UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA CIVIL EN BIOINFORMÁTICA

Aplicaciones de estructuras de grafos y aprendizaje profundo a sistemas de clasificación de interacción antígeno-anticuerpo

Por: Claudio Guevara Vásquez

Julio 2022

Talca, Chile

Profesor Guía: Álvaro Olivera Nappa
Profesor Co-Guía: David Medina Ortiz
Profesor Informante: Fabio Duran Verdugo

CONSTANCIA

La Dirección del Sistema de Bibliotecas a través de su unidad de procesos técnicos certifica que el autor del siguiente trabajo de titulación ha firmado su autorización para la reproducción en forma total o parcial e ilimitada del mismo.

Talca, 2022

Agradecimientos

Primero agradecer a mis tutores Álvaro Olivera y David Medina quienes me dieron la posibilidad de trabajar en esta investigación y me apoyaron a lo largo de todo de este camino. Todo lo aprendido en este tiempo me ayudo a perfeccionarme en mi futuro laboral.

Agradecer a la Universidad de Chile, la Universidad de Magallanes, la Universidad de Talca, el Centro de Biotecnología y Bioingenería, la Universidad de Antofagasta y la Universidad de los Lagos. Todas estas entidades hacieron posible la realización de este proyecto.

Agradecer al profesor Fabio Durán quien fue mi informante en mi proyecto de memoria y que me apoyo siempre que lo necesite, no solo para mi tesis, sino que también a lo largo de toda mi instancia en la Universidad.

Agradecer a todas las personas que colaboraron de una u otra forma en el proyecto, pero un especial agradecimiento a Yasna Barrera y Francisca Rodríguez, quienes me apoyaron en la realización y comprensión de tareas relacionadas a la bioinfórmatica estructural.

Por último, agradecer a mi familia que me ha apoyado a lo largo de toda mi vida. Son ellos los que me motivan a esforzarme cada día y, este proyecto, la Universidad y lo que me deparé en el futuro, es una pequeña forma de devolver todo lo que me han dado hasta el momento.

A todos los que participaron, ya sea de forma directa o indirecta en el proyecto, mis amigos y mi familia, los quiero y gracias.

Powered@NLHPC: This research was partially supported by the supercomputing infrastructure of the National Laboratory for High-Performance Computing, NLHPC (ECM-02), Chile.

Resumen

Las interacciones proteína-proteína son de real importancia para la ingeniería de proteínas debido a que forman parte esencial en la mayoría de los procesos moleculares. Un caso particular es la interacción antígeno-anticuerpo, la cual cumple con el rol de inhibir o neutralizar agentes patógenos que afectan negativamente la homeostasis normal del cuerpo. Conocer el funcionamiento específico de un anticuerpo es de gran interés en áreas como la medicina y la farmacología, ya que facilita el diseño de vacunas y medicamentos.

Variados métodos experimentales se han desarrollado con el fin de estudiar las interacciones proteína-proteína. Algunos de los ejemplos clásicos corresponden a los microarrays de ADN y proteína, la espectroscopia de masas (MS) y la letalidad sintética. Sin embargo, estos métodos se caracterizan por tener un alto costo de producción y tiempo de desarrollo, ser susceptibles al error humano, y muchas veces, requieren de un elevado conocimiento. Para solventar este problema, cada vez se aplican más técnicas basadas en Machine Learning y Deep Learning, como es el caso de AlphaFold y su capacidad de predecir la estructura secundaria. Sin embargo, la rama de la inteligencia artificial aún debe ser más estudiada y aplicada en la investigación científica.

Con base en esto, se realizó una investigación con el fin de predecir la interacción antígeno-anticuerpo por medio de Graph Neural Network. Para lograr esto, las proteínas se representaron por medios de estructuras de grafos, en donde los nodos correspondían a los residuos de las proteínas, mientras que las aristas a las distancias euclidianas entre aminoácidos. Además, se contó con un clasificador de interacción para cada complejo. En general, se obtuvo un rendimiento alrededor del 0,51 y se planteó una serie de puntos a tratar en futuros trabajos para el perfeccionamiento de los modelos, los cuales tienen que ver con arquitecturas de redes neuronales, representación de grafos, métodos de codificación y predicción de complejos proteicos. Demostrando que, a pesar de que los resultados no fueron los esperados, el camino por delante es extenso y queda un largo desarrollo por realizar, con el fin de llegar a elaborar sistemas predictivos en base a esta arquitectura de deep learning.

Abstract

Protein-protein interactions are significant for protein engineering because they are essential to most molecular processes. A particular case is an antigen-antibody interaction, which fulfills the role of inhibiting or neutralizing pathogenic agents that negatively affect the normal homeostasis of the body. Knowing the specific functioning of an antibody is of great interest in areas such as medicine and pharmacology since it facilitates the design of vaccines and drugs.

Various experimental methods have been developed to study protein-protein interactions. Some classic examples are DNA and protein microarrays, mass spectroscopy (MS), and synthetic lethality. However, these methods are characterized by high production cost and development time, being susceptible to human error, and often requiring a high level of knowledge. To solve this problem, more and more techniques based on Machine Learning and Deep Learning are being applied, such as AlphaFold and its ability to predict secondary structure. However, the branch of artificial intelligence still needs to be more studied and used in scientific research.

Based on this, an investigation was carried out to predict the antigen-antibody interaction using the Graph Neural Network. Proteins were represented by employing graph structures, where nodes corresponded to protein residues while edges to Euclidean distances between amino acids. In addition, there was an interaction classifier for each complex. In general, a performance of around 0.51 was obtained, and a series of points were raised to be dealt with in future works to improve the models, which have to do with neural network architectures, graph representation, coding methods, and prediction of protein complexes. Even though the results were not as expected, the road ahead is long, and there is a long development to be done to develop predictive systems based on this deep learning architecture.

Índice general

Índice de cuadros

Índice de figuras

Capítulo 1

Introducción

1.1. Marco Teórico

El sistema inmune es un conjunto de células, órganos y tejidos que tiene como función proteger al huésped de agentes extraños (antígenos), como virus, bacterias o sustancias tóxicas. El sistema inmune se divide en dos grandes sistemas: i) el sistema inmunológico innato, el cual está conformado por la protección proveniente de la piel, mucosas, enzimas de las lágrimas, entre otras, y ii) el sistema inmunológico adaptativo, el cual está compuesto por células y procesos sistemáticos para inhibir o eliminar un antígeno por medio de su reconocimiento. El principal componente de este sistema son los linfocitos, los cuales son un tipo de glóbulo de blanco. Existen dos tipos principales de linfocitos: i) los linfocitos B, que generan anticuerpos para interactuar y luchar con antígeno de manera específica, y ii) los linfocitos T, que se encargan de realizar actividades fagocitarías hacia la sustancia extraña. A continuación, se describirán más en detalle cada uno de los componentes principales, así como sus características particulares.

1.1.1. Sistema Inmune

Los seres humanos viven en un mundo que está poblado tanto de microbios patógenos como no patógenos, así como también de sustancias tóxicas o alérgicas que amenazan la homeostasis normal del cuerpo humano (DD, 2010). En este contexto, el sistema inmune juega un rol fundamental como agente defensivo y de aprendizaje para subsistir ante estas amenazas.

Por definición, el sistema inmune corresponde a una colección de células, productos químicos y procesos de regulación que funcionan para proteger la piel, las vías respiratorias, el tracto intestinal y diferentes áreas, de patógenos externos como los microbios (bacterias, hongos y parásitos), virus, cáncer, células y toxinas, así como también reacciones ante componentes tóxicos internos Marshall JS (2018). Este sistema de protección está conformado por barreras estructurales y químicas. Además, normalmente se aprecia la existencia de dos “líneas de defensa", las cuales corresponden a la inmunidad innata y la inmunidad adaptativa Marshall JS (2018).

1.1.1.1. Sistema inmunológico innato

Este tipo de mecanismo, corresponde a la primera línea de defensa contra los agentes patógenos que ingresan al cuerpo. Responde de la misma forma para todos los elementos reconocidos como extraños, por lo que también se le denomina sistema inmunológico “inespecífico". Se caracteriza por actuar muy rápido. Sin embargo, tiene un poder limitado para detener la propagación de los agentes externos. Este sistema consta de la protección ofrecida por la piel, las mucosas, las células del sistema inmunológico (células de defensa) y las proteínas (Colonia, 2020), como pueden ser las enzimas líticas que se encuentran en las lágrimas, las mucosidades o la saliva (Luna, 2010).

1.1.1.2. Sistema inmunológico adaptativo

El sistema inmunológico adaptativo proporciona la capacidad de reconocer y responder a una variedad de patógenos. Tiene como componente principal los linfocitos, los cuales son un tipo de célula inmunitaria que se produce en la médula ósea, y que se encuentra en la sangre y en el tejido linfático INSTITUYE (2021). Existen dos tipos principales de linfocitos, las células B y las células T. Las primeras maduran en la médula ósea y participan principalmente en la creación de los anticuerpos, mientras que los linfocitos T maduran en el timo (órgano linfoide) y participan principalmente en respuestas inmunitarias mediadas por células. Más en detalle, las células B expresan el receptor de las células B (BCR), el cual es una molécula de anticuerpo unida a la membrana, y que tiene la función de reconocer y unirse a las moléculas extrañas (antígenos). Los anticuerpos unidos a moléculas extrañas los convierten en blancos fáciles para los linfocitos T, los cuales realizan

funciones fagocitarias y otros mecanismos de eliminación. La ventaja de este tipo de sistema es que las células B activadas se diferencian en células de memoria específicas con el antígeno que interactuaron, es decir, los anticuerpos recuerdan el antígeno eliminado para futuras infecciones, permitiendo la memoria inmunológica. Las células B permanecen en circulación por un largo período y respondiendo rápidamente si el mismo patógeno vuelve a infectar al huésped Whitacre et al. (2012). La comparación de este sistema, con respecto al sistema innato se puede apreciar en el figura 1.1.1.

Un ejemplo que aplica este tipo de mecanismo son las vacunas, las cuales tienen la función de generar un efecto inmune en el cuerpo de manera controlada Pollard (2021). Para lograr esto, las vacunas incorporan un adyuvante (estimula la inmunidad innata) y un antígeno (estimula la inmunidad adaptativa). En este sentido, las vacunas más eficaces son aquellas que imitan o mejoran la inmunidad natural estimulando los anticuerpos y los linfocitos T, que posteriormente eliminarán o bloquearán el contagio o enfermedad Dobaño (2021).

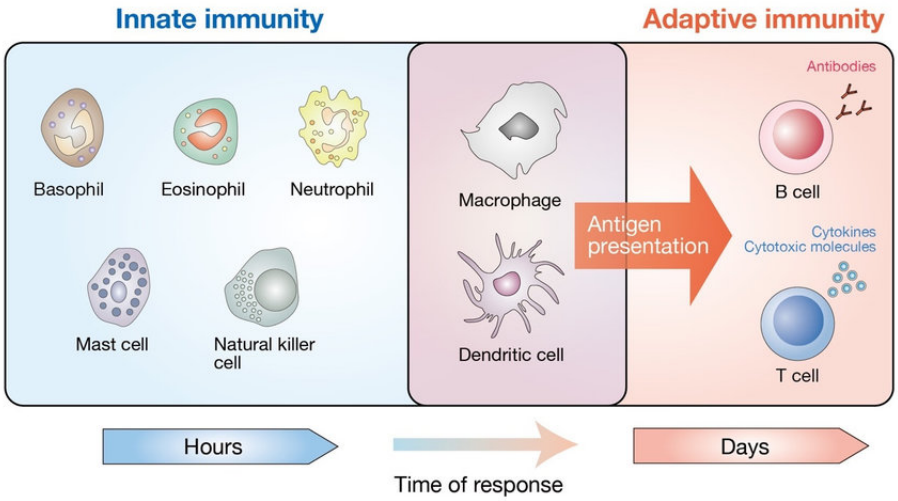

Figura 1.1.1: Comparación entre sistemas inmunes. El sistema inmune innato esta compuesto por células del cuerpo que actúan como primera defensa ante una enfermedad, mientras que el sistema inmune adaptativo se encuentra conformado principalmente por los anticuerpos, donde su interacción con los antígenos permite inhibir el patógeno y generar memoria inmunológica..

1.1.2. Interacción antígeno-anticuerpo

En la sección anterior se dieron a conocer dos conceptos que son de vital importancia para el presente estudio: los anticuerpos y los antígenos. Dicha

interacción se conoce como interacción proteína-proteína (PPI por sus siglas en inglés), las cuales son importantes en muchos procesos biológicos como en tareas de señalización, traducción de señales, transporte, entre otras. La presente sección tiene como objetivo profundizar en ambos componentes y entender cómo interactúan (asociación de proteínas) de una forma específica para poder controlar o eliminar una enfermedad VS. et al. (2014).

1.1.2.1. Anticuerpo

Los anticuerpos, también denominados inmunoglobulinas, son un tipo de proteína protectora del sistema inmunológico en respuesta a la presencia de una sustancia extraña (antígenos). Los anticuerpos reconocen y se adhieren a los antígenos, para que posteriormente sean eliminados o neutralizados Britannica (2021a).

Estructuralmente, los anticuerpos son proteínas con forma de *"Y"* compuestas por dos cadenas ligeras idénticas (L) con un peso molecular de entre 23 a 26 kilo dalton (kDa), y dos cadenas pesadas idénticas (H) con un peso de entre 55 y 77 kDa. En los sistemas naturales, el emparejamiento de una cadena L con una cadena H decanta en un hetero dímero idéntico para formar la inmunoglobulina intacta mediante interacciones electroestáticas débiles, así como enlaces disulfuro Chiu M (2019). Cada una de las cadenas se compone de dos regiones, una región constante (C) y una región hipervariable (V). Funcionalmente, la región constante confiere propiedades efectoras tales como unión al complemento, duración de la vida media, interacción con receptores celulares y la clase o isotipo de la inmunoglobulina (Ig). En cambio, la región variable confiere especificidad al anticuerpo al permitir la interacción con un antígeno en particular Janda (2016). La representación estructural del anticuerpo se puede apreciar en la Figura 1.1.2.

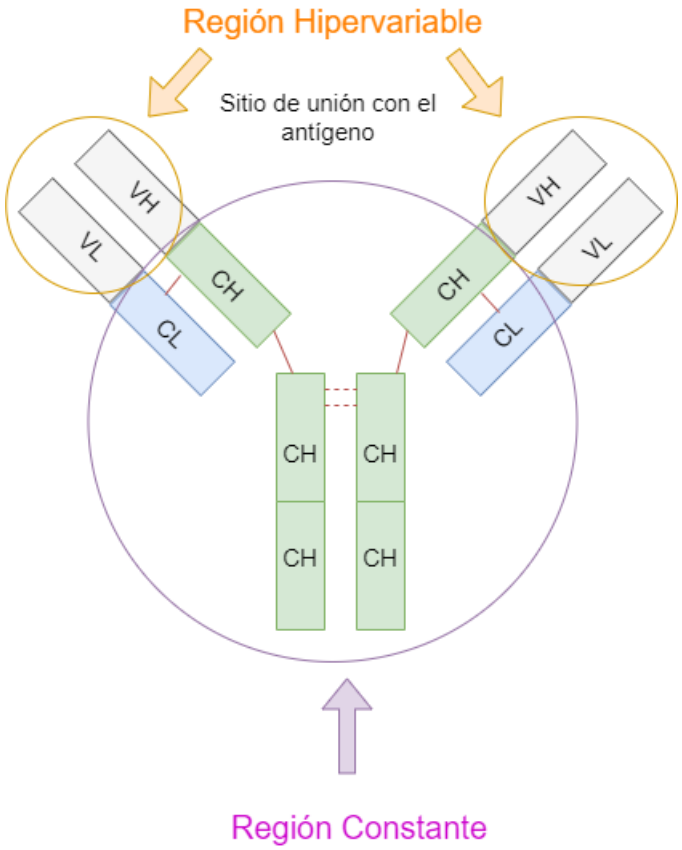

Figura 1.1.2: Representación estructural del anticuerpo. Los anticuerpos son proteínas en forma de *"Y"* conformadas por dos cadenas ligeras idénticas (L) y dos cadenas pesadas idénticas (H), además de una región constante (C) y una región variable (V) por cada cadena. La región constante revela la clase o tipo de anticuerpo (IgA, IgD, IgE, IgG e IgM), mientras que la región variable otorga la especificidad al momento de interactuar con un antígeno en particular. Las dos cadenas se emparejan mediante enlaces disulfuro para así conformar un hetero dímero idéntico correspondiente al anticuerpo.

1.1.2.2. Antígeno

Un antígeno es una molécula que es capaz de activar los linfocitos B para así estimular una respuesta inmune. En general se reconocen dos divisiones principales de antígenos Britannica (2021b):

- **Antígenos extraños (o hetero antígeno):** Se originan fuera del cuerpo, como es el caso de los virus o microorganismos como bacterias y protozoos, así como sustancias en el veneno de una serpiente, ciertas proteínas de alimentos y componentes del suero y glóbulos rojos de otros individuos.

- **Autoantígenos:** Se originan dentro del cuerpo. Esto se da en personas con trastornos autoinmunes, en donde las sustancias corporales normales provocan una respuesta inmune, lo que lleva la generación de autoanticuerpos y desencadena el proceso de autoreactividad, siendo uno de los ejemplos más comunes los pacientes con cáncer linfático.

