



Universidad
Nacional
de San Martín

Evaluación de modelos de aprendizaje profundo para el reposicionamiento de compuestos bioactivos

Autor: Sebastián Jinich

Dirección: Dr. Fernán Agüero

Lugar: Laboratorio de Genómica y Bioinformática de Tripanosomátidos, Instituto de Investigaciones Biotecnológicas, UNSAM

Tesina de grado para optar por el título de Licenciado en Biotecnología

Índice

Índice.....	2
0. Abreviaturas y glosario.....	5
Abreviaturas.....	5
Glosario.....	6
1. Introducción.....	9
1.1. Machine Learning.....	9
1.1.1. Aplicación del Machine Learning.....	9
Etapas centradas en los datos.....	10
Partición de datos.....	10
Etapas centradas en el modelo.....	11
Controles y validación.....	12
Sobreajuste y subajuste.....	12
Detección y manejo del sobreajuste.....	13
Ajuste fino del modelo.....	13
<i>Performance y predicción</i>	14
1.1.2. Modelos de machine learning: Redes Neuronales.....	14
Entrenamiento de redes neuronales.....	16
Monitoreo de entrenamiento.....	17
1.1.3. Redes neuronales convolucionales.....	18
Convoluciones.....	18
Arquitectura general de las CNN de imágenes.....	20
1.2. Descubrimiento de drogas.....	21
1.2.1. Proceso completo de descubrimiento de drogas.....	21
1.2.2. Descubrimiento de hits.....	22
Diseño de drogas asistido por computadora basado en ligandos.....	22
Modelos de relaciones estructura-actividad.....	23
Obtención y curado de datos.....	23
Partición de conjuntos de datos químicos.....	24
Representación molecular.....	25
Validación de modelos.....	26
Similitud química como herramienta de LB-CADD.....	28
1.2.3. Descubrimiento de drogas para enfermedades tropicales desatendidas.....	30
Enfermedades tropicales desatendidas.....	30
Enfermedad de Chagas.....	30
<i>Trypanosoma cruzi e implicancias clínicas</i>	31

Tratamientos contra la enfermedad de Chagas.....	31
Estrategias de reposicionamiento.....	32
Planteo del trabajo.....	32
2. Objetivos.....	33
3. Resultados.....	34
3.1. Esquema general.....	34
3.2. Obtención de datos para el entrenamiento de modelos clasificadores de bioactividad de compuestos químicos frente a proteínas aisladas.....	35
3.2.1. Filtrado inicial de la base de datos ChEMBL.....	35
3.2.2. Selección de targets para entrenar modelos de machine learning.....	36
3.2.3. Detección de sesgos de similitud química.....	38
3.2.4. Partición de datos con Scaffold Splitter para maximizar la independencia entre set de datos de entrenamiento y testeo.....	40
Discusión de la sección 3.2.....	41
3.3. Desarrollo de modelos clasificadores para screening de compuestos químicos in silico....	42
3.3.1. Implementación de un modelo basado en similitud química como baseline para comparación con redes neuronales.....	42
3.3.2. Desarrollo de TrypanoDEEPscreen, una estrategia de redes neuronales convolucionales basadas en estructuras en 2D de compuestos químicos para la predicción de bioactividad.....	43
Adaptación y reimplementación de DEEPscreen como base para el desarrollo de TrypanoDEEPscreen.....	43
Exploración de enfoques alternativos en la implementación para mejorar el rendimiento de las redes neuronales.....	44
Evaluación de métricas y redefinición del enfoque de clasificación para mejorar la priorización de compuestos activos.....	44
Estrategia de ensamblado de modelos para prevenir el sobreajuste y aumentar la robustez del modelo.....	45
3.3.3. Comparación del desempeño de los modelos desarrollados.....	46
Los modelos de redes neuronales convolucionales tienen una performance similar al baseline de similitud química en términos de métricas de AUROC y AUROC 15%.....	46
Los modelos de redes neuronales no superaron al baseline en términos de capacidad de priorización de compuestos activos en screening in silico.....	47
Los modelos de redes neuronales no superan al baseline en como clasificadores.....	48
Análisis de sobreajuste de DEEPscreen y TrypanoDEEPscreen.....	48
<i>TrypanoDEEPscreen es 3 órdenes de magnitud más costoso computacionalmente que el baseline.....</i>	50
El modelo TrypanoDEEPScreen y el baseline tienen criterios diferentes para definir el score de bioactividad para cada compuesto.....	51

Discusión de la sección 3.3.....	53
3.4. Uso de los algoritmos con datasets de descubrimiento de antibióticos.....	55
3.4.1. El set de entrenamiento es altamente desbalanceado y muestra un sesgo de similitud química en los compuestos activos.....	55
3.4.2. En una campaña de screening in silico con experimentación TrypanoDEEPscreen presenta resultados similares a los modelos Chemprop y rankea mejor a la halicina.....	56
Discusión de la sección 3.4.....	58
3.5. Aplicación de los modelos para el descubrimiento de tratamientos contra Trypanosoma cruzi.....	60
Evaluación de los modelos con compuestos tripanocidas.....	60
Discusión de la sección 3.5.....	62
4. Conclusiones.....	62
5. Perspectivas futuras.....	63
6. Métodos.....	65
6.1. Cómputo y principales librerías utilizadas.....	65
6.2. Construcción de datasets.....	65
6.2.1. Filtrado y curado de datos de ChEMBL.....	65
6.2.2. Partición de datasets.....	66
6.3. Baseline basado en similitud química.....	67
6.3.1. Desarrollo y selección del modelo.....	67
6.3.2. Selección final del modelo.....	67
6.4. Modelos TrypanoDEEPscreen.....	68
6.4.1. Arquitectura.....	68
6.4.2. Entrenamiento y optimización por hiperparámetros.....	69
6.4.3. Ensamblado.....	69
7. Referencias.....	70
8. Reconocimientos.....	77
Agradecimientos.....	78

0. Abreviaturas y glosario

Abreviaturas

ANN	Redes Neuronales Artificiales (Artificial Neural Networks)
ASP	<i>All-Shortest Path</i>
AUROC	Área Bajo la Curva ROC (Area Under the ROC Curve)
CADD	Diseño de Drogas Asistido por Computadora (Computer-Aided Drug Design)
CNN	Redes Neuronales Convolucionales (Convolutional Neural Networks)
ETD	Enfermedades Tropicales Desatendidas
FNN	Redes Neuronales de Propagación hacia Adelante (<i>Feed-forward Neural Network</i>)
HTS	<i>Screening</i> de Alto Rendimiento (<i>High-Throughput Screening</i>)
<i>LB-CADD</i>	Diseño de Drogas Asistido por Computadora Basado en Ligandos
MCC	Coeficiente de Correlación de Matthews (<i>Matthews Correlation Coefficient</i>)
ML	Aprendizaje Automático (<i>Machine Learning</i>)
MLP	Perceptrón Multicapa (<i>Multilayer Perceptron</i>)
ReLU	Unidad Lineal Rectificada (<i>Rectified Linear Unit</i>)
SAR	Relación Estructura-Actividad (<i>Structure-Activity Relationships</i>)

Glosario

Variable objetivo

La variable objetivo es la salida o respuesta que un modelo de aprendizaje automático supervisado pretende predecir; constituye el valor de referencia contra el cual se ajustan los parámetros durante el entrenamiento.

Features

Las características (*features*) son las variables de entrada que codifican información relevante sobre cada observación y a partir de las cuales un modelo infiere patrones para predecir la variable objetivo.

Parámetros

Variables internas que el modelo ajusta durante el aprendizaje de los datos de entrenamiento.

Hiperparámetros

Variables externas al modelo (configuraciones, preferencias) que controlan el proceso de aprendizaje.

Épocas (*epochs*)

Una iteración completa sobre todo el conjunto de datos de entrenamiento durante el proceso de ajuste del modelo.

Target

Entidad biológica (como una biomolécula o una célula) cuya actividad se busca modular mediante un compuesto químico. En el contexto del *drug discovery*, los modelos de aprendizaje automático intentan predecir la interacción entre compuestos y estos *targets*.

Logit

Logit en *Machine Learning* se refiere al vector de predicciones sin normalizar generado por un modelo de clasificación. Es decir, el valor crudo returnedo por una red neuronal clasificadora [1].

Farmacóforo

Farmacóforo es una descripción abstracta de las características moleculares que son necesarias para el reconocimiento molecular de un ligando por una macromolécula biológica [2].

Sobreajuste (*Overfitting*) y Subajuste (*Underfitting*)

El sobreajuste ocurre cuando un modelo aprende no solo los patrones generales, sino también el ruido específico del conjunto de entrenamiento, perdiendo capacidad de generalización; el subajuste refleja un modelo demasiado simple para capturar las relaciones subyacentes en los datos.

Función de pérdida (*Loss Function*)

Es una función escalar que cuantifica el error entre las predicciones del modelo y los valores verdaderos, y cuya minimización mediante algoritmos de optimización guía el ajuste de parámetros.

Ensamblado de Modelos (*Ensemble*)

Es una estrategia que combina las predicciones de múltiples modelos base para reducir varianza, sesgo o errores de modelado.

Baseline

Es un modelo de referencia, generalmente simple o heurístico, que establece un umbral mínimo de desempeño contra el cual se comparan modelos más sofisticados.

Embedding

Es una representación vectorial continua en un espacio latente de menor dimensión que preserva relaciones semánticas o estructurales del objeto original, como una molécula, una palabra o un nodo.

Relación Estructura-Actividad (SAR)

La relación estructura-actividad (SAR) es una representación del vínculo entre la estructura química de un compuesto y su actividad biológica. Su propósito es comprender cómo las modificaciones químicas en la estructura de una molécula influyen en su actividad.

Fingerprint Molecular

Es una representación computacional de una molécula basada en un vector (usualmente binario o entero) que codifica la presencia de fragmentos químicos específicos, utilizada para calcular similitud estructural.

Scaffold (Esqueleto Molecular) y Scaffold Splitting

El *scaffold* representa la estructura central o núcleo de una molécula; el *scaffold splitting* es una técnica de partición de datos que separa compuestos con esqueletos distintos en diferentes subconjuntos.

Hit y Lead

Un *hit* es un compuesto identificado como activo en un ensayo primario, mientras que un *lead* es un derivado optimizado en términos de potencia, selectividad y propiedades farmacocinéticas para avanzar en el proceso de descubrimiento de drogas.

IC50 / EC50 / Ki / Kd

IC50 es la concentración de inhibidor que reduce la actividad del sistema al 50%; *EC50* es la concentración de agonista que provoca el 50% del efecto máximo observado; *Ki* es la constante de equilibrio que mide la afinidad intrínseca del inhibidor por la enzima (disociación del complejo enzima-inhibidor); *Kd* es la constante de disociación ligando-receptor, definida como la concentración de ligando libre necesaria para que el 50% de los receptores estén ocupados en equilibrio.

Precipios de Actividad (*Activity Cliffs*)

Son pares de compuestos altamente similares estructuralmente pero con diferencias marcadas en actividad biológica, lo que genera discontinuidades locales en el espacio químico explorado durante el descubrimiento de drogas.

BEDROC

Boltzmann-Enhanced Discrimination of ROC es una métrica que pondera más las predicciones tempranas en un *ranking*, capturando la capacidad del modelo para priorizar correctamente compuestos activos en los primeros puestos [3].

1. Introducción

1.1. Machine Learning

El aprendizaje automático (ML, *machine learning* en inglés) es la disciplina que otorga a las computadoras la capacidad de aprender a partir de información, sin necesidad de ser programadas explícitamente para cada tarea. Esto se realiza a través de la identificación de patrones estadísticos en datos y su posterior generalización para realizar predicciones [4].

El uso de ML en el campo del descubrimiento de drogas es un abordaje que transformó la manera en la cual se desarrollan nuevos compuestos terapéuticos. Históricamente este era un campo dependiente de metodologías basadas en la prueba y error [5]. No obstante, la creciente disponibilización de datos e incremento en el poder de cómputo, abrió la puerta a la aplicación de algoritmos de ML [4, 6].

1.1.1. Aplicación del Machine Learning

El ML puede clasificarse en distintos tipos según la naturaleza del problema y los datos disponibles. En particular, el aprendizaje supervisado consta de algoritmos entrenados con conjuntos de datos donde se conoce la relación entre los datos de entrada y los resultados de salida. Por otro lado, los métodos no supervisados exploran las estructuras subyacentes de los datos al agruparlos o reduciendo su dimensionalidad, para descubrir patrones en datos que no están necesariamente etiquetados [4, 7, 8].

El proceso general de ML supervisado (Figura 1), busca extraer información latente en conjuntos de datos para posteriormente realizar predicciones con datos desconocidos. El proceso comienza definiendo el problema a resolver con algoritmos de aprendizaje automático. Esto consiste en seleccionar la variable objetivo a predecir, e identificar mediante qué información se pretende realizar el aprendizaje. También consiste en definir qué tarea se busca realizar: clasificación (por ejemplo, asignar un objeto a una clase: positivo o negativo), ordenamiento (ranquear objetos con algún criterio), realizar una regresión (predecir el valor para una variable objetivo y en función del valor conocido para otra variable x), entre otros. Además de definir la tarea, es necesario evaluar y definir las estrategias de validación deseables para dicho problema.

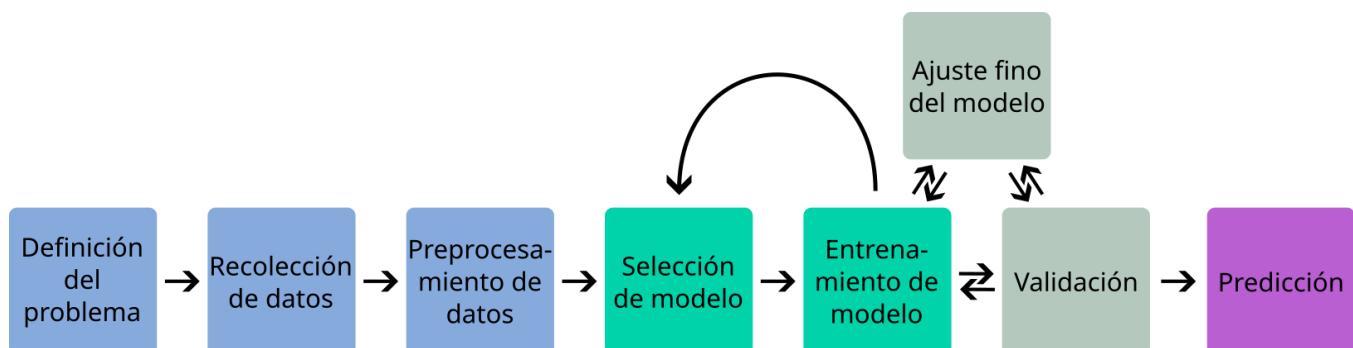


Figura 1. Esquema general de un *pipeline* de ML de aprendizaje supervisado (Basado en de Sousa et al. [4]). Las distintas etapas están separadas por colores: Etapas centradas en los datos (azul), centradas en el modelo (verde), en los controles y la validación (gris) y predicción (violeta)

Etapas centradas en los datos

En las etapas de manejo de datos, se procede a obtener datos a partir de bases de datos disponibles o se generan específicamente para el estudio. La calidad y la cantidad de estos datos son factores clave para asegurar la robustez y relevancia de la información empleada. En general, contar con una mayor cantidad de datos permite alcanzar una mejor capacidad de generalización [4, 7, 9]. Ligado al volumen de datos, es crucial que los datos sean representativos de los patrones que se buscan generalizar (es decir que contengan información), de lo contrario es poco probable realizar predicciones precisas con los modelos entrenados [7, 10].

La instancia de preprocesamiento de datos es una etapa integral en *machine learning*, ya que la calidad de los datos y la utilidad de la información contenida en ellos afectan directamente la capacidad de un modelo para aprender. Esta etapa, también conocida como ingeniería de variables o atributos (del inglés *feature engineering*), consiste en limpiar, estandarizar y analizar los datos para obtener variables o atributos (*features*) útiles para las tareas de aprendizaje [4]. Naturalmente, si los datos de entrenamiento contienen errores, *outliers* y ruido, resulta más difícil identificar patrones subyacentes. Es por eso que, para entrenar modelos eficaces, es fundamental seleccionar, extraer y crear *features* que reflejen los comportamientos que se busca predecir, con el objetivo de obtener modelos capaces de generalizar adecuadamente [7].

Partición de datos

Una vez obtenidas y procesadas las *features*, el siguiente paso consiste en dividir el conjunto de datos en subconjuntos de observaciones que cumplen funciones específicas dentro del proceso de entrenamiento (Figura 2). Habitualmente, los datos se dividen en tres subconjuntos: entrenamiento, validación y testeо. El conjunto de entrenamiento se utiliza para ajustar los parámetros del modelo; el conjunto de validación permite evaluar, monitorear y optimizar la fase de entrenamiento; y el conjunto de testeо se reserva para obtener una estimación “*no sesgada*” del rendimiento final sobre datos no vistos [5].

La partición puede llevarse a cabo de forma aleatoria o racional. La partición aleatoria se basa en la suposición de que las instancias del conjunto de datos son independientes e idénticamente distribuidas (i.i.d.). La aleatorización ayuda a garantizar que los subconjuntos sean representativos del conjunto de datos completo, lo que a su vez permite una estimación no sesgada del error de generalización del modelo.

Cuando no se puede garantizar i.i.d es más útil realizar una partición racional. Esto se debe considerar en conjuntos de datos con correlaciones intrínsecas, y también en conjuntos de datos pequeños o desbalanceados, donde un muestreo aleatorio podría estar generando una sobreestimación del rendimiento del modelo [11].

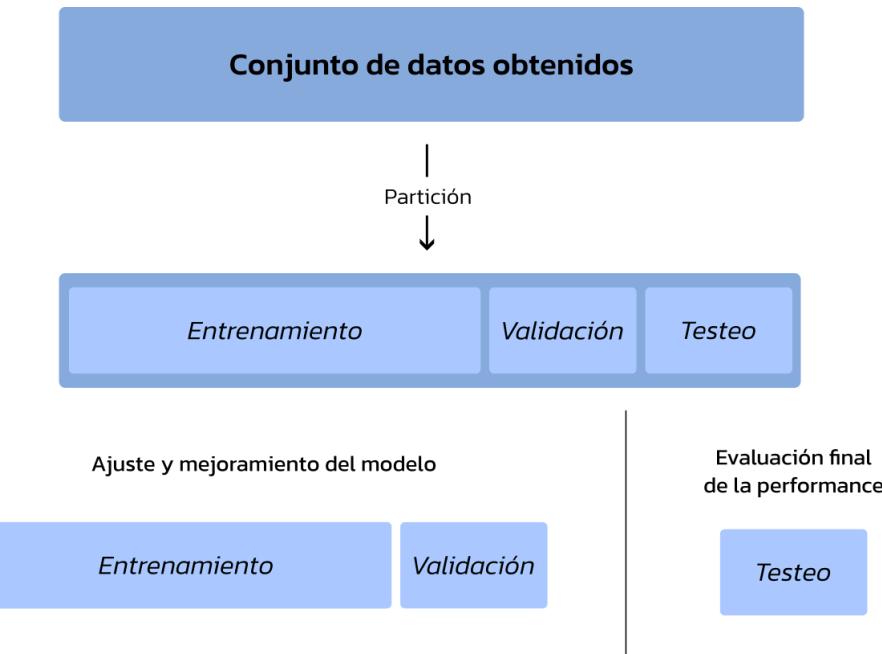


Figura 2. Esquema de partición del conjunto de datos en entrenamiento, validación y testeo. El conjunto de testeo se mantiene separado durante todo el proceso y se utiliza únicamente para la evaluación final del desempeño del modelo.

El particionado racional implica una etapa previa de agrupamiento (*clustering*), en la que cada observación es asignada a un grupo en función de la distribución de las *features*. A partir de esta agrupación, se puede realizar una partición estratificada que conserva la proporción original de cada grupo en todos los subconjuntos generados, asegurando una representación equitativa en cada división [12, 13].

También es posible realizar una partición no equitativa, cuyo objetivo es maximizar las diferencias entre el conjunto de entrenamiento y el de testeo. Esta estrategia permite evaluar la capacidad del modelo para generalizar en condiciones desafiantes, validando que el modelo no haya simplemente memorizado los datos de entrenamiento [14].

Etapas centradas en el modelo

Existen diversas metodologías para el modelado, especializadas en distintos comportamientos de datos (modelos lineales, modelos logísticos, árboles de decisiones, redes neuronales, etc.). Esencialmente, modelar es obtener una representación simplificada de los datos; existen datos con comportamientos lineales, no lineales, con distribuciones espaciales, temporales, categóricas, etc. Cuando se selecciona una metodología para modelar un *dataset* se está realizando una suposición implícita de los comportamientos de los datos. Según el teorema “*No Free lunch*” [15], no hay razón para preferir *a priori* un tipo de modelado por sobre otro sin realizar estas suposiciones. La única manera de conocer cuál es la mejor estrategia es probando todas; como esto no es realizable en la práctica, se evalúa una cantidad razonable de modelos y se selecciona el mejor de estos, en base al desempeño del modelo y las necesidades del problema [7]. En esta

tesina, como prueba de concepto, se desarrollaron y entrenaron modelos con redes neuronales convolucionales, que se introducen en la siguiente [subsección](#).

Durante la fase de entrenamiento, el modelo es expuesto a muchas observaciones de los valores de distintas *features*, y aprende a relacionarlos con la variable objetivo planteada. El propósito del modelo en esta instancia es minimizar la diferencia entre los *outputs* predichos y los *outputs* reales (la variable objetivo), ajustando los parámetros internos en función de la optimización de una métrica que refleja esta diferencia [4]. La optimización puede realizarse usando una ecuación de forma cerrada, donde se obtiene directamente el resultado que mejor ajusta los datos (por ejemplo, una regresión lineal), o bien en forma iterativa optimizando los parámetros del modelo para minimizar en cada iteración una función de coste, como es el caso de las redes neuronales [7].

[Controles y validación](#)

Durante la fase de entrenamiento se aplican distintas métricas para monitorear el progreso de aprendizaje del modelo. Para ello se utiliza un conjunto de datos adicional que no es el utilizado para entrenar el modelo (*set* de validación). A diferencia del conjunto de datos de entrenamiento, que se utiliza para ajustar los parámetros del modelo, el conjunto de datos de validación se utiliza solamente para evaluar el progreso del aprendizaje. Sobre estos datos, el modelo hace predicciones y se miden las discrepancias entre las predicciones y el resultado real (conocido). Esto representa una medida directa de las capacidades de generalización del modelo [4]. En este sentido, surgen el [sobreajuste y el subajuste](#) (*overfitting* y *underfitting* en inglés respectivamente) [16].

[Sobreajuste y subajuste](#)

El sobreajuste y el subajuste son dos fenómenos opuestos que pueden ocurrir a la hora de entrenar modelos de ML [4]. El sobreajuste esencialmente sucede cuando el algoritmo aprende a identificar patrones en los datos de entrenamiento llevando a captar ciertos patrones que pueden deberse al azar (ruido), o aprende exclusivamente los datos de entrenamiento, en lugar de reflejar propiedades comunes a otros datos similares [16, 17]. Es decir que en lugar de aprender una simplificación de los datos que permita generalizarlos, el modelo sobre-aprende o “memoriza” el conjunto de datos de entrenamiento. Entonces, el modelo tiene obviamente buen rendimiento en el conjunto de entrenamiento pero no tiene capacidades para generalizar y extender adecuadamente las predicciones a nuevos datos. Esto usualmente sucede cuando el modelo es demasiado complejo en relación a la cantidad y el ruido del conjunto de entrenamiento, por sobreoptimización, o por datos muy ruidosos [7].

Contrariamente el subajuste ocurre cuando el modelo es demasiado simple para capturar la complejidad de la información. Por eso, falla a la hora de aprender los patrones intrínsecos de los datos y no puede aprender de ellos. Entonces, el modelo performance insatisfactoriamente en cualquier conjunto de datos [4, 7, 18].

Detección y manejo del sobreajuste

La manera de detectar el sobreajuste es evaluando la *performance* del modelo con datos que no fueron utilizados para el entrenamiento. Estos datos pueden provenir de una o más particiones del conjunto de datos inicial (como la partición de validación mencionada previamente), o bien utilizarse métodos de validación cruzada con re-muestreo como el *k-fold cross-validation* [19]. Esta metodología (véase Figura 3) se basa en fraccionar los datos en k particiones, entrenar con $k-1$ particiones y validar con la partición separada. Luego repetir el entrenamiento k veces, dejando en cada instancia separada una de las particiones para la validación y entrenando con el resto de las $k-1$ particiones [17]. Esta forma de validar suele ser más robusta que una validación de una única partición ya que es menos propensa a verse afectada por sesgos propios de un submuestreo.

		Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Cálculo de performance
Entrenamiento	Split 1	Validación	Entrenamiento	Entrenamiento	Entrenamiento	Entrenamiento	→ 0.89
	Split 2	Entrenamiento	Validación	Entrenamiento	Entrenamiento	Entrenamiento	→ 0.78
	Split 3	Entrenamiento	Entrenamiento	Validación	Entrenamiento	Entrenamiento	→ 0.92
	Split 4	Entrenamiento	Entrenamiento	Entrenamiento	Validación	Entrenamiento	→ 0.62
	Split 5	Entrenamiento	Entrenamiento	Entrenamiento	Entrenamiento	Validación	→ 0.83
							Promedio → 0.8

Figura 3. Evaluación de la *performance* por 5-fold cross validation. En cada entrenamiento, se realiza un *split* donde quedan 4 *folds* para entrenar y 1 para validar. Se calcula la *performance* del modelo en cada *split*, con su respectivo *set* de validación y finalmente se promedia el resultado en los 5 splits.

Existen diversas estrategias para lidiar con el sobreajuste, pero en resumidas cuentas se basan en modificar el modelo para optimizar la generalización, conseguir más datos de entrenamiento o reducir el ruido [7]. Modificar el modelo puede suponer elegir otra estrategia de modelado más simplificada, o bien ajustar la forma en la que modela alterando los hiperparámetros (ver Glosario), a esto se lo conoce como el ajuste fino del modelo.

Ajuste fino del modelo

Los *hiperparámetros* son parámetros externos en los modelos de *machine learning* que controlan el proceso y la estrategia del modelado. No son los parámetros que se ajustan durante el entrenamiento, sino que determinan cómo el modelo lleva a cabo el aprendizaje. Los *hiperparámetros* que se pueden ajustar varían considerablemente según el método de modelado, pero suelen controlar aspectos como la arquitectura del modelo, la cantidad de iteraciones de optimización (epochs) y la implementación de distintas estrategias de regularización [20] que restringen el entrenamiento para minimizar el *sobreajuste*. Usualmente, la manera de ajustar estos *hiperparámetros* consiste en probar múltiples configuraciones y evaluar el rendimiento

utilizando un conjunto de datos de validación distinto del conjunto de entrenamiento y del de testeo final. Idealmente, se emplean técnicas de *cross-validation* para minimizar también el *sobreajuste* respecto a los *hiperparámetros*. Esta exploración puede realizarse por fuerza bruta (conocida como *grid search*) o mediante algoritmos de búsqueda con menores requerimientos computacionales [7].

Cabe resaltar que las etapas de validación, ajuste fino y entrenamiento son procesos iterativos donde se busca mejorar el modelado hasta alcanzar las capacidades de generalización que requiera el problema. Como se mencionó previamente, no hay un tipo de modelado que sea el definitivo para todos los problemas, ni siquiera para los mismos tipos de datos. Durante el desarrollo de modelos de ML siempre que la *performance* no sea suficiente, se puede volver a la instancia anterior para abordar el problema de una manera diferente y así lograr llegar a un modelo satisfactorio.

Performance y predicción

Una vez completadas las etapas de entrenamiento, validación y ajuste de hiperparámetros, el modelo se considera apto para generar predicciones. En esta etapa, se ingresan los datos de entrada al modelo, que utiliza los patrones aprendidos durante el entrenamiento para producir una predicción. Esta fase de medición de la *performance* constituye la penúltima instancia del proceso de *machine learning*, en la cual el modelo entrenado es aplicado sobre los datos no vistos previamente (el *set* de testeo) con el objetivo de medir el desempeño del modelo. Se miden las discrepancias entre los valores predichos y los valores reales (conocidos) y se calculan diferentes métricas de desempeño.

Por último, una vez que el modelo superó el criterio de *performance* aceptable para la tarea que se busca resolver con aprendizaje automático, está listo para predecir los datos a los que se busque analizar, “confiando” en los resultados que el mismo otorga.

1.1.2. Modelos de *machine learning*: Redes Neuronales

Las redes neuronales artificiales (ANN, por sus siglas en inglés) son modelos computacionales inspirados en el funcionamiento de las redes neuronales biológicas, que procesan información mediante unidades básicas denominadas neuronas. Cada neurona (Figura 4), conocida también como perceptrón, recibe estímulos en forma de valores numéricos ($x \in \mathbb{R}^n$) como entrada, procesa cada valor multiplicándolo por pesos ajustables ($w \in \mathbb{R}^n$) y suma estos resultados junto con un sesgo ($b \in \mathbb{R}$, *bias*). El resultado ($z \in \mathbb{R}$) de esta suma lineal se transforma posteriormente mediante una función de activación, generando así una salida escalar ($y \in \mathbb{R}$). El ajuste de estos pesos (w) y sesgo (b) permiten a una neurona aprender representaciones útiles para resolver clasificaciones binarias simples y lineales [11, 21].

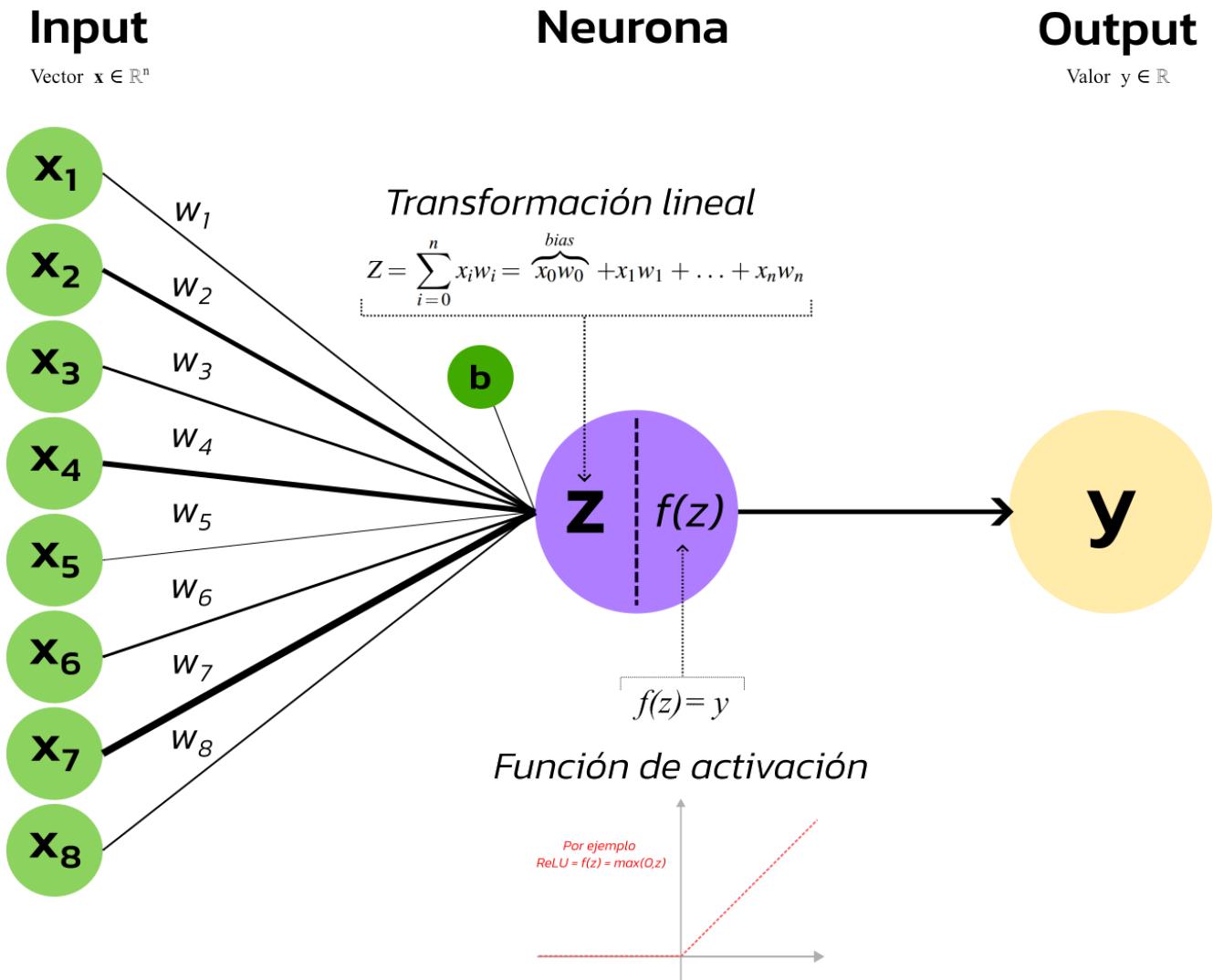


Figura 4. Esquema de funcionamiento de un perceptrón (Basado en de Sousa et al. [4]). La entrada numérica (x) se transforma linealmente mediante pesos ajustables (w) y un sesgo (b), para luego aplicar una función de activación (por ejemplo, una función ReLU [22]) que produce la salida escalar (y).

Un único *perceptrón* permite resolver problemas de clasificación binaria lineal. Para abordar tareas más complejas, se pueden combinar múltiples perceptrones en arquitecturas más complejas. Las *redes neuronales profundas* (*Deep Neural Networks*, Figura 5) se construyen mediante el apilamiento de varias capas de perceptrones totalmente conectados, lo que permite capturar relaciones no lineales en los datos. En particular, las *redes neuronales multicapa* (*Multilayer Perceptron*, MLP) constituyen una extensión directa del perceptrón simple, al incluir una o más capas ocultas. Estas capas transforman sus entradas mediante funciones de activación no lineales, como la sigmoidal o la *Rectified Linear Unit* (ReLU), y permiten al modelo aproximar funciones de creciente complejidad [7].

Las redes neuronales de propagación hacia adelante (*feed-forward neural networks*, FNN) son MLP organizadas secuencialmente: una capa de entrada, capas ocultas y una capa de salida (Figura 5). En estas arquitecturas, la información fluye únicamente en dirección hacia adelante,

sin conexiones cíclicas. Las FNN pueden ser usadas como modelos de aprendizaje supervisado, en los cuales un vector de entrada se transforma progresivamente a través de cada capa, permitiendo que el modelo aprenda representaciones internas adecuadas para predecir etiquetas o resultados específicos [4].

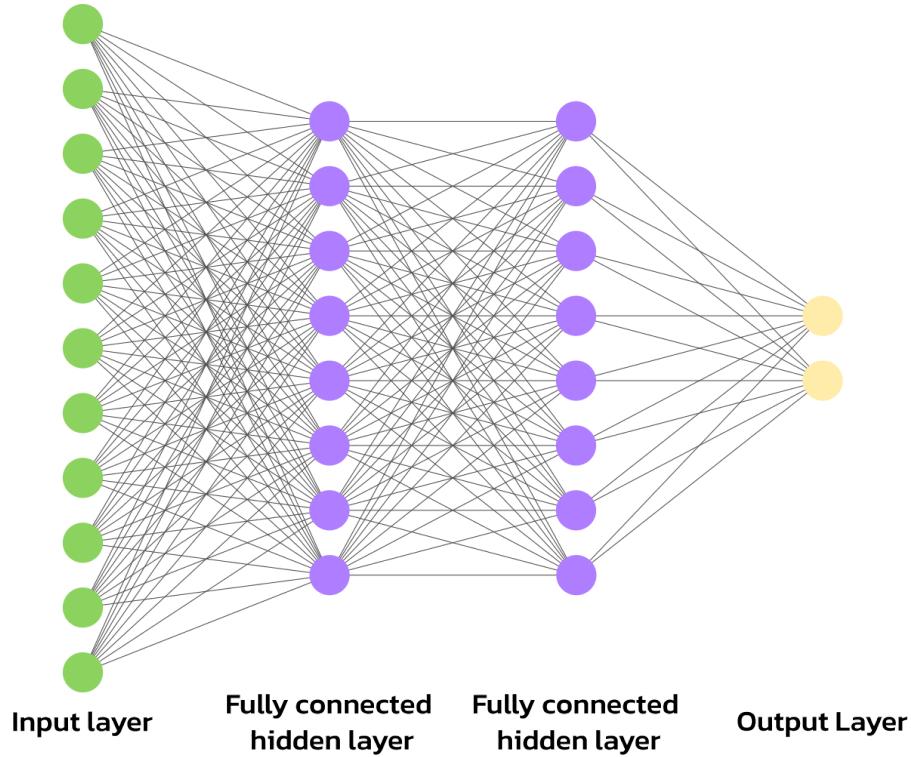


Figura 5. Arquitectura esquemática de una *Feedforward Neural network* para clasificación binaria. En verde la *input layer*, que recibe un vector de 11 dimensiones, en lila dos capas completamente conectadas de 7 neuronas y en amarillo la capa de salida de 2 dimensiones (1 por cada categoría de la clasificación binaria).

Entrenamiento de redes neuronales

El proceso de entrenamiento busca que la red neuronal ajuste sus parámetros internos que son los pesos (w) y los sesgos (b), de manera que la diferencia entre las salidas (y) que predice el modelo y las salidas reales observadas en los datos de entrenamiento sea mínima. Estos pesos y sesgos son los parámetros del modelo que se aprenden durante el entrenamiento [4, 7].

El entrenamiento de una red neuronal constituye un proceso iterativo que consiste en pasar repetidamente el conjunto de datos de entrenamiento a través de la red mientras se ajustan sus parámetros. Cada pasada completa sobre la totalidad del conjunto de datos se denomina una época (*epoch*). Usualmente la duración del entrenamiento (el número de *épocas*) puede ser controlado, siendo éste uno de los hiperparámetros del modelo. La actualización de los pesos y sesgos se realiza a partir de la comparación entre las predicciones del modelo y los valores reales presentes en los datos de entrenamiento. Dicha comparación se cuantifica mediante una función de pérdida (*loss function*), que mide la discrepancia entre las predicciones del modelo y los valores de la variable objetivo. Durante el entrenamiento, el objetivo principal es minimizar esta función de pérdida, lo cual se logra habitualmente mediante un algoritmo de optimización, típicamente basado en *descenso de gradiente* [23]. El gradiente indica la dirección de máximo

incremento de una función continua; por tanto, desplazarse en la dirección opuesta permite aproximarse a un mínimo. En este contexto, se calcula el gradiente de la función de pérdida respecto de cada uno de los pesos y sesgos de la red [7, 16]. Por ejemplo, una función de coste utilizada frecuentemente para clasificación es la entropía cruzada (*cross entropy*), que mide cuán bien se ajusta la distribución del conjunto de probabilidades estimadas a la distribución de las probabilidades reales, penalizando de forma más significativa cuando se asigna baja probabilidad a la clase real [7].

El algoritmo fundamental para calcular estos gradientes de manera eficiente en redes neuronales, incluso en arquitecturas profundas con múltiples capas, es el de retropropagación (*backpropagation*). La retropropagación opera en dos fases: una pasada hacia adelante (*forward pass*) donde se calculan las salidas del modelo y la función de coste, y una pasada hacia atrás (*backward pass*) donde se propaga el error desde la capa de salida hacia atrás a través de las capas, calculando la contribución al error de cada parámetro [4, 11, 21]. Como el descenso de gradiente completo es costoso computacionalmente [11], suelen utilizarse distintas versiones del algoritmo de optimización, como por ejemplo el *ADAM* (*Adaptive Moment Estimation*) que adapta la tasa de aprendizaje para cada parámetro de modelo individualmente, acelerando el entrenamiento y mejorando la convergencia [7, 11].

Monitoreo de entrenamiento

La selección del "mejor" modelo entre aquellos obtenidos en las distintas *épocas* se basa en el rendimiento observado sobre el conjunto de validación. A medida que el entrenamiento avanza, el modelo tiende a mejorar su ajuste a los datos de entrenamiento, lo cual suele manifestarse en una disminución de la función de pérdida. No obstante, en etapas avanzadas del entrenamiento, el modelo puede comenzar a ajustarse excesivamente a las particularidades o al ruido presente en los datos de entrenamiento generando un sobreajuste. Esto se traduce en una disminución del rendimiento al ser evaluado sobre datos no vistos, es decir, sobre el conjunto de validación (Figura 6). Una estrategia habitual para mitigar el sobreajuste es la interrupción del entrenamiento (*early stopping*), que requiere monitorear el rendimiento del modelo sobre el conjunto de validación a lo largo del entrenamiento. El procedimiento se interrumpe cuando dicho rendimiento deja de mejorar o comienza a deteriorarse durante un número consecutivo de *épocas*, aun cuando las métricas calculadas sobre el conjunto de entrenamiento continúen mejorando [4, 7]. La interrupción temprana y el criterio de degradación de la *performance* (número consecutivo de épocas) son hiperparámetros.

Otra estrategia, de mayor costo computacional y riesgo de sobreajuste, es entrenar por una cantidad de épocas establecida y conservar el modelo óptimo (el *checkpoint*) de la época con mejor rendimiento en validación [11].

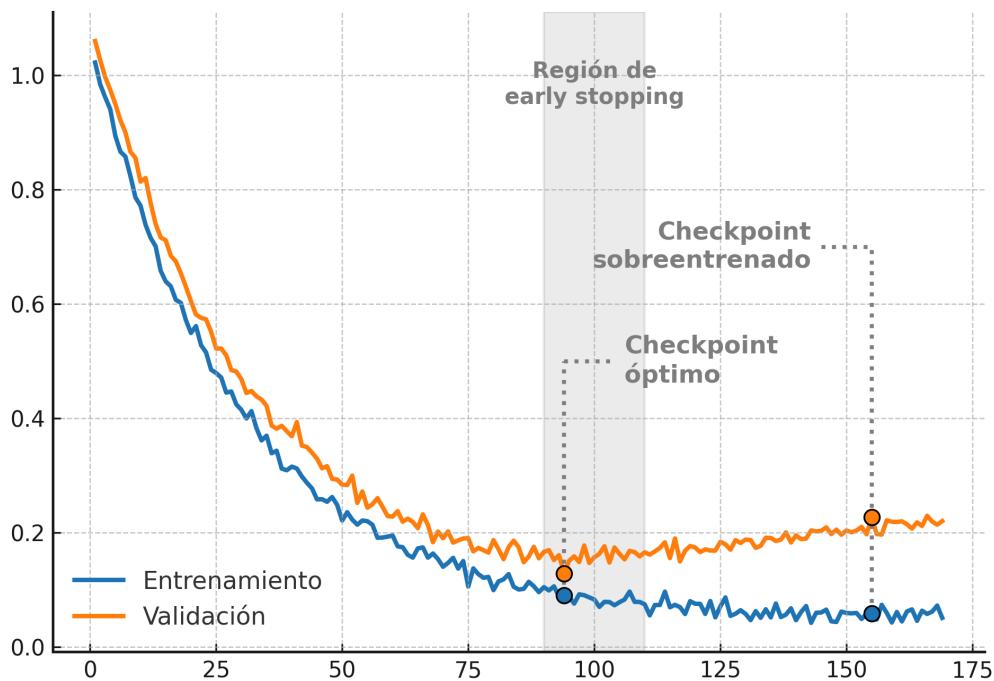


Figura 6. Evolución de la función de pérdida (*loss*) durante el entrenamiento del modelo en los conjuntos de entrenamiento (azul) y validación (naranja). Se indican el *checkpoint* óptimo (mínimo de pérdida en validación), el *checkpoint* sobreentrenado y la región correspondiente al criterio de *early stopping* utilizado para interrumpir el entrenamiento antes de que se produzca sobreajuste.

1.1.3. Redes neuronales convolucionales

Las redes neuronales convolucionales (*CNN*, por sus siglas en inglés) constituyen un tipo especializado de redes neuronales diseñadas específicamente para procesar datos estructurados en cuadrículas, tales como imágenes. Inspiradas en el córtex visual humano, estas redes logran identificar patrones visuales reconociendo características presentes en distintas posiciones espaciales de los datos de entrada. Su uso es central en áreas como la visión por computadora, donde destacan en tareas de clasificación de imágenes mediante la identificación jerárquica de patrones visuales, desde características simples a patrones más complejos [7, 16].

Convoluciones

Una imagen es representada por la computadora como una matriz numérica, donde cada elemento indica la intensidad de un *pixel* (Figura 7). En el caso de las imágenes en color, esta representación se compone de tres matrices superpuestas correspondientes a los canales rojo, verde y azul (RGB), cuya combinación permite reconstruir el color de cada *pixel*.

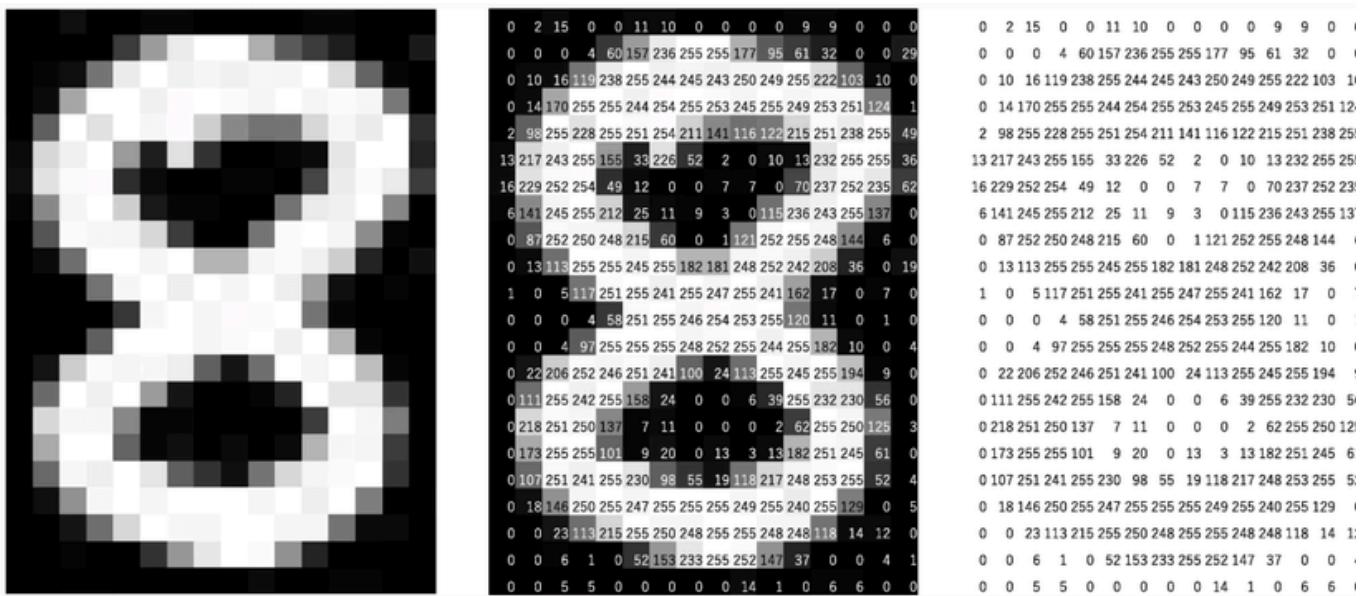


Figura 7. Interpretación de una imagen por una computadora (Extraído de Hidayat et al. [24]). La imagen del número 8 en negro y blanco, se representa como una matriz de valores numéricos que indican la intensidad de cada *pixel* con valores que van desde el 0 (completamente oscuro) al 255 (completamente claro).

La convolución 2D es una operación fundamental en el procesamiento de imágenes. En su forma más general, implica deslizar una función sobre otra y medir la integral de su multiplicación punto a punto [7]. En la práctica, para una imagen, esto se traduce en aplicar un "filtro" o "kernel" (una pequeña matriz de pesos) sobre la imagen de entrada. Para cada posición (una ventana deslizante), se multiplican los elementos correspondientes del filtro y la región de la imagen que cubre, y luego se suman los resultados para producir un único valor en la salida (Figura 8A) [16]. Si se agrega, por ejemplo, una capa de valores nulos rodeando la imagen (Figura 8B), se logra obtener una imagen de igual tamaño, a la que se le aplicó un filtro (tal como los filtros aplicados en los softwares de edición de imágenes). Este filtro actúa resaltando características locales específicas en la imagen, como bordes verticales, horizontales o pequeñas formas, al destacar las áreas donde estos patrones coinciden (Figura 8C) [16].

