179	0.9939
256	20	esm2_t12	0.8402	0.9345	0.9960	0.9175	0.9936
256	20	esm2_t30	0.8752	0.9510	0.9961	0.9389	0.9926
256	20	esm2_t33	0.9086	0.9655	0.9956	0.9582	0.9907
256	20	esm2_t36	0.9126	0.9672	0.9966	0.9604	0.9907

Anexo G. Métricas de rendimiento obtenidas en el conjunto de prueba reducido compuesto por secuencias de hasta 30 aminoácidos por los 45 modelos GAT construidos

Tabla 13. Métricas de rendimiento obtenidas en el conjunto de prueba reducido compuesto por secuencias de hasta 30 aminoácidos por los 45 modelos GAT construidos.

Parámetros			Conjunto de prueba reducido (hasta 30 aminoácidos)				
HDD	**Umbral**	**ESM-2**	**MCC**	**ACC**	**AUC**	**SP**	**SN**
64	10	esm2_t6	0.7419	0.8716	0.9851	0.8294	0.9836
64	10	esm2_t12	0.8207	0.9171	0.9897	0.8906	0.9873
64	10	esm2_t30	0.8358	0.9260	0.9881	0.9053	0.9812
64	10	esm2_t33	0.7481	0.8763	0.9838	0.8373	0.9798
64	10	esm2_t36	0.7650	0.8871	0.9845	0.8530	0.9775
64	15	esm2_t6	0.8595	0.9386	0.9869	0.9241	0.9770
64	15	esm2_t12	0.8296	0.9223	0.9886	0.8991	0.9841
64	15	esm2_t30	0.8231	0.9191	0.9867	0.8954	0.9822
64	15	esm2_t33	0.8484	0.9327	0.9876	0.9150	0.9798
64	15	esm2_t36	0.8515	0.9348	0.9882	0.9194	0.9756
64	20	esm2_t6	0.8153	0.9146	0.9882	0.8885	0.9841
64	20	esm2_t12	0.7475	0.8741	0.9881	0.8315	0.9873
64	20	esm2_t30	0.8203	0.9180	0.9871	0.8947	0.9798
64	20	esm2_t33	0.7682	0.8883	0.9832	0.8537	0.9803
64	20	esm2_t36	0.7259	0.8609	0.9843	0.8142	0.9850
128	10	esm2_t6	0.8567	0.9364	0.9906	0.9185	0.9841
128	10	esm2_t12	0.7602	0.8817	0.9915	0.8414	0.9887
128	10	esm2_t30	0.8336	0.9239	0.9909	0.8996	0.9883
128	10	esm2_t33	0.8080	0.9099	0.9916	0.8804	0.9883
128	10	esm2_t36	0.8139	0.9131	0.9885	0.8846	0.9887
128	15	esm2_t6	0.8157	0.9145	0.9894	0.8876	0.9859
128	15	esm2_t12	0.7965	0.9028	0.9920	0.8696	0.9911
128	15	esm2_t30	0.8328	0.9245	0.9895	0.9033	0.9808
128	15	esm2_t33	0.8180	0.9163	0.9897	0.8913	0.9827
128	15	esm2_t36	0.7610	0.8821	0.9902	0.8417	0.9892
128	20	esm2_t6	0.8601	0.9383	0.9910	0.9218	0.9822
128	20	esm2_t12	0.8317	0.9233	0.9894	0.9001	0.9850
128	20	esm2_t30	0.8256	0.9200	0.9909	0.8954	0.9855
128	20	esm2_t33	0.7799	0.8939	0.9874	0.8587	0.9873
128	20	esm2_t36	0.8566	0.9364	0.9896	0.9187	0.9836
256	10	esm2_t6	0.7247	0.8584	0.9896	0.8082	0.9916
256	10	esm2_t12	0.6916	0.8353	0.9887	0.7758	0.9934

256	10	esm2_t30	0.7264	0.8594	0.9893	0.8095	0.9920
256	10	esm2_t33	0.7898	0.8989	0.9923	0.8640	0.9916
256	10	esm2_t36	0.8346	0.9235	0.9927	0.8966	0.9948
256	15	esm2_t6	0.7178	0.8532	0.9924	0.8005	0.9934
256	15	esm2_t12	0.7421	0.8699	0.9888	0.8245	0.9906
256	15	esm2_t30	0.6906	0.8344	0.9902	0.7743	0.9939
256	15	esm2_t33	0.7452	0.8710	0.9910	0.8248	0.9939
256	15	esm2_t36	0.7530	0.8764	0.9888	0.8329	0.9920
256	20	esm2_t6	0.7243	0.8573	0.9911	0.8058	0.9944
256	20	esm2_t12	0.7342	0.8636	0.9906	0.8140	0.9953
256	20	esm2_t30	0.7580	0.8794	0.9916	0.8366	0.9930
256	20	esm2_t33	0.8149	0.9133	0.9914	0.8843	0.9906
256	20	esm2_t36	0.7266	0.8596	0.9871	0.8100	0.9916

Anexo H. Medida de desacuerdo entre modelos

Sean D_i y D_k dos clasificadores, para calcular la medida de desacuerdo entre ellos, primero se crea una matriz de confusión para el par de clasificadores (ver Tabla 14). Posteriormente, la medida de desacuerdo se determina utilizando la siguiente ecuación (ver ecuación (25)):

$$Dis_{i,\,k} = \frac{N^{10} + N^{01}}{N^{11} + N^{10} + N^{01} + N^{00}} \tag{25}$$

Tabla 14. Matriz 2 × 2 de la relación entre el par de clasificadores $\boldsymbol{D_i}$ y $\boldsymbol{D_k}$.