Los antígenos poseen en su estructura regiones proteicas que pueden desencadenar una respuesta inmune medida por células T o B, denominadas epítopos. Los epítopos de células T suelen ser péptidos derivados de antígenos proteicos presentados por moléculas MHC, mientras que los epítopos de células B son péptidos o residuos de superficie de proteínas que se unen a un anticuerpo. Un antígeno puede poseer más de un epítopo, y estos son *específicos* a la región hipervariable de algún anticuerpo, permitiendo, por lo tanto, la interacción especifica antígeno-anticuerpo Roman Kogay (2019).

Existen dos tipos de epítopos, los continuos y los discontinuos. Los epítopos de las células B son comúnmente discontinuos (también llamados conformacionales o ensamblados), y consisten en segmentos múltiples cadenas reunidas por el plegamiento del antígeno. Con respecto a los epítopos continuos, o también llamados lineales, solo el 10 % de todos los epítopos reconocidos son de esta clase. Tanto los epítopos lineales como conformacionales participan en las interacciones antígeno-anticuerpo Luštrek (2013). En la Figura 1.1.3 se aprecia la diferencia estructural entre los epítopos lineales y conformacionales.

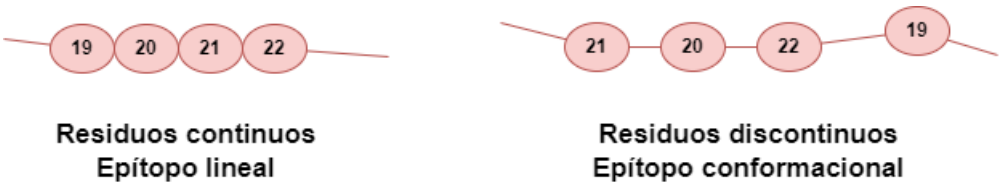

Figura 1.1.3: Representación estructural de los epítopos. Los antígenos poseen una o varias regiones proteicas denominadas epítopos, las cuales son reconocidas específicamente por los anticuerpos (región hipervariable). Hay dos tipos de epítopos, los epítopos continuos o lineales y los epítopos discontinuos o conformacionales. Estos últimos son los más comunes y se diferencian por la disposición que adoptan los residuos a lo largo de la cadena, en donde la posición de los aminoácidos varía a lo largo de la cadena a diferencia de los lineales.

Con base en lo expuesto en la sección anterior y en la sección actual, en la Figura 1.1.4 se representa la unión estructural que se da entre un antígeno y un

anticuerpo, la cual se caracteriza por ser una interacción débil, como pueden ser las interacciones electroestáticas, puentes de hidrógeno, fuerzas de Van der Waals e interacciones hidrofóbicas; y específica, debido a que la región hipervariable del anticuerpo reacciona de manera específica con uno de los epítopos del antígeno, como se explicó en la sección anterior contributors (2021).

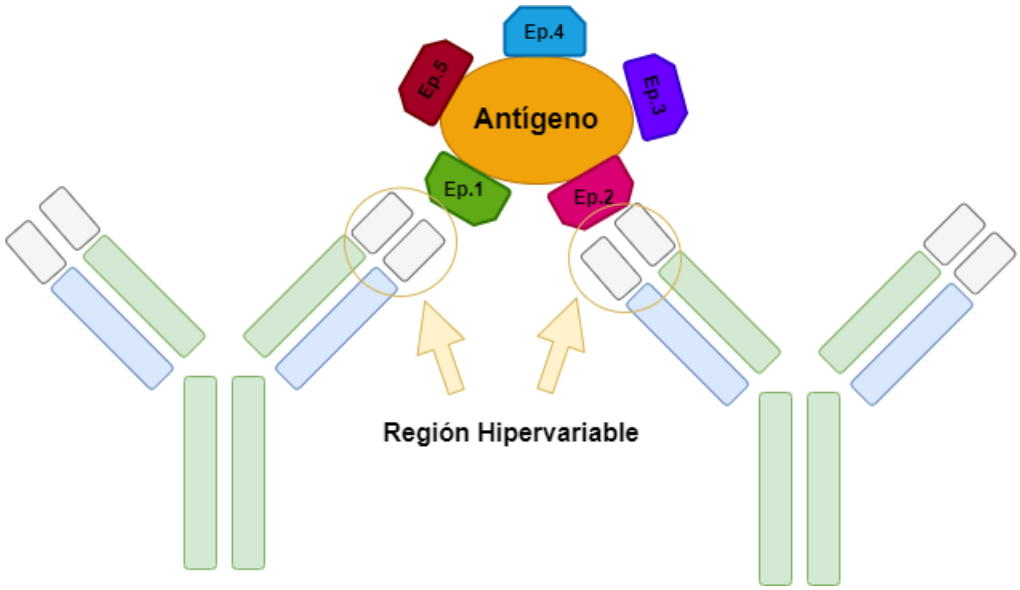

Figura 1.1.4: Representación de la interacción antígeno-anticuerpo. Los antígenos tienen regiones proteicas denominadas epitopos, los cuales interactúan de forma *específica* con la región hipervariable de un anticuerpo. En el caso de la Figura 1.1.4, el antígeno posee 5 epítopos (Ep), en donde dos de ellos están interactuando con un anticuerpo en particular.

1.1.3. Autoinmunidad

La autoinmunidad se refiere a una falla del sistema inmune al momento de reconocer sus propias células y tejidos. Esto conduce a que los anticuerpos producidos por los linfocitos B ataquen a sus propias células y tejidos como si fueran invasoras, en donde el proceso de identificación de estas células se denomina autorreactividad Dr. Mandal (2021). Esta clase de anticuerpos se les denomina autoanticuerpos, mientras que las propias células identificadas como invasoras, producto del deterioro del sistema inmune, se les denomina autoantígenos.

Existen más de 80 tipos de enfermedades inmunes, donde las más comunes son: la enfermedad inflamatoria intestinal, la diabetes tipo I, la artritis reumatoide, la celiaquía y el lupus. Otras menos frecuentes como la enfermedad de Addison o

el vitíligo. No se conocen las causas de las enfermedades autoinmunes, aunque tienden a ser hereditarias, y se cree que algunos virus, como el rotavirus y el virus de la influenza A (IAV) Smatti (2019), bacterias intestinales HERALDO (2018), y fármacos, como medicamentos de la nueva clase de tratamientos contra el cáncer Barker (2019), originan alteraciones que causan su aparición TOPDOCTORS (2021).

1.1.4. Leucemia

La leucemia es un tipo de cáncer de los tejidos que se forma en la sangre del cuerpo, así como el sistema linfático y la medula ósea MAYOCLINIC (2021). Dentro de los principales factores que generan esta enfermedad se encuentra: la exposición a sustancias radioactivas, el tabaquismo, el síndrome de Down y el síndrome mielodisplásico Sanitaria (2021). Existen distintos tipos de leucemias, las cuales se pueden dividir en dos grandes grupos: con respecto a la rapidez que propagación (leucemias agudas o graves) o con base en las células alteradas (leucemias mieloides o linfoides) Pacientes (2021). Por último, para poder tratar la leucemia existen diferentes tratamientos como inmunoterapia, radioterapia, quimioterapia, entre otros; aunque cada uno va a depender de diferentes factores como la edad y salud del paciente Clinid (2021).

1.1.5. Machine Learning y Deep Learning aplicado a Ingeniería de Proteínas

La inteligencia artificial es una área de la informática que busca simular el comportamiento humano en las máquinas. Esta área está compuesta por varios campos, pero el que más destaca entre ellos es el del Machine Learning, el cual tiene como objetivo diseñar algoritmos que permitan a los sistemas o máquinas *aprender*, tanto para tareas de predicción como de clasificación. Algunos de estos algoritmos son los árboles de decisión, modelos de regresión, modelos de clasificación, clustering, entre otras. Sin embargo, los que más han ganado fuerza este último tiempo han sido las redes neuronales, así como también el empleo de representaciones no estructuradas vectorialmente de conjuntos de datos para entrenamiento de modelos predictivos, siendo ejemplos de esto las imágenes y los grafos con sus diferentes topologías. A continuación, se detallarán los principales componentes del Machine Learning, así como también sus ventajas y desventajas

y su forma de uso.

1.1.5.1. Machine Learning e Inteligencia Artificial

La inteligencia Artificial (IA) es el término utilizado para describir el uso de computadoras y tecnología para simular un comportamiento inteligente y un pensamiento crítico comparable al del ser humano. John McCarthy Amisha (2019) describió por primera vez la IA en 1956 como la ciencia y la ingeniería para fabricar máquinas inteligentes. A su vez, la inteligencia artificial está conformada por diferentes subcampos, como puede ser el reconocimiento de voz (**Speech Recognition)**, la interpretación de texto (**Natural Lenguage Processing**), generación de máquinas que asemejen el comportamiento humano (**Robótica**) o el procesamiento de imágenes o vídeos (**Computer Vision**). Sin embargo, hay un área que destaca por sobre el resto y que es muy utilizada hoy en día, el **Machine Learning**, la cual se especializa en el diseño, implementación y validación de algoritmos capaces de *aprender* Fernández (2021).

El Machine Learning o aprendizaje automático, tiene como objetivo el que las máquinas o sistemas aprendan a partir de los datos; sin ser explícitamente programadas para ello BBVA (2019). Dicho en otras palabras, el Machine Learning se diferencia de la Inteligencia Artificial en que no es lo mismo programar un robot para que realice una acción determinada (IA), a que el robot aprenda cómo hacer esa misma tarea (ML). Para lograr esto, en la actualidad se utiliza una amplia variedad de algoritmos de aprendizaje automático programados a partir de modelos matemáticos, los cuales, por medio de ecuaciones matemáticas, buscan hacer una representación de la realidad Roldán (2020). La elección de un modelo particular para un problema dado está determinado por las características de los datos, así como por el tipo de resultado deseado Nichols (2019). Así también, el Machine Learning se divide en cuatro distintos tipos de aprendizaje, los cuales son datos.gob.es (2020):

- **Aprendizaje Supervisado:** El aprendizaje supervisado se basa en el desarrollo de modelos a partir de set de datos etiquetados, es decir, el valor a predecir, o columna respuesta, es conocida. Un ejemplo práctico de esto es predecir la venta de cervezas para un local específico en un día en particular a partir de temperatura, ciudad, mes, día de la semana, etc. En este caso, la etiqueta corresponde a la venta de cervezas en un día en específico.

Dicha etiqueta puede ser de dos tipos, categórica o numérica, donde por cada una habrá un tipo de modelo en particular:

- **Modelos de clasificación:** Los modelos de clasificación requieren de etiquetas categóricas. Las predicciones que realizan este tipo de modelos pueden ser de dos tipos: binarias, como es el caso de verdadero o falso, si tiene o no una enfermedad, etc.; o pueden ser multiclase cuando la cantidad de clases a predecir es superior a dos, como en el caso de la clasificación de tipos de clientes en una empresa, el tipo de condena a aplicar a un delincuente, entre otros. Para este tipo de modelos se encuentran los árboles de decisión, máquinas de soporte vectorial (SVM), K-vecinos (K-NN), entre otros.
- **Modelos de regresión:** Producen como salida un valor real, como por ejemplo, la cantidad de completos a vender para un local comercial dada información temporal y espacial. Dentro de este grupo se encuentra modelos de regresión y sus diferentes variaciones, así como también algoritmos similares a los nombrados en los métodos de clasificación. Pero, con respuesta generada por ponderación, como en el caso de los SVM o K-NN.

- **Aprendizaje No Supervisado:** El aprendizaje no supervisado utiliza algoritmos de aprendizaje automático para analizar y agrupar conjuntos de datos sin etiquetar. Dichos algoritmos descubren patrones ocultos o agrupaciones de datos sin necesidad de intervención humana. Los modelos de aprendizaje no supervisados se utilizan en tareas de agrupamiento, asociación y reducción de dimensionalidad, siendo una solución ideal para el análisis de datos exploratorios, estrategias de venta cruzada, segmentación de clientes y reconocimiento de imágenes, entre otros. Dentro de los algoritmos más conocidos se encuentran los agrupamientos jerárquicos y K-Medias Education (2020).
- **Aprendizaje Semi-Supervisado:** El aprendizaje automático semi-supervisado es una combinación del aprendizaje supervisado y no supervisado. Utiliza una pequeña cantidad de datos etiquetados y una gran cantidad de datos no etiquetados, lo que proporciona los beneficios de ambos tipos de aprendizaje, así como evitar los desafíos de encontrar una gran cantidad

de datos etiquetados Algorithmia (2020). Esta técnica consiste en primero etiquetar manualmente algunos datos. Luego, entrenar dicho conjunto por medio de modelos de aprendizaje supervisado. El modelo resultante de dicho entrenamiento se utiliza para caracterizar el conjunto de datos no etiquetado. Por último, se entrena el conjunto de datos por medio de algoritmos supervisados, usando como valores a predecir, las etiquetas generadas manualmente, como las etiquetas generadas por los modelos anteriores datos.gob.es (2020).

- **Aprendizaje Reforzado:** El aprendizaje por refuerzo es el entrenamiento de modelos de aprendizaje automático para tomar una secuencia de decisiones. El agente aprende a lograr un objetivo en un entorno incierto y potencialmente complejo. En el aprendizaje por refuerzo, la inteligencia artificial se enfrenta a una situación similar a un juego, en donde una computadora emplea prueba y error para encontrar una solución al problema. Para que la máquina haga lo que quiera el programador, la inteligencia artificial obtiene recompensas o penalizaciones por las acciones que realiza hasta maximizar la recompensa total Osiński (2018).

Los modelos resultantes, ya sea por Machine Learning deben ser evaluados para así determinar si predicen de forma correcta. Una herramienta muy útil para evaluar es la matriz de confusión, la cual sirve para mostrar de forma explícita cuándo una clase es confundida con otra, permitiendo, por lo tanto, trabajar de forma separada con distintos tipos de errores Barrios (2020).

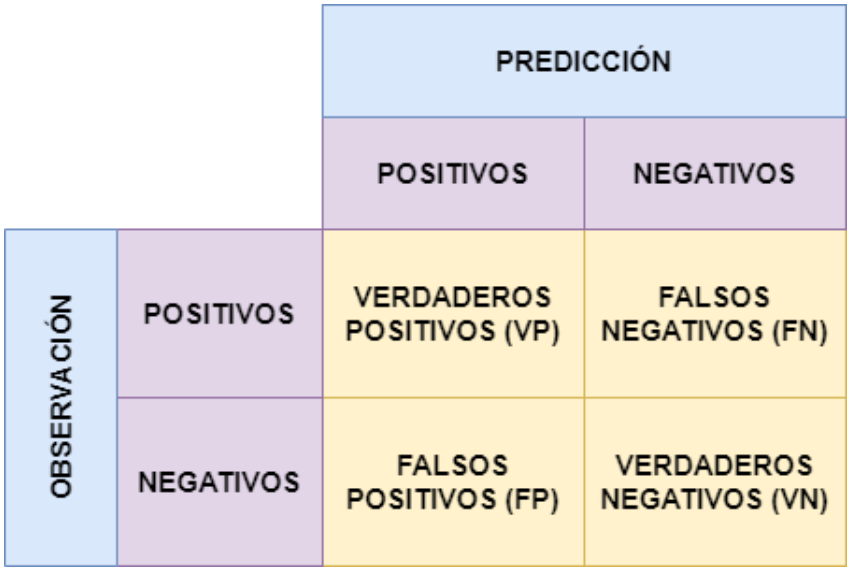

Figura 1.1.5: Matriz de confusión general. La matriz de confusión es una herramienta muy útil para determinar la cantidad de clases que fueron predichas correctamente. Las filas de la matriz corresponden a los valores observables, mientras que las columnas a los valores predichos por el modelo resultante. La intersección de filas y columnas genera cuatro estimadores: Verdaderos Positivos (VP), Falsos Negativos (FN), Falsos Positivos (FP) y Verdaderos Negativos (VN). A partir de estos valores, se pueden determinar cinco variables o métricas de la matriz de confusión, como lo son: Exactitud (Accuracy), Precisión, Sensibilidad (Recall), Especificidad y el Coeficiente de Correlación de Matthews (MCC). Cada uno de estos estimadores permite evaluar que tan bien predice un modelo de predicción.

En la Figura 1.1.5 se puede apreciar la estructura general de una matriz de confusión, donde las filas corresponden a los valores observados o reales, mientras que las columnas corresponden a los valores que el modelo predijo. Además, la intersección de ellos da el resultado de cuatro estimadores Barrios (2021):

- **Verdadero Positivo:** El valor real es positivo y el modelo lo predice como positivo, es decir, en el caso de un modelo de diagnóstico clínico, la persona está enferma y el modelo predice que la persona si está enferma.
- **Verdadero Negativo:** El valor real es negativo y el modelo lo predice como negativo, es decir, en el caso de un modelo de diagnóstico clínico, la persona no está enferma y el modelo así lo demuestra.
- **Falso Negativo:** El valor real es positivo y el modelo lo predice como negativo, es decir, en el caso de un modelo de diagnóstico clínico, la persona está enferme, pero el modelo predice que la persona no está enferma. Se le

denomina **error tipo II**.

- **Falso Positivo:** El valor real es negativo, pero el modelo lo predice como positivo, es decir, en el caso de un modelo de diagnóstico clínico, la persona no está enferma, pero el modelo predice que si esta enferma. Se le denomina **error tipo I**.