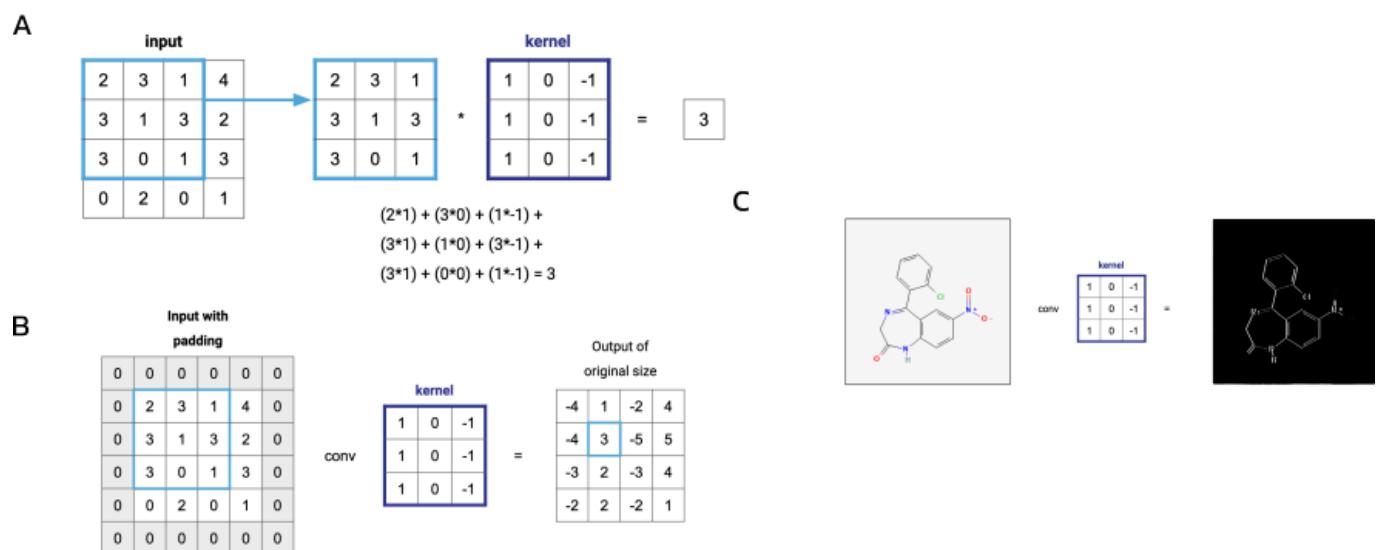


Figura 8. Ejemplo de operación convolucional (Basado en [25]). (A) Cálculo de la convolución entre una región de la imagen y un *kernel* 3×3 , mediante multiplicación elemento a elemento y suma. (B) Al aplicar *padding* (agregado de ceros en el borde) se conserva el tamaño original de la imagen tras la convolución. (C) Aplicación del mismo *kernel* a una imagen de una estructura de Lewis: se resaltan los bordes (enlaces) y se atenúan regiones como los átomos de oxígeno en grupos carbonilo.

Arquitectura general de las CNN de imágenes

La arquitectura típica de una *CNN* (Figura 9) se caracteriza por capas convolucionales y capas de agregación (*pooling*), seguidas por capas completamente conectadas. Las capas convolucionales, bloques centrales de esta arquitectura, procesan los datos de entrada mediante la aplicación de filtros o *kernels* que identifican patrones específicos, tales como bordes o texturas. La convolución consiste en deslizar estos filtros sobre la imagen de entrada, generando mapas de características que resaltan áreas donde estos patrones aparecen. La operación convolucional es típicamente seguida por una función de activación no lineal, como ReLU (*Rectified Linear Unit*), que introduce no linealidad, permitiendo a la red aprender funciones más complejas.

Las capas de agregación o *pooling*, por otro lado, realizan un submuestreo de estos mapas de características, disminuyendo sus dimensiones espaciales para reducir la complejidad computacional, minimizar el riesgo de sobreajuste y dotar al modelo de cierta invarianza frente a desplazamientos menores del patrón detectado. Finalmente, los mapas de características resultantes son aplandados y procesados por capas completamente conectadas de tipo FNN para generar la clasificación final.

El proceso de ajuste o entrenamiento de una *CNN* implica aprender los valores de los pesos (incluidos los de los filtros/kernels en las capas convolucionales y los pesos en las capas completamente conectadas) y los sesgos de la red. Esto se logra minimizando una función de coste (o pérdida) que cuantifica la diferencia entre las predicciones de la red y las etiquetas reales de las imágenes de entrenamiento. Es decir que la red aprende la manera más optimizada de representar las imágenes para clasificarlas, y la manera más optimizada de clasificarlas.

En conjunto, esta estructura jerárquica permite a las *CNN* capturar patrones visuales con una gran eficacia, siendo especialmente apropiadas para aplicaciones que requieren identificar objetos o características dentro de imágenes o datos organizados espacialmente [7, 11, 16].

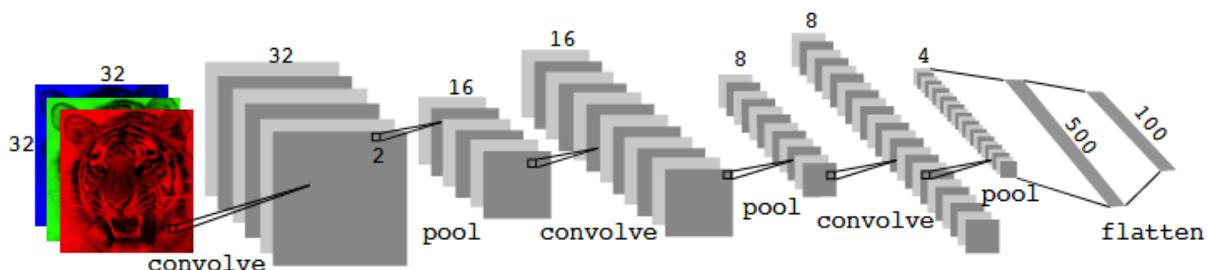


Figura 9. Arquitectura de una *CNN* (Extraído de James et al. [16]) que utiliza kernels de 2×2 , *pooling layers* de 2×2 y 1 capa completamente conectada de 500 neuronas seguido de una capa *output* de 100 neuronas.

1.2. Descubrimiento de drogas

1.2.1. Proceso completo de descubrimiento de drogas

Una droga o fármaco puede definirse como cualquier sustancia capaz de generar un cambio en un proceso biológico mediante acciones químicas. Generalmente, la molécula (o ligando) actúa como agonista (activador) o antagonista (inhibidor) al interactuar con una o más moléculas *target* (receptores) que desempeñan un papel específico en el sistema biológico. Los efectos de una droga se producen a partir de su unión con la macromolécula receptora, alterando la actividad bioquímica o biofísica de esta última. El peso molecular, la estructura y la carga eléctrica de una droga determinan su afinidad y unión con los posibles sitios del receptor. La mayoría de los receptores de drogas clínicamente relevantes son proteínas [26].

El descubrimiento y desarrollo de un nuevo fármaco (o droga) es un proceso largo, riesgoso y costoso. Típicamente, el ciclo de su descubrimiento lleva entre 10 y 18 años [27] y cuesta entre 161M y 1800M de dólares capitalizados [28]. Dicho proceso (Figura 10), comienza con la identificación y validación de un blanco terapéutico (*target*) mediante técnicas como minería de datos, correlaciones de expresión de proteínas con enfermedades, estudios de asociaciones genéticas con polimorfismos, *screening* fenotípico, entre otros. Luego se desarrolla un ensayo específico para realizar una búsqueda intensiva de moléculas bioactivas ("hits") utilizando estrategias como *screening* de alto rendimiento (HTS) o *screening* virtual. Estos compuestos químicos iniciales ("hits") se optimizan química y farmacológicamente en una fase denominada "hit-to-lead", donde se mejoran características como potencia, selectividad, solubilidad y propiedades farmacocinéticas. Posteriormente, los compuestos más prometedores entran en la etapa de optimización de leads (candidatos prometedores a ser una droga exitosa), donde se profundiza en la relación estructura-actividad, se reducen efectos secundarios y toxicidades y se ajustan propiedades de absorción, distribución, metabolismo y excreción. Finalmente, las moléculas candidatas se someten a estudios preclínicos (*in vitro* e *in vivo*) para caracterizar toxicidad, estabilidad química y farmacología detallada, antes de iniciar ensayos clínicos en humanos. En caso de superar la fase preclínica y clínica la droga pasa por un extenso proceso regulatorio antes de ser aprobada y comercializada en el mercado [29].

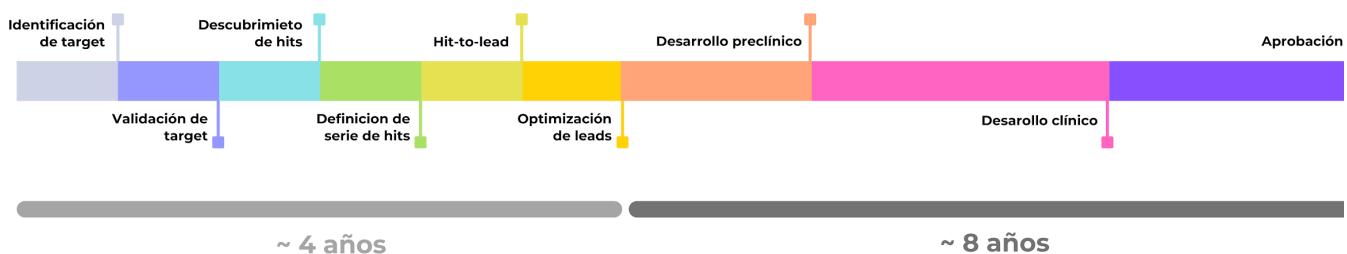


Figura 10. Esquema del proceso de descubrimiento de drogas basado en Hughes et al. [29]. No está a escala.

1.2.2. Descubrimiento de hits

El concepto de una molécula *hit* varía según diferentes investigadores, pero en este trabajo el término se refiere a un compuesto químico que tiene la actividad deseada en un ensayo de *screening* y dicha actividad es confirmada en un retesteo [29, 30], y tiene la potencialidad de convertirse en *lead* [31].

El proceso de descubrimiento de *hits* puede abordarse con o sin información previa sobre la relación entre la estructura química y la actividad biológica de los compuestos. En ausencia de información, se recurre a estrategias como el *high-throughput screening* (HTS), que consiste en la evaluación masiva de bibliotecas de compuestos químicos de bajo peso molecular (generalmente <500 Da) en un ensayo biológico definido [29, 32]. El objetivo de este procedimiento es identificar moléculas capaces de modular un proceso biológico determinado, que puedan posteriormente desarrollarse como compuestos *lead*.

Este tipo de abordaje es llevado a cabo principalmente por la industria farmacéutica [26], dado que requiere una infraestructura costosa, tanto por la necesidad de automatización y miniaturización de los ensayos, como por los recursos necesarios para sintetizar y mantener bibliotecas químicas extensas y diversas. La eficacia del HTS también depende en gran medida de la serendipidad, ya que se explora una gran cantidad de compuestos sin un conocimiento previo específico que guíe la selección [31] generando una tasa de hits muy baja y alta tasa de falsos positivos y falsos negativos [33]. Asimismo, para maximizar las probabilidades de éxito, las bibliotecas deben cubrir un espacio químico amplio, lo que incrementa considerablemente los tiempos de desarrollo y los costos asociados a la síntesis y validación experimental [30, 32]. La ventaja principal de este enfoque es su capacidad para revelar nuevas interacciones biológicas de manera poco sesgada, especialmente útil cuando no se conoce el tipo de estructura química que podría ser activa frente al blanco terapéutico. Sin embargo, este método suele generar una gran cantidad de compuestos con actividad inespecífica o limitada, requiriendo posteriores etapas de validación y optimización para seleccionar candidatos viables [29, 32].

Cuando existe información previa sobre interacciones entre compuestos y *targets*, se pueden utilizar estrategias de *screening* enfocado, donde se aprovechan datos estructurales, modelos computacionales y conocimientos previos sobre el [farmacóforo](#) para identificar compuestos con mayor probabilidad de éxito [29]. Esta metodología de *screening* está generalmente ligada al diseño de drogas asistido por computadora (*CADD*, por sus siglas en inglés), que con metodologías basadas en estructuras y en ligandos, acelera considerablemente el tiempo de desarrollo y reduce los costos del descubrimiento de drogas [32, 34].

Diseño de drogas asistido por computadora basado en ligandos

El diseño de drogas asistido por computadora basado en ligandos (*ligand-based computer-aided drug discovery, LB-CADD*) es una estrategia que se fundamenta en el análisis de compuestos conocidos por interactuar con un blanco biológico de interés. A diferencia de los enfoques basados en la estructura del *target*, *LB-CADD* no requiere información estructural del mismo, por

lo que constituye un enfoque indirecto para el descubrimiento de fármacos. Este enfoque se basa en el principio de similitud de propiedades [35], que sostiene que moléculas estructuralmente similares tienden a compartir propiedades biológicas. A partir de este principio, se utilizan estructuras de referencia con actividad conocida para construir representaciones moleculares que exhiben las propiedades fisicoquímicas relevantes para la interacción deseada, descartando información no pertinente [34].

Modelos de relaciones estructura-actividad

Uno de los enfoques utilizados dentro del *LB-CADD* es la construcción de modelos de relación estructura-actividad (structure–activity relationships, SAR) [34]. Estos modelos buscan establecer relaciones entre la estructura química de los compuestos y una propiedad biológica o fisicoquímica de interés, utilizando modelos estadísticos o de *machine learning* [34, 36]. De forma general, el proceso de desarrollo de un modelo SAR (Figura 11), es análogo al proceso de ML explicado en secciones anteriores (Figura 1). Incluye la recolección y curado de datos, la partición del conjunto en subconjuntos de entrenamiento, validación y testeо, la selección de una representación molecular, el entrenamiento de un modelo predictivo y finalmente una etapa de validación de dicho modelo.

Una particularidad de estos modelos es el marco del problema, que algunos autores lo denominan como un problema de “reconocimiento temprano” [3]. El objetivo de realizar un *screening* virtual, es ranquear en los primeros lugares a los compuestos activos dentro de una biblioteca de muchos compuestos (por ejemplo de más de 1.000.000 de compuestos), ya que se busca testear experimentalmente sólo una pequeña porción de los compuestos de la biblioteca.

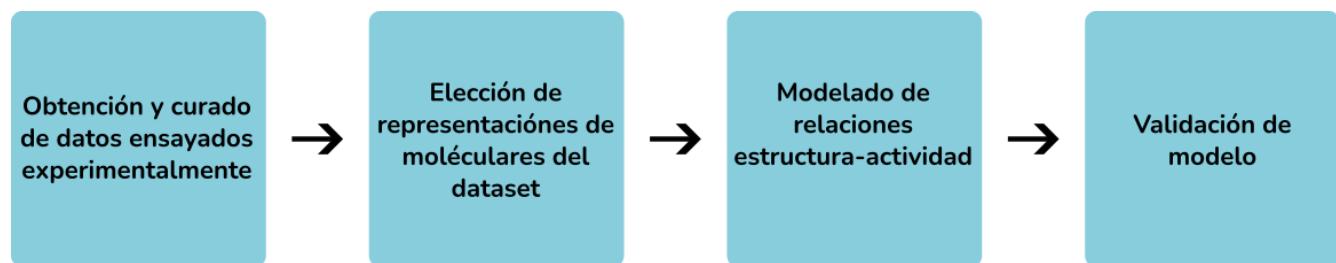


Figura 11. Proceso simplificado de desarrollo de modelos de relaciones estructura-actividad [36]. Se puede considerar un caso de uso específico del *pipeline* de ML de la Figura 1.

Obtención y curado de datos

La construcción de un modelo SAR comienza con la obtención de un conjunto de datos compuesto por moléculas con actividad biológica medida experimentalmente. Estos datos pueden derivar de ensayos realizados por el mismo grupo de investigación [37], aunque con mayor frecuencia se obtienen de la bibliografía o de bases de datos de ensayos químicos [36, 38–40]. Los datos deben ser rigurosamente inspeccionados y curados, dado que los modelos dependen de la diversidad química del conjunto de datos del que aprende [34], y la presencia de errores o inconsistencias puede comprometer su capacidad predictiva [36, 41].

Estos conjuntos de datos suelen obtenerse de bases de datos públicas. Una de estas bases de datos es ChEMBL, una plataforma curada manualmente que compila información de bioactividad de

compuestos químicos extraída de literatura científica, patentes y datasets depositados. ChEMBL contiene actualmente más de 20 millones de mediciones de bioactividad correspondientes a más de 1,6 millones de ensayos, con foco en targets moleculares como proteínas aisladas, complejos proteicos o líneas celulares [42].

En este trabajo se utilizaron datos provenientes de ensayos integrados en ChEMBL de tipo *binding* contra proteínas aisladas. Estos ensayos de *binding* miden la afinidad de un compuesto por su blanco molecular y reportan resultados como *Ki* (constante de inhibición), *Kd* (constante de disociación), *IC₅₀* (concentración inhibitoria media) y *EC₅₀* (concentración efectiva media), todos valores cuantitativos ampliamente utilizados en farmacología para estimar la potencia de interacción entre un compuesto y un blanco.

El valor de *IC₅₀* indica la concentración del inhibidor necesaria para reducir en un 50% una función enzimática o biológica medida, y se obtiene a partir de una curva de respuesta-inhibidor ajustada sobre la base de velocidades iniciales de reacción. A pesar de su uso extendido, el *IC₅₀* no representa una constante fundamental del sistema, ya que depende de múltiples factores experimentales como la concentración de sustrato, las condiciones del medio, el tipo de célula o proteína ensayada, y el modelo matemático utilizado para el ajuste.

Por el contrario, los valores de *Ki* y *Kd* son constantes de disociación bajo condiciones de equilibrio, que reflejan de manera directa la afinidad entre una molécula y su blanco. El cálculo de *Ki* requiere mayor esfuerzo experimental, ya que implica medir velocidades iniciales a múltiples concentraciones de sustrato e inhibidor bajo modelos cinéticos definidos [43].

Un desafío que plantea la obtención de datos de bioactividad para modelos de ML de bases de datos es el sesgo de publicación positiva. A pesar de que en los ensayos de *screening* los compuestos activos son eventos de baja ocurrencia [44], se observa que en las bases de datos masivas como ChEMBL existe una tendencia a encontrar más compuestos activos observados que los esperados. Inclusive suelen encontrarse targets con mayor proporción de compuestos activos que inactivos, algo poco esperable en un muestreo del espacio químico [45]. Este sesgo es especialmente perjudicial a la hora de entrenar modelos de ML que se benefician de datasets balanceados para poder generalizar y realizar predicciones sin sesgo [5, 7, 45]. Es por eso que el desbalance en los datos de entrenamiento es un factor clave a considerar a la hora de realizar *LB-CADD*, y se lo debe tener en cuenta en la elección y preparación del *dataset*, estrategia de modelado y en las métricas de validación utilizadas.

Partición de conjuntos de datos químicos

Como se mencionó previamente, los conjuntos de datos químicos en el descubrimiento de fármacos, suelen ser desafiantes debido a su tamaño limitado y desequilibrio de clases. Para evaluar la capacidad de generalización de los modelos de *LB-CADD* en este contexto, se utilizan particionados de datos racionales, como por ejemplo el "*scaffold splitting*" [4, 46].

El "*scaffold splitting*" se basa en la identificación de esqueletos moleculares (*Bemis-Murcko scaffolds*) [47, 48]. Un esqueleto molecular se define como la unión de los sistemas de anillos y

los átomos conectores de una molécula, ignorando las cadenas laterales y los detalles de los tipos de átomos o los órdenes de enlace en su definición más genérica [49].

En la práctica del descubrimiento de fármacos *in silico*, el *scaffold splitting* se emplea para dividir las moléculas en conjuntos de entrenamiento, validación y prueba, garantizando que compuestos con esqueletos químicos idénticos permanezcan agrupados y no sean asignados a particiones distintas. Esta estrategia proporciona un criterio de evaluación más exigente que la división aleatoria, ya que la partición aleatoria podría incluir moléculas estructuralmente similares en los conjuntos de entrenamiento y testeo, situación que no refleja adecuadamente la diversidad química que se espera en compuestos realmente novedosos. Utilizando *scaffold splitting*, se pretende evitar una sobreestimación del desempeño del modelo causada por la exposición del predictor durante el testeo, a moléculas excesivamente parecidas a las del conjunto de entrenamiento [4, 46, 47].

Representación molecular

Para que una estructura química pueda utilizarse en métodos computacionales, es necesario representarla de forma estandarizada (Figura 12). Habitualmente, las bases de datos químicas almacenan esta información en formato *SMILES* (*Simplified Molecular Input Line Entry System*), que codifica la estructura molecular como una cadena lineal de caracteres (Figura 12A). Este formato permite representar compactamente la conectividad atómica y la estereoquímica del compuesto [50, 51].

En el contexto de *LB-CADD*, la representación seleccionada debe permitir codificar cada compuesto químico en un vector numérico adecuado para su procesamiento mediante modelos de *machine learning*. Para este propósito, una opción tradicional es el uso de descriptores moleculares (Figura 12B). Los descriptores moleculares son representaciones numéricas de una molécula química que se calculan mediante reglas predefinidas, extrayendo información cuantitativa sobre su estructura y propiedades [52]. Un subtipo de descriptor habitualmente utilizado son las *fingerprints* moleculares (Figura 12C), vectores binarios que codifican la presencia o ausencia de fragmentos estructurales específicos, permitiendo representar la molécula en un formato de longitud fija apropiado para algoritmos de aprendizaje automático. Un ejemplo utilizado en esta tesis es la *fingerprint* topológica *All-Shortest Path* (ASP), que codifica exclusivamente los caminos más cortos entre todos los pares de átomos dentro de la molécula, capturando la topología molecular y demostrando buenos resultados en tareas de predicción de bioactividad [36].

Además de estas representaciones clásicas, existen metodologías basadas en redes neuronales que extraen automáticamente las *features* relevantes a partir de estructuras químicas. Por ejemplo, las redes convolucionales sobre grafos, como *Chemprop* (Figura 12D), generan *embeddings* numéricos que capturan información estructural mediante el procesamiento directo del grafo molecular [37, 53]. Alternativamente, las redes neuronales convolucionales en 2D como *DEEPScreen* (Figura 12E) generan *embeddings* a partir de imágenes de estructuras químicas en

formato de Lewis, codificando automáticamente patrones espaciales visuales relevantes para la predicción de actividad biológica [39].

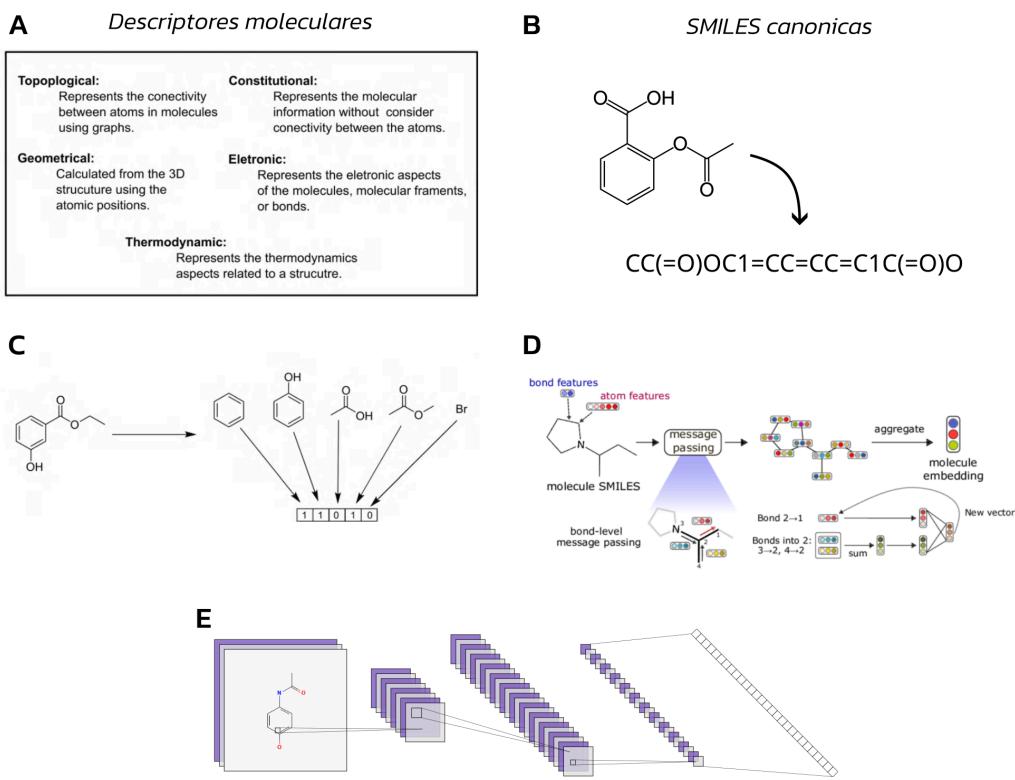


Figura 12. Representaciones computacionales de moléculas químicas. (A) Definición de las distintas clases de descriptores moleculares [36]. (B) Serialización de una molécula química a una *SMILES*. (C) Representación esquemática de una *fingerprint* molecular, donde una molécula es comparada a distintos fragmentos y esta información es combinada en un vector de valores binarios denominado *fingerprint*. (D) Esquema del *embedding* generado por redes neuronales de grafos utilizado en Chemprop [53]. (E) Esquema de *embedding* generado por redes neuronales convolucionales en 2D a partir de imágenes de estructuras de Lewis [39].