	$\boldsymbol{D_k}$ **correcto (1)**	$\boldsymbol{D_k}$ **incorrecto (0)**
$\boldsymbol{D_i}$ **correcto (1)**	N^{11}	N^{10}
$\boldsymbol{D_i}$ **incorrecto (0)**	N^{01}	N^{00}

N^{11}: número de instancias donde ambos clasificadores predicen correctamente.
N^{10}: número de instancias donde el clasificador D_i es correcto y el clasificador D_k es incorrecto.
N^{01}: número de instancias donde el clasificador D_i es incorrecto y el clasificador D_k es correcto.
N^{00}: número de instancias donde ambos clasificadores predicen incorrectamente.
$N = N^{11} + N^{10} + N^{01} + N^{00}$ es el número total de instancias.

No se ha establecido en la literatura un valor óptimo de desacuerdo; sin embargo, se considera que un valor superior al 10% indica desacuerdo.

Anexo I. Predicciones de los tres mejores modelos individuales y de los modelos de consenso

https://docs.google.com/spreadsheets/d/1lOJo41uDug-7pQePcvSHgK77GeaQ49PH/edit?usp=sharingouid=114384123402188922133yrtpof=trueysd=true

Anexo J. Ficheros FASTA correspondientes a conjuntos negativos construidos aplicando diferentes umbrales de similitud

https://drive.google.com/drive/folders/1lRK7P-3yyIVyrEbLpUCmDtqYjKTvz8nS?usp=sharing

Anexo K. Valores de especificidad obtenidos por los tres mejores modelos individuales y el modelo de consenso de los mismos sobre los conjuntos de datos construidos aplicando varios umbrales de similitud

Tabla 15. Valores de especificidad obtenidos por los tres mejores modelos individuales y el modelo de consenso de los mismos sobre los conjuntos de datos construidos aplicando varios umbrales de similitud.

Similitud (%)	Numero de secuencias	Modelo 2	Modelo 3	Modelo 5	Modelo de consenso
0.1	25	0.8	0.8	0.28	0.96
0.2	417	0.8801	0.9544	0.8561	0.9976
0.3	3304	0.8018	0.8148	0.8417	0.9431
0.4	6576	0.6542	0.6696	0.7587	0.879
0.5	8272	0.6118	0.631	0.7071	0.8399
0.6	9543	0.6121	0.6417	0.7001	0.8394
0.7	10958	0.6156	0.6391	0.701	0.8494
0.8	12741	0.6214	0.6576	0.6922	0.8484
0.9	15379	0.6349	0.6674	0.6996	0.8601

Anexo L. Estadísticas de la prueba de Friedman

1 Average rankings of Friedman test

Average ranks obtained by each method in the Friedman test.

Algorithm	Ranking
0	1.1333
70	2.7
80	3
90	3.1667

Table 1: Average Rankings of the algorithms (Friedman)

Friedman statistic (distributed according to chi-square with 3 degrees of freedom): 46.84.
P-value computed by Friedman Test: 0.

Iman and Davenport statistic (distributed according to F-distribution with 3 and 87 degrees of freedom): 31.47266.
P-value computed by Iman and Daveport Test: 0.

Figura 24. Ranking promedio obtenido por cada modelo en la prueba de Friedman. El algoritmo 0 representa los modelos construidos a partir de grafos basados en distancia (valores de referencia). Los algoritmos 70, 80 y 90 representan los modelos construidos a partir de mapas de contacto ESM-2, aplicando umbrales de 70, 80 y 90, respectivamente.

Anexo M. Estadísticas de la prueba post hoc de Holm

2 Post hoc comparison (Friedman)

P-values obtained in by applying post hoc methods over the results of Friedman procedure.

i	algorithm	$z = (R_0 - R_i)/SE$	p	Holm
3	90	6.1	0	0.016667
2	80	5.6	0	0.025
1	70	4.7	0.000003	0.05

Table 2: Post Hoc comparison Table for $\alpha = 0.05$ (FRIEDMAN)

Figura 25. Valores p obtenidos al aplicar el método post hoc de Holm sobre los resultados del procedimiento de Friedman. Los algoritmos 70, 80 y 90 representan los modelos construidos a partir de mapas de contacto ESM-2, con umbrales de 70, 80 y 90, respectivamente.

Anexo N. Producción científica del autor

Publicaciones

1. **Cordoves-Delgado, G.**, y García-Jacas, C. R. (2024). Predicting Antimicrobial Peptides Using ESMFold-Predicted Structures and ESM-2-Based Amino Acid Features with Graph Deep Learning. *Journal of Chemical Information and Modeling*, *64*(10). https://doi.org/10.1021/acs.jcim.3c02061

2. Martínez-Mauricio K, L.; García-Jacas, C. R.; y **Cordoves-Delgado, G**. (2024). Examining Evolutionary Scale Modeling-derived Different-dimensional Embeddings in the Antimicrobial Peptide Classification through a KNIME Workflow. *Protein Science*, *33*(4). https://doi.org/10.1002/pro.4928

Presentación de conferencias

1. **Cordoves-Delgado, G.**; y García-Jacas, C. R. Graph deep learning-driven framework for classification of antimicrobial peptides from ESMFold-predicted structures and ESM-2 embeddings. *ACS Fall 2024*.

Producción de software

1. **Cordoves-Delgado, G.**, y García-Jacas, C. R. Framework esm-AxP-GDL. Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California, México. 2024. https://github.com/cicese-biocom/esm-AxP-GDL.git