A partir de estas cuatro variables, surgen diferentes métricas asociadas a la matriz de confusión, las cuales son:

- **Exactitud (Accuracy):** La exactitud es una medida que se calcula con base en las predicciones correctas con respecto al total de datos. Es una buena medida cuando todas las clases del conjunto de datos tienen la misma importancia.

$$\textbf{Accuracy} = (\text{VP}/\text{VN})/(\text{VP}+\text{FP}+\text{FN}+\text{VN})$$

- **Precisión:** La precisión es una medida que determina que tan bien predijo un modelo. Se calcula con base en las predicciones positivas clasificadas correctamente, con respecto al total de muestras clasificadas positivamente (correcta o incorrectamente).

$$\textbf{Precisión} = \text{VP}/(\text{VP}+\text{FP})$$

- **Sensibilidad (Recall):** La sensibilidad mide la capacidad de un modelo de predecir clases positivas correctamente, estableciendo la relación entre las muestras correctamente clasificadas como positivas, con respecto al total de muestras positivas.

$$\textbf{Recall} = \text{VP}/(\text{VP}+\text{FN})$$

- **Especificidad:** La especificidad corresponde a la proporción de muestras correctamente clasificadas como negativas, con respecto al total de muestras negativas (correctas e incorrecta).

$$\textbf{Especificidad} = \text{VN}/(\text{VN}+\text{FP})$$

- **Matthews Correlation Coefficient (MCC):** El coeficiente de correlación de Matthews mide las diferencias que existen entre los valores reales y las predicciones realizadas, tomando en consideración, los verdaderos negativos y positivos, así como los falsos negativos y positivos Shmueli (2019).

$$\mathbf{MCC} = \frac{VPXVN - FPXFN}{\sqrt{(VP+FP)(VP+FN)(VN+FP)(VN+FN)}}$$

Uno de problemas que sufren los modelos de machine learning se denomina sobre ajuste, u *overfitting*. El sobre ajuste se da cuando un modelo está muy ajustado a los datos de entrenamiento, impidiendo que generalice bien a los datos de prueba. La idea de un modelo de aprendizaje automático es el de obtener patrones de los datos de entrenamiento de cara a predecir o inferir correctamente datos *nuevos*. Esto ocurre a raíz de que un modelo se entrenó demasiado o con datos anómalos; llevando a aprender de características específicas pero no de patrones generales. Un caso similar es en los seres humanos, donde el sobre ajuste se genera cuando se aprenden las cosas de memoria, sin entender el concepto Gonzalo (2020).

Existen diferentes métodos para poder evitar el sobre ajuste, los cuales se mencionan a continuación Chuan-En Lin (2020):

- **Hold-out**: Dividir el conjunto de datos en dos: entrenamiento y prueba. Una proporción de división común es 80 % entrenamiento y 20 % para pruebas, siendo esta la proporción estándar pero no la única.
- **Cross-validation**: Dividir el conjunto de datos en k grupos. La idea es que uno de los grupos sea el conjunto de prueba y los otros formen el conjunto de entrenamiento, y repetir este proceso hasta que cada grupo individual se haya utilizado como conjunto de prueba (k repeticiones), la idea es que en cada iteración del entrenamiento, el modelo estará prediciendo valores que no conoce. En la figura 1.1.6 se puede apreciar una representación básica de una validación cruzada.
- **L1/L2 regularization**: La regularización L1 o L2 consiste en agregar penalización en la función de costo, en donde L2 permite que los pesos decaigan hacia cero, pero sin llegar a cero, mientras que la regulación L1 si permite llegar a cero.
- **Remove layers**: Eliminar capas y reducir el tamaño del modelo, así como también reducir el número de neuronas por capa.
- **Dropout**: Las capas ignoran un subconjunto de unidades de la red con una probabilidad establecida.

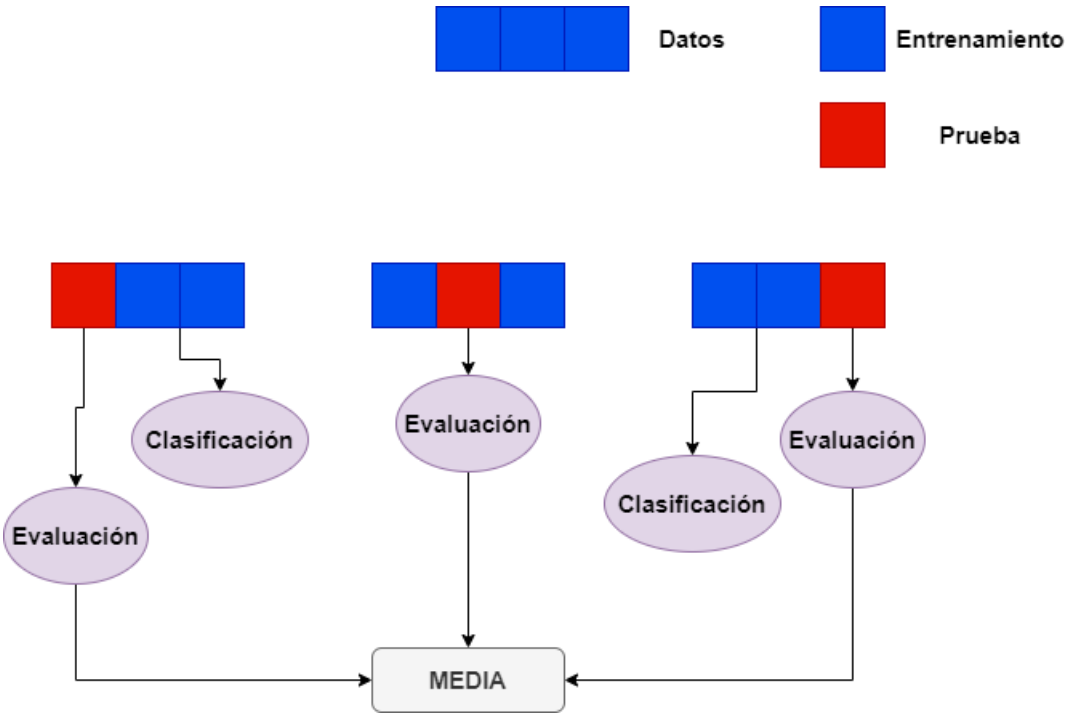

Figura 1.1.6: Representación de la validación cruzada (Cross-validation). El conjunto de datos se divide en k grupos, donde un gran parte de ese conjunto se ocupa para entrenar, mientras que la restante de prueba. Este proceso se repite k veces, para que así cada grupo individual sea utilizado de prueba.

1.1.5.2. Estrategias de representación de proteínas

Uno de los problemas más complejos de tratar a la hora de aplicar métodos computacionales basados en ML y minería de datos a proteínas es su representación numérica. Existen diferentes estrategias que han sido desarrollados a lo largo del tiempo y que han permitido el desarrollo satisfactorio de sistemas de predicción. No obstante, pese a los grandes avances, el problema aún persiste, lo cual hace necesario diseñar e implementar nuevas metodologías para solventar este problema Wittmann et al. (2021).

Las principales estrategias de representación pueden clasificarse en tres: Numéricas, empleando imágenes, y aplicando estructuras de grafos (Figura 1.1.7). Dependiendo del tipo de formato en el que se encuentre la secuencia, es posible aplicar variadas estrategias de representación. Para el caso de secuencias lineales (solo la estructura primaria) normalmente son aplicadas representaciones numéricas tales como: One Hot Encoder, aplicación de propiedades fisicoquímicas y recientemente el uso de transformadores basados en Natural Language Processing como es el

caso de BERT Rao et al. (2019a). Además, propiedades fisicoquímicas se han combinado junto con técnicas de digital signal processing para simular los efectos y relaciones estructurales a nivel lineal Medina-Ortiz et al. (2020). Por otro lado, la representación de proteínas mediante imágenes ha ido adquiriendo mayor relevancia últimamente. Ejemplos clásicos han usado la estructura 3D como input, no obstante, estudios recientes han empleado propiedades fisicoquímicas, e incluso representaciones numéricas para obtener imágenes desde la secuencia lineal ST et al. (2021). Finalmente, las estructuras de grafos han permitido representar las proteínas a nivel estructural, facilitando la identificación de patrones y el entrenamiento de modelos predictivos. Normalmente, se emplean métricas de distancia para establecer las conexiones entre los nodos. Sin embargo, no existe un consenso sobre qué estrategia utilizar, cuáles son las distancias óptimas o cómo representar interacciones electrostáticas débiles Gaudelet et al. (2021).

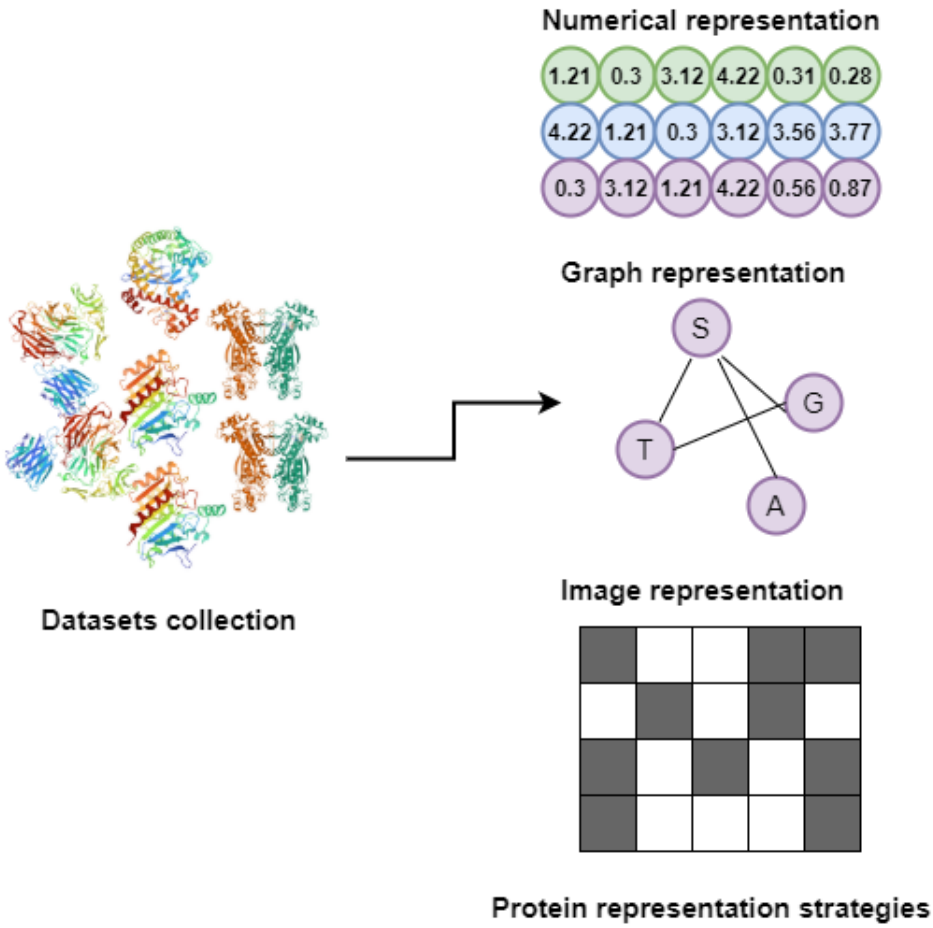

Figura 1.1.7: Principales estrategias de representación de proteínas. La representación de proteínas es de vital importancia a la hora de desarrollar modelos predictivos o identificar patrones. Es posible dividirlas en tres grandes grupos. Numéricas, empleando imágenes, y aplicando estructuras de grafos. Dentro de las numéricas se encuentran las conocidas como One Hot Encoder, el uso de propiedades fisicoquímicas, la combinación de Digital signal processing y recientemente, el uso de transformadores basados en procesamiento de lenguaje natural. Por otro lado, el uso de imágenes puede provenir desde la estructura 3D o empleando técnicas de fingerprinters. Finalmente, las estrategias de grafos facilitan la aplicación de modelamiento matemático discreto con el fin de identificar patrones o características similares.

1.1.5.3. Grafos

Para entender que son los grafos, primero se debe viajar al año 1736, más específicamente a Königsberg (actualmente Kaliningrado, Rusia). Esta era una ciudad cercana a la costa báltica y fue fundada en el siglo XIII en el río Pregolya de Prusia. La ciudad se extendía por las dos orillas del río y las islas intermedias; en total, tenía cuatro regiones de tierra, conectadas por siete puentes S (2019).

Dada la distribución geográfica de la ciudad, los habitantes se plantearon el desafío de recorrer cada una de las regiones2 pasando solo una vez por cada puente y regresar al punto de origen.

Este caso llamó la atención del matemático Leonard Euler, quién se encontraba en Königsberg trabajando en la Academia Prusiana de las Ciencias. Euler resolvió este problema a fuerza bruta, es decir, comprobó todas y cada una de las posibles combinaciones, y demostró que este problema no tenía solución. Ante esto, Euler busco resolver este problema representando la ciudad de Königsberg como un grafo, centrándose en las regiones a unir (**Nodos**), los puentes (**Aristas**) y el **grado** al número de aristas que salen de un nodo Luisyep (2019a). Esto dio el inicio a la teoría de grafos, en la cual expico que para hacer el recorrido deseado por los habitantes, se debía pasar al menos 2 veces por alguno de los puentes o nodos. Esta teoría establece las bases de lo que son y como se comportan los grafos, y forman parte fundamental de la informática y de la gestión de bases de datos Luisyep (2019b).

A partir de la teoría, Eulier definió un grafo como una representación gráfica de la realidad conformada por nodos o vértices y sus lazos o aristas que sirven para establecer relaciones entre ellos Rochina (2017). Desde una mirada matemática, un grafo $G = (V,E)$ es una pareja formal ordenada en la que V es el conjunto de vértices y E es un conjunto de aristas. Además, E consta de pares no ordenadas de vértices, tales como $x,y \in E$. En la figura 1.1.8 se puede apreciar una representación con base en un grafo, en donde los nodos corresponden a las ciudades de España y las aristas a la distancia que la separa. Como se puede apreciar, hay nodos que solo tienen un único tipo de relación (simple), como la ciudad de Cádiz, mientras que otros nodos presentan más de un tipo de relación (múltiple), como es el caso de Granada.

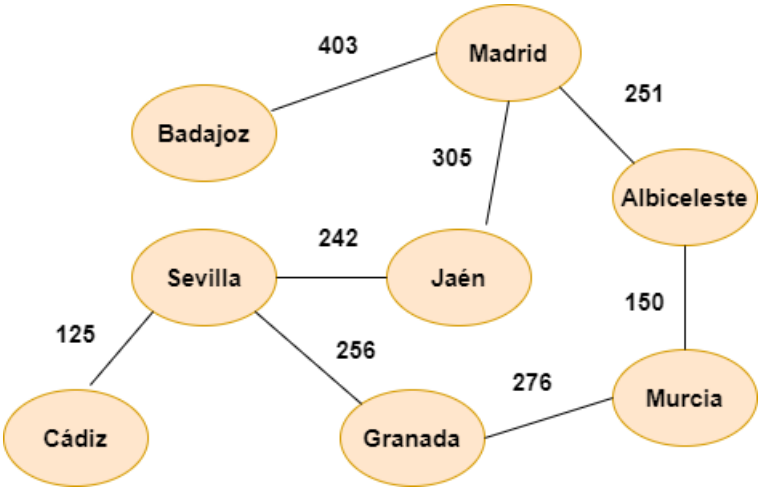

Figura 1.1.8: Ejemplo de un grafo. Ejemplo básico de un grafo, en donde los nodos corresponden a ciudades de España y las aristas a la distancia (Km) que las separa. Se puede apreciar que hay nodos que solo poseen una relación (simple), como la ciudad de Cádiz, mientras que hay otros nodos que mantienen más de una relación (múltiple), como la ciudad de Granada y Sevilla (Francisco J. Gil Gala, 2015).

1.1.5.4. Deep Learning

Anteriormente, se explicaron las diferentes técnicas para cubrir los distintos tipos de aprendizaje, como pueden ser los modelos de regresión, modelos de clasificación, clustering, etc.; pero existe una técnica que cada vez ha adquirido mucha más fama y fuerza dentro del Machine Learning, las denominadas redes neuronales.

Las redes neuronales artificiales (RNA) son un tipo de modelo que busca reflejar el comportamiento del cerebro humano, más en específico, el funcionamiento de las neuronas. Tienen como unidad fundamental los nodos (neuronas), las cuales tienen la labor de *aprender*. La acumulación de nodos se denomina una capa, donde entre más capas tenga una red neuronal, mayor será la capacidad de aprendizaje y profundidad de la red neuronal Group (2019).

En la figura 1.1.9 se aprecia un esquema básico de una red neuronal, en donde se puede distinguir los **nodos de entrada** que reciben una serie de datos desde el exterior. Luego, estos datos son enviados a los **nodos ocultos**, en los cuales la información es procesada, modificada y se transfiere de una capa a otra (puede haber muchas capas ocultas). Este proceso se conoce como aprendizaje, pues cada nodo oculto va aprendiendo de las capas más externas. Finalmente, se encuentra la **capa de salida**, la cual corresponde al modelo resultante del proceso de

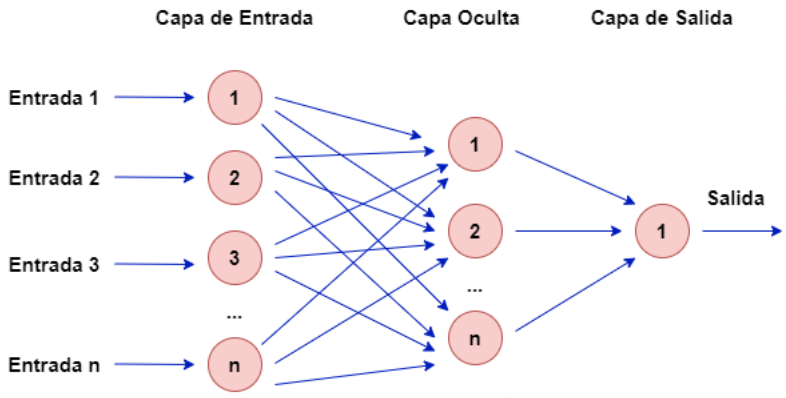

Figura 1.1.9: Esquema de una red neuronal. La red neuronal más básica se compone por tres capas: una capa de entrada con los datos sin procesar. Una capa oculta, la cual procesa, modificar y transfiere la información de una capa a otra. Este proceso se conoce como aprendizaje. Y por último, una capa de salida, la cual corresponde al modelo resultado del proceso de aprendizaje por parte de las capas ocultas.

aprendizaje y que tiene como finalidad ayudar en la predicción de algún problema en particular.