Validación de modelos

La validación de modelos SAR implica determinar si un modelo posee la capacidad adecuada para predecir actividades biológicas de compuestos químicos basándose en su estructura molecular (en el contexto de una campaña de *screening in silico*). Este proceso suele involucrar dos etapas: la validación interna y la validación externa.

La validación interna evalúa el rendimiento del modelo utilizando únicamente el conjunto de datos de entrenamiento, verificando así su robustez y capacidad para ajustar correctamente los datos empleados en su creación. Generalmente, se implementan estrategias como la validación cruzada (*cross-validation*), donde el conjunto de entrenamiento se divide en subconjuntos para validar la capacidad predictiva del modelo internamente [36].

La validación externa, utilizada en esta tesina, es crucial para evaluar la capacidad predictiva real del modelo sobre datos no utilizados en ninguna etapa del entrenamiento o ajuste del modelo. Este método implica predecir la actividad de compuestos de un conjunto de testeo

independiente, permitiendo determinar objetivamente el desempeño predictivo mediante métricas específicas. La calidad predictiva determinada mediante la validación externa es la que define la utilidad práctica del modelo SAR en contextos reales de *screening in silico* [13].

Para la validación de modelos SAR se emplean métricas provenientes del *machine learning* junto con métricas diseñadas específicamente para el *virtual screening*. Entre las primeras destacan la precisión y la exhaustividad (*recall*, sensibilidad), definidas respectivamente como:

$$\text{Precisión} = \frac{TP}{TP+FP} \quad \text{Recall} = \frac{TP}{TP+FN} \quad [54]$$

donde *TP*, *FP* y *FN* representan respectivamente verdaderos positivos, falsos positivos y falsos negativos. La precisión (también llamado valor positivo predicho) representa la fracción de instancias recuperadas que son relevantes, mientras la exhaustividad o *recall* (también llamado sensibilidad) es la fracción de instancias relevantes que han sido recuperadas.

Como se mencionó previamente, los datos de los modelos SAR suelen estar desbalanceados, es por eso que también se suelen utilizar métricas adicionales como el *F1-score* y el coeficiente de correlación de *Matthews* (*MCC*), definidos respectivamente como:

$$F1 \text{ score} = \frac{2 \times \text{Precisión} \times \text{Recall}}{\text{Precisión} + \text{Recall}} \quad MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN-FP)(TN-FN)}}$$

Estas métricas son menos sensibles al desbalance de clases [4]. El *F1-Score*, es la media armónica entre la Precisión y la Exhaustividad [55], mientras que el *MCC* puede entenderse como un coeficiente de correlación entre las clasificaciones predichas por el modelo y las reales, con un rango de -1 a +1. Un *MCC* de +1 indica predicciones perfectas, 0 indica que no son mejores que una predicción aleatoria, y -1 indica una discrepancia total entre predicciones y valores reales.

Junto a ellas se emplea la curva ROC y el área bajo la curva ROC (*AUROC*), que cuantifica la capacidad del modelo para distinguir entre compuestos activos e inactivos a diferentes umbrales. La curva ROC (*Receiver Operating Characteristic*) se llama así por su origen en la detección de señales de radares, e históricamente se utilizaba para definir las características operativas de los receptores de señales [56]. En una curva ROC (Figura 13) la coordenada *x* es la probabilidad condicional de falsos positivos (tasa de falsos positivos), y la coordenada *y* es la probabilidad condicional de un acierto (tasa de falsos negativos). De esta manera, la curva ROC evalúa el equilibrio entre la sensibilidad y la especificidad del modelo. El área bajo la curva (*AUROC*) resume globalmente este desempeño: un valor cercano a 1 indica una capacidad predictiva perfecta, mientras que un valor de 0.5 representa una predicción aleatoria, equivalente a elegir al azar entre las clases.

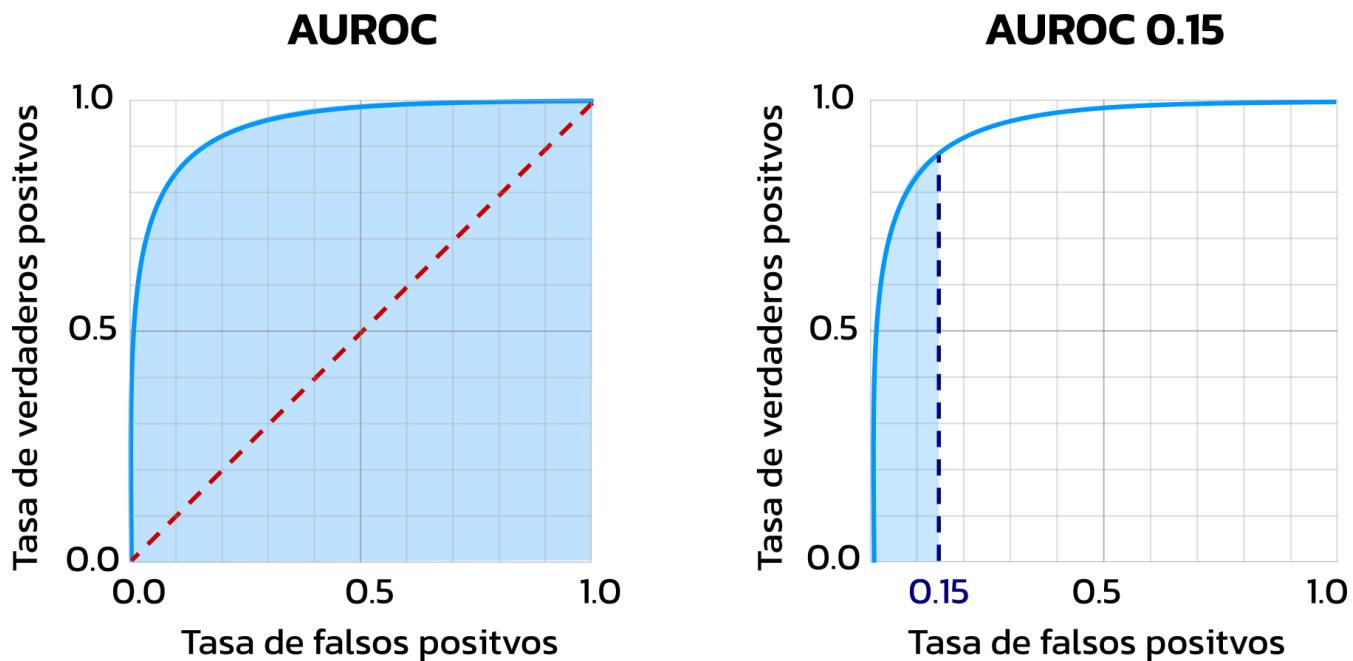


Figura 13. La figura muestra la curva ROC (Receiver Operating Characteristic), que evalúa la capacidad de un modelo para discriminar entre clases mediante la relación entre la tasa de verdaderos positivos y la de falsos positivos. El área bajo la curva ROC (ROC-AUC, AUROC) representa una medida directa de la capacidad predictiva global del modelo. A la izquierda se representa el ROC-AUC global, mientras que a la derecha se muestra el ROC-AUC 0.15, que calcula el área bajo la curva restringida a una tasa de falsos positivos menor al 15 %, enfocándose en las predicciones de mayor señal del modelo.

En el contexto específico de descubrimiento de fármacos, y en escenarios de *screening* virtual, los conjuntos de datos suelen estar marcadamente desbalanceados, con una proporción muy baja de compuestos activos en el mejor de los casos; por tanto la métrica *AUROC* puede sobreestimar el desempeño de los modelos predictivos, sobre todo al considerar la lista completa de predicciones. Por motivos prácticos y económicos, en estos casos suele ser preferible priorizar la detección temprana de compuestos activos. Para esto se emplean variantes de la *AUROC* como la métrica *BEDROC* [3] o la variante parcial *AUROC 0.15*, que calcula el área bajo la curva solo hasta un 15 % de falsos positivos (Figura 13). Ambos enfoques asignan mayor peso a los verdaderos positivos situados al inicio del *ranking*, es decir, a los compuestos con el *score* más alto. En contraste, *F1* y *MCC*, aunque robustas frente al desbalance de clases, no incorporan esta prioridad temprana en su formulación. La combinación de métricas globales (*precision*, *recall*, *AUROC*) y de reconocimiento temprano (*AUROC 0.15*, *BEDROC*) ofrece una evaluación equilibrada del desempeño del modelo SAR, aunque la validación definitiva depende de comprobar experimentalmente la actividad de los compuestos priorizados.

Similitud química como herramienta de LB-CADD

El principio de “similitud de las propiedades” establece que compuestos estructuralmente parecidos tienden a exhibir propiedades biológicas semejantes, y constituye uno de los pilares fundamentales de *LB-CADD*. En este contexto, la similitud química se emplea como un método directo para priorizar y seleccionar compuestos con probabilidad de poseer la actividad deseada frente a un blanco biológico, aun cuando no se disponga de información estructural del *target* biológico [34, 35, 57, 58].

La base teórica de esta estrategia parte de la consideración de que en general pequeñas modificaciones en la estructura de un fármaco tienden a producir cambios graduales en la actividad biológica, de modo que compuestos similares presentan respuestas similares. La limitante de esta premisa yace en los llamados “precipicios de actividad” (en inglés, *activity cliffs*), donde pequeñas modificaciones estructurales alteran drásticamente la actividad biológica. De cualquier manera, suelen existir regiones de “baja pendiente” donde la similitud química, es una buena medida para encontrar compuestos análogos, siempre y cuando la manera de representar las moléculas y calcular dicha similitud favorezca dichas topografías SAR [57, 59, 60].

Para cuantificar la similitud entre moléculas suelen utilizarse *fingerprints* o descriptores moleculares que codifican la información química relevante (p. ej., presencia de subestructuras, propiedades topológicas o rasgos farmacofóricos) en forma de vectores numéricos o cadenas binarias como se explica en la Figura 12. Una vez definidas dichas representaciones, la comparación entre compuestos se lleva a cabo utilizando coeficientes de similitud como el índice de Tanimoto (Jaccard) o el coeficiente de Braun-Blanquet, que permiten estimar el grado de solapamiento estructural entre las moléculas evaluadas [57, 60].

En la práctica, el enfoque de *screening virtual* basado en similitud comienza seleccionando un conjunto de ligandos de referencia con actividad confirmada contra el blanco de interés. A partir de ellos se calculan las *fingerprints* (o de descriptores) que capturen sus rasgos estructurales y/o fisicoquímicos determinantes de la bioactividad. Luego, cada molécula de la biblioteca candidata se compara con los ligandos de referencia según la métrica elegida y se obtiene un *score* de similitud. Finalmente, rankean los compuestos en base a este *score*, y se seleccionan los compuestos que superen un umbral (o *cutoff*) de similitud, asumiendo que estos presentarán propiedades biológicas análogas. Esta metodología se encuentra sintetizada en la Figura 14. Con esta estrategia se pueden descartar de manera rápida y eficiente enormes cantidades de compuestos poco prometedores, reduciendo significativamente la carga experimental y a un costo computacional bajo [58, 61].

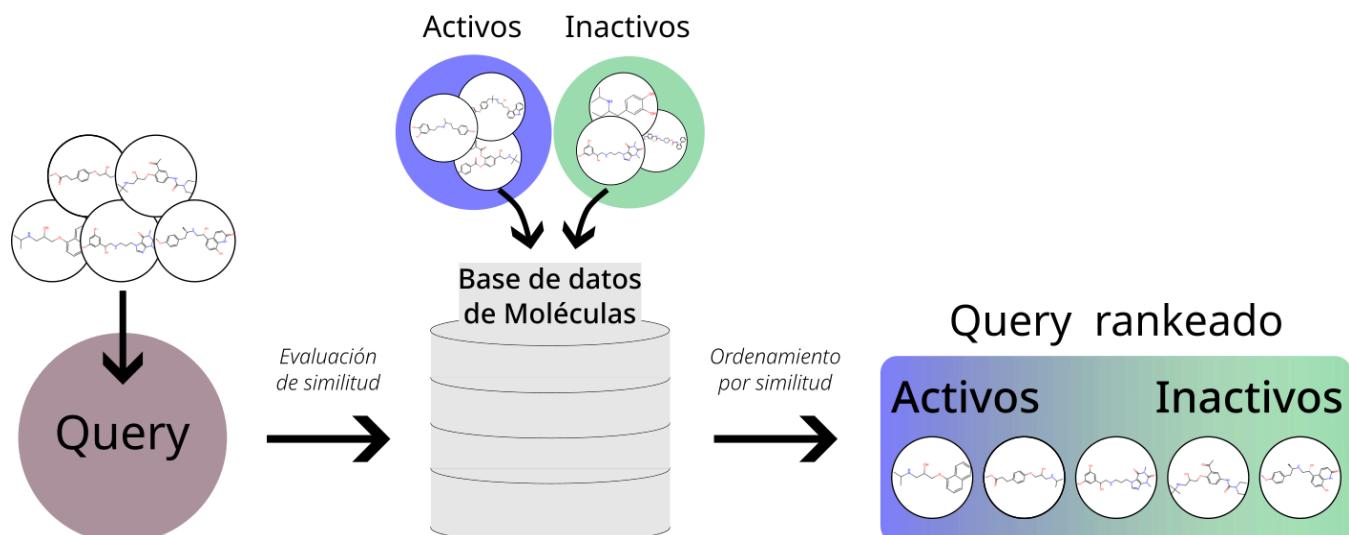


Figura 14. Protocolo de *screening in silico* basado en similitud molecular (Adaptado de Cai et al. [57]). Las estructuras query se evalúan por similitud contra una base de datos de moléculas activas e inactivas, para obtener un *ranking* de

actividad potencial.

1.2.3. Descubrimiento de drogas para enfermedades tropicales desatendidas

Enfermedades tropicales desatendidas

La Organización Mundial de la Salud (OMS) ha identificado como Enfermedades Tropicales Desatendidas (ETD) a un grupo de 20 enfermedades infecciosas que afectan principalmente a poblaciones marginadas en regiones tropicales y subtropicales, y que han sido históricamente descuidadas en los programas de salud pública globales [62]. Estas enfermedades representan aproximadamente el 11% de la carga de enfermedades a nivel mundial y afectan a más de 1000 millones de personas en todo el mundo [63].

A pesar de su impacto sanitario, el desarrollo de terapias novedosas para las ETD ha sido históricamente escaso, en gran medida por la limitada rentabilidad que estas patologías ofrecen a la industria farmacéutica [63–65]. A modo de ejemplo, entre 2000 y 2011, únicamente cinco de los 850 productos terapéuticos aprobados ($\approx 0,6\%$) se destinaron a ETD, y ninguno constituyó una nueva entidad química; todos fueron reposicionamientos, reformulaciones o combinaciones de fármacos existentes [63, 66].

Dado el rol crucial de las ETD en la salud global, la OMS estableció en 2021 una hoja de ruta para las ETD, en busca de controlarlas y eliminarlas para 2030 [62]. El documento resalta la necesidad de intervenciones terapéuticas innovadoras, seguras, eficaces y de bajo costo, e insta a intensificar la investigación y el descubrimiento de nuevos fármacos. Dentro de las ETD, las enfermedades causadas por protozoarios y helmintos, requieren tratamientos quimioterápicos intensivos para su manejo y prevención; sin embargo, el desarrollo de nuevas drogas para estos cuadros ha estado históricamente limitado por la falta de innovación. Afortunadamente, en los últimos años asociaciones entre ONGs, institutos de investigación y la industria farmacéutica han impulsado el desarrollo de drogas contra estas enfermedades. Iniciativas como la *Drugs for Neglected Diseases initiative* (DNDi), el financiamiento estratégico de la *Bill & Melinda Gates Foundation* y los programas del *Wellcome Centre for Anti-Infectives Research* de la Universidad de Dundee han impulsado el descubrimiento innovador de compuestos mediante consorcios de optimización de activos (*lead optimization*), redes de *screening* compartido y acuerdos de cooperación académica e industrial [65]. Por ejemplo, en 2015 *Glaxo Smith Kline* (GSK) compartió los resultados de un *screening* HTS que realizó contra las tres enfermedades causadas por parásitos kinetoplástidos más relevantes [67].

Enfermedad de Chagas

Entre las ETD causadas por protozoarios kinetoplástidos se encuentra la Enfermedad de Chagas la cual es endémica en 21 países de la región incluyendo a la Argentina, afectando a aproximadamente 6-7 millones de personas y causando 12.000 muertes al año [68]. Esta enfermedad es reconocida por la OMS como una de las ETD más comunes en las zonas de mayor pobreza [69]. Además, existe un número creciente de casos en regiones no endémicas [70–72].

Trypanosoma cruzi e implicancias clínicas

El agente etiológico causante de Enfermedad de Chagas es *Trypanosoma cruzi* es un protozoario de la clase Kinetoplastida, familia Trypanosomatidae. Este parásito tiene un ciclo de vida que alterna entre hospedadores invertebrados (hematófagos de la familia *Triatominae*) y mamíferos. El ciclo de vida del parásito es complejo. El parásito prolifera en un estadio no infectivo en el tracto gastrointestinal del vector, y luego en respuesta a estrés nutritivo se diferencia a su forma infectiva, que es liberada en las heces del vector cuando el mismo se alimenta de la sangre de un mamífero. Luego el parásito alcanza el torrente sanguíneo a través de lesiones en la piel o mucosas.

Una vez en el torrente sanguíneo el parásito invade células nucleadas. Luego de la invasión a la célula blanco el parásito escapa de los lisosomas y se diferencia en el citosol a una forma proliferativa intracelular, que luego de sucesivas divisiones destruyen la célula huésped alcanzando nuevamente el torrente sanguíneo e invadiendo otras células. Los principales órganos afectados en la fase crónica son el corazón y el sistema digestivo, lo que puede derivar en cardiomiopatías, megaesófago o megacolon. Aproximadamente entre el 30-40% de los pacientes en fase crónica tras décadas de infección desarrollan estas patologías, causando alteraciones graves en los tejidos afectados [73].

Tratamientos contra la enfermedad de Chagas

La enfermedad de Chagas es clínicamente curable durante la fase aguda si el diagnóstico se hace tempranamente, y en casos de contagio congénito. Actualmente para el tratamiento de la enfermedad se utilizan dos drogas: Nifurtimox (Lampit; Bayer 2502) y Benznidazol (Abarax; ELEA) [74]. Estas drogas son efectivas en la fase aguda y no durante la fase crónica de la enfermedad, además de resultar inconvenientes para el paciente debido a su alta toxicidad generando una baja adherencia a los programas de tratamiento [75]. A pesar de su menor efectividad durante fase crónica, existe una tendencia a indicar el tratamiento también en estos pacientes [76, 77].

Se cree que el efecto del nifurtimox se debe a que la droga sufre procesos de óxido-reducción que generan especies reactivas tales como radicales superóxido, peróxido de hidrógeno y radicales hidroxilo. Estas especies producen estrés oxidativo que puede dañar el DNA o lípidos de las membranas celulares [78]. Además, el nifurtimox inhibe a la enzima tripanotionina reductasa, lo cual resulta en la inhibición de la formación de tripanotionina. Esta molécula es un tiol de bajo peso molecular, exclusivo de tripanosomátidos, implicado en la defensa contra oxidantes, xenobióticos y proteínas regulatorias, y es esencial para la supervivencia del parásito [79].

En cuanto al benznidazol, su acción tripanocida se debe a su función como prodroga, activada por una nitroreductasa tipo I (NTR1) tripanosómica dependiente de NADH, ausente en eucariotas superiores. Esta enzima cataliza una reacción independiente de oxígeno que genera metabolitos reductores altamente reactivos. Estos compuestos inducen múltiples daños en el parásito, incluyendo la formación del dialdehído glioxal (capaz de generar aductos con guanosina), la

depleción de tioles, la modificación covalente de macromoléculas celulares y, fundamentalmente, la inducción de roturas de doble cadena en el ADN nuclear [80, 81].

Estrategias de reposicionamiento

Debido a las limitaciones de tiempo y recursos económicos, el descubrimiento *de novo* ha sido históricamente poco frecuente en el ámbito de las enfermedades desatendidas [82, 83]. En su lugar, han predominado enfoques que reutilizan desarrollos existentes o esfuerzos compartidos, como la extensión de indicaciones o el *piggy-back drug discovery*. A estos enfoques se los denomina reposicionamiento.

El reposicionamiento puede abordarse desde distintos enfoques, dependiendo de la entidad: drogas completas, compuestos candidatos (leads), blancos terapéuticos específicos o incluso familias completas de blancos [83]. En particular, para enfermedades tropicales desatendidas resulta especialmente relevante el enfoque conocido como "*piggy-back drug discovery*", en el cual compuestos originalmente desarrollados para enfermedades comercialmente más atractivas se reposicionan hacia otros patógenos debido a la conservación del blanco terapéutico [82].

Dentro del reposicionamiento es fundamental distinguir dos tipos principales: reposicionamiento de drogas y *targets*. El primero consiste en la identificación de nuevas indicaciones para fármacos ya aprobados, como es el caso de la eflornitina, originalmente desarrollada como agente antitumoral y posteriormente aprobada para el tratamiento de la tripanosomiasis africana [84]. Por otra parte, el reposicionamiento de blancos terapéuticos implica aprovechar un mecanismo de acción ya conocido y validado en una enfermedad para identificar compuestos activos en otros patógenos que comparten dicho *target*. En este caso no se busca un reposicionamiento directo sino una optimización de los compuestos en pos de maximizar la actividad o especificidad por el *target* reposicionado [83].

Planteo del trabajo

En base al estado actual de las técnicas de *machine learning* y su aplicación en el descubrimiento de drogas, sumado a la necesidad de identificar nuevos compuestos terapéuticos y aprovechando las ventajas que ofrece el reposicionamiento de fármacos existentes, en esta tesis exploramos el uso de métodos computacionales basados en redes neuronales convolucionales para la predicción de la bioactividad de compuestos químicos frente a proteínas (*targets*) aisladas.

El objetivo final del trabajo del laboratorio es el de poder desarrollar modelos capaces de clasificar y priorizar compuestos potencialmente activos, con el objetivo último de ensayar experimentalmente estos candidatos contra *Trypanosoma cruzi*. A pesar de que este objetivo es de mayor alcance y excede el trabajo de Tesis, aquí sentamos las bases de la caracterización y desarrollo de modelos de aprendizaje automático existentes y proveemos un *baseline* comparador para futuros desarrollos.

2. Objetivos

Generales

- Generar métodos computacionales para colaborar en la identificación de potenciales compuestos reposicionables hacia nuevos blancos proteicos.

Específicos

- Identificar conjuntos de compuestos activos e inactivos contra proteínas únicas (cada conjunto asociado a una proteína/*target* diferente).
- Desarrollo de un método simple de referencia (“*baseline*”) basado en similitud química para clasificar compuestos activos e inactivos y usarlo como punto de comparación.
- Entrenar modelos de redes neuronales convolucionales específicas para los blancos proteicos con los sets de datos generados. Esto incluye diseñar, optimizar y programar la arquitectura de las redes neuronales, y poner a punto el sistema automatizado de ajuste de parámetros e hiperparámetros para el fine tuning.
- Estudiar la *performance* de los modelos entrenados en conjuntos de datos novedosos, por comparación con distintas estrategias de modelado. Analizar la *performance* comparativa de modelos frente a conjuntos de datos de bioactividad reportada (antibacterianos y tripanocidas).

3. Resultados

3.1. Esquema general

La base de datos ChEMBL almacena información de moléculas pequeñas y su actividad biológica (bioactividad), extraída de artículos publicados en las principales revistas de química medicinal y los integra junto con otras bases de datos mayores como PubChem BioAssay y BindingDB [85]. Los ensayos de tipo “*Binding*”, que están presentes en ChEMBL, son aquellos que miden la interacción de compuestos químicos con targets moleculares [86]. Dentro de estos ensayos, los de tipo “*Single Protein*”, son bioactividades medidas contra proteínas aisladas (por ejemplo proteínas naturales o recombinantes purificadas). Estas bioactividades, constan de compuestos activos e inactivos contra diferentes proteínas, algunas de las cuales pueden ser *targets* biológicos de patógenos causantes de distintas enfermedades. Es decir que cada bioactividad consta de 1 compuesto químico evaluado en 1 *target*, que en este caso es 1 proteína aislada.

Estos conjuntos de datos de *Binding* de actividad e inactividad contra targets *Single Protein* serán el punto de partida de la exploración del uso de *Deep Learning* para el reposicionamiento de compuestos bioactivos. A partir de los datos filtrados y curados de interacciones compuesto-target, se seleccionaron algunos targets específicos para entrenar modelos clasificadores predictores de bioactividad basados en redes neuronales convolucionales. En primer lugar se entrenaron modelos para validar y poner a punto el funcionamiento de las redes neuronales. Luego se estudió la *performance* de los modelos en comparación con distintas metodologías de modelado y también en conjuntos de datos que difieren de los objetivos iniciales del modelado. Por último se analizó el comportamiento de los modelos en conjuntos de datos de ensayos de actividades tripanocida, para finalizar la evaluación de la metodología de modelado y su aplicabilidad en el campo del *screening in silico* aplicado a enfermedades desatendidas.

3.2. Obtención de datos para el entrenamiento de modelos clasificadores de bioactividad de compuestos químicos frente a proteínas aisladas

3.2.1. Filtrado inicial de la base de datos ChEMBL

Se extrajeron datos de bioactividad a partir de ChEMBL v34, seleccionando únicamente aquellos correspondientes a ensayos de *binding* asociados a *targets* de proteínas aisladas (*Single Protein*). Tras un proceso de filtrado y curado, se obtuvo un conjunto final compuesto por bioactividades binarias (activo/inactivo), correspondientes a ensayos de compuestos químicos en *targets* proteicos. Es decir que cada *target* virtual, corresponde a una lista de compuestos activos e inactivos contra una única proteína aislada.

La Figura 15 resume gráficamente los resultados obtenidos en cada etapa del filtrado, desde un total inicial de 20.772.701 bioactividades, hasta la conformación final del conjunto depurado de 1.285.624 bioactividades binarias de compuestos químicos, distribuidas en 6.140 targets proteicos. El detalle completo del procedimiento aplicado, incluyendo los criterios de exclusión, estandarización y deduplicación, se encuentra en la sección de [Metodología](#).

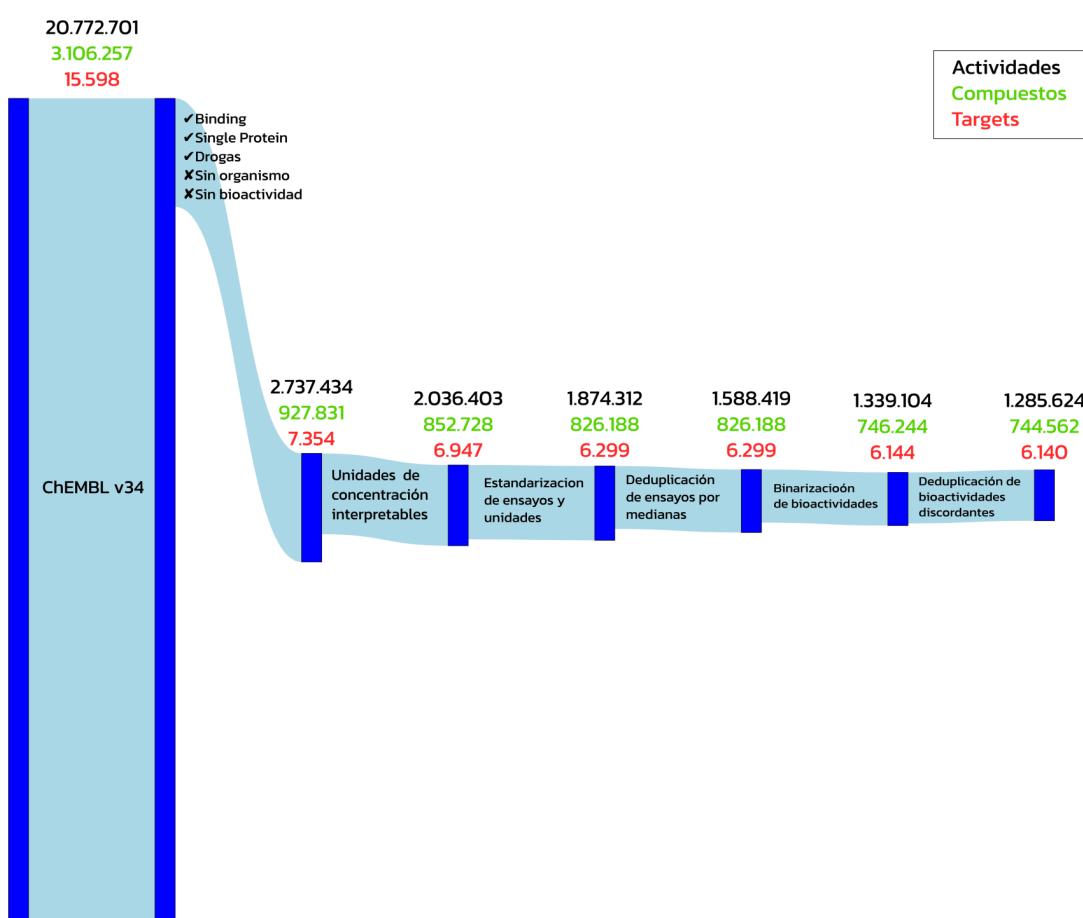


Figura 15. Reducción del número de bioactividades tras el *pipeline* de filtrado y curado de datos. Cada etapa de filtrado y curado descarta bioactividades, reduciéndose la cantidad de compuestos químicos y targets.

3.2.2. Selección de targets para entrenar modelos de machine learning

El conjunto de datos depurado, posee información sobre la bioactividad de compuestos químicos contra 6.140 targets, siendo cada *target* una proteína única con distintas bioactividades binarias (activas e inactivas) reportadas. A continuación se seleccionaron los targets que poseían entre 700 y 3500 bioactividades. Se buscó filtrar *targets* en este rango, para tener una cantidad mínima aceptable [9] de datos para el entrenamiento, y una cantidad máxima de datos adaptada a las capacidades de cómputo que se dispusieron para el trabajo. Luego se descartaron todos los targets con un ratio activos/inactivos de entre 0.3 - 4.5, para no disponer de targets con un desbalance mayor que impida realizar entrenamientos. Se puede observar en la Figura 16 un esquema con todos los targets que superaron los filtros y curados hasta esta instancia.

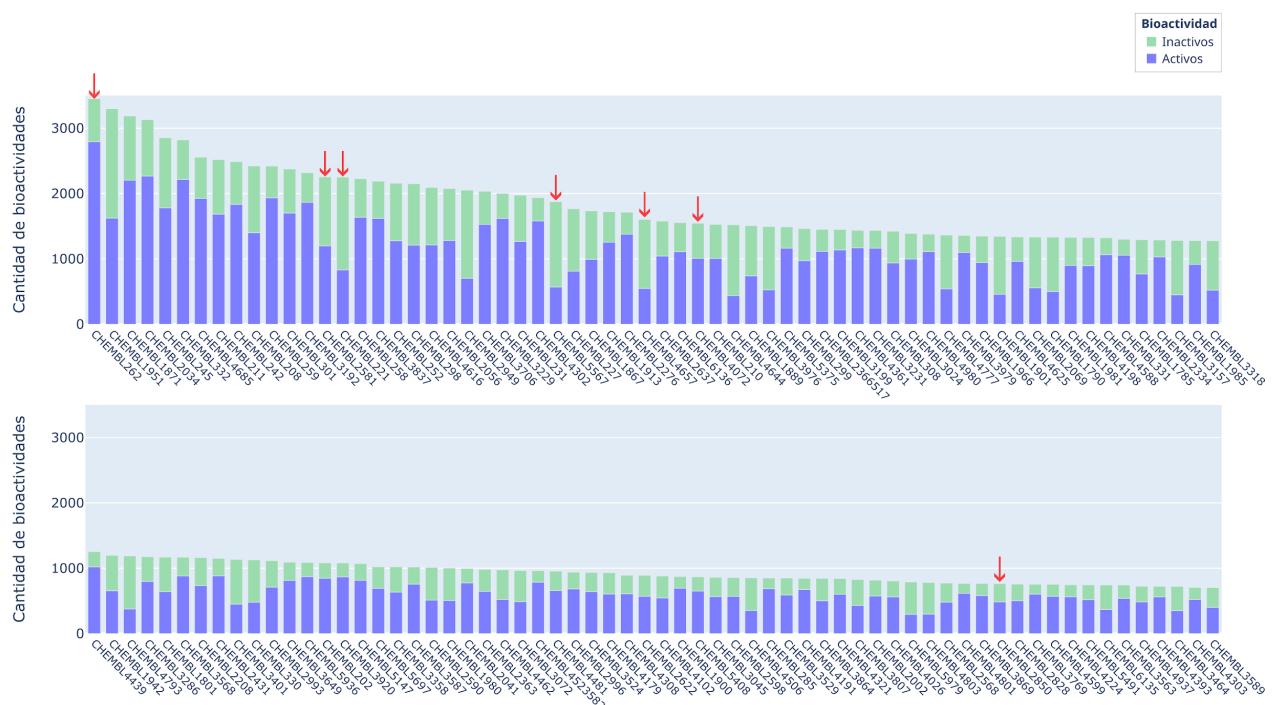


Figura 16. Distribución de la cantidad de bioactividades por *target*, separadas en activas (en azul) e inactivas (en verde). El gráfico se divide en dos niveles para optimizar la visualización: el gráfico inferior es la continuación hacia la derecha del gráfico superior. Los *targets* destacados con flechas rojas representan aquellos seleccionados por los criterios específicos para su análisis posterior.

A continuación se seleccionaron 7 *targets* (Tabla 1) dentro de los criterios establecidos, basándose en el balance entre los compuestos activos e inactivos y la cantidad de datos disponibles para cada *target*. Se observa en la Figura 17, cómo se seleccionaron targets de distintas cantidades de bioactividades (desde 3452 a 764) y distintas proporciones de activos/inactivos. Estos targets fueron seleccionados independientemente del organismo de origen, con el objetivo de ser utilizados para validar el adecuado funcionamiento de los modelos que se entrenaron posteriormente. En todo el manuscrito se mencionará a estos *targets* por su CHEMBL ID sin contemplar a qué proteína representa.

Tabla 1. Targets seleccionados para la validación de los modelos.

Enzima	CHEMBL ID	Organismo de origen
Glycogen synthase kinase-3 beta	CHEMBL262	<i>Homo sapiens</i>
Cathepsin D	CHEMBL2581	<i>Homo sapiens</i>
Cyclooxygenase-1	CHEMBL221	<i>Homo sapiens</i>
Luciferin 4-monoxygenase	CHEMBL5567	<i>Photinus pyralis</i> (luciérnaga)
Dipeptidyl peptidase VIII	CHEMBL4657	<i>Homo sapiens</i>
Glycogen synthase kinase-3 alfa	CHEMBL2850	<i>Homo sapiens</i>
Cathepsin B	CHEMBL4072	<i>Homo sapiens</i>

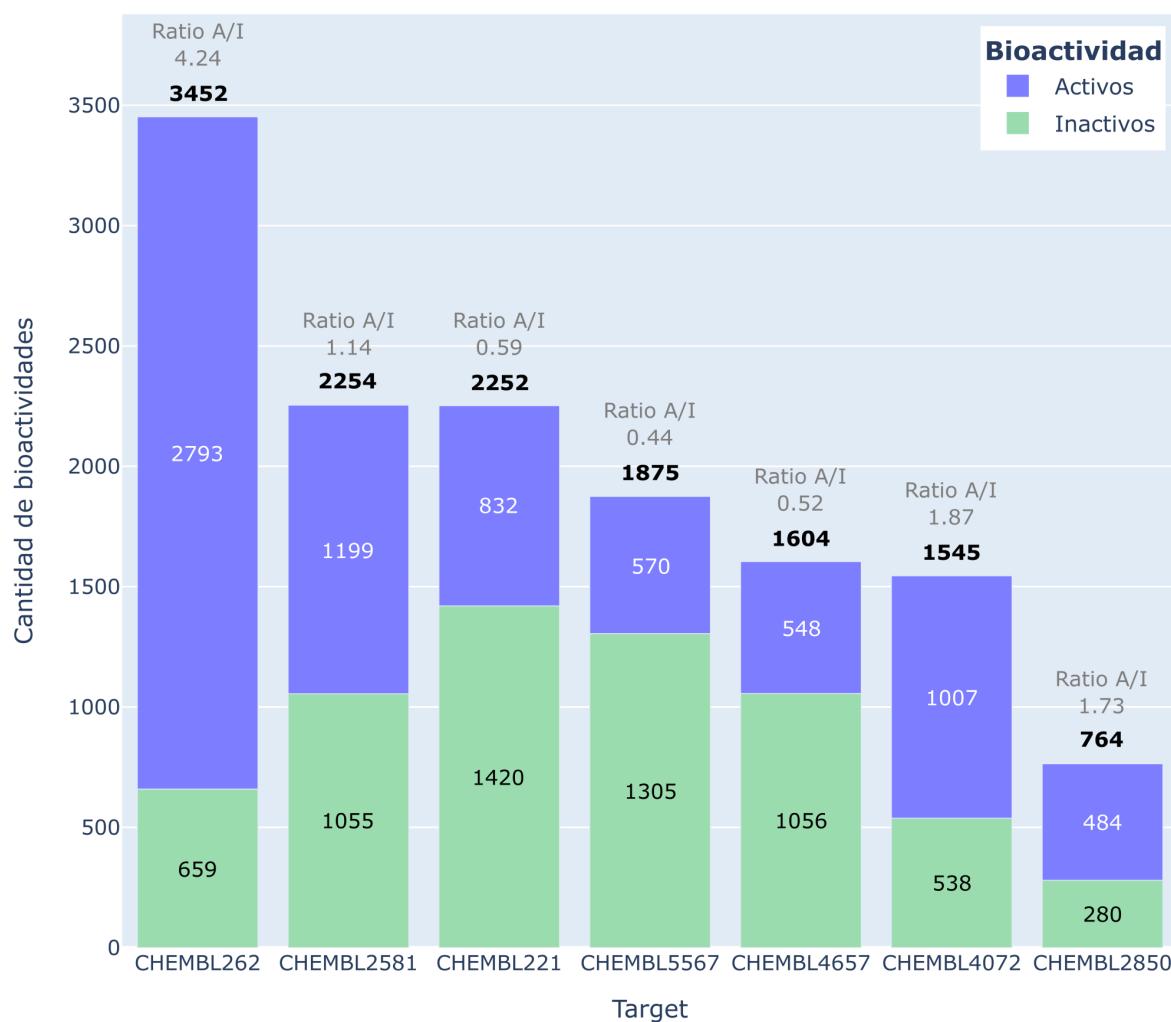


Figura 17. Cantidad total de bioactividades por target seleccionado, separadas en activas (en azul) e inactivas (en verde). Sobre cada barra se indica el total de bioactividades (en negrita) y el ratio activos/inactivos (A/I). Los targets seleccionados presentan una variedad tanto en la cantidad total de bioactividades (rango de 3452 a 764) como en las proporciones de activos respecto a inactivos.

3.2.3. Detección de sesgos de similitud química

Se estudió la diversidad química de los *datasets* de cada *target*, calculando la similitud química entre todos los compuestos activos entre sí y entre todos los compuestos inactivos entre sí. Los resultados de estas evaluaciones de similitud química se plasmaron en histogramas (Figura 18) que muestran el número de eventos de similitud entre los pares de compuestos, separados entre los activos y los inactivos. Es decir, de haber n compuestos activos y m compuestos inactivos, habrán n^2 eventos de similitud de activos, y m^2 eventos de similitud de inactivos.

Este análisis se realizó para estudiar la posible presencia de sesgo de similitud química en los datos disponibles para cada *target*. Por ejemplo, es posible que todos los compuestos activos sean muy similares entre sí debido a sesgos experimentales de origen en los *screenings* de series químicas reportados en ChEMBL. En tal caso, cualquier modelo debería aprender muy fácilmente que moléculas activas deben ser muy parecidas a esta serie química. En estos casos, no existe una ventaja real en aplicar métodos complejos de aprendizaje automático, ya que simplemente se estaría evaluando la similitud química, la cual ya puede medirse a través de numerosos métodos, como el índice de Tanimoto o el coeficiente de Braun-Blanquet.

Para la mayoría de los targets seleccionados, se encontró una media de similitud baja (alrededor de 0.2). Sin embargo, la catepsina D (CHEMBL2581) presentó una gran cantidad de compuestos activos más similares entre sí, en comparación con los inactivos. A su vez, la glucógeno sintasa quinasa-3 alfa (CHEMBL2850) presentó en menor medida un pico de compuestos de alta similitud. Estos picos de similitud supondrían cierto sesgo en los datasets, que podrían influenciar el desempeño de los modelos. De cualquier manera, no se descartaron dichos datasets, ni se realizó algún curado individual para apaciguar el sesgo; por el contrario, se buscó estudiar el comportamiento de los modelos en este tipo de *datasets*.

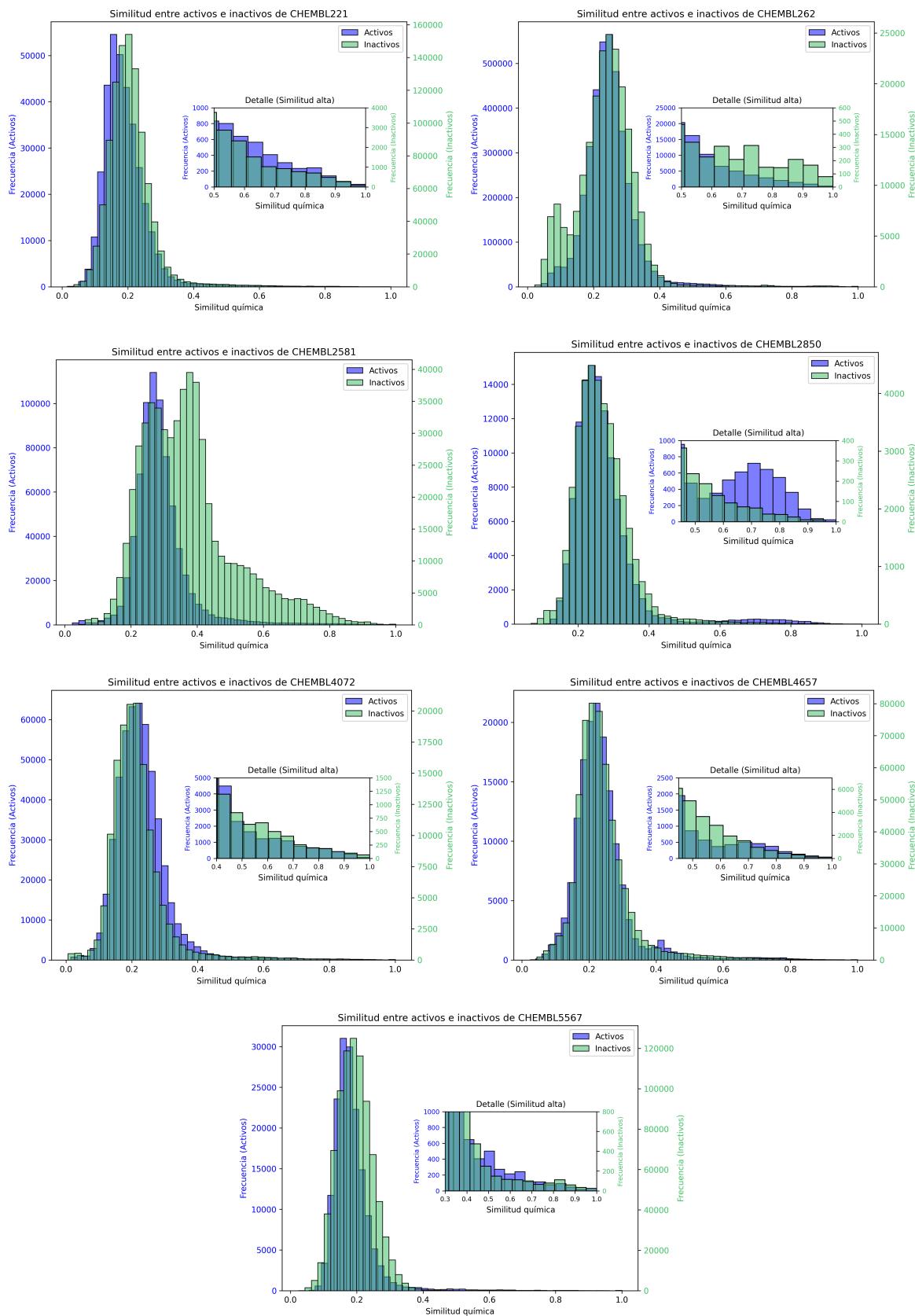


Figura 18. Estudio de similitud química de los diferentes targets. En celeste se encuentran graficados los histogramas de las similitudes entre todos los activos y en verde entre todos los inactivos. Cabe resaltar que dados los diferentes balances de moléculas activas e inactivas, los ejes Y para cada subconjunto son diferentes, para poder apreciar las distribuciones de similitud internas de cada conjunto.

3.2.4. Partición de datos con *Scaffold Splitter* para maximizar la independencia entre set de datos de entrenamiento y testeo

Se partió cada conjunto de datos en: datos de entrenamiento, validación y testeo. Se realizó una partición basada en *scaffolds* de Bemis-Murcko, agrupando las moléculas según esqueletos compartidos (representaciones donde se remueven cadenas laterales de las moléculas para generar estos esqueletos o “*scaffolds*”). Este método de partición genera subconjuntos de datos lo más distintos entre sí posible, lo que dificulta al modelo la predicción en el conjunto de testeo, proporcionando así una evaluación que lo fuerza a utilizar información “aprendida” en lugar de información “memorizada”. Se utilizó esta metodología para lograr detectar el sobreajuste y poder calcular la *performance* sin sobreestimar la capacidad de los modelos.

Es importante señalar que el conjunto de validación se emplea durante el entrenamiento para ajustar los hiperparámetros y seleccionar el modelo óptimo, mientras que el conjunto de testeo permanece completamente independiente y se utiliza al final del desarrollo para evaluar el desempeño final del modelo.

En la Figura 19 se presentan las particiones realizadas para cada uno de los conjuntos de datos de los targets. Se puede visualizar como el *scaffold splitter* logra construir conjuntos de datos con ambas clases (activos e inactivos), y en cierta medida mantiene los ratios originales de activos e inactivos, especialmente en los conjuntos de entrenamiento que son los que poseen mayor cantidad de bioactividades. Esto se puede observar comparando los ratios (activos/inactivos), entre la Figura 17 y la Figura 19.

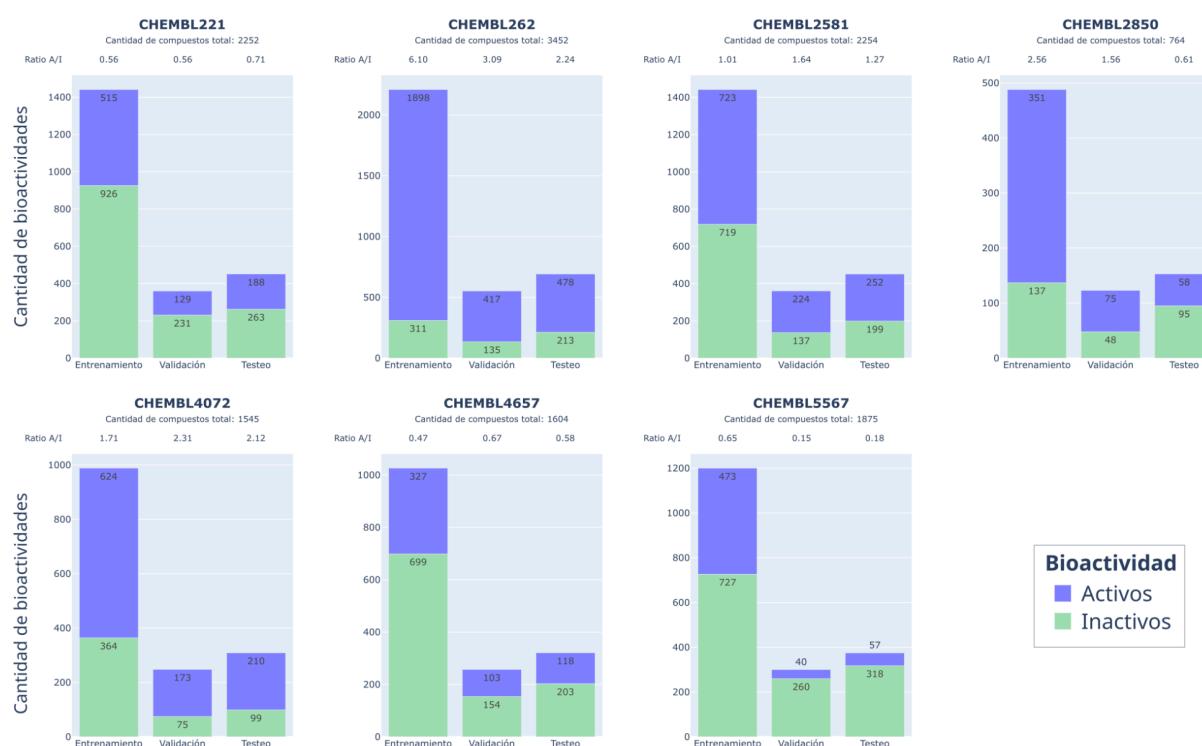


Figura 19. Balance de bioactividades de los distintos targets, en las distintas particiones de datos. Se observa como el *scaffold splitter* distribuye los compuestos activos e inactivos en las distintas particiones manteniendo en cierta medida las proporciones originales.