Como se explicó anteriormente, la red neuronal más básica se compone de tres capas: una capa input, para las variables de entrada, una capa oculta, donde se realiza la etapa de aprendizaje, y una capa output, la cual genera el modelo resultante. A partir de esto, existe un caso particular de las redes neuronales, el **Deep Learning, o Aprendizaje Profundo**, el cual se distingue por tener en su estructura, muchas capas ocultas. Los modelos de deep learning se caracterizan por poseer variadas arquitecturas para las redes neuronales y generar un procesamiento no lineal de los datos ITELLIGENT (2018).

Los modelos de deep learning requieren de datos etiquetados, es decir, necesitan una columna respuesta conocida que predecir. Este set de datos es ingresado a la capa de entrada, para posteriormente ser procesados por las capas ocultas. Una vez que se realiza toda la etapa de aprendizaje, la capa de salida devuelve el modelo resultante. Un ejemplo práctico de esto es el procesamiento de imagen, donde en una primera instancia la imagen es procesada a una matriz de píxeles e ingresada a la capa de entrada. Luego a partir de esa matriz se pueden desprender característica como la aparición o no de ejes, en una segunda instancia la unión de dichos ejes, en tercer lugar realizar combinaciones que corresponderían a partes del objeto, así sucesivamente. Lo que más destaca al deep learning, es que el proceso

de aprendizaje se realiza sin intervención humana, sino aprendiendo directamente de los datos brutos.

El aprendizaje profundo consta de redes neuronales con arquitecturas variables, en donde estás redes se basan en varias capas que procesan datos: una capa de entrada (datos sin procesar), capas ocultas (procesan y combinan datos de entrada) y una capa de salida (produce el resultado: clasificación, estimación, pronóstico, etc.). Existen diferentes tipos de redes neuronales, y las arquitecturas de aprendizaje profundo se basan en estas redes Lisowski (2020).

Una arquitectura define la forma en que se estructura un modelo de aprendizaje profundo, y lo que es más importante, definir para qué esta diseñado, que se busca predecir, lo que se espera como variable input y output y la combinación de capas ocultas y como fluyen los datos dentro de ellas. Algunas de las arquitecturas más importantes son Pingel (2021):

- **Convolutional neural networks (CNN):** Las CNN constan de muchas capas, pero siguen algo parecido a un patrón de convolución ReLU Brownlee (2019) que se repite una y otra vez. Son útiles para clasificar imágenes porque son muy buenos en la coincidencia de patrones espaciales locales. Otra razón es que las CNN se basan en la convolución, donde al convertir una imagen de entrada en varios filtros, se resaltan las características de la imagen sin perder la interacción espacial entre los píxeles adyacentes.

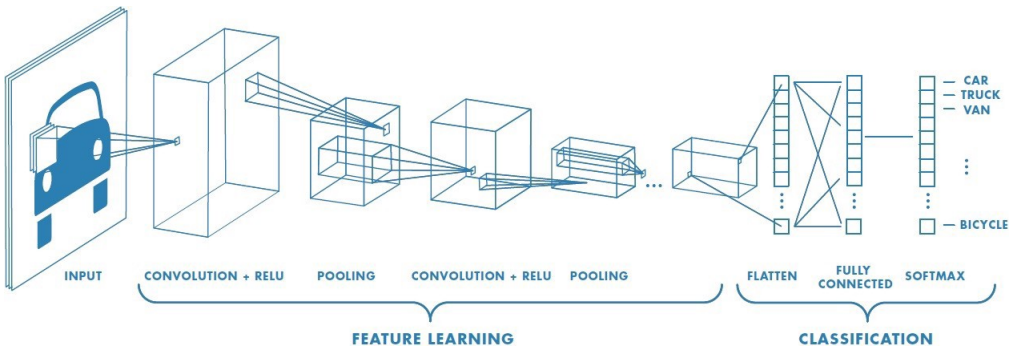

Figura 1.1.10: Arquitectura base para una Convolutional Neural Network (Mayank Mishra, 2020).

- **Recurrent neural networks (RNN):** Los RNN tienen conexiones que realizan un seguimiento de la información anterior para realizar predicciones futuras. A diferencia de las CNN, donde se supone que cada entrada es un

evento independiente, las RNN pueden procesar secuencias de datos que podrían afectarse entre sí. Un ejemplo es el procesamiento del lenguaje natural, donde las palabras anteriores influyen en la probabilidad de lo que sigue.

- **Long short-term memory (LSTM):** Las redes de memoria a corto plazo se asocian principalmente con series de tiempo y datos de secuencia. Las redes LSTM recuerdan una parte de los datos antes de tomar desiciones; ven los datos en contexto, lo que ayuda a hacer mejores asociaciones.
- **Generative adversarial networks (GAN):** Las GAN pueden generar nuevos datos basados en datos existentes, por ejemplo, imágenes de personas que en realidad no son personas reales.
- **Autoregressive Models:** Los modelos autorregresivos profundos son modelos secuenciales, pero de avance (es decir, no recurrentes). Los modelos generativos, pero supervisados son una alternativa convincente a las RNN para los datos secuenciales y a las GAN para las tareas de generación geprgeho (2019).
- **Graph Neural Networks (GNN):** Las redes neuronales de grafos, o Graph Neural Networks (GNN), son un tipo de red neuronal que funciona con grafos como input o variable de entrada, a diferencia de otras redes neuronales que reciben vectores, tensores o matrices, como es el caso de las imágenes. El uso de grafos permite que las GNN puedan aprender información más compleja, ya que no corresponden a números aislados (nodos), sino que también las relaciones (aristas) entre ellos PhD. Maldonado (2021). Las redes neuronales de grafos (GNN) se basan en tareas de clasificación de nodos, predicción de enlaces (aristas) y la agrupación de clústeres Zhou et al. (2020).
- **Graph Convolutional Neural Network (G-CNN):** Las GCN es una variante de las GNN, en donde la red neuronal ocupa capas de convolución y puede trabajar directamente con grafos y aprovechar su información estructural. Este tipo de arquitectura resuelve el problema de clasificar nodos (como documentos) en un grafo (red de citas), donde las etiquetas solo están disponibles para un pequeño subconjunto de nodos Pham (2020).

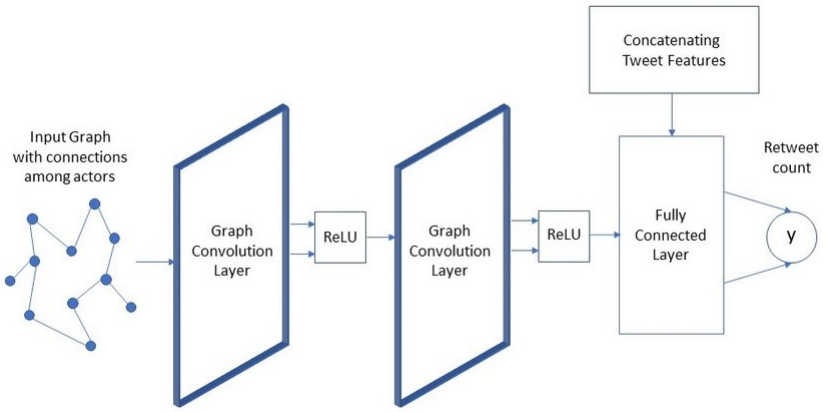

Figura 1.1.11: Arquitectura base para una Graph Neural Network (George Mohler, 2018).

1.2. Estado del Arte

Las redes neuronales de grafos tienen variadas aplicaciones en ingeniería de proteínas y la bioinformática, como por ejemplo, la determinación de SNP (Polimorfismo de Nucleótido Único) para estimar el impacto de las variantes no codificantes en la metilación del ADN L. et al. (2013), así como la predicción de la función, estructura e interacciones proteína-proteína (PPI) VS. et al. (2014). Con respecto a esta última, una de las aplicaciones más interesantes es ProteinSolver, la cual es una red neuronal con la que se ha demostrado que se puede diseñar con precisión secuencias que se plieguen en una forma predeterminada. Dicha red fue entrenada por más de 70.000.000 secuencias de proteínas reales correspondientes a más de 80.000 estructuras. ProteinSolver diseña rápidamente nuevas secuencias de proteínas y las compara *in silicio* utilizando puntuaciones basadas en energía, dinámica molecular y métodos de predicción de estructuras. Un ejemplo de esto fue la generación de secuencias que coinciden con la estructura de albúmina sérica, para luego ser sintetizadas con mayor puntuación y validadas *in vitro* mediante dicroísmo circular Strokach et al. (2020).

Otra aplicación de Deep Learning es DeepFRI, la cual se enfoca en la predicción de la función de las proteínas. DeepFRI es una red convolucional para predecir las funciones de las proteínas aprovechando las características de la secuencia extraídas de un modelo de lenguaje LSTM de proteínas y de estructuras de las proteínas. Supera a los principales métodos actuales y a las redes neuronales convolucionales basadas en secuencias y se adapta al tamaño de los repositorios de secuencias

habilitados a la fecha. DeepFRI tiene una importante capacidad de *de-noising* y su mapeo de activación de clases permite la predicción de funciones a una alta resolución, permitiendo anotaciones específicas con respecto a residuos de forma automatizada. Su utilidad y alto rendimiento (0,70) ha sido probada haciendo la predicción de proteínas funcionales de estructuras de PDB y de SWISS-MODEL V et al. (2021).

Por otro lado, se tiene a AlphaFold, el cual es un programa de inteligencia artificial (IA) que realiza predicciones de la estructura de proteínas mediante aprendizaje profundo. AlphaFold fue diseñado por DeepMind y ha sido entrenado con más de 170.000 proteínas de los depósitos públicos de secuencias y estructura de proteínas. Este programa se destaca, ya que en la CASP14 fue el método de predicción de la estructura de proteínas mejor clasificado por un amplio margen, produciendo predicciones con gran precisión. Dentro de algunos datos, el 88 % de las predicciones de AlphaFold tuvieron una puntuación por sobre los 80 puntos, donde el RMSD promedio para la posición de los átomos fue de 2.1 Å. Además, el programa logró determinar cuatro estructuras que hasta entonces no se habían podido elucidar de *novo* con métodos experimentales, como fue el caso de Af1503, una proteína de membrana estudiada durante 10 años AlphaFold (2022).

Una aplicación de graph neural network fue la otorgada por el equipo de K et al. (2019), quienes utilizaron GCN para predecir la interacción entre proteínas utilizando la información estructural de la proteína y características de la secuencia. Los grafos se construyeron a partir de los archivos PDB que contenían las coordenadas 3D de los átomos, logrando así generar una red de aminoácidos o residuos conectados entre sí por un umbral de distancia (6 angstroms). Para poder extraer características de los nodos, utilizaron un modelo de lenguaje de las proteínas en donde la entrada era la secuencia aminoacídica y la salida un vector de características de cada aminoácido de la secuencia subyacente mediante propiedades fisicoquímicas. Dicha metodología fue validada en dos conjuntos de datos de PPI en humanos y *S. cerevisiae*, obteniendo resultados con una eficacia de hasta 0,90.

Para el caso en particular del sistema inmune, se han utilizado técnicas de aprendizaje profundo aplicado al desarrollo de fármacos para anticuerpos, haciendo uso de métodos de Lead Optimization, Ens-Grad, DeepInterface, MaSIF-Search y TopNetTree J. et al. (2020). Sin embargo, la interacción antígeno-anticuerpo

sigue siendo un campo poco abordado por la ciencia, y menos con el uso de Deep Learning, debido principalmente a la generalización de los comportamientos y la complejidad de la extracción de patrones, en particular gracias a las infinitas combinaciones de interacciones, la variabilidad en el proceso de maduración de los anticuerpos y la falta de estimadores robustos de sistemas de regiones epítopos para poder incorporarlos como método de información al sistema.

Para poder representar las proteínas como grafos, los aminoácidos de las proteínas se utilizan como nodos, mientras que las aristas pueden venir a partir de interacciones electroestáticas o por las distancias, euclidianas, coseno, etc.; entre los residuos Brinda and Vishveshwara (2005). Con respecto a la presentación de los aminoácidos, normalmente se expresan a través de una representación vectorial de los elementos codificados. En este sentido, los métodos de codificación más comunes es por: one-hot, VHSE8 (propiedades fisicoquímicas), BLOSUM (capturar relaciones evolutivas), ProtVec (representación vectorial aprendida de cien millones para k-mers superpuestos fijos de aminoácidos) y mediante métodos de learning embedding H. et al. (2020).

Lo que respecta a la composición de las redes neuronales de grafos, estas utilizan arquitecturas basadas en métodos espaciales y espectrales principalmente, siendo estas últimas las más usadas debido a su menor consumo y demanda de requerimientos al trabajar en cuanto a nodos por medio del método del traspaso de mensaje, a diferencia de los métodos espaciales que hacen uso de todo el grafo a través de transformadas de Laplace. Por otro lado, al momento de querer clasificar nodos, se optan por arquitecturas basadas en GraphSAGE, VR-GCN y PinSAGE principalmente, mientras que para clasificar grafos se utilizan metodologías basadas en ClusterGCN y GraphSAINT Karagiannakos (2021). Generalmente, se hacen uso de alrededor 6 capas ocultas (llegado a un punto la red deja de aprender por el método de traspaso de mensaje), empleando una función de activación ReLU, además de emplear comúnmente un batch size de 32, iterando 100 épocas y esperando una función de pérdida mínima de 0.6.

1.3. Justificación del problema

Diferentes estrategias de machine learning y deep learning han sido aplicadas a variadas tareas de ingeniería de proteínas, tales como la predicción de

cambios termodinámicos producidos por mutaciones, efectos clínicos de variantes, clasificación de familias o funcionalidades, entre otros. No obstante, uno de los problemas más complejos asociados a la predicción se centra en la evaluación de complejos de interacción proteína-proteína, siendo de especial interés para áreas como la biotecnología y la medicina. Para el caso en concreto de la interacción antígeno-anticuerpo, esta es de suma importancia para el sistema inmunológico y de gran relevancia para herramientas diagnósticas, biotecnológicas y terapéuticas. Comprender como los anticuerpos interactúan específicamente con sus antígenos puede permitir un mejor diseño de vacunas y medicamentos, así como proporcionar conocimientos sobre la inmunidad natural. Para lograr esto se necesitan métodos que generen predicciones más precisas que los métodos experimentales, con un tiempo de ejecución óptimo, con bajo margen de error y con menos costo de producción.

Para abordar las interacción proteína-proteína, se han aplicado varios enfoques desde el punto de vista bioinformático, como el uso de estrategias de simulación molecular para comprender las dinámicas de interacción. Sin embargo, su costo computacional es elevado y requiere de ajustes directos sobre cada sistema. No obstante, no han sido explorados enfoques de Graph Neural Network como estrategia de representación de proteínas, debido a que no hay consenso sobre la forma del diseño del grafo, así como sus aplicaciones en ingeniería de proteínas. Además, no se han explorado estrategias de generación de grafos a partir de la información lineal de secuencias.

Tomando en consideración los antecedentes, se plantea la exploración de estructuras de grafo como método de representación numérica para el diseño e implementación de modelos predictivos de interacción antígeno-anticuerpo que combine estrategias de Deep Learning y Graph Neural Network, con el fin de generar sistemas de clasificación de la intensisdad de interacción entre antígenos y anticuerpos.

Capítulo 2

Hipótesis y objetivos

2.1. Hipótesis

La aplicación de estructuras de grafos como estrategia de representación de complejos antígeno-anticuerpo, facilita el desarrollo de modelos predictivos basados en técnicas de Deep Learning.

2.2. Objetivo General

Crear modelos predictivos para estudiar la interacción proteína-proteína mediante la aplicación de representación de grafos, modelamiento matemático discreto y técnicas de aprendizaje profundo.

2.3. Objetivos Específicos

A partir del objetivo general, nacen los siguientes objetivos específicos:

1. Elaborar estrategias de predicción de complejos proteína-proteína.
2. Evaluar diferentes estrategias de representación numérica de proteína combinadas con estructuras de grafos.
3. Explorar arquitecturas de Deep Learning basada en graph neural network para el entrenamiento de modelos de predicción.