Discusión de la sección 3.2

El filtrado de ChEMBL entregó un conjunto depurado, aunque con desequilibrios que varían según el *target*: en algunos casos predominan los compuestos activos y en otros los inactivos. Esto es un factor clave a tener en cuenta al leer las métricas de validación, sobre todo las que ponderan el hallazgo temprano de activos como *BEDROC*, para no sobrevalorar a los modelos.

La similitud estructural entre activos e inactivos permitió detectar posibles series químicas de alta similitud estructural. Esto es importante a la hora de evaluar los modelos, ya que es posible que estén sesgados por esos conjuntos de datos similares. Si bien el uso de histogramas para visualizar estos sesgos de similitud en datos fue útil, se podría evaluar el uso de métricas especializadas para evaluar la similitud de conjuntos de datos o métricas de disimilitud entre conjuntos de datos químicos para describir con mayor fidelidad la diversidad molecular [87]. También la utilización de *clustering* podría sumar al entendimiento de la diversidad química de los conjuntos.

Se optó por la partición por *scaffolds* en lugar de validación cruzada debido a las restricciones de tiempo y cómputo disponible para entrenar las redes convolucionales. Si bien acelera el entrenamiento, este esquema puede sobreestimar la *performance* de los modelos al conservar *scaffolds* similares en distintas particiones generando una falsa expectativa de validación [88]. A su vez, esta técnica de *splitting* por definición debería generar datasets de testeo muy diferentes al *set* de datos de entrenamiento. Esto recae sobre la falsa idea que los modelos de ML son capaces de generar un aprendizaje trascendental sobre datos encontrando patrones indetectables de otra manera, cuando en realidad los modelos de ML son técnicas de aprendizaje estadístico aplicado [16] y no suelen ser buenos extrapolando observaciones muy distintas a los datasets de entrenamiento [89].

3.3. Desarrollo de modelos clasificadores para screening de compuestos químicos *in silico*

Se desarrollaron modelos clasificadores para ser entrenados con los conjuntos de datos obtenidos en las instancias anteriores. En primer lugar, se diseñó un modelo que emplea la similitud química tradicional, sin estrategias de aprendizaje automático. Este modelo se utilizó como “*baseline*” de la evaluación, es decir la *performance* mínima que se debería esperar de un modelo de aprendizaje automático para ser aceptable. Luego se implementaron los modelos de redes convolucionales *DEEPscreen* publicados en Rifaioglu *et al.* [39], que aplicando cambios y mejoras dieron a nuestros modelos *TrypanoDEEPscreen*. Durante todo el manuscrito, se referirá al modelo basado en similitud química como “*baseline*”, a los modelos de Rifaioglu *et al.* [39] como “*DEEPscreen*” y a los modelos modificados por nosotros como “*TrypanoDEEPscreen*”.

3.3.1. Implementación de un modelo basado en similitud química como *baseline* para comparación con redes neuronales

Bajo el principio de similitud propiedad (los compuestos químicos estructuralmente parecidos tienden a exhibir propiedades biológicas semejantes) [35, 58, 90], se desarrolló un algoritmo predictivo basado en la similitud química de los compuestos (Figura 20). El algoritmo calcula la similitud química del conjunto de compuestos químicos a predecir, contra todos los compuestos activos y contra todos los compuestos inactivos de un *target*, por separado. Luego se aplica la función *max()* a los resultados de similitud de cada conjunto, obteniendo el valor máximo de similitud, correspondiente al compuesto químico más similar al compuesto evaluado. Finalmente, ambos resultados de cada conjunto (activos/inactivos) se procesan por la función *softmax()*, obteniéndose valores complementarios que van del 0 al 1. Este algoritmo resultó similar a estrategias publicadas de *screening in silico* [57, 60, 90, 91], aunque difiere en la manera de utilizar el *set* de entrenamiento e integra la señal con la función *softmax*. En la metodología se detalla cómo evaluamos distintos tipos de *fingerprints*, coeficientes de similitud y metodologías de *scoring*, para el diseño del algoritmo, hasta lograr llegar a la estructura actual.

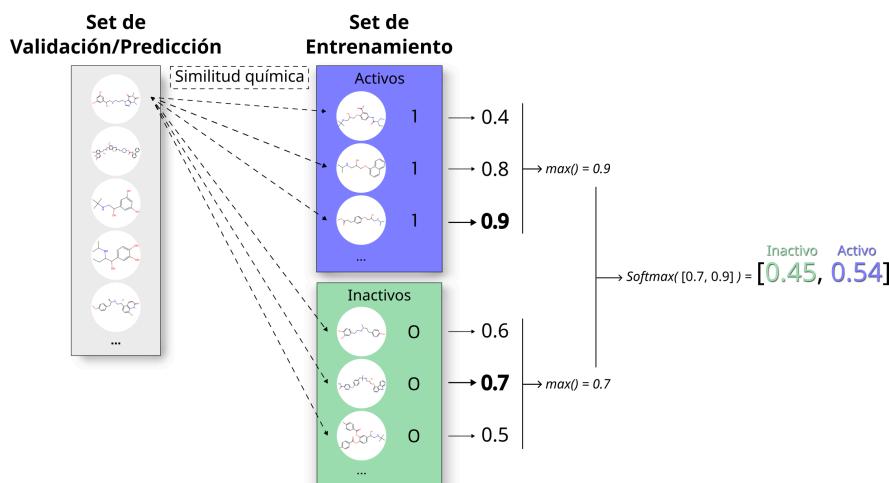


Figura 20. Esquema del algoritmo predictivo basado en similitud. Se observa como a partir de un conjunto de datos químicos con actividad biológica, se clasifica un conjunto de datos novedoso sin ningún paso de ajuste.

3.3.2. Desarrollo de TrypanoDEEPscreen, una estrategia de redes neuronales convolucionales basadas en estructuras en 2D de compuestos químicos para la predicción de bioactividad

Adaptación y reimplementación de DEEPscreen como base para el desarrollo de TrypanoDEEPscreen

Los modelos *DEEPscreen*, desarrollados por Rifaioglu *et al.* [39], consisten en 704 redes neuronales convolucionales 2D diseñadas para predecir la bioactividad de compuestos frente a 704 *targets* proteicos. Estos modelos procesan representaciones en formato de estructuras de Lewis de compuestos químicos (imágenes de 200x200 px) y, mediante datos de bioactividad asociados, aprenden a clasificar compuestos desconocidos como activos o inactivos (Figura 21).

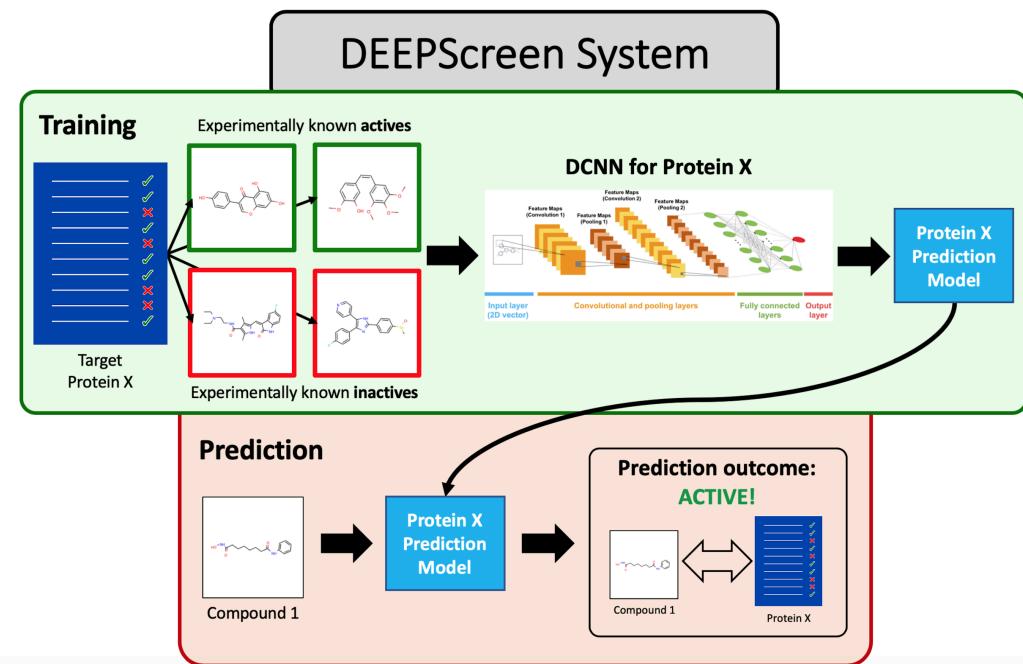


Figura 21. Abstracto gráfico de los modelos *DEEPScreen*. Extraído de Rifaioglu et al. [39] y modificado para mejorar la comprensión. Los compuestos químicos activos e inactivos biológicamente representados como imágenes de las estructuras de Lewis entran al modelo que aprende a diferenciarlos únicamente a partir de las imágenes. Luego se puede utilizar el modelo para predecir imágenes de compuestos novedosos.

Los autores de la publicación ponen a disposición el código utilizado para entrenar sus modelos en un repositorio citado en el artículo. Como punto de partida, se buscó reproducir los resultados reportados, tanto como ejercicio de aprendizaje como con el objetivo de adaptar los modelos a *targets* específicos del laboratorio. Durante este proceso, se identificaron algunas diferencias entre el código disponible y su descripción en el artículo, así como ciertos desafíos técnicos que ofrecieron la oportunidad de explorar enfoques alternativos basados en la propuesta original.

Exploración de enfoques alternativos en la implementación para mejorar el rendimiento de las redes neuronales

Inicialmente, se intentó usar el código original sin modificaciones. Sin embargo, dado que los modelos ya entrenados no estaban disponibles y los hiperparámetros optimizados utilizados para entrenar los modelos no se encontraban especificados, fue necesario entrenar los modelos desde cero. Para ello, se replicó la metodología de ajuste de hiperparámetros por "grid search" mencionada en la publicación, evaluando combinaciones exhaustivas de hiperparámetros (tabla 2). Este proceso resultó computacionalmente intensivo, requiriendo aproximadamente una semana de entrenamiento por modelo con una *GPU* NVIDIA A100.

Para optimizar el entrenamiento, se reimplementó la arquitectura utilizando *PyTorch Lightning* [92], lo que facilitó la integración de *RayTune* [93] como herramienta para la optimización de hiperparámetros. *RayTune*, mediante el algoritmo ASHA (*Asynchronous Successive Halving Algorithm*) [94], combinó un muestreo aleatorio de configuraciones con la terminación anticipada de aquellas subóptimas, reduciendo significativamente el tiempo de ajuste. *RayTune* habilitó la capacidad de parallelizar tanto el entrenamiento como las instancias de predicción, acelerando de manera considerable la etapa de desarrollo. Esta configuración permitió llevar a cabo el entrenamiento en *CPU* dentro de un tiempo razonable gracias a la parallelización, incrementando la eficiencia general del proceso a menos de 2 días por modelo.

Por otro lado, se identificaron diferencias entre la arquitectura descrita en el artículo y la implementada en el repositorio. Mientras que el artículo describe una red con cinco capas convolucionales y una capa completamente conectada, con filtros de 3x3 o 5x5 [39], el código en el repositorio incluye una arquitectura con dos capas completamente conectadas y filtros de 2x2 [103]. Para esta tesis, se decidió emplear la arquitectura disponible en el repositorio, ya que representaba una implementación funcional y accesible.

Evaluación de métricas y redefinición del enfoque de clasificación para mejorar la priorización de compuestos activos

El planteamiento original de *DEEPscreen* aborda el problema del *screening in silico* como una tarea de clasificación binaria, evaluando las predicciones casi únicamente con métricas como el coeficiente de matthews (MCC) y F1. La capa final de los modelos retornaba un vector con dos *logits*, uno para cada categoría (activo/inactivo), clasificando los compuestos según el *logit* más alto. Este esquema, aunque efectivo para clasificación, limitaba su aplicación en tareas de priorización de compuestos.

Para *TrypanoDEEPscreen*, se exploró adicionar un enfoque distinto inspirado en métodos convencionales de *screening* [3, 12, 95], tratando el problema como una tarea de *ranking*. La capa final fue modificada para incluir una función *softmax*, que convierte los *logits* en probabilidades normalizadas [96]. Esto permite ordenar las moléculas según su probabilidad predicha de ser activas, ampliando la utilidad del modelo en contextos de priorización.

Este nuevo enfoque motivó la incorporación de métricas específicas para evaluar tanto la clasificación como la capacidad de priorización del modelo. Entre ellas, se utilizó el área bajo la curva ROC (AUROC) y su fracción inicial (AUROC 15% FPR), que se enfoca en las predicciones con mayor puntuación. Además, se implementaron métricas propias del *screening in silico*, como *BEDROC*, que otorga mayor peso a las primeras posiciones del *ranking* [3]. Estas métricas permitieron una evaluación más completa y adecuada de los modelos en el contexto de priorización.

Estrategia de ensamblado de modelos para prevenir el sobreajuste y aumentar la robustez del modelo

Con el objetivo de prevenir el sobreajuste, se modificó el proceso de selección de modelos durante el entrenamiento de *DEEPscreen*. En la implementación original, se seleccionaba el modelo con el mejor MCC en el conjunto de validación dentro de los 200 *epochs* de entrenamiento predeterminados. Esta metodología es susceptible a sobreajuste, ya que puede seleccionar un modelo que presenta un MCC elevado solo en el *set* de validación, pero no en otros conjuntos de datos. Al analizar las curvas de entrenamiento de estos modelos (Figura S2), se observó que varios *epochs* mostraban resultados igualmente válidos en términos de AUROC en el conjunto de validación. Además, en la mayoría de los casos, a partir del *epoch* 40 aproximadamente, la métrica AUROC alcanzaba un *plateau*, lo que indicaba que continuar el entrenamiento hasta 200 *epochs* no resultaba productivo. Estos hallazgos motivaron la implementación de criterios de selección diferentes buscando obtener mayor robustez en la predicción de los modelos.

Por lo tanto, se implementó una estrategia de [ensamblado](#) por promedio en *TrypanoDEEPscreen*. Se optó por almacenar los cinco mejores modelos (*epochs*) de cada entrenamiento y considerar también distintas configuraciones de hiperparámetros para realizar las predicciones. De esta forma, se dispondría de varios modelos que predicen el mismo conjunto de datos. Los resultados de estas predicciones (probabilidades de que un compuesto sea activo) se promediaron, obteniendo la probabilidad media entre todos los modelos combinados.

Para determinar la cantidad de modelos a ensamblar, se graficó el AUROC 15% en función de la cantidad de modelos ensamblados con respecto a la predicción conjunto de validación de todos los targets juntos (cada uno con su respectivo modelo, pero todos compilados). Se procedió de esta manera para buscar generalizar una cantidad de modelos a ensamblar para todos los targets, en pos de minimizar el sobreajuste en cuanto a la cantidad de modelos ensamblados. En la Figura 22 se puede observar cómo mejora el AUROC 15% al aumentar la cantidad de modelos ensamblados, y luego de un valle entre los primeros 25 modelos, adopta un *plateau* de *performance*. Se optó por elegir a 26 como la cantidad de modelos a ensamblar ya que es el punto inicial de dicho *plateau*.



Figura 22. Impacto del ensamblado de modelos en la métrica AUROC 15%. Se muestra la evolución del rendimiento al aumentar la cantidad de modelos ensamblados. Se observa un plateau de *performance* a partir de 26 modelos ensamblados, lo que motivó la selección de esta cantidad para la estrategia final.

3.3.3. Comparación del desempeño de los modelos desarrollados

La evaluación de los modelos desarrollados se realizó prediciendo el conjunto de testeo con cada uno de los modelos para los distintos targets seleccionados. A partir de estas predicciones, se calcularon las métricas de *performance* (AUROC, AUROC 15, *BEDROC*, precisión y *recall*) para cada *target* de manera independiente. Se promediaron los resultados de las métricas mencionadas, entre los distintos targets para observar el rendimiento general de los modelos. Para establecer un límite inferior de desempeño esperable, se incluyó un predictor aleatorio sesgado (*biased random predictor*). Este modelo generó predicciones positivas y negativas en proporciones equivalentes a las observadas en el conjunto de datos de prueba, y como era de esperar, mostró valores bajos en todas las métricas. Esto permitió contextualizar el rendimiento de los demás modelos al proporcionar una referencia mínima de comparación.

Aunque los promedios generales permiten identificar tendencias en el rendimiento de los modelos, es importante reiterar que los targets son independientes entre sí. Por esta razón, los resultados específicos de cada *target* se presentan en el material suplementario (S3). Estos gráficos desglosados permiten evaluar el comportamiento particular de cada modelo frente a cada *target* sin extrapolar conclusiones generales inapropiadas.

Los modelos de redes neuronales convolucionales tienen una performance similar al baseline de similitud química en términos de métricas de AUROC y AUROC 15%

El análisis global de las métricas promedio entre los targets mostró un desempeño consistente del modelo *baseline*, que presentó los valores promedios más altos en la mayoría de las métricas (Figura 23). Específicamente, obtuvo un AUROC promedio de 0.84 y un AUROC 15 promedio de 0.75. En comparación, *DEEPscreen* y *TrypanoDEEPscreen* lograron AUROC promedio de 0.74 y 0.76, respectivamente, mientras que los valores de AUROC 15 fueron de 0.61 y 0.63. Esto sugiere

que tanto el modelo *DEEPscreen* como su versión modificada *TrypanoDEEPscreen* no lograron superar al modelo *baseline* en términos de clasificación global. Dada la dispersión de los valores de *performance* entre los distintos targets, que son modelos entrenados independientemente, no se puede asegurar que el *baseline* es mejor que los otros modelos significativamente. Sin embargo, observando las *performance* de cada *target* por separado en S3, se detecta cierta tendencia entre los distintos targets, donde la *performance* del *baseline* es mejor o igual que los otros modelos en casi todos los casos.

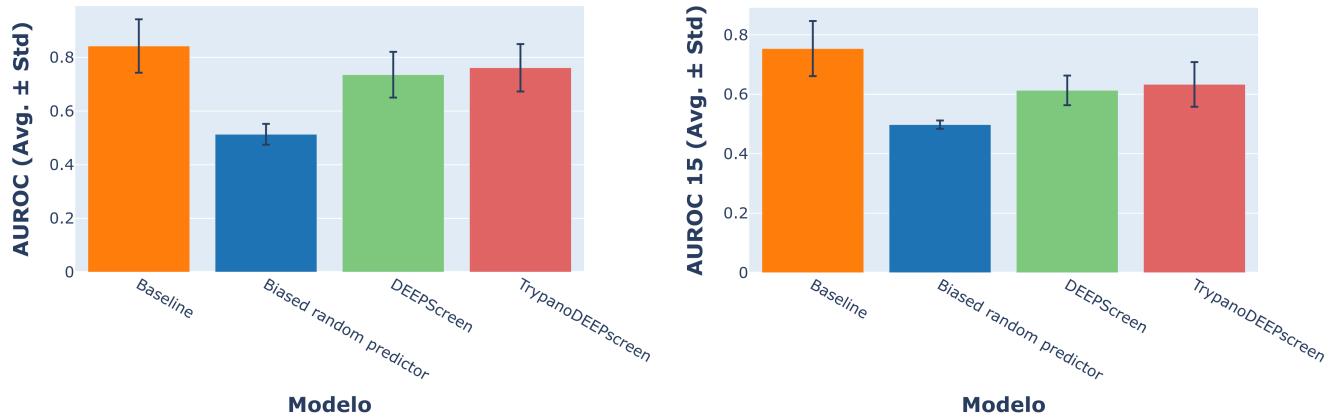


Figura 23. Comparación de desempeño entre los modelos en términos de AUROC y AUROC 15%. Se presentan los valores de las métricas promedio entre los distintos modelos entrenados para los distintos targets, mostrando el rendimiento general en clasificación binaria y priorización de compuestos.

Los modelos de redes neuronales no superaron al baseline en términos de capacidad de priorización de compuestos activos en screening in silico

En el análisis de priorización, la métrica *BEDROC* reflejó un comportamiento similar al AUROC (Figura 24). El modelo *baseline* mostró un promedio de 0.90, superando tanto a *DEEPscreen* como a *TrypanoDEEPscreen*, que obtuvieron valores promedio de 0.77. Aunque los modelos basados en redes neuronales convolucionales alcanzaron un desempeño consistente superior a no utilizar una metodología de priorización (predictor random), el modelo tradicional basado en similitud química resultó al menos igual de efectivo para priorizar compuestos activos en un contexto de *screening in silico*.

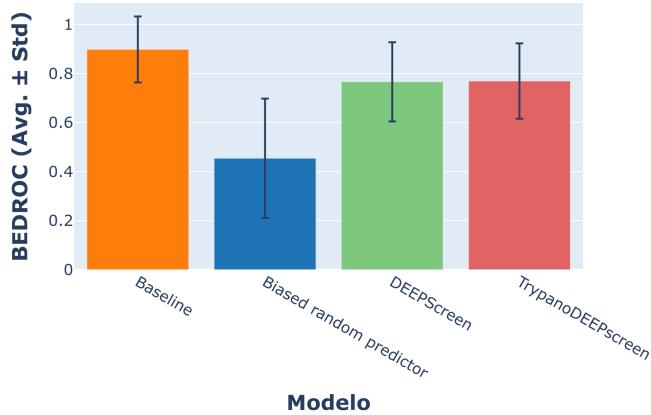


Figura 24. Evaluación de priorización de compuestos utilizando la métrica *BEDROC*. Se comparan los distintos modelos en su capacidad de asignar mayores probabilidades a compuestos activos en los primeros percentiles del *ranking*.

Los modelos de redes neuronales no superan al baseline en como clasificadores

La evaluación de precisión y *recall* arrojó resultados alineados con las otras métricas. *TrypanoDEEPScreen* no demostró un desempeño superior a *DEEPScreen*, con promedios de precisión y *recall* de 0.71 y 0.70, respectivamente, frente a 0.61 y 0.68. Sin embargo, el modelo *baseline* continuó destacándose con valores promedio de precisión de 0.69 y *recall* de 0.77. Estos resultados refuerzan la robustez del enfoque basado en similitud química, que presenta una tendencia a ser superior a los otros enfoques si se observa individualmente cada *target* (Figura S3).

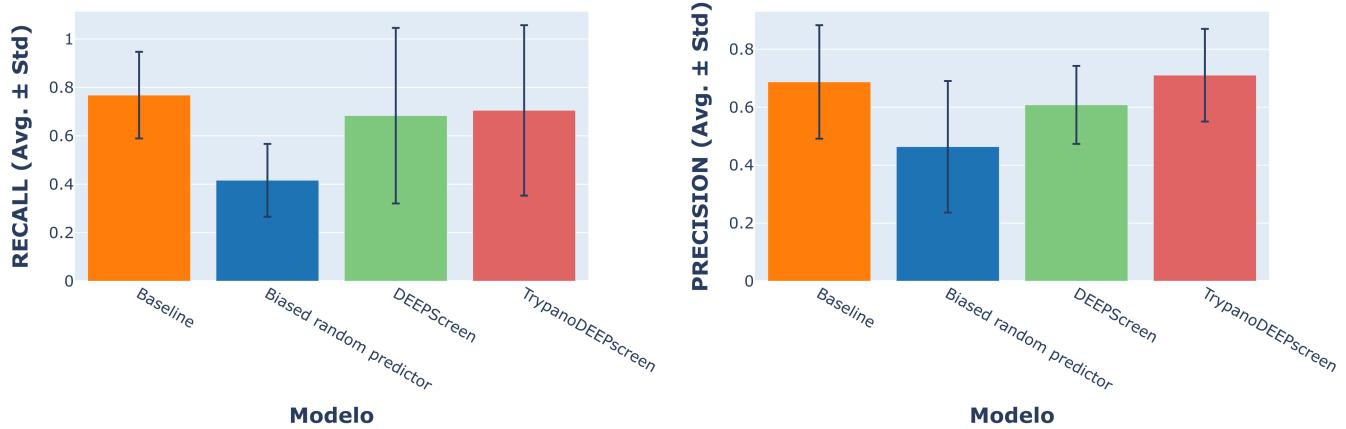


Figura 25. Análisis del desempeño de los modelos en términos de precisión y *recall*. Se presentan los valores promedio obtenidos por cada modelo

Análisis de sobreajuste de DEEPScreen y TrypanoDEEPScreen

Para evaluar la presencia de *sobreajuste* se comparó el rendimiento de los modelos en los conjuntos de entrenamiento, validación y testeо. Un modelo sin sobreajuste debería mostrar métricas de magnitud similar en los tres conjuntos; de este modo, la representación aprendida a partir de los datos de entrenamiento generaliza adecuadamente a datos no vistos. En cambio, un

modelo sobreajustado tiende a alcanzar valores muy altos en entrenamiento, producto de “memorizar” tanto la señal como el ruido de ese conjunto, y fracasa al predecir los ejemplos de validación o testeo.