Capítulo 3

Metodología

El objetivo de la tesis fue buscar la mejor forma de representación y clasificación para complejos de interacción proteína-proteína asociados a interacciones antígeno-anticuerpo, empleando estructuras de grafos y técnicas de Deep Learning, enfatizando en Graph Neural Netowork (GNN) y Graph Convolutional Neural Network (GCNN). Como ya se explicó anteriormente, los grafos y el deep learning se pueden trabajar a partir de 3 enfoques: nodos, aristas y grafos; donde para efectos de la investigación se aplicaron los enfoques de los nodos y los grafos. En la Figura 3.0.1 se aprecia un resumen de las actividades realizadas, donde ambas comparten similitudes, pero se diferenciaron en el procesamiento inicial de los datos. El detalle de la metodología se describe a continuación:

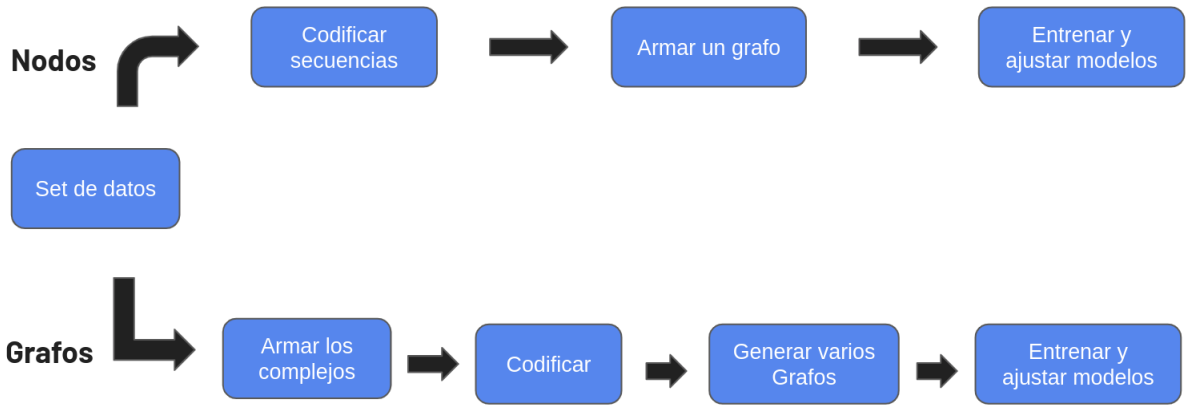

Figura 3.0.1: Metodología. Actividades a realizar para ambos enfoques a la hora de trabajar con grafos y deep learning.

3.1. Conjunto de datos

El conjunto de datos proviene de un estudio previo de la Universidad Leiden, Alemania, el cual recopiló repertorios de anticuerpos sobre pacientes leucémicos. Este set de datos contiene información relacionada con complejos de interacción antígeno-anticuerpo, y está conformado de aproximadamente 45 anticuerpos (cadena pesada y cadena ligera) provenientes de personas con leucemia y 8.000 antígenos provenientes de enfermedades autoinmunes (autoantígenos). Los experimentos fueron desarrollados empleando tecnologías de ProtoArray y procesados para su estimación numérica de intensidad. Luego, se eliminaron elementos ruidosos empleando la metodología propuesta por Torres Jorge et al. (2019). Finalmente, empleando representaciones estadísticas, se categorizaron los valores de intensidad en tres tipos de interacción: Fuerte, Mediana y Débil, la cuales hacen referencia al nivel de interacción que hay entre las proteínas, donde por ejemplo: en una interacción de tipo fuerte, se entiende que hay una mayor cantidad de enlaces interactuando. En total se registraron 200.000 interacciones aproximadamente. Al mismo tiempo, debido a trabajos previos realizados por el grupo de investigación, se cuenta con la información de las regiones epítopos y los segmentos CDR para las secciones hipervariantes de los anticuerpos, los cuales fueron utilizados para el desarrollo de los complejos de interacción mediante técnicas de bioinformática estructural.

3.2. Enfoques de trabajo

3.2.1. Enfoque: Nodos

Este enfoque se centra en las secuencias amino acídicas de las proteínas y consistió en armar un único grafo que represente toda la red de proteínas. En la Figura 3.2.1 se aprecia un esquema representativo del grafo a generar, en donde los nodos corresponden a proteínas, en este caso antígenos y anticuerpos, y las aristas al nivel de intensidad o el tipo de interacción que poseen entre ellas, es decir, el target o clasificador. Al mismo tiempo, los nodos son caracterizados por las codificaciones de las secuencias de las proteínas, es decir, empleando técnicas de representación numérica de secuencias lineales de aminoácidos.

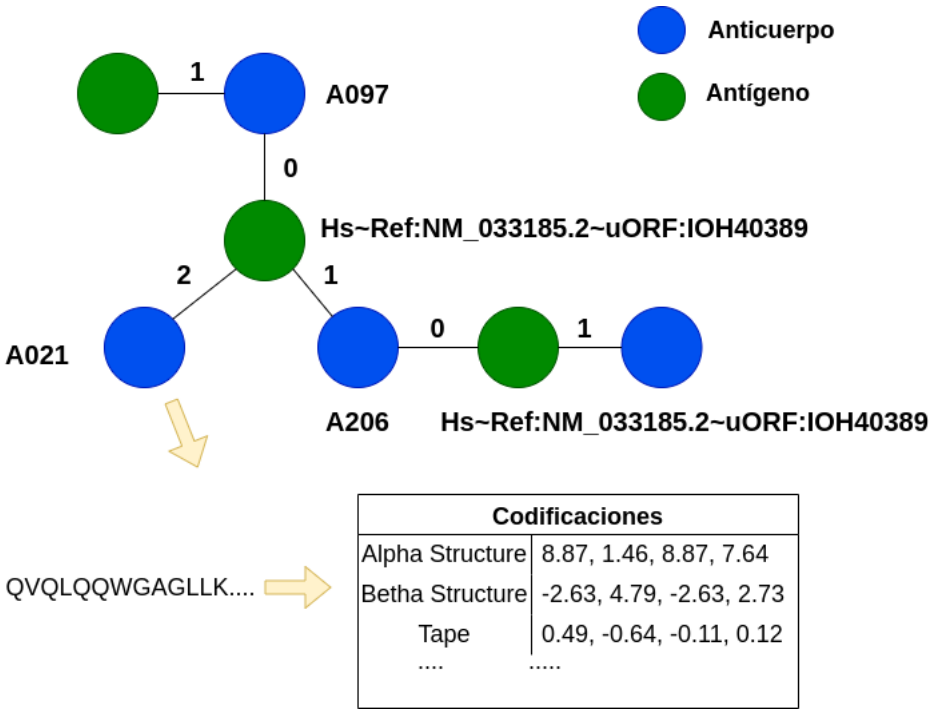

Figura 3.2.1: Metodología. El enfoque de los nodos consiste en armar un grafo que represente toda la red de proteínas, en donde los nodos corresponden a la secuencia codificada de las proteínas, mientras que las aristas a los niveles de interacción entre ellas.

Para lograr este objetivo, los pasos a seguir fueron los siguientes:

1. Dado que este enfoque hace uso de las secuencias amino acídicas, este punto se encuentra cubierto, ya que proviene del set de datos inicial.

2. Las secuencias de las proteínas fueron codificadas, esto debido a que una red neuronal de grafos no se puede entrenar con letras. Para este punto se aplicaron tres métodos:

   - **One Hot Encoding:** Esta estrategia consiste en crear una columna para cada valor distinto que exista en la característica que estamos codificando y, para cada registro, marcar con un 1 la columna a la que pertenezca dicho registro y dejar las demás con 0 InteractiveChaos (2021).

   - **Propiedades fisicoquímicas:** A partir de 8 propiedades fisicoquímicas propuestas en Fontaine and Cadet (2019) se va a obtener una codificación de la misma secuencia por propiedad, para así, posteriormente, aplicarles a cada una de ellas la Transformada Rápida

de Fourier (FFT). Dichas propiedades fisicoquímicas serán: alpha structure group, betha structure group, energetic group, hydropathy group, hydrophobicity group, index group, secondary structure properties group y volume group. Dichas propiedades fueron obtenidas por medio de una investigación sobre la base de datos AAindex Kawashima and Kanehisa (2020), la cual almacena información de las propiedades fisicoquímicas de los aminoácidos. Dichas descripciones fueron agrupadas, y luego de que se les aplicará un análisis de componentes principales (PCA), se determinó que las propiedades que más aportaban a la estructura de una proteína, eran las 8 ya mencionadas.

- **Embedding:** Los embeddings son un mapeo de una variable discreta a un vector de números continuos. Los embeddings son útiles porque pueden reducir la dimensionalidad de variables discretas y representar categorías de manera significativa en el espacio transformado Koehrsen (2018). Para generar dichos embeddings, se hará uso de la herramienta Tasks Assessing Protein Embeddings (TAPE), la cual es una biblioteca de código abierto implementada en Python y que posee un conjunto de cinco tareas de aprendizaje semi-supervisadas biológicamente relevantes distribuidas en diferentes dominios de la biología de las proteínas Rao et al. (2019b). Dicha biblioteca permite codificar secuencias proteicas en embeddings utilizando métodos de Natural Language Processing.

3. A partir de las secuencias codificadas se armó el grafo mediante la librería PyTorch y dicho grafo actuó de input para la red neuronal.

3.2.2. Enfoque: Grafos

El enfoque de grafos se centra en las estructuras de los complejos proteicos y consistió en armar un grafo por cada PPI, es decir, un grafo por cada interacción antígeno-anticuerpo. Para realizar esto, los nodos se representan por los residuos de las proteínas, mientras que las aristas por las distancias euclidianas que separan a los residuos. Además, cada grafo es etiquetado de forma independiente con base en el nivel de interacción que haya entre las proteínas que conforman el grafo, y al igual que en el caso anterior, los nodos son codificados. En la figura 3.2.2 se ve un bosquejo de los grafos a generar, donde este caso, se aprecia un grafo para 3

complejos antígeno-anticuerpo.

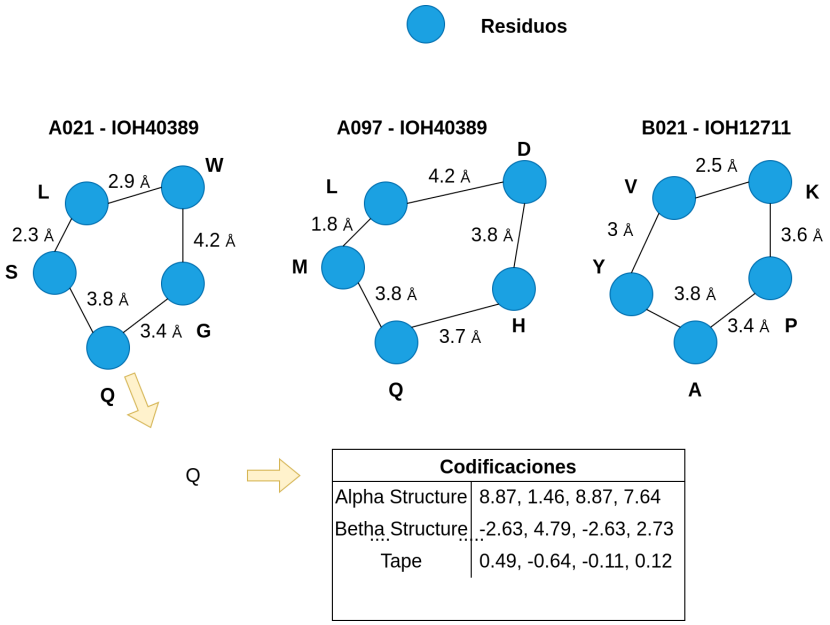

Figura 3.2.2: Metodología. El enfoque de los grafos consiste en armar un grafo por cada interacción antígeno-anticuerpo, en donde los nodos corresponde a los residuos de las proteínas, mientras que las aristas a las distancias euclidianas que los separan.

Para lograr este objetivo, los pasos a seguir son los siguientes:

1. Dado que no se cuenta con las estructuras de los complejos PPI, se busco un método para poder predecir su estructura a partir de las secuencias de antígeno-anticuerpo. Para esto se hizo uso de Rosetta, el cual es un proyecto de computación distribuida para la predicción de la estructura de proteínas dirigido por el laboratorio Baker de la Universidad de Washington Sircar et al. (2010).

2. Las estructuras predichas fueron codificadas por los mismos métodos explicados en el punto 2 anterior.

3. A las estructuras codificadas se le calcularon las distancias euclidianas entre todos los residuos mediante las posiciones tridimensionales que poseen los residuos para cada uno de sus átomos que se encuentran en sus archivos PDB predichos. A partir de esto, se calcularon las distancias de las siguientes formas:

    - Calculó de distancias entre los carbonos *alfa* de cada residuo.

- Calculó de distancias entre los centroides de cada residuo.

Para poder obtener las coordenadas de los residuos se hizo uso de BioPython, mientras que para calcular las distancias se usa la librería de Numpy. Además, cabe recalcar que solo se consideraron las distancias más relevantes mediante un análisis de distribución.

4. Ya con las estructuras codificadas y las distancias calculadas, se generaron los grafos por cada PPI mediante Pytorch.

3.2.3. Comparación entre enfoques

Nodos	Grafos
Clasificador a nivel de nodos	Clasificador a nivel de grafos
Requiere de secuencias aminoacídicas	Requiere de estructuras (PDB)
1 grafo	Muchos grafos (1 por cada complejo)
Los nodos son proteínas	Los nodos son residuos
Las aristas son el nivel de interacción	Las aristas son distancias euclidianas
El grafo no es etiquetado	Los grafos son etiquetados
Rápidos de entrenar	Requieren de un mayor tiempo
Requieren menos recursos computacionales	Requieren más recursos computacionales

Cuadro 3.2.1: Comparación entre ambos enfoques descritos para la generación de grafos.

3.3. Entrenamiento de modelos

Independiente del enfoque, el resultado fueron grafos que actuaron de input para la red neuronal. Para esto, el entrenamiento de los modelos de predicción se hizo por medio de PyTorch Paszke et al. (2019), la cual es una librería escrita en Python especializa en el trabajo con redes neuronales. El objetivo de este proceso fue poder clasificar los complejos de interacción transformados y procesados a grafos en las etapas anteriores.

Ya entrando en el entrenamiento de los modelos como tal, los grafos se dividieron en un conjunto de entrenamiento y otro de testing, siguiendo una analogía de 80/20 respectivamente. Para cada tipo de conjunto se generaron lotes o batchs, los cuales corresponden a agrupaciones de grafos antes de ser ingresados a una red neuronal. El objetivo es garantizar la utilización completa de la GPU, para así

impedir una sobrecarga computacional o de memoria y lograr una paralelización de varios ejemplos. PyTorch posee una función integrada para lograr esto denominada *batch_ size*, en donde el funcionamiento interno consiste en apilar las matrices de adyacencias en forma diagonal (se crea un grafo gigante que contiene múltiples subgrafos), mientras que los nodos y los targets se concatenan en la dimensión del nodo. En la figura 3.3.1 se ve reflejado este proceso. Además de esto, se puede indicar por medio de la función *shuffle*, si se desea o no que el set de datos sea "barajado".

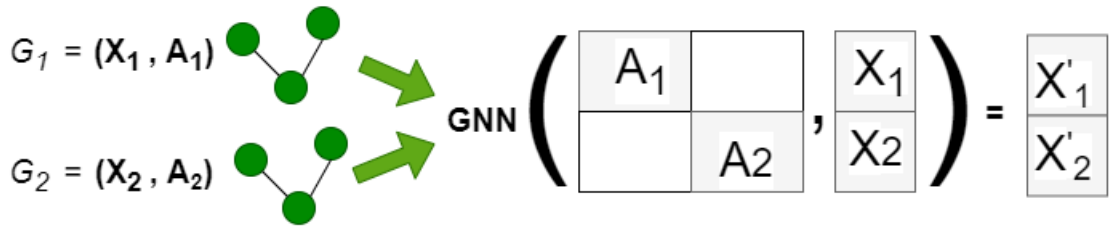

Figura 3.3.1: Generación de batchs. PyTorch apila las matrices de adyacencias en forma diagonal (se crea un grafo gigante que contiene múltiples subgrafos), mientras que los nodos y los targets se concatenan en la dimensión del nodo.

Ya con los batchs establecidos, empieza el entrenamiento de la red neuronal. El detalle del proceso se menciona a continuación:

1. Los grafos ingresan a la capa input de la red neuronal.
2. Se establecen los pesos iniciales y se ingresa a las capas ocultas (capas de convolución).
3. En la primera capa de convolución se forma un embedding, el cual es un vector unidimensional que resume todas las características de los nodos. Este embedding se puede formar partir de procesos como transferencias de mensajes o por estructura, para luego pasar por una función de activación.
4. El proceso descrito en el paso anterior se repite para cada capa de convolución.
5. Los embeddings obtenidos se resumen en un único valor en la capa Readout. Existen diferentes métodos para resumir un embedding, pero el más común es determinar el promedio de los embeddings.
6. Se emplea un clasificador lineal, esto para que la red prediga como tal.

7. Se calcula la función de pérdida o loss, la cual permite establecer el error de predicción de una red neuronal.

8. A partir de la función de pérdida se calculan los gradientes, los cuales corresponden a las direcciones y magnitudes calculadas durante el entrenamiento de la red neuronal que se utilizan para actualizar los pesos en la red en la dirección y cantidad correcta.

9. El proceso mencionado hasta el momento se denomina **Forward Propagation**.

10. Una vez calculados los gradientes se actualizan los pesos y se vuelve a realizar el proceso de la red neuronal. Este proceso se denomina **Backpropagation**.

11. El proceso termina hasta que la función de pérdida es minimizada y adecuada para el modelo.

El proceso descrito se puede ver reflejado en la figura 3.3.2. Una vez que el modelo es entrenado, se emplea para clasificar el conjunto de testing generando anteriormente, donde se calcularán métricas de desempeño para evaluar el poder de predicción de la red neuronal. Para dar validez estadística, este proceso se repite N veces, lo que se conoce como épocas o epochs.

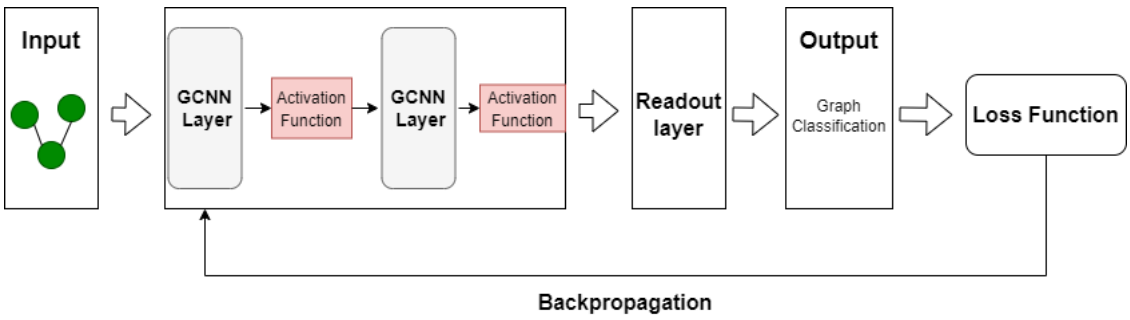

Figura 3.3.2: Arquitectura para una red neuronal. Los grafos en su forma matricial ingresan a la capa input de la red neuronal. Luego, en las capas ocultas se generan embedding, los cuales resumen las características de la información traspasada entre nodos o las estructuras, para posteriormente pasar a una función de activación. Una vez realizada la predicción, se calcula la función de pérdida para estimar el error de predicción. Finalmente, se calculan los gradientes para actualizar los pesos de la red en un proceso denominado Backpropagation. Este proceso se repite hasta que la función de pérdida es reducida.

3.4. Ajuste de hiperparámetros

Cabe recordar que la tesis fue un proceso de investigación, en donde se realizarán constantes pruebas para diferentes funcionalidades dentro de una red neuronal, como lo son DeBlois-Beaucage (2021):

- **Número de capas:** La generación de embeddings se puede repetir tantas veces como se desee, basándonos en el número de capas de convolución. El número óptimo se encuentra entre 3 y 5, ya que un mayor número de capas tiende a producir embeddings inespecíficos.
- **Función de pérdida (loss):** La función de pérdida posee un hiperparámetro fijo Δ, el cual posee un óptimo en un valor de 0,266. Un delta demasiado bajo conduce a un aprendizaje deficiente, ya que el modelo no otorga puntuaciones bajas a los pares negativos, mientras que un delta demasiado positivo conduce a que el modelo no aprendió en absoluto, ya que el problema es muy difícil de procesar para el modelo.
- **Dimensión de los embeddings:** Al generar embeddings, se tiene que definir el tamaño del espacio latente oculto y el espacio de salida. Para el espacio latente el valor óptimo es de alrededor 256 dimensiones, mientras que para el espacio de salida es de alrededor de 128 dimensiones. Un valor bajo para la cantidad de dimensiones conduce a embeddings de mala calidad, mientras que un valor muy alto ralentizan el entrenamiento del modelo sin mejorar su rendimiento.
- **Otras consideraciones:** Se deben probar diferentes funciones de para la capa de Readout, como el máximo y mínimo, así como diferentes funciones de activación (Lineal, Umbral, Sigmoide, Tangente hiperbólica, ReLu y Softmax).

Capítulo 4

Resultados y Discusiones

4.1. Conjunto de datos

El conjunto de datos inicial está conformado por, 228.629 complejos antígeno-anticuerpo, que a su vez está compuesto por 81 y 8221 secuencias amino acídicas únicas de anticuerpos y antígenos, respectivamente, en donde los anticuerpos presentan su secuencia para la cadena pesada y ligera. Tras un análisis del set de datos se identificaron 44 anticuerpos, de los cuales 7 no presentaban su secuencia, ya sea para la cadena pesada o ligera. Con base en esto, el conjunto de datos depurado está conformado por 74 y 8221 secuencias de anticuerpos y antígenos, respectivamente, en donde se identifican 37 anticuerpos con sus respectivas cadenas pesada y ligera.

Con base en lo anterior y tras un proceso de depuración y eliminación de duplicados, el set de datos final pasó de, 228629 a 183113 complejos antígeno-anticuerpo. Dicho dataset presenta el antígeno y el anticuerpo del complejo, las secuencia para la cadena pesada y ligera del anticuerpo, además de la secuencia del antígeno, el valor de intensidad para cada complejo y la categoría a la cual pertenecen, la cual puede ser de tres tipos, Fuerte (0), Mediana (1) y Débil (2). Con respecto a esto último, los complejos están divididos en, 44295, 45778 y 93040 para las categorías 2, 1 y 0, respectivamente.

El set de datos descrito fue la base para el desarrollo de las actividades de la investigación.

4.2. Predictor para complejos Antígeno-Anticuerpo

Como ya se explicó anteriormente, se posee las secuencias amino acídicas para los anticuerpos (cadena pesada y ligera) y los antígenos. Sin embargo, lo que se requiere es el complejo estructural antígeno-anticuerpo para así poder generar grafos de distancias a partir de los complejos estructurales que servirán de input para el entrenamiento de las redes neuronales de grafos (GNN). Para poder predecir dichas estructuras, se evaluaron y emplearon diferentes metodologías, donde cada una de ellas se explica a continuación:

4.2.1. Rosetta

Rosetta es un proyecto de computación distribuida para la predicción de la estructura de proteínas dirigido por el laboratorio Baker de la Universidad de Washington. Rosetta tiene como objetivo predecir el acoplamiento proteína-proteína y diseñar nuevas proteínas con la ayuda de, aproximadamente, 55000 computadoras Sircar et al. (2010). A partir de esta tecnología, se elaboró un pipeline integrado con Rosetta para predecir los complejos antígeno-anticuerpo. Para esto se hizo uso de las secuencias de antígenos y anticuerpos, además de las regiones CDR y epítopos provenientes del set de datos Yasna (2021). Para más detalles de la arquitectura y funcionamiento del pipeline, ver la Figura 4.2.1.

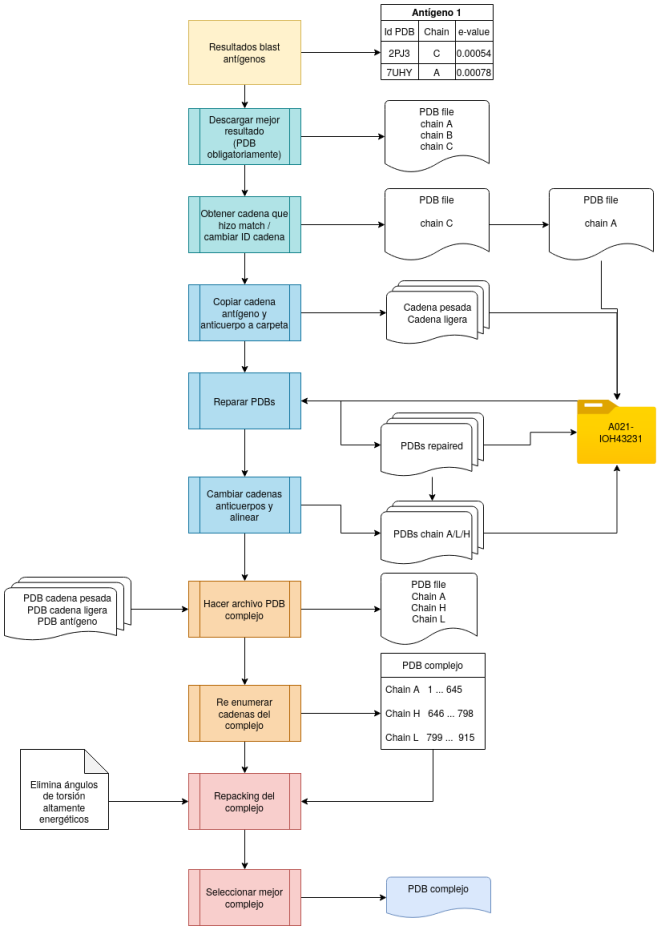

Figura 4.2.1: Pipeline de Rosetta para complejos Antígeno-Anticuerpo.

Uno de los problemas que presentó el pipeline fue la alta demanda computacional que conllevaba ejecutar scripts basados en Rosetta, donde en un computador con bajos o medianos recursos, los tiempos de cálculo y predicción llegaban hasta 40 horas por complejo. Dado que no era rentable y eficiente obtener un modelo por día, se hizo uso de un servidor perteneciente a NLHPC (Laboratorio Nacional de Computación de Alto Rendimiento), empresa que posee la red científica y el supercomputador más poderoso de Chile NLPHC (2021), actualmente. Con base en esto, una de las labores desarrolladas durante este proceso de ejecución del pipeline fue administrar dicho servidor y preparar un entorno para dejar en ejecución el script. Dicha tarea se cumplió, y por 1 mes se administró de forma personal el pipeline de Rosetta, dando como resultado 427 complejos antígeno-anticuerpo.

Si bien el funcionamiento de Rosetta fue "Exitoso", los resultados no fueron los esperados, ya que tras un mes de ejecución se esperaba tener un mayor de número de complejos predichos. Además de esto, se sumaron otras problemáticas a la ya

mencionada costo computacional. Estas fueron:

- NLHPC es un servidor compartido, esto significa que muchos usuarios hacían uso de los recursos del servidor. Esto imposibilitaba definir una partición y núcleos fijos para un script, ya que el consumo de recursos del servidor era muy dinámico.
- El pipeline no estaba automatizado, es decir, había que correr el script complejo a complejo.

A largo plazo, estas dos problemáticas se volvieron muy tediosas y frustrantes, ya que consumían mucho tiempo personal. Es por esto que se vio la necesidad de buscar una alternativa mucho más automatizada que permita predecir los complejos de interacción antígeno-anticuerpo.

La solución fue la creación de un pipeline alternativo que tomaba como base el ya existente, solo que automatizando los procesos y manejando de mejor forma los posibles errores a la hora de predecir. De esta forma se logró desarrollar un script que podía predecir la cantidad de complejos que uno quisiera con la sola ejecución de un comando. Con respecto al servidor, dado que NLHPC era compartido, se optó por la contratación de 3 sistemas VPS, sacrificando un poco de rendimiento pero ganando en la ejecución de procesos de forma ininterrumpida.

A pesar de este nuevo pipeline, surgió una nueva problemática, la cual fue: *"¿Cómo controlar los complejos que va a correr cada servidor, evitando que se repitan?"*. La solución ante esto fue el desarrollo de una base de datos que registrara la información de los complejos que se iban ejecutando. Dicha base de datos se desarrolló en MongoDB y se subió a su servicio en la nube MongoDB Atlas, así cada script se conecta a la nube y no hay necesidad de instalar la base de datos localmente en cada servidor. El funcionamiento general del nuevo pipeline se puede apreciar en la figura 4.2.2.

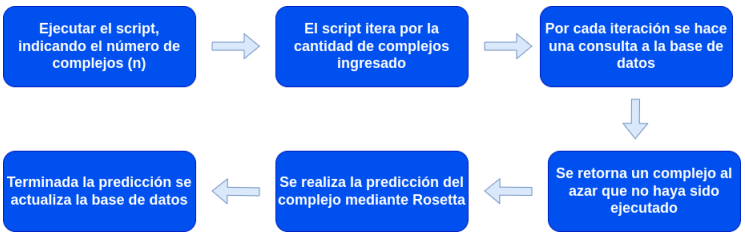

Figura 4.2.2: Pipeline actualizado para la predicción antígeno-anticuerpo.

Los scripts quedaron en ejecución y en constante monitorio durante todo el semestre, obteniendo un total de 8630 modelos estructurales, separados en 1923 para interacción fuerte, 2274 de interacción mediana y 4433 de interacción débil.

4.2.2. AlphaFold

AlphaFold es un programa de inteligencia artificial (IA) desarrollado por DeepMind de Alphabets/Google que realiza predicciones de la estructura de proteínas mediante el sistema de aprendizaje profundo AlphaFold (2022). Posee una base de datos pública con más de 992.316 predicciones de estructuras de proteínas para el proteoma humano y otras proteínas clave de interés, para acelerar la investigación científica.

A partir de AlphaFold se buscó una alternativa a Rosetta para predecir los complejos de interacción antígeno-anticuerpo. El primer intento fue automatizar mediante web scraping, utilizando la librería Playwright Playwright (2022), un colab que utilizaba por debajo AlphaFold. Este método no tuvo los resultados esperados y se descartó esta alternativa, ya que no se pudo automatizar los procesos y los tiempos de predicción de un complejo eran muy largos (2 horas por complejo aproximadamente). Por otro lado, se contará con el apoyo de la Universidad de Magallanes (UMAG) para predecir complejos empleando AlphaFold para futuros proyectos, ya que ellos cuentan con la maquinaria para poder dejar corriendo los modelos de manera local.

4.3. Codificaciones

Las codificaciones consistieron en transformar las secuencias de aminoácidos en vectores numéricos, empleando las estrategias nombradas en la metodología para los diferentes casos de aplicación.

Para el caso del enfoque de los nodos, las secuencias fueron codificadas por propiedades fisicoquímicas y TAPE. Para los primeros, se generaron 8 codificaciones distintas, donde los antígenos tuvieron una representación vectorial con un largo de 241, mientras que los antígenos con un largo de 1021. Por otro lado, la codificación por TAPE resulto en una representación vectorial de 768 tanto para antígenos y anticuerpos. Cabe mencionar que la codificación por One Hot no se tomó en cuenta, ya que la representación vectorial resultante fue de un largo superior

a los 30.000, por lo que el poder de cómputo no cubría las necesidades de esa codificación, así como también, carece de una interpretación biológica.

Para el enfoque por grafos, los residuos se codificaron por One Hot y por propiedades fisicoquímicas. Para el primer caso, los vectores resultantes tuvieron un largo de 20, mientras que los vectores de propiedades fisicoquímicas tuvieron una representación vectorial de 8. Cabe mencionar que la codificación por TAPE no se realizó debido a que internamente trabaja con Natural Language Processing y dado que el input era un residuo, la información para codificar por este método era muy escasa.

4.4. Cálculo de distancias

A las estructuras PDB predichas se le extrajeron las coordenadas (x,y,z) para los carbonos alfa y los centroides de cada residuo de las proteínas. A partir de dichas coordenadas se calcularon las distancias euclidianas entre residuos haciendo uso de Numpy (*np.linalg.norm()*) y se filtraron distancias menores que 12 debido a la tendencia a formar interacciones electrostáticas débiles en dicha instancia. No obstante, no se toma en consideración los tipos de residuos, así como sus propiedades a la hora de evaluar el posible tipo de interacción en sí, debido a la falta de información que se tiene del sistema y a que solo corresponde a un método exploratorio.

4.5. Generación de grafos

Para ambos enfoques, los grafos fueron generados por medio de la librería PyTorch. Para crear los grafos se tuvieron que indicar las siguientes características:

- **edge_index:** Se hace referencia a los nodos, es decir, los antígenos y anticuerpos para el enfoque en nodos y los residuos para el enfoque en grafos.
- **x:** Corresponden a las características de los nodos, es decir, las codificaciones de las secuencias de los aminoácidos y los residuos.
- **y:** Corresponde al target, es decir, el nivel de interacción entre proteínas.

- **edge_attr:** Se le asignan características a las aristas, en este caso, las distancias calculadas entre residuos.

4.6. Resultados del entrenamiento

El entrenamiento de redes neuronales de grafos se realizó mediante la librería PyTorch, los cuales utilizan los grafos generados en etapas anteriores como input.

Con respecto a la construcción de modelos, y con base en referencia bibliográfica, se hicieron modelos de entre 1-10 capas ocultas, con 64 nodos por cada capa oculta, una learning curve de 0.01, se aplicó una función de activación ReLU y las redes se iteraron en 100 epochs.

Ya entrando más en detalle, para cada enfoque se hicieron las siguientes arquitecturas:

4.6.1. Enfoque en nodos

Para el primer enfoque se diseñaron dos arquitecturas. La primera consistió en una arquitectura base conformada por capas de convolución y funciones de activación ReLU. Por otro lado se tiene la segunda arquitectura de la Figura 4.6.1, la cual es un poco más sofisticada que la primera, ya que a las ya conocidas capas de convolución y función de activación ReLU, se agrega una capa de Dropout (evaluar sobreajuste) y una capa de Softmax (generar predictor final). Cabe recalcar que cada arquitectura se construyó con 1-10 capas ocultas.

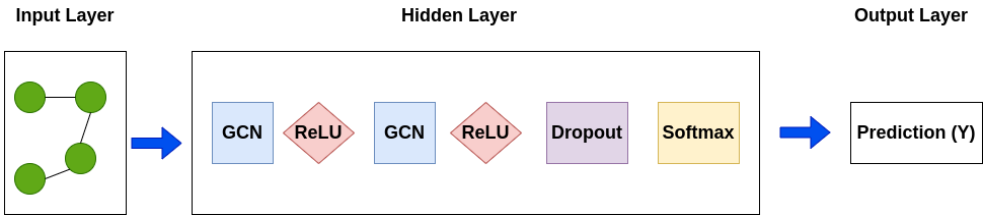

Figura 4.6.1: Arquitectura Enfoque en Nodos. Arquitectura aplicada para el entrenamiento de GNN con un enfoque en nodos, la cual se encuentra conformada por capas de convolución, funciones de activación ReLU, capas de Dropout para evaluar sobreajuste y una capa de Softmax final para generar el clasificador.

En el global, no se notaron diferencias marcadas entre ambas arquitecturas, en donde se obtuvo un accuracy entre 0,49 y 0,52 (0,50 como media) y una precisión entre 0,25 y 0,35 (0,32 como media). Por otro lado, las codificaciones que mejor presentaron resultados fueron las desarrolladas por propiedades fisicoquímicas, más específicamente la Alpha Structure y Betha Structure, mientras que las peores resultados provienen de la codificación por TAPE y la propiedad fisicoquímica Index.