La Figura 26 ilustra esta comparación mediante dos métricas centradas en el enriquecimiento temprano, criterio especialmente relevante para el *screening in silico*. En ambos casos, *TrypanoDEEPscreen* y *DEEPScreen* exhiben un *sobreajuste* marcado: las métricas en entrenamiento superan ampliamente a las obtenidas en validación y prueba, y en ocasiones se aproximan a 1, indicando la separación perfecta entre compuestos activos e inactivos.

Estos resultados indican que las estrategias de ensamblado y el esquema de entrenamiento alternativo implementados en *TrypanoDEEPscreen* no mitigaron el *sobreajuste* respecto de *DEEPScreen*. Además, el desempeño en el conjunto de validación para ambos modelos es, en la mayoría de los casos, igual o ligeramente superior al de prueba, lo que sugiere un sobreajuste adicional asociado a la optimización de hiperparámetros; no obstante, este efecto es menor que el derivado del proceso de entrenamiento.

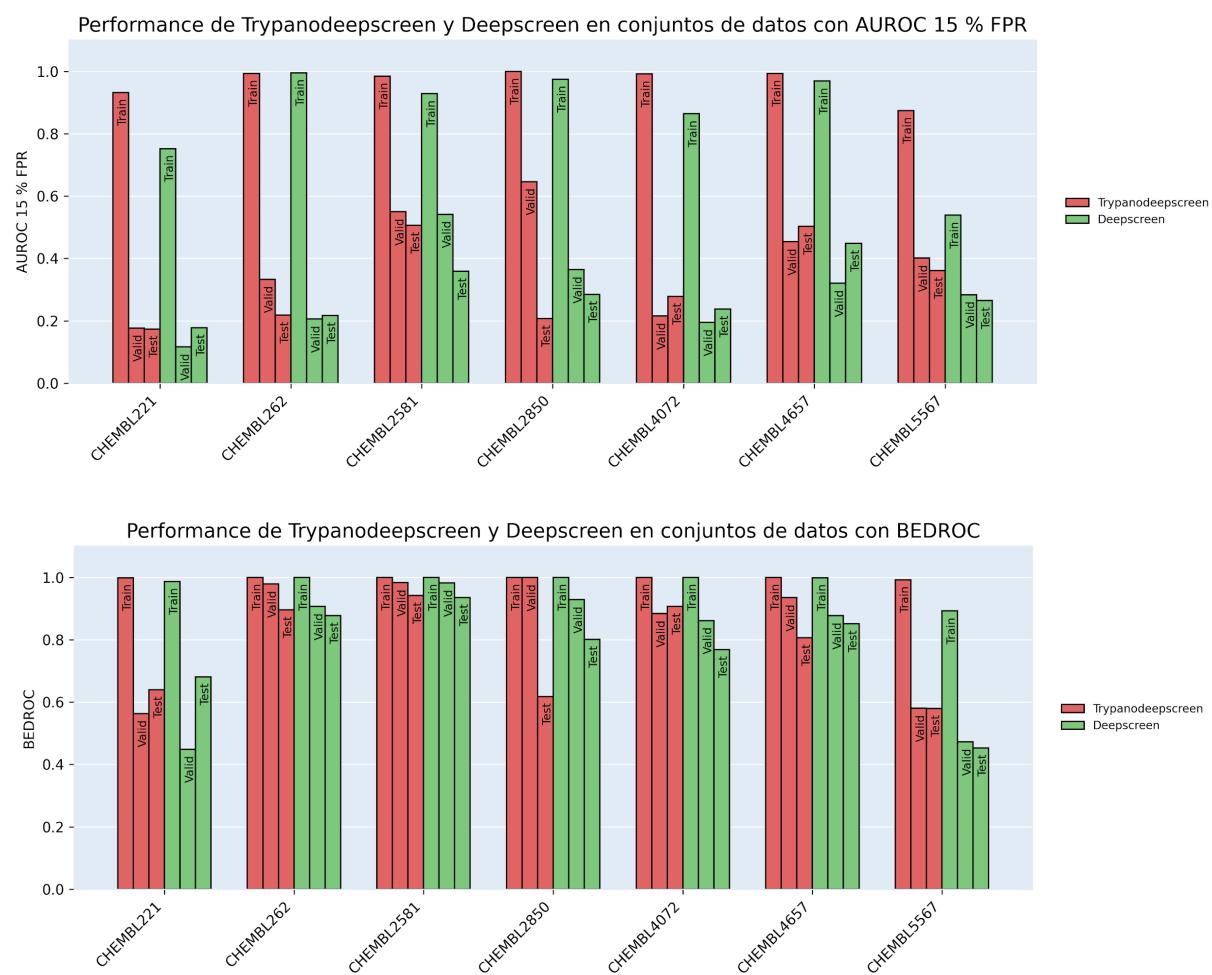


Figura 26. Performance de *TrypanoDEEPscreen* (rojo) y *DEEPScreen* (verde) en los conjuntos de entrenamiento, validación y testeo para los siete blancos analizados. De arriba hacia abajo se muestran los valores de AUROC al 15% de *FPR* y de *BEDROC*. La diferencia entre los resultados de entrenamiento y los de validación/prueba evidencia el sobreajuste presente en ambos modelos.

TrypanoDEEPscreen es 3 órdenes de magnitud más costoso computacionalmente que el baseline

En la Tabla 2 se presenta un *benchmarking* computacional entre el modelo *TrypanoDEEPscreen* y el *baseline*. El mismo se realizó comparando el tiempo requerido para predecir un conjunto de datos utilizando el modelo basado en similitud química (*baseline*) y el modelo basado en redes neuronales (*TrypanoDEEPscreen*). Se evaluó el tiempo total necesario para completar la predicción de los compuestos en el conjunto de testeo para cada *target*, midiendo el tiempo desde la carga de datos hasta la obtención de las predicciones finales, incluyendo en el caso de *TrypanoDEEPscreen* el entrenamiento y ajuste de hiperparámetros.

Los resultados muestran que el modelo *baseline* presenta tiempos de predicción significativamente menores en comparación con *TrypanoDEEPscreen*, con diferencias de hasta tres órdenes de magnitud entre ambos enfoques. Mientras que el modelo basado en similitud química requiere menos de dos minutos en todos los casos, *TrypanoDEEPscreen* demanda entre 465 y 1173 minutos, dependiendo del *target* analizado. Esta diferencia de tiempos resalta la mayor complejidad computacional del modelo basado en aprendizaje profundo y pone en evidencia la necesidad de evaluar la relación entre el costo computacional y la mejora en la *performance* predictiva para aplicaciones de *screening in silico*.

Tabla 2. Comparación del tiempo de cómputo requerido para la predicción entre los modelos *Baseline* y *TrypanoDEEPscreen*. Los valores se expresan en minutos y corresponden al tiempo necesario para predecir los conjuntos de testeo de cada *target*. Para el modelo *Baseline*, el tiempo reportado incluye el procesamiento del conjunto de testeo mediante el algoritmo utilizando como referencia el *dataset* de entrenamiento, realizado con *CPU* en el servidor especificado en la metodología, sin paralelización. En el caso de *TrypanoDEEPscreen*, los tiempos incluyen tanto el entrenamiento del modelo como la posterior predicción, ambos realizados utilizando una *GPU* del servidor CHE descrito en la metodología.

CHEMBL ID	Baseline (min)	TrypanoDEEPscreen (min)
CHEMBL221	0.76	621.57
CHEMBL262	1.81	465.73
CHEMBL2581	1.03	740.88
CHEMBL2850	0.28	560.23
CHEMBL4072	0.49	1013.65
CHEMBL4657	0.53	867.97
CHEMBL5567	0.58	1173.18

El modelo TrypanoDEEPScreen y el baseline tienen criterios diferentes para definir el score de bioactividad para cada compuesto

En la Figura 27 se muestra un diagrama de dispersión en el que cada punto representa un compuesto del conjunto de testeо, posicionado de acuerdo con el puntaje asignado por el *baseline* (eje X) y el de *TrypanoDEEPscreen* (eje Y). Los compuestos están coloreados según su clasificación real (activos en azul e inactivos en verde), lo que permite visualizar zonas de coincidencia (cuando ambos modelos otorgan puntajes similares) y de discrepancia (cuando uno asigna valores altos y el otro bajos). Este comportamiento evidencia que el método basado en similitud química y el modelo de *TrypanoDEEPscreen* emplean criterios diferentes para estimar la probabilidad de bioactividad y, por ende, podrían priorizar de forma distinta los compuestos durante un *screening in silico*.

En la mayoría de los casos, el *baseline* logra separar los compuestos activos e inactivos, lo que se refleja en una distribución diferenciada a lo largo del eje X. En contraste, *TrypanoDEEPscreen* solo muestra una separación visualmente clara en los casos de CHEMBL4657 y CHEMBL2581. Sin embargo, en el cuadrante superior derecho de los resultados de CHEMBL4657, CHEMBL5567, CHEMBL221, CHEMBL2581 y CHEMBL4072, donde se encuentran los compuestos priorizados por ambos modelos, se observa una mayor concentración de compuestos activos en comparación con los inactivos. Esto indica que *TrypanoDEEPscreen* tiende a reducir los falsos positivos del *baseline*, lo que se traduce en un aumento de la especificidad a expensas de una reducción en la sensibilidad (*recall*).

Un comportamiento particular del modelo *baseline* se observa en CHEMBL221 y CHEMBL226. En estos casos, aunque la separación entre compuestos activos e inactivos es clara, el punto de corte óptimo para maximizar la especificidad no se encuentra en 0.5, sino en 0.6. Dado que el *baseline* asigna una puntuación basada en la similitud máxima de cada compuesto con los activos e inactivos del conjunto de entrenamiento, este desplazamiento sugiere que el modelo es capaz de diferenciar compuestos activos e inactivos incluso cuando presentan mayor similitud con los compuestos activos. Específicamente, un compuesto con una puntuación superior a 0.5 indica que es más similar a un compuesto activo que a uno inactivo, pero el *baseline* logra diferenciar entre los más y menos activos.

TrypanoDEEPScreen en algunos casos aislados funciona mejor que el *baseline* para una cantidad reducida de compuestos. En CHEMBL2581, CHEMBL221 y CHEMBL262 hay compuestos activos correctamente clasificados por *TrypanoDEEPScreen*, con *scores* altos, e incorrectamente clasificados por el *baseline*, con *scores* bajos. Nuevamente, el *score* bajo del *baseline* indica alta similitud con un compuesto negativo, en comparación con los compuestos activos. Esta observación, muestra que *TrypanoDEEPScreen* logró dentro de su aprendizaje, encontrar particularidades dentro de los compuestos negativos que no afectan a la actividad del compuesto con la proteína. No obstante, no alcanza dicho aprendizaje para hacer un *screening* efectivo ya que la concentración de falsos positivos es muy alta en los cuadrantes superiores, por lo que en una campaña de *screening* real, estos compuestos que el modelo convolucional logró separar, tenderían a pasar desapercibidos.

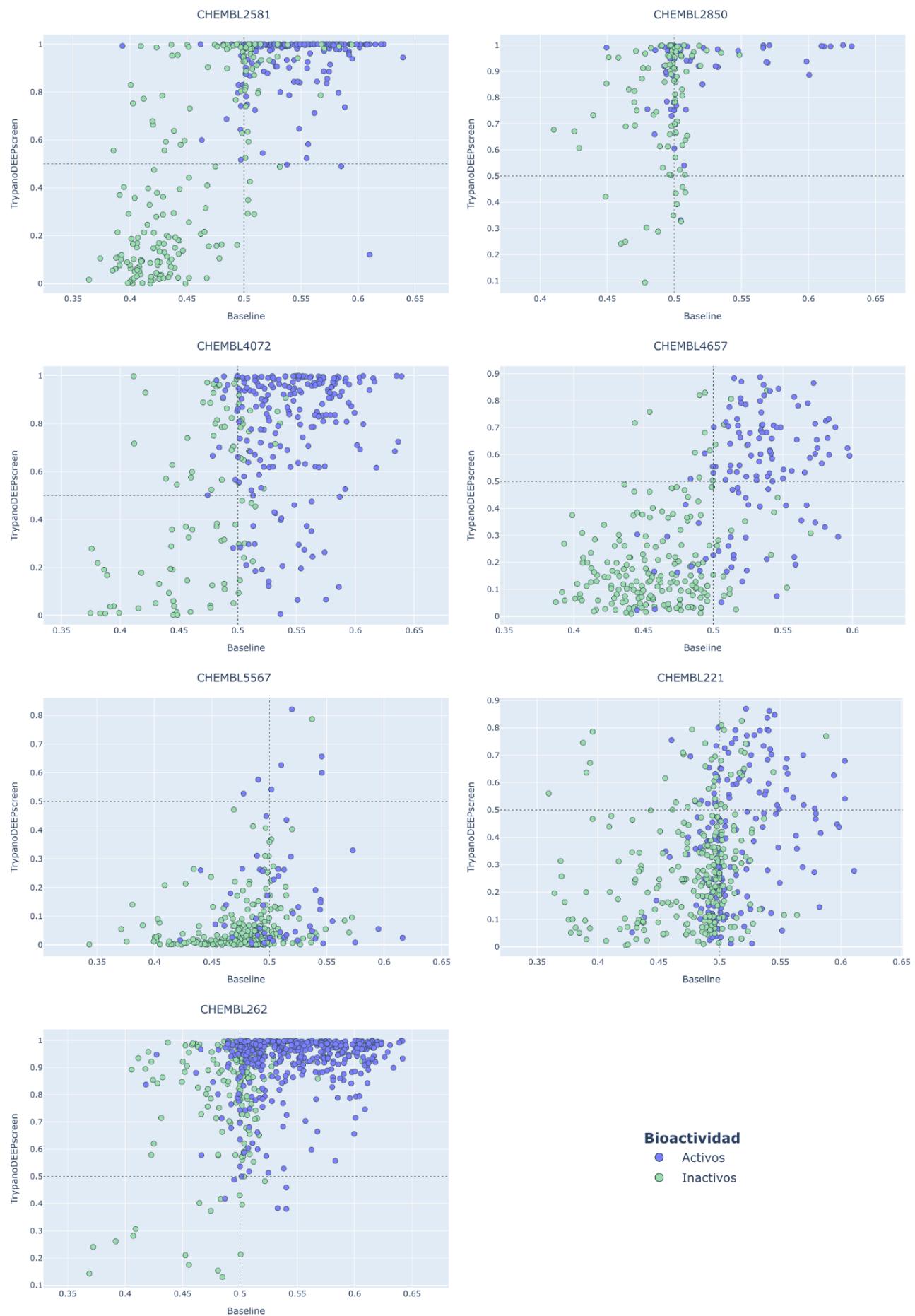


Figura 27. Comparación de los scores predichos por compuestos. Se grafica los Scores escalados entre 0 y 1, de los resultados de la predicción del *set* de testeo por los modelos *TrypanoDEEPscreen* y el modelo *baseline*. Se observa como modelos utilizan distintos criterios para definir la bioactividad de un compuesto predicho.

Discusión de la sección 3.3

Las adaptaciones introducidas en *DEEPscreen*, especialmente la migración del código a *PyTorch Lightning* y la optimización de hiperparámetros con *Ray Tune*, permitieron reducir significativamente el tiempo necesario para entrenar cada modelo. Originalmente, el entrenamiento tomaba alrededor de una semana utilizando una *GPU* y era incapaz de paralelizarse; tras estas mejoras, dicho proceso se redujo a dos días utilizando una *GPU*, o bien a un tiempo equivalente utilizando paralelización sobre 42 hilos de *CPU*. Sin embargo, la ganancia computacional no se tradujo en un aumento significativo del rendimiento predictivo.

El ensamblado no logró corregir el sobreajuste heredado del esquema de entrenamiento. Tanto *DEEPscreen* como *TrypanoDEEPscreen* exhiben métricas cercanas a 1 en el conjunto de entrenamiento, pero presentan una caída marcada en los conjuntos de validación y testeо (véase la Figura 26). Esta diferencia indica que la selección del mejor *epoch* usando el conjunto de validación funciona como un *cherry-picking*. La ausencia de técnicas como *early stopping* o regularización adicional facilita que los modelos memoricen características específicas del conjunto de entrenamiento, efecto que persiste incluso tras promediar los checkpoints de varios modelos con diferentes hiperparámetros en el ensamblado. Adicionalmente, quedaron sin explorar otras variantes arquitectónicas para las redes convolucionales, tales como diferentes tamaños de kernels o estructuras alternativas de convolución, que podrían potencialmente mejorar estos resultados. Por último, aunque a un costo computacional mayor, también se podrían evaluar metodologías alternativas de representación molecular, incluyendo representaciones tridimensionales o superposiciones de estructuras de Lewis en combinación con redes convolucionales.

Reformular el problema como uno de *ranking* permitió evaluar métricas centradas en la detección temprana (AUROC 15% y *BEDROC*), que reflejan de modo más realista las necesidades de un *screening* virtual. Aun así, el modelo *baseline* de similitud química siguió liderando en cuanto a capacidades predictivas en el *set* de testeо. El coste computacional refuerza esta conclusión. El tiempo medio de inferencia del *baseline* oscila entre 0,3 y 1,8 min por *target*, mientras que *TrypanoDEEPscreen* requiere entre 460 y 1170 min, es decir, tres órdenes de magnitud más. En campañas de *screening* sobre bibliotecas extensas, esta diferencia compromete la escalabilidad de la propuesta basada en *CNN*.

Existen, no obstante, indicios de aprendizaje no trivial en las redes convolucionales. En casos puntuales (por ejemplo, CHEMBL2581, CHEMBL221 y CHEMBL262) *TrypanoDEEPscreen* asignó puntuaciones altas a compuestos activos que el *baseline* descartó. Esto sugiere que la red capta señales latentes diferentes de la similitud estructural; aun así, el efecto es marginal y queda enmascarado por la elevada proporción de falsos positivos y el sobreajuste mencionado.

En conjunto, el enfoque basado en similitud química resulta más robusto, eficiente y, en la mayoría de los casos, más preciso que las variantes convolucionales evaluadas. Para que las *CNN* aporten una ventaja competitiva será necesario (i) incorporar criterios de parada temprana y técnicas de regularización que limiten el sobreajuste; (ii) explorar arquitecturas o representaciones moleculares que capturen mejor la diversidad química.

3.4. Uso de los algoritmos con datasets de descubrimiento de antibióticos

Para evaluar el desempeño del modelo *TrypanoDEEPscreen*, se realizó una comparación con los modelos de aprendizaje profundo desarrollados en el estudio de Stokes et al. [37] (*A Deep Learning Approach to Antibiotic Discovery*). En dicho trabajo, los autores generaron datasets de entrenamiento mediante un *screening* experimental de 2.335 compuestos de una biblioteca de fármacos aprobados por la FDA y productos naturales, evaluando su capacidad para inhibir el crecimiento de *Escherichia coli*. Con estos datos, entrenaron modelos *Chemprop* de redes neuronales profundas basadas en *directed message passing neural networks* para predecir capacidad de inhibición bacteriana de nuevas moléculas.

El modelo resultante entrenado fue un ensamble de 20 modelos de redes neuronales *Chemprop*, que utilizaban los *embeddings* de grafos junto con información adicional de propiedades fisicoquímicas de las moléculas. Este modelo se utilizó para predecir la actividad antibacteriana de la biblioteca de compuestos *Drug Repurposing Hub*, que contiene 6.111 moléculas en distintas fases de investigación para enfermedades humanas. Se predijeron puntuaciones para cada compuesto y posteriormente, se seleccionaron los 99 compuestos con puntuación más alta y los 63 con puntuación más baja, para ser ensayados experimentalmente. A partir de estos ensayos, se identificó a la *halicina* (un thiadiazol) como un antibiótico de amplio espectro con actividad en modelos murinos de infección. Un aspecto relevante de la *halicina* es su divergencia estructural con respecto a los compuestos del conjunto de entrenamiento. Su similitud de Tanimoto (sobre *fingerprints Morgan*) con la molécula más cercana en dicho conjunto fue de 0.37, y con el antibiótico más cercano, metronidazol, fue de 0.21. Cabe resaltar que, a diferencia de los modelos trabajados en esta tesis, estos modelos fueron entrenados con compuestos químicos activos o inactivos (antibióticos o inocuos) contra una célula entera (*E. coli*) en lugar de una proteína aislada.

3.4.1. El set de entrenamiento es altamente desbalanceado y muestra un sesgo de similitud química en los compuestos activos

Para realizar la comparación de los modelos *Chemprop* de Stokes et al. con *TrypanoDEEPscreen*, se utilizó el mismo conjunto de datos de entrenamiento empleado en el estudio original (2.335 compuestos). El mismo no se encontraba particionado, por lo que se llevó a cabo una partición aleatoria del *dataset*, separándolo en subconjuntos de entrenamiento, validación y testeо (64-16-20). La Figura 28A muestra la distribución de compuestos en estos conjuntos de datos, reflejando el balance entre compuestos activos e inactivos en el conjunto completo. Se observa que el ratio entre activos e inactivos es muy bajo, indicando que la proporción de compuestos activos es reducida. La Figura 28B presenta un análisis de similitud química, evidenciando que los compuestos activos comparten cierto grado de similitud estructural entre sí, y los inactivos no.

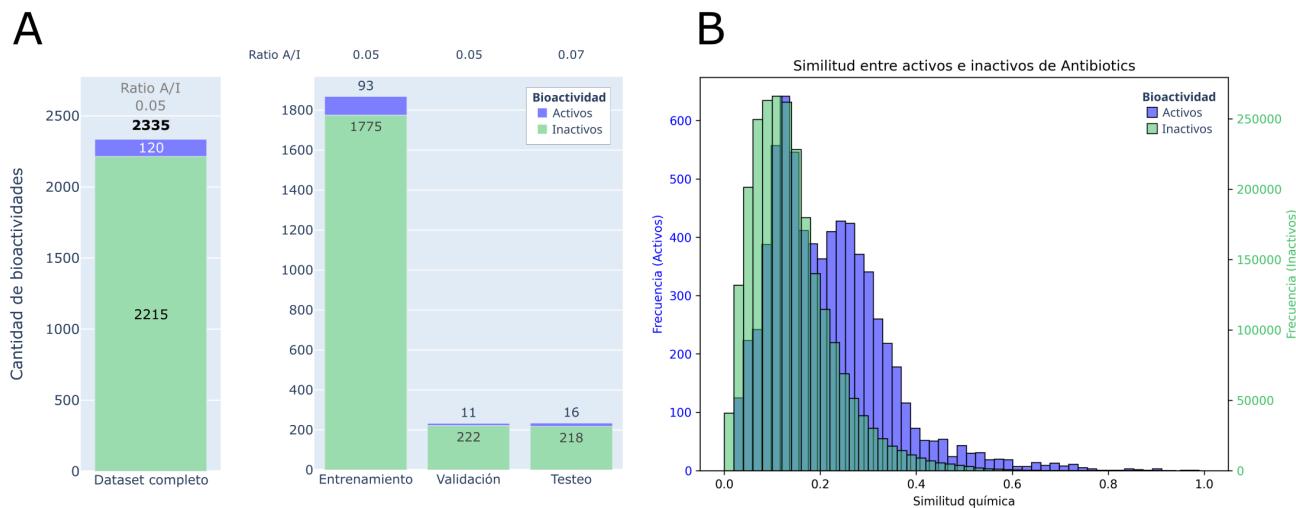


Figura 28. (A) Distribución de compuestos en el conjunto de datos utilizado para el entrenamiento del modelo. Se muestra la proporción de compuestos activos e inactivos en los subconjuntos de entrenamiento, validación y testeо. (B) Distribución de similitud química entre compuestos activos e inactivos.

3.4.2. En una campaña de screening *in silico* con experimentación

TrypanoDEEPscreen presenta resultados similares a los modelos *Chemprop* y rankea mejor a la halicina.