Por el lado de los hiperparámetros, el hecho de agregar más capas ocultas no representó una mejora en los modelos, donde posterior a las 5 capas la red se estancó en su rendimiento. Al mismo tiempo, una cantidad de 100 epochs resulta representativa a la hora de determinar si una métrica alcanza un estándar mantenible. En la Tabla 4.6.1 se puede apreciar un resumen del mejor resultado obtenido para una codifación por Alpha Structure, mientras que en las figuras 4.6.2 se puede apreciar el rendimiento de la red neuronal por cada epoch para el accuracy.

Alpha Structure	**Loss**	**Accuracy**	**Precision**
Min	1.033	0.316	0.263
Mean	1.040	0.505	0.291
Max	1.104	0.516	0.613
STD	0.014	0.029	0.067

Cuadro 4.6.1: Mejor resultado para el enfoque en nodos proveniente de la codificación fisicoquímica Alpha Structure.

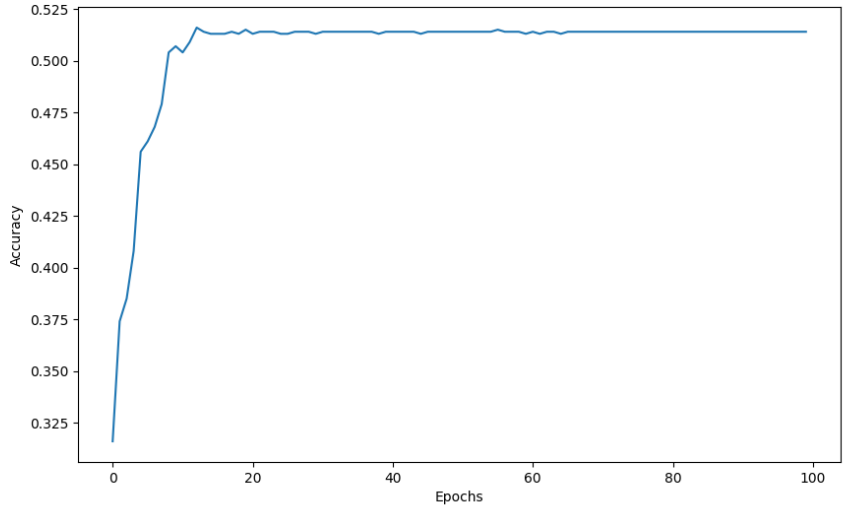

Figura 4.6.2: Mejor rendimiento para el accuracy a lo largo de las 100 epochs, tomando en consideración una arquitectura basada en capas de convolución, funciones ReLU, capas de Dropout, una capa Softmax y una codificación fisicoquímica Alpha Structure.

Con respecto a los resultados, un mal accuracy denota una mala distribución de clases del set de datos, mientras que una mala precisión marca una alta tasa de falsos positivos a la hora de hacer predicciones. Esto puede indicar que el conjunto de datos se encuentra desbalanceado, o, por otro lado, que el clasificador empleado es erróneo, donde una solución alternativa sería pasar de un target terciario (Fuerte, Mediano y Débil) a uno binario (Interactúa y No Interactúa). Con esto último, se busca evitar una sobre estandarización por parte del clasificador, evitando posible ruido, y, por lo tanto, un sobre aprendizaje por parte de las redes neuronales.

Por otro lado, las formas de representación de aminoácidos como vectores numéricos presentaron una mala performance. Es por esto, que se deben buscar más alternativas de codificación, como las mencionadas en el segmento 1.2. Del mismo modo, se deben explorar más arquitecturas de Graph Neural Network basadas en clasificadores de nodos, así como librerías alternativas a PyTorch (TensorFlow o Keras), pero la limitante documentación en esta área es un gran problema para la investigación al ser un campo poco explorado a la fecha.

Si bien se deben explorar los puntos ya mencionados, el enfoque por nodos no resultó ser una buena forma de trabajo a la hora de trabajar con grafos y proteínas. Esto se debe principalmente a que este tipo de enfoque condensa mucho

la información y como se genera un único grafo, la red neuronal no puede realizar una buena etapa de aprendizaje o discriminar entre clases. Además, este tipo de grafos solo ofrece información con respecto a la interacción, más no entrega información estructural de las proteínas, por lo que la información que se cuenta para entrenar en estos casos resulta no ser la idónea.

4.6.2. Enfoque en grafos

Para el enfoque en grafos se aplicaron dos tipos de arquitecturas, donde ambas comparten en común el uso de funciones de activación ReLU, una capa de *global mean pool* para obtener un valor promedio de los embeddings, una capa de Dropout para evaluar sobre ajuste y una función lineal final para obtener el clasificador, sin embargo, lo que las diferencia es el tipo de capa oculta utilizada, donde para la primera arquitectura se utilizó GNN, mientras que para la segunda GCN. Además, y al igual que para el primer enfoque, se hicieron redes neruonales de hasta 1-10 capas ocultas. En la figura 4.6.3 se puede apreciar un esquema de la arquitectura implementada.

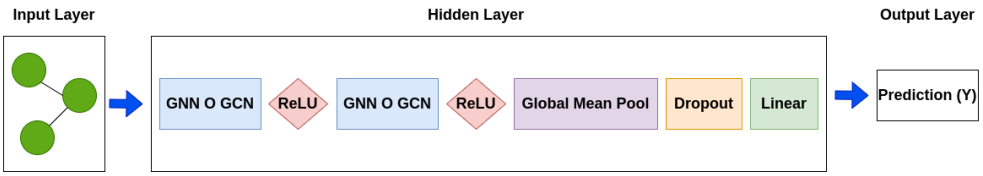

Figura 4.6.3: Arquitectura Enfoque en Grafos. Arquitectura para el enfoque en grafos basada en capas de convolución, ya sea GNN o GCN, funciones de activación ReLU, una capa global mean pool para obtener un valor promedio de los embeddings generados, una capa de Dropout y una capa Linear para generar la predicción.

Tomando en consideración ambas arquitecturas, no se notaron diferencias notorias entre ellas, donde el rendimiento del accuracy tuvo una media cercana al 0,51, mientras que el precisión alcanzo valores muy cercanos al 0,40. No se pudo notar una diferencia marcada entre codificaciones, pero si se evidenció una leve mejora por parte de los grafos calculados a partir de carbonos alfa, alcanzando un 0,10 de rendimiento por sobre los centroides. Por otro lado, las Graph Convolutional Neural Network presentaron una pequeña mejora que las Graph Neural Network, pero no lo suficiente para concluir que una arquitectura es mejor que otra. En la Tabla 4.6.2 se puede ver un resumen del mejor rendimiento obtenido por parte de una GCN haciendo uso de carbonos alfa y una codificación fisicoquímica, mientras que en la Figura 4.6.4 se puede apreciar el rendimiento de la red neuronal por cada epoch.

Item	Accuracy	Precision
Min	0.2020	0.2488
Mean	0.5194	0.3969
Max	0.5344	0.5182
STD	0.0560	0.0635

Cuadro 4.6.2: Mejor resultado para el enfoque en nodos proveniente de la codificación fisicoquímica Alpha Structure.

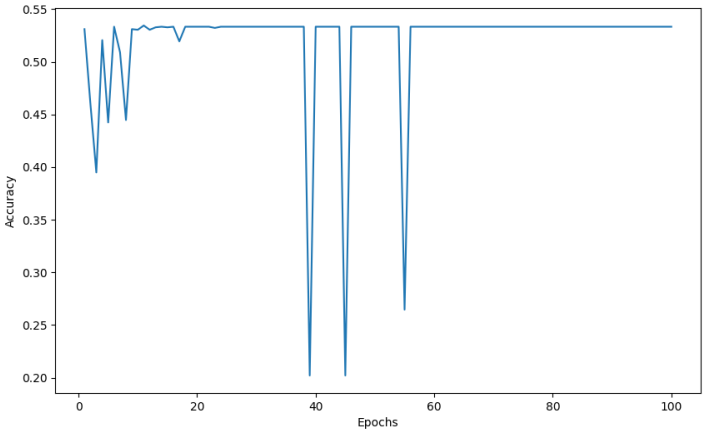

Figura 4.6.4: Rendimiento del accuracy por parte del enfoque de los grafos a lo largo de cada epoch.

Al igual que en el enfoque en nodos, el entrenamiento de la red resultó con un rendimiento del 0,51 aproximadamente. Esto reafirma la hipótesis de un posible desbalance del conjunto de datos, o por otro lado, reafirmar la necesidad de replantear el clasificador de los complejos antígeno-anticuerpo a un sistema binario.

Como ya se había mencionado, se deben explorar nuevos métodos de codificación como BLOSUM, así como buscar nuevas formas de representación de las aristas. Con respecto a esto último, no se notaron diferencias entre carbonos alfa y centroides, en donde los primeros presentaron un leve rendimiento superior (0.1) que los segundos. Un método alternativo a explorar para este punto es considerar distancias diferentes a las euclidianas, como es el caso de las distancias de manhattan o coseno. Por otro lado, se deben explorar nuevas formas de representación de aristas, como por ejemplo, las interacciones electroestáticas entre aminoácidos.

Otra posible explicación del mal rendimiento de la red neuronal son las predicciones realizadas por Rosetta. Para poder indagar en este punto, se hicieron algunas predicciones de complejos antígeno-anticuerpo a partir de AlphaFold. Con esto, los modelos de Rosetta y AlphaFold fueron comparados por el software Prosa, el cual es un servicio web para el reconocimiento de errores en estructuras tridimensionales de proteínas M and MJ. (2007). En la figura 4.6.5 y 4.6.6 se puede apreciar los gráficos de energía para los modelos de AlphaFold y Rosetta respectivamente,

donde este último presentó una gran cantidad de zonas positivas, es decir, zonas con problemas energéticos o erróneas. Por otro lado, los niveles de energía para AlphaFold fueron más negativas, y por la tanto, más estabilizadas. Resultados como esto da a pensar que los modelos usados para generar grafos y entrenar la red neuronal no fueron los idóneos y sería una buena alternativa realizar el entrenamiento a partir de complejos predichos por AlphaFold.

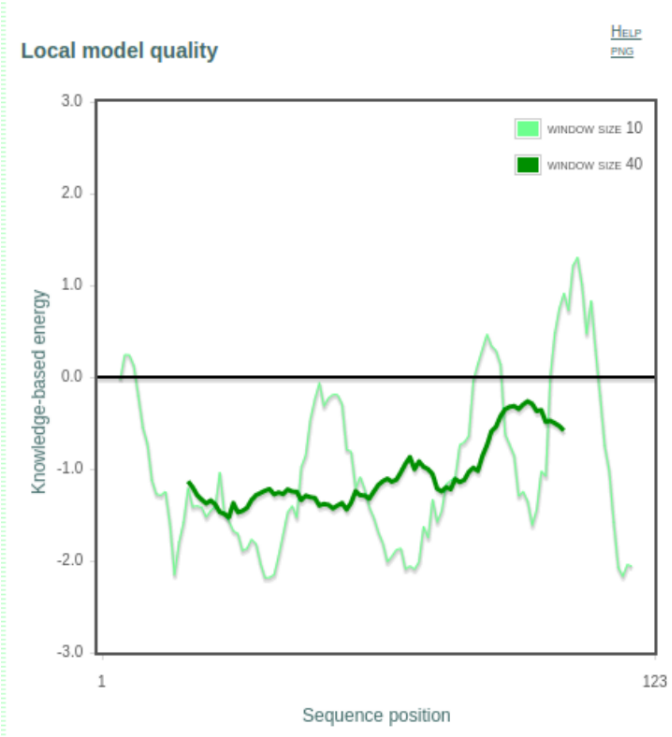

Figura 4.6.5: Gráfico resultante de Prosa para un complejo predicho por AlphaFold.

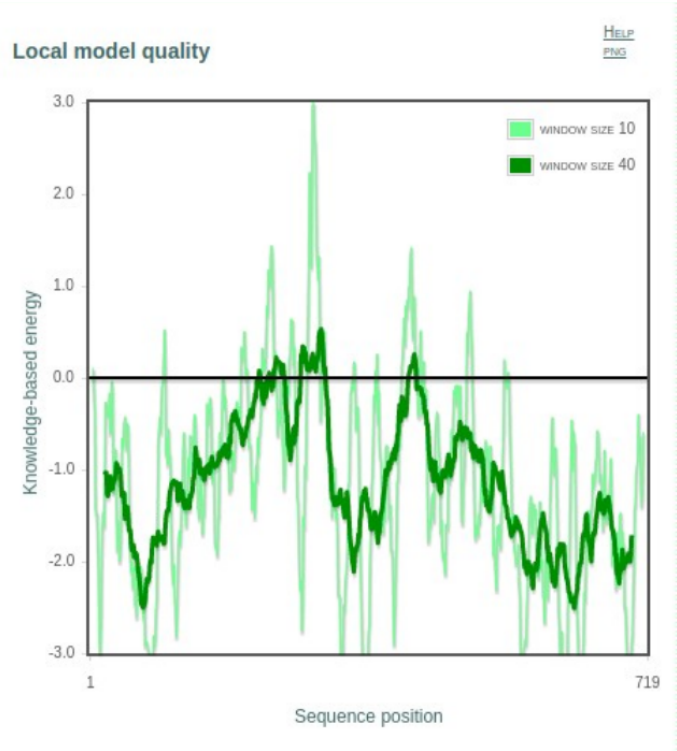

Figura 4.6.6: Gráfico resultante de Prosa para un complejo predicho por Rosetta.

Si bien los resultados no fueron los esperados, el enfoque en grafos sigue llamando la atención debido a que permite utilizar de mejor forma la información de las proteínas, así como generar una mayor data para el entrenamiento de la red, a diferencia del enfoque en nodos. Se deben tratar los puntos descritos anteriormente, dando énfasis en la predicción de estructuras por parte de AlphaFold.

Capítulo 5

Conclusiones y futuros trabajos

Con base en los trabajos realizados, el uso de grafos no significó la generación de buenos modelos de predicción para complejos PPI de interacción antígeno-anticuerpo haciendo uso de Graph Neural Network y Graph Convolutional Neural Network. A partir de todos los entrenamientos realizados, en general se obtuvieron rendimientos muy por debajo de los esperados alcanzando medias de hasta 0,51 para valores como el accuracy, es decir, modelos muy poco confiables a la hora de generar predicciones. A partir de esto, surgen muchas posibles hipótesis que explican el mal rendimiento de los modelos y que son foco de análisis y exploración para futuros trabajos.

El primer punto a tratar son los tipos de codificaciones empleadas, es decir, las formas de representación de números a letras. Para la investigación se emplearon 3 tipos de codificaciones: One Hot, Propiedades Fisicoquímicas y TAPE. La primera de ellas tuvo el problema de generar vectores de representación muy grandes, llegando a generar matrices de hasta 30.000 columnas. Esto genero un claro consumo computacional con el cual no se pudo lidiar. Por otro lado, TAPE se vio limitado ante la poca información que entrega un residuo, por lo que las codificaciones generadas eran muy precarias. Por último, las propiedades fisicoquímicas no presentaron problemas, ni de codificación como de rendimiento, pero no se obtuvieron buenos resultados a la hora de entrenar modelos. Para trabajos a futuro, se recomienda seguir haciendo pruebas con este tipo de codificación, sumando de forma alternativa codificaciones como BLOSUM y ProtVec.

Otro punto a tocar es la representación de las aristas de los grafos. Estas se generaron a partir de las distancias euclidianas entre residuos, haciendo uso de las coordenadas (x,y,z) de los carbonos alfa y centroides para cada aminoácido. Con respecto a esto último, no se diferencian entre ambos, por lo que se recomienda seguir haciendo pruebas con ellos. Por otro lado, es posible explorar más métodos para el cálculo de distancias, como puede ser la distancia de Manhattan, coseno, entre otras. Además, se deben explorar formas alternativas para la representación de aristas, como por ejemplo: aristas a partir de interacciones electroestáticas entre residuos, las cuales pueden ser obtenidas por software como WHATIF WHATIF (2021).

A los puntos anteriores se suma la exploración de más arquitecturas basadas en GNN y GCNN, así como librerías alternativas a PyTorch como lo son TensorFlow y Keras. Con respecto a esto, y más específicamente para las arquitecturas, el hecho de agregar más capas ocultas no significo un mejor rendimiento de la red neuronal. Se recomienda realizar entrenamiento con 4-5 capas ocultas. Por otro lado, un entrenamiento de 100 epochs resultó significativo a la hora de diagnosticar el rendimiento de la red, por lo que se recomienda seguir optando por realizar 100 iteraciones por entrenamiento.

Hasta el momento, todas las hipótesis planteadas son considerando una mal etapa de entrenamiento de la red neuronal, pero existe la posibilidad de que la falla este a partir del conjunto de datos. Como se explicó, las estructuras fueran predichas con Rosetta, donde un método alternativo a esto es el uso de la inteligencia artificial AlphaFold. Con el apoyo de la UMAG se realizaron algunas predicciones de complejos antígeno-anticuerpo y se compararon con respecto a los modelos generados con Rosetta. Dichas comparaciones se realizaron mediante software como Prosa (Reconocimiento de errores en estructuras tridimensionales) y ProqServer (clasifica según el nivel de predicción de un modelo). Con respecto a Prosa, los complejos de AlphaFold presentaron mejores mínimos de energía, en donde los complejos PPI predichos por la inteligencia artificial alcazaron valores de -5.53, mientras que los modelos de Rosetta tuvieron valores cercanos a -11. Sin embargo, la diferencia más notoria fue por parte de ProqServer, en donde los modelos de AlphaFold alcazaron valores cercanos a 8.9, mientras que Rosetta obtuvo apenas un puntaje de -0.83. Esto lleva a pensar a que los complejos predichos por Rosetta son de mala calidad, y, por lo tanto, los grafos generados para el entrenamiento

de modelos no fue el idóneo y habré la posibilidad de que las redes neuronales no tuvieron la culpa de los malos resultados generados. Es por esto que se plantea hacer el entrenamiento otra vez, pero ahora con estructuras PDB predichas por AlphaFold. En la figura 5.0.1 se puede apreciar el puntaje obtenido para un modelo de AlphaFold, mientras que él figura 5.0.2 para un modelo de Rosetta.

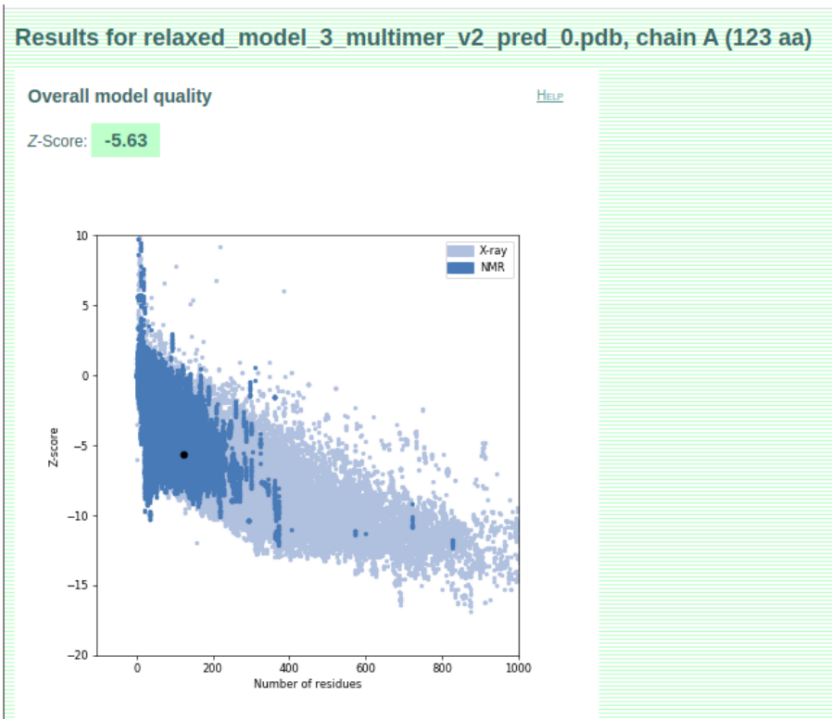

Figura 5.0.1: Resultado para un modelo de AlphaFold. Resultado de Prosa para un complejo de AlphaFold, alcanzando un puntaje de -5.63.

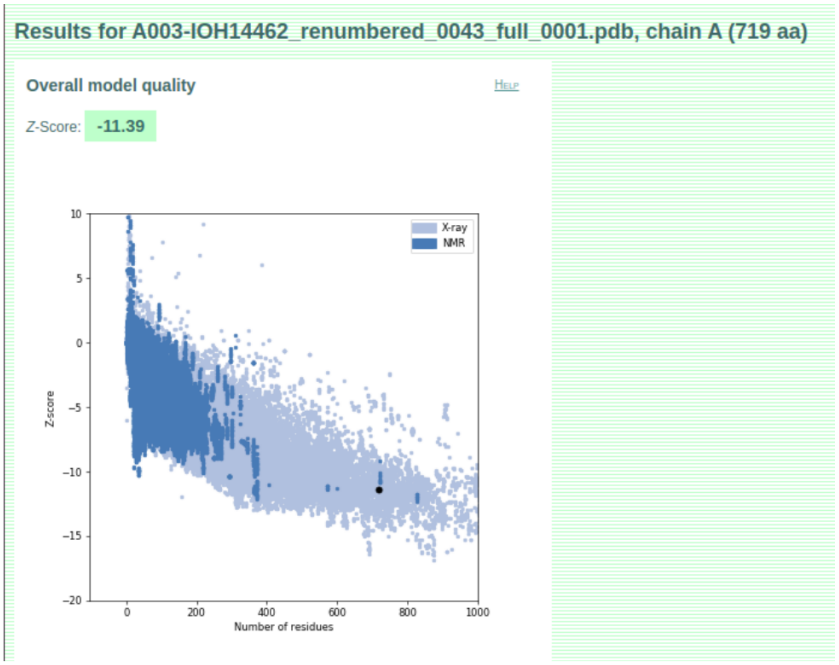

Figura 5.0.2: Resultado para un modelo de Rosetta. Resultado de Prosa para un complejo de Rosetta, alcanzando un puntaje de -11.39.

Con respecto a los enfoques, se recomienda indagar en el planteamiento para grafos, ya que es más susceptible a probar nuevas variables, formas de representación y metodologías, a diferencia del enfoque en nodos, donde exceptuando las formas de codificación, no existe mucho más que explorar. Por otro lado, este último enfoque no resultó ser una buena forma de trabajo para esta investigación en concreto, ya que se condensa mucho la información y no se aprovecha la información estructural que se puede extraer de las proteínas.

En conclusión, si bien no se estuvieron los resultados esperados, el conocimiento adquirido en Graph Neural Network y los resultados preliminares sobre complejos predichos a partir de AlphaFold, mantienen y permiten creer que los grafos pueden ser una buena estrategia de representación de proteínas para la predicción de interacción proteína-proteína, campo que se debe seguir explorando y que es de mucha importancia para la bioinformática y la generación de fármacos.

Bibliografía

Algorithmia (2020). https://algorithmia.com/blog/semi-supervised-learning. https://algorithmia.com/blog/semi-supervised-learning.

AlphaFold (2022). Alphafold protein structure database. https://alphafold.ebi.ac.uk/.

Amisha, Malik, P. (2019). Overview of artificial intelligence in medicine. https://doi.org/10.4103/jfmpc.jfmpc_440_19.

Barker, M. (2019). Autoinmunidad inducida por fármacos. https://www.msdmanuals.com/es-cl/professional/trastornos-endocrinol.

Barrios, J. (2020). La matriz de confusión y sus métricas. https://www.juanbarrios.com/la-matriz-de-confusion-y-sus-metricas/.

Barrios, J. (2021). La matriz de confusión y sus métricas. https://www.juanbarrios.com/la-matriz-de-confusion-y-sus-metricas/.

BBVA (2019). 'machine learning': ¿qué es y cómo funciona? https://www.bbva.com/es/machine-learning-que-es-y-como-funciona/.

Brinda, K. and Vishveshwara, S. (2005). A network representation of protein structures: Implications for protein stability. *Biophysical Journal*, 89(6):4159–4170.

Britannica, T. E. o. E. (2021a). antibody. https://www.britannica.com/science/antibody.

Britannica, T. E. o. E. (2021b). Antígeno. https://www.britannica.com/science/antigen.

Brownlee, J. (2019). A gentle introduction to the rectified linear unit (relu). https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/.

Chiu M, Goulet D, T. A. (2019). Antibody structure and function: The basis for engineering therapeutics. https://doi.org/10.3390/antib8040055.

Chuan-En Lin, D. (2020). Simple techniques to prevent overfitting. https://towardsdatascience.com/8-simple-techniques-to-prevent-overfitting-4d443da2ef7d.

Clinid, M. (2021). Leucemia. https://www.mayoclinic.org/es-es/diseases-conditions/leukemia/diagnosis-treatment/drc-20374378.

Colonia, A. (2020). El sistema inmunológico innato y adaptativo. https://www.ncbi.nlm.nih.gov/books/NBK279396/.

contributors, W. (2021). Antigen-antibody interaction.

datos.gob.es (2020). ¿cómo aprenden las máquinas? machine learning y sus diferentes tipos. https://datos.gob.es/es/blog/como-aprenden-las-maquinas-machine-learning-y-sus-diferentes-tipos.

DD, C. (2010). Descripción general de la respuesta inmune. https://doi.org/10.1016/j.jaci.2009.12.980.

DeBlois-Beaucage, J. (2021). Building a recommender system using graph neural networks. https://medium.com/decathlontechnology/building-a-recommender-system-using-graph-neural-networks-2ee5fc4e706d.

Dobaño, C. (2021). Estos son los mecanismos de las vacunas para activar el sistema inmunitario. https://www.vacunacovid.gob.es/voces-expertas/estos-son-los-mecanismos-de-las-vacunas-para-activar-el-sistema-inmunitario.

Dr. Mandal, A. (2021). ¿cuál es autoinmunidad? https://www.news-medical.net/health/What-is-Autoimmunity-(Spanish).aspx.

Education, I. C. (2020). https://www.ibm.com/cloud/learn/unsupervised-learning. https://www.ibm.com/cloud/learn/unsupervised-learning.

Fernández, A. (2021). Tecnologías de inteligencia artificial y sus categorías. https://www.auraquantic.com/es/tecnologias-de-inteligencia-artificial-y-sus-categorias/.

Fontaine, N. T. and Cadet, X. F.and Vetrivel, I. (2019). Novel descriptors and digital signal processing- based method for protein sequence activity relationship study. https://doi.org/10.3390/ijms20225640.

Gaudelet, T., Day, B., Jamasb, A. R., Soman, J., Regep, C., Gertrude Liu, J. B. R. H., Vickers, R., Roberts, C., Tang, J., Roblin, D., Blundell, T. L., Bronstein, M. M., and Taylor-King, J. P. (2021). Utilizing graph machine learning within drug discovery and development. https://doi.org/10.1093/bib/bbab159.

geprgeho (2019). Autoregressive models in deep learning — a brief survey. https://www.georgeho.org/deep-autoregressive-models/.

Gonzalo, A. (2020). ¿qué es el sobreajuste u overfitting y por qué debemos evitarlo? https://machinelearningparatodos.com/que-es-el-sobreajuste-u-overfitting-y-por-que-debemos-evitarlo/.

Group, E. A. (2019). Redes neuronales artificiales y deep learning, explicado para dummies. https://blog.enzymeadvisinggroup.com/redes-neuronales-artificiales-y-deep-learning.

H., E., Y., B., and A., H. (2020). Amino acid encoding for deep learning applications. *BMC Bioinformatics*.

HERALDO (2018). Las bacterias intestinales causan enfermedades autoinmunes. https://www.heraldo.es/noticias/sociedad/2018/03/08/las-bacterias-intestinales-causan-enfermedades-autoinmunes-1229020-310.html.

INSTITUYE, N. C. (2021). lymphocyte. https://www.cancer.gov/publications/dictionaries/cancer-terms/def/lymphocyte.

InteractiveChaos (2021). One hot encoding. https://interactivechaos.com/es/manual/tutorial-de-machine-learning/one-hot-encoding.

ITELLIGENT (2018). Deep learning & convolutional neuronal network: qué es y en qué consiste. https://itelligent.es/es/deep-learning-convolutional-neuronal-network-cnn-consiste/.

J., G., J., B., E., P., N, M., SV., P., B., M., and M, B. (2020). A review of deep learning methods for antibodies.

Janda, A. (2016). Ig constant region effects on variable region structure and function. https://doi.org/10.3389/fmicb.2016.00022.

Jorge, T.-A., David, M.-O., Diego, A.-S., Águila Guerrero. Julio, Álvaro, O.-N., and Marcelo, N. (2019). Pattern recognition on antigen-antibody interactions from protein microarrays based on data mining and bioinformatics analysis. In *2019 38th International Conference of the Chilean Computer Science Society (SCCC)*, pages 1–8.

K, J., S, S., and H, S. (2019). Prediction of protein–protein interaction using graph neural networks.

Karagiannakos, S. (2021). Best graph neural network architectures: Gcn, gat, mpnn and more. https://theaisummer.com/gnn-architectures/.

Kawashima, S. and Kanehisa, M. (2020). Aaindex: amino acid index database. https://doi.org/10.1093/nar/28.1.374.

Koehrsen, W. (2018). Neural network embeddings explained. https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526.

L., M., T., L., and G., F. (2013). Dna methylation and its basic function. neuropsychopharmacol.

Lisowski, E. (2020). Deep learning architectur. https://addepto.com/deep-learning-architecture/.

Luisyep (2019a). El problema de los puentes de königsberg. https://ingenieriabasica.es/el-problema-de-los-puentes-de-konigsberg/.

Luisyep (2019b). El problema de los puentes de königsberg. https://ingenieriabasica.es/el-problema-de-los-puentes-de-konigsberg/.

Luna, P. (2010). Una enzima presente en lágrimas podría convertirse en los nuevos antibióticos. https://www.bbc.com/mundo/noticias/2010/10/101004_enzimas_lagrimas_antibioticos_pl.

Luštrek, M. (2013). Epitope predictions indicate the presence of two distinct types of epitope-antibody-reactivities determined by epitope profiling of intravenous immunoglobulins. https://doi.org/10.1371/journal.pone.0078605.

M, W. and MJ., S. (2007). Prosa-web: interactive web service for the recognition of errors in three-dimensional structures of proteins.

Marshall JS, Warrington R, W. W. (2018). Introducción a la inmunología y la inmunopatología. https://doi.org/10.1186/s13223-018-0278-1.

MAYOCLINIC (2021). Leucemia. https://www.mayoclinic.org/es-es/diseases-conditions/leukemia/symptoms-causes/syc-20374373.

Medina-Ortiz, D., Contreras, S., Amado-Hinojosa, J., Torres-Almonacid, J., and A, J. (2020). Combination of digital signal processing and assembled predictive models facilitates the rational design of proteins. https://arxiv.org/abs/2010.03516.

Nichols, J. (2019). Machine learning: applications of artificial intelligence to imaging and diagnosis. https://doi.org/10.1007/s12551-018-0449-9.

NLPHC (2021). Faqs — nlhpc,. https://www.nlhpc.cl/.

Osiński, B. (2018). What is reinforcement learning? the complete guide. https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/.

Pacientes, R. (2021). ¿cómo se clasifican y cuantos tipos existen de leucemias? https://rochepacientes.es/cancer/leucemia-linfatica-cronica/tipos.html.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Pham, C. (2020). Graph convolutional networks (gcn). https://www.topbots.com/graph-convolutional-networks/.

PhD. Maldonado, A. (2021). Redes neuronales de grafos, ¿qué son? https://www.ciiia.mx/noticiasciiia/redes-neuronales-de-grafos-qu-son.

Pingel, J. (2021). Deep learning questions asked and answered. https://explore.mathworks.com/all-about-deep-learning-network-architectures.

Playwright (2022). Playwright enables reliable end-to-end testing for modern web apps. https://playwright.dev/.

Pollard, A.J; Bijker, E. (2021). A guide to vaccinology: from basic principles to new developments. https://doi.org/10.1038/s41577-020-00479-7.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., Abbeel, P., and Song, Y. S. (2019a). Evaluating protein transfer learning with tape. https://arxiv.org/abs/1906.08230.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., Abbeel, P., and Song, Y. S. (2019b). Evaluating protein transfer learning with tape. *Advances in Neural Information Processing Systems*.

Rochina, P. (2017). El análisis de redes sociales mediante la teoría de grafos. https://revistadigital.inesem.es/informatica-y-tics/teoria-grafos/.

Roldán, P. (2020). Modelo matemático. https://economipedia.com/definiciones/modelo-matematico.html.

Roman Kogay, C. S. (2019). Encyclopedia of bioinformatics and computational biology.

S, O. (2019). Science and culture: Can the principles of topology help improve the world's slums? https://doi.org/10.1073/pnas.1904847116.

Sanitaria (2021). Leucemias. redaccionmedica.com/recursos-salud/diccionario-enfermedades/leucemias.

Shmueli, B. (2019). Matthews correlation coefficient is the best classification metric you've never heard of. https://medium.com/analytics-vidhya/mae-mse-rmse-coefficient-of-determination-adjusted-r-squared-which-metric-is-better-cd0326a569

Sircar, A., Chaudhury, S., Kilambi, K., Berrondo, M., and Gray, J. (2010). A generalized approach to sampling backbone conformations with rosettadock for capri rounds 13-19.

Smatti, M. (2019). Viruses and autoimmunity: A review on the potential interaction and molecular mechanisms. https://doi.org/10.3390/v11080762.

ST, S., MM, H., A, A., and S, S. (2021). Convolutional neural networks with image representation of amino acid sequences for protein function prediction. 10.1016/j.compbiolchem.2021.107494.

Strokach, A., Becerra, D., Corbi-Verge, C., Perez-Riba, A., and Kim, P. M. (2020). Fast and flexible protein design using deep graph neural networks. *Cell Systems*, 11(4):402–411.e4.

TOPDOCTORS (2021). ¿qué son las enfermedades autoinmunes? https://www.topdoctors.es/diccionario-medico/enfermedades-autoinmunes.

V, G., P, R., and T, K. (2021). Structure-based protein function prediction using graph convolutional networks.

VS., R., K., S., GN., S., and GN., K. (2014). Protein-protein interaction detection: methods and analysis.

WHATIF (2021). What if homepage. https://swift.cmbi.umcn.nl/whatif/.

Whitacre, J., Lin, J., and Harding, A. (2012). T cell adaptive immunity proceeds through environment-induced adaptation from the exposure of cryptic genetic variation. https://doi.org/10.3389/fgene.2012.00005.

Wittmann, B. J., Johnston, K. E., Wu, Z., and Arnold, F. H. (2021). Advances in machine learning for directed evolution. https://arxiv.org/abs/1906.08230.

Yasna, B. (2021). Análisis de autorreactividad de anticuerpos leucémicos soportado por estrategias de inteligencia artificial.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020). Graph neural networks: A review of methods and applications. https://www.sciencedirect.com/science/article/pii/S2666651021000012.