Posteriormente, se evaluó el rendimiento de los modelos entrenados utilizando la partición de testeо. La Figura 29A compara el desempeño de *DEEPscreen* y *TrypanoDEEPscreen* contra el *baseline* en la partición de testeо, mostrando los tres modelos un rendimiento similar en métricas de AUROC, AUROC 15% y BEDROC. Se observa también que el modelo *TrypanoDEEPscreen* mejora la capacidad de predicción en comparación con el *baseline* en términos de *precisión*, en cambio este último es superior en términos de *recall* (sensibilidad). En la Figura 29B se representa la correlación entre los valores de predicción obtenidos por *TrypanoDEEPscreen* y el modelo *baseline* para los compuestos de la partición de testeо; se muestra una tendencia en la que *TrypanoDEEPscreen* asigna mayores puntajes a compuestos que efectivamente presentan actividad inhibitoria. Por otro lado el *baseline* presenta varios falsos positivos.

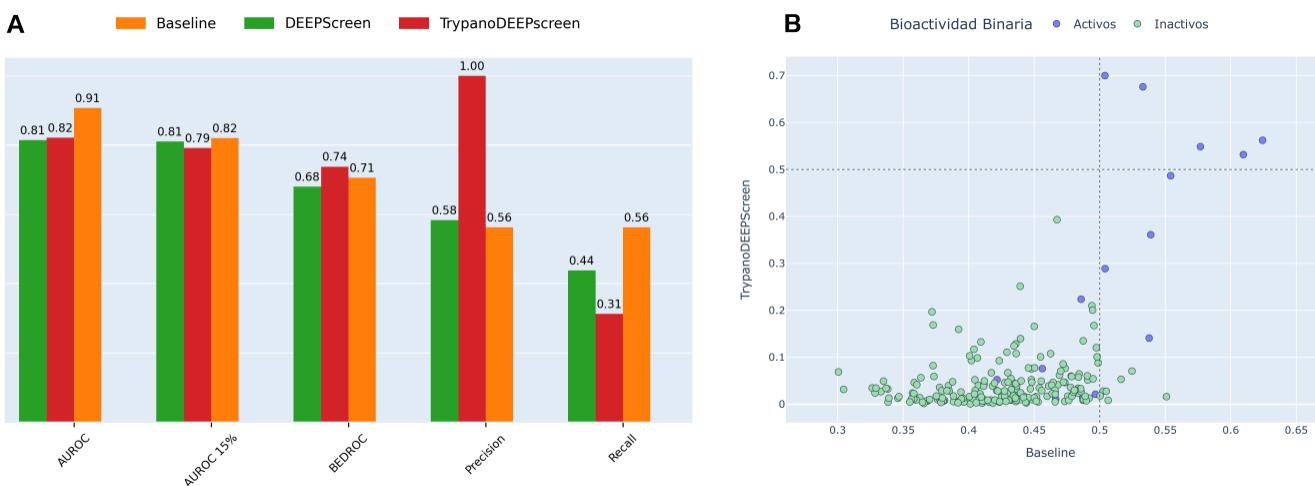


Figura 29. (A) Comparación del desempeño de *TrypanoDEEPscreen* y un *baseline* en el conjunto de prueba. (B) Correlación entre las puntuaciones de predicción del conjunto de testeо obtenidas por *TrypanoDEEPscreen* y el *baseline*.

A continuación, se aplicó *TrypanoDEEPscreen* para predecir la actividad antibacteriana en la biblioteca *Drug Repurposing Hub* (6.111 compuestos), siguiendo la misma metodología utilizada en el estudio de Stokes *et al.* Se generaron puntuaciones para hacer un *ranking* de los compuestos y se compararon los resultados con los de los modelos empleados en el estudio original. En este análisis, se observó que *halicina* fue clasificada en la posición 62 por *TrypanoDEEPscreen*, mientras que el modelo de Stokes *et al.* la ubicó en la posición 89. Además, la molécula se situó en las posiciones 1427 y 1693 en *DEEPscreen* y el *baseline*, respectivamente, lo que refleja las diferencias en la priorización de compuestos divergentes estructuralmente del *set* de entrenamiento, según el modelo empleado.

La Figura 30 compara a los modelos sobre los compuestos que fueron ensayados del *Drug Repurposing Hub*. En la Figura 30A se observa que *TrypanoDEEPscreen* y *Chemprop* superan al resto de los modelos en la métrica *BEDROC*, diseñada específicamente para evaluar el enriquecimiento temprano de compuestos activos. La Figura 30B muestra que *TrypanoDEEPscreen* asigna a *halicina* una puntuación alta, superior a la del *baseline*, en el que la molécula no se distingue de los compuestos inactivos. Se evidencia que *TrypanoDEEPscreen* tiene una menor sensibilidad (*recall*) que el *baseline*, aunque este último muestra menor precisión, ya que la molécula con mayor puntuación en el *baseline* está mal clasificada.

La Figura 30C compara los modelos *TrypanoDEEPscreen* y *Chemprop* únicamente sobre los compuestos ensayados, destacando discrepancias en la priorización, en particular para los compuestos inactivos. Finalmente, la Figura 30D extiende este análisis al resto de la biblioteca que no fue ensayada en el estudio de Stokes *et al.*, revelando que *TrypanoDEEPscreen* asigna puntuaciones elevadas a compuestos que el modelo de Stokes valora menos, lo que sugiere posibles candidatos antibióticos no priorizados previamente, ya que *TrypanoDEEPscreen* parecería ser preciso según las otras evaluaciones.

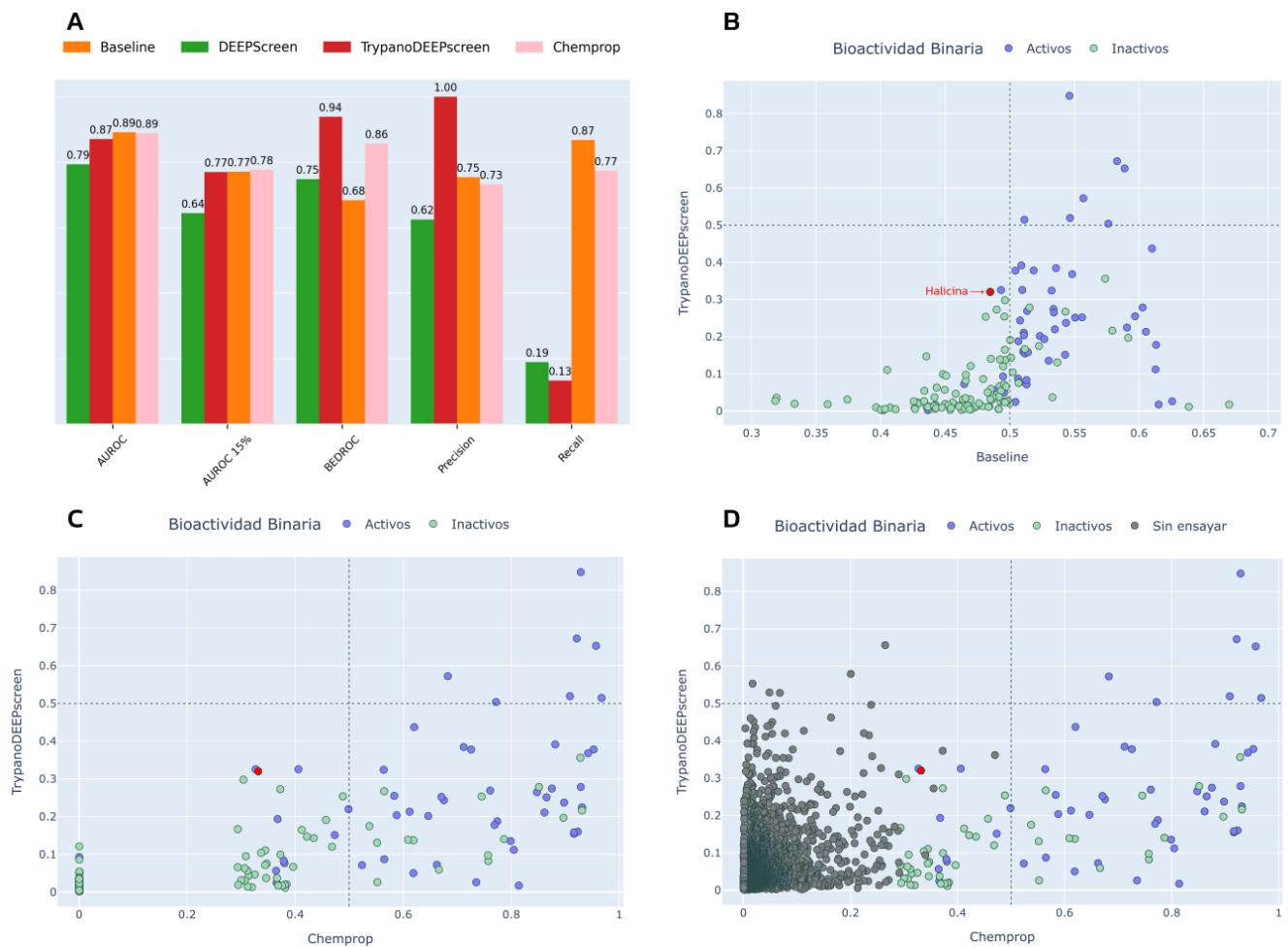


Figura 30. (A) Evaluación de los modelos sobre el conjunto de datos de descubrimiento de antibióticos evaluado experimentalmente en Stokes et al. (B) Comparación de puntuaciones de predicción entre *TrypanoDEEPscreen* y el *baseline* para los compuestos ensayados experimentalmente. (C) Comparación de puntuaciones de predicción entre *TrypanoDEEPscreen* y el modelo de Stokes et al. (D) Comparación de puntuaciones incluyendo compuestos no ensayados experimentalmente en el estudio original.

Discusión de la sección 3.4

La evaluación comparativa del modelo *TrypanoDEEPscreen* frente a los modelos Chemprop desarrollados por Stokes *et al.* reveló que, a pesar de utilizar una arquitectura más sencilla, *TrypanoDEEPscreen* alcanzó resultados comparables en métricas clave como AUROC y BEDROC. Este resultado es destacable debido a la mayor sofisticación de Chemprop, basado en redes neuronales profundas con *directed message passing neural networks*, que ganó relevancia en la bibliografía recientemente como una forma interesante de representar a las moléculas químicas [10, 97–99].

Un punto central de la comparación fue la identificación de la *halicina*, una molécula estructuralmente diferente al conjunto de entrenamiento utilizado. La *halicina* representa un caso emblemático del estudio original por su diversidad química y su actividad antibiótica confirmada experimentalmente. Tanto Chemprop como *TrypanoDEEPscreen* identificaron esta molécula entre las mejor clasificadas, con *TrypanoDEEPscreen* ubicándola incluso más alto en el *ranking*.

Otro punto destacable es como el *baseline* solo logró superar a los otros modelos en términos de sensibilidad, sugiriendo que al menos en estos *set* de datos, una clasificación por similitud química no es suficiente para obtener resultados de relevancia como la *halicina*.

3.5. Aplicación de los modelos para el descubrimiento de tratamientos contra *Trypanosoma cruzi*

Durante este trabajo se desarrollaron y evaluaron los modelos *TrypanoDEEPscreen*, junto con el *baseline*, en contextos de proteínas aisladas y células completas (bacterias). El objetivo final de estos desarrollos, es modelar la actividad de compuestos químicos de bibliotecas de compuestos de actividad desconocida para obtener *hits* putativos para ensayar experimentalmente con una mayor tasa de éxito. Especialmente se busca realizar este *screening* virtual para reposicionar compuestos activos contra *T. cruzi*. Por eso, se planteó un análisis de los modelos de proteínas aisladas ya entrenados, con bibliotecas de compuestos que no fueron validados experimentalmente con esos *targets*.

Evaluación de los modelos con compuestos tripanocidas

La *Chagas-Box* [67] es un conjunto específico de moléculas activas contra *Trypanosoma cruzi* generado mediante *screening* fenotípico de alto rendimiento (HTS) realizado por GlaxoSmithKline (GSK). Este *screening* consistió en la evaluación fenotípica de aproximadamente 1.8 millones de compuestos contra células infectadas con *T. cruzi*, seguido de ensayos confirmatorios para validar actividad, especificidad y toxicidad. Los compuestos (activos) finalmente incluidos en la *Chagas-Box* fueron seleccionados considerando estrictamente su potencia, baja citotoxicidad y propiedades fisicoquímicas favorables. Adicionalmente, mediante análisis bioinformáticos complementarios, se identificaron potenciales targets moleculares involucrados en la actividad observada, incluyendo quinasas, proteasas y citocromos, lo que permitió sugerir posibles mecanismos de acción para estos compuestos [67].

Los modelos de proteínas aisladas desarrollados en esta tesina fueron evaluados en la tarea de predecir la actividad biológica de compuestos químicos pertenecientes a la *Chagas-Box* [67]. El objetivo central de este análisis fue explorar la posibilidad de identificar nuevos targets putativos reposicionables; si algún compuesto de la *Chagas-Box* presentaba un *score* predictivo alto en alguno de los modelos entrenados, esto indicaría la relevancia potencial del *target* correspondiente en *T. cruzi*, suponiendo la existencia de un ortólogo de dicha proteína en el parásito.

Las predicciones realizadas por los modelos *TrypanoDEEPscreen* y el *baseline* de similitud química fueron representadas en gráficos que comparan los *scores* obtenidos en ambos enfoques (Figura 31). Los resultados muestran un comportamiento heterogéneo de los modelos entrenados según el *target* evaluado. A continuación, se describen algunos casos representativos, mientras que las figuras para todos los targets evaluados se encuentran disponibles en material complementario (Figuras S5).

Para los targets CHEMBL4657 y CHEMBL2850 (Figura 31A y C), ambos modelos asignaron puntajes altos a una gran cantidad de compuestos, indicando una limitada especificidad. En estos

casos, la elevada proporción de compuestos rankeados con alta actividad potencial tanto por *TrypanoDEEPscreen* como por el modelo *baseline*, evidencia una reducida capacidad de discriminación de ambos métodos. 6 de los 7 modelos presentaron este comportamiento errático.

En contraste, en el caso del *target* CHEMBL5567 (Figura 31B), se observó un mejor rendimiento relativo de *TrypanoDEEPscreen*, que asignó puntuaciones elevadas solo a un número reducido de compuestos, como es deseable en campañas de *screening* virtual. Este resultado sugiere que en ciertos casos específicos, el modelo basado en redes neuronales convolucionales puede capturar patrones más complejos que el modelo basado en similitud química, aunque estos casos parecen ser excepcionales y no la regla. El *baseline* por el contrario, asignó un puntaje mayor a 0.5 a una gran cantidad de compuestos, sugiriendo un mal desempeño del modelo. Sin embargo, si observamos también se puede observar que asignó con un alto puntaje relativo a unos pocos compuestos, aunque estos son compuestos diferentes que los predichos con alto *score* en el caso de *TrypanoDEEPscreen*.

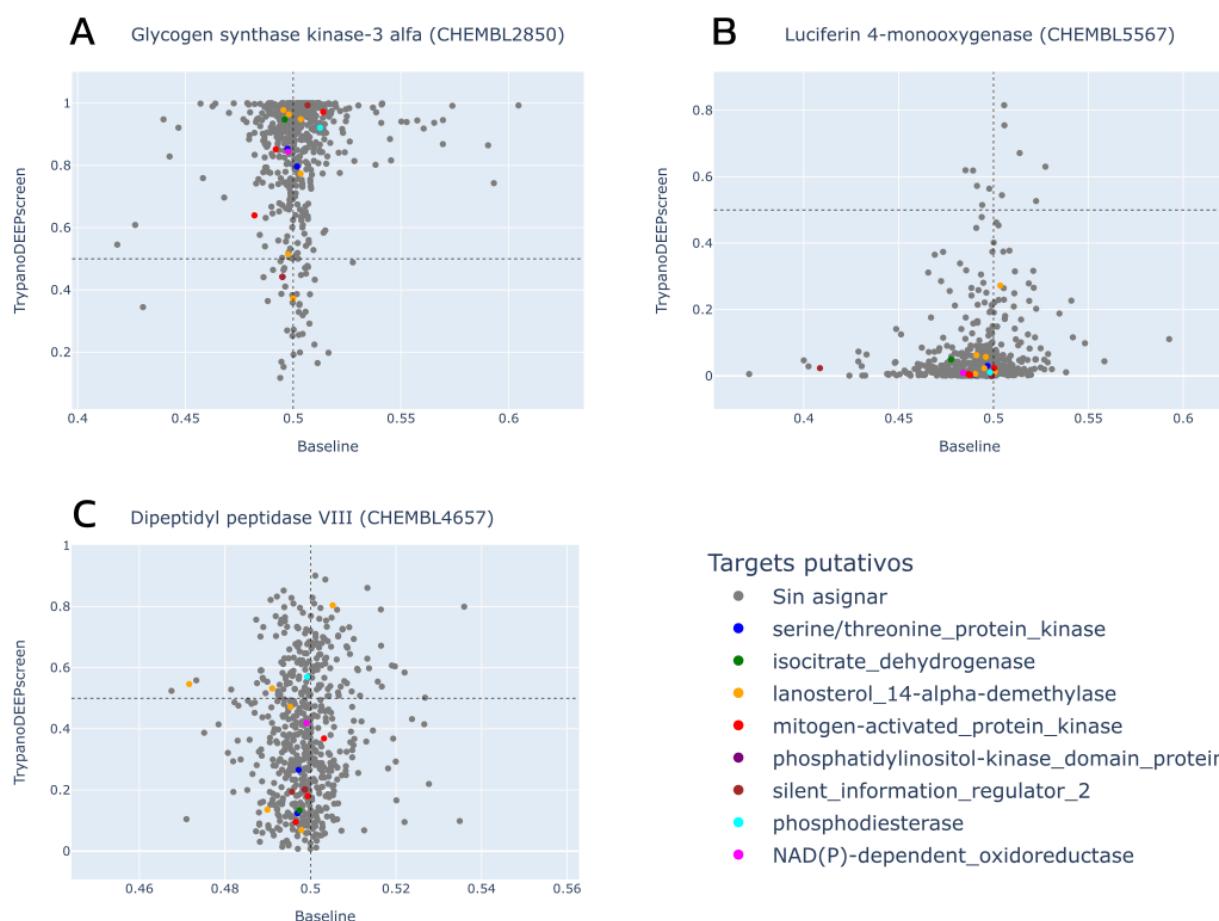


Figura 31. Predicciones comparativas de actividad biológica en proteínas aisladas realizadas por los modelos *TrypanoDEEPscreen* y *Baseline* de similitud química para compuestos de la *Chagas-Box*. (A) Glycogen synthase kinase-3 alfa (CHEMBL2850); (B) Luciferin 4-monooxygenase (CHEMBL5567); (C) Dipeptidyl peptidase VIII (CHEMBL4657). Los compuestos están coloreados según los targets putativos identificados experimentalmente en el estudio original

Discusión de la sección 3.5

La evaluación de los modelos desarrollados frente a un conjunto completamente desconocido de compuestos, como es la *Chagas-Box*, reveló comportamientos diversos y, en algunos casos, limitaciones significativas. En términos generales, es esperable que modelos efectivos para *screening* retornen pocos compuestos como potencialmente activos, ya que típicamente la bioactividad es una propiedad química rara. Sin embargo, en la mayoría de los casos analizados, ambos modelos, *TrypanoDEEPscreen* y el *baseline* de similitud química, tendieron a asignar puntuaciones altas a muchos compuestos, lo cual no es deseable en campañas reales de *screening* debido a la limitada especificidad observada.

Este fenómeno se observó claramente en los casos emblemáticos de CHEMBL2850 y CHEMBL4657 (y se observa lo mismo para el resto de los targets en la figura S5), donde ambos métodos indicaron alta actividad potencial para una gran cantidad de compuestos. Estos resultados sugieren que, pese a las métricas positivas obtenidas en las particiones de entrenamiento y validación, los modelos pueden estar influenciados por sesgos derivados de la diversidad química de los conjuntos de entrenamiento, afectando negativamente su capacidad para generalizar adecuadamente cuando enfrentan sets de datos con diferentes composiciones químicas.

Sin embargo, el análisis del *target* CHEMBL5567 mostró una mejor capacidad discriminativa del modelo *TrypanoDEEPscreen*, sugiriendo que bajo ciertas condiciones particulares de composición química, las redes neuronales convolucionales podrían capturar patrones más complejos y relevantes para el *screening*. Este resultado indica que, aunque no generalizable a todos los casos, en escenarios específicos los modelos convolucionales tienen potencial para ser una herramienta efectiva, destacando la importancia de evaluar cuidadosamente la aplicabilidad del método caso por caso.

4. Conclusiones

En este trabajo se desarrollaron y evaluaron modelos computacionales para la predicción de bioactividad química, con foco en el reposicionamiento de compuestos.

Se implementó un modelo de referencia (*baseline*) basado en similitud química y se adaptó el método *DEEPscreen* para generar el modelo *TrypanoDEEPscreen*, empleando representaciones bidimensionales de estructuras químicas y redes neuronales convolucionales.

Se filtró y curó la base ChEMBL para obtener conjuntos binarios de actividad/inactividad contra proteínas aisladas, seleccionando *targets* con cantidades y balances de datos adecuados. Se utilizaron estrategias de partición de datos específicas para datos químicos (*scaffold splitting*) y se utilizaron para entrenar y validar los modelos.

Mediante el uso del modelo de referencia (*baseline*) demostramos que en promedio, el modelo *baseline* alcanza métricas iguales o superiores a las de los modelos convolucionales en

clasificación y priorización de compuestos activos, con un costo computacional significativamente menor. Para algunos conjuntos de datos *TrypanoDEEPscreen* clasificó correctamente compuestos activos no identificados por el *baseline*, incluyendo a la *halicina*.

TrypanoDEEPscreen obtuvo resultados comparables a modelos con arquitecturas diferentes (Chemprop) frente a la evaluación con un conjunto de compuestos antibacterianos. Al aplicar los modelos a la biblioteca *Chagas-Box*, ambos métodos (*baseline* y *TrypanoDEEPscreen*) mostraron limitaciones en la especificidad.

En conjunto, se estableció un punto de partida metodológico para la evaluación comparativa de enfoques basados en similitud química y en redes convolucionales, y se generó un repositorio reproducible con los modelos y *pipelines* utilizados.

5. Perspectivas futuras

La estrategia empleada en este trabajo evidenció un problema recurrente de sobreajuste en los modelos convolucionales desarrollados. Resulta prioritario incorporar técnicas como el *early stopping*, que detienen tempranamente el entrenamiento basado en el rendimiento del conjunto de validación, para evitar la memorización de los datos de entrenamiento y favorecer una generalización efectiva.

Por otra parte, la representación utilizada en este trabajo, basada en imágenes bidimensionales de estructuras de Lewis, aunque poco convencional, mostró indicios interesantes de aprendizaje. Resultaría valioso explorar en mayor profundidad otras formas más completas y sofisticadas de representar moléculas químicas. Entre estas, se destacan representaciones tridimensionales que capturan información espacial adicional, o representaciones basadas en grafos que permiten incorporar de forma explícita relaciones estructurales complejas.

En este sentido, las técnicas recientes basadas en redes neuronales convolucionales de grafos, como las *Directed Message Passing Neural Networks* (DMPNN) implementadas en Chemprop, surgen como alternativas particularmente prometedoras. Estos modelos tienen la capacidad de extraer automáticamente características moleculares relevantes directamente desde la estructura química, prescindiendo del uso de *fingerprints* tradicionales. Dicha capacidad para generar representaciones latentes directamente a partir de los datos crudos podría incrementar significativamente la precisión y flexibilidad de los modelos en distintos contextos, lo que amerita ser considerado en futuras investigaciones.

Además, un enfoque centrado en los datos (*data-centric*), priorizando la curación, selección y preparación de conjuntos de datos más robustos y representativos, aparece como esencial. Es evidente que gran parte del potencial de mejora radica en la calidad y diversidad de los datos empleados, destacando la importancia de dirigir esfuerzos hacia el enriquecimiento y balance de los conjuntos utilizados.

Finalmente, este trabajo permitió comprender con mayor claridad la relevancia de la priorización de targets moleculares desde el punto de vista del aprendizaje automático. Se observó que no todos los targets presentan datos suficientes o adecuados para justificar la aplicación de modelos complejos. Futuras investigaciones deberían considerar un análisis previo detallado para determinar qué blancos proteicos son apropiados para la implementación efectiva de técnicas avanzadas de *Machine Learning*, maximizando así el rendimiento y el impacto real de los modelos generados.

6. Métodos

Para garantizar reproducibilidad, todos los scripts, *notebooks* y sets de datos utilizados para el desarrollo de este trabajo, se encuentran disponibles en un repositorio en *github* de la tesina (https://github.com/sebastianjinich/trypnodeepscreen_dev). Además se encuentran disponibles implementaciones empaquetadas en contenedores *Docker* para entrenar y predecir modelos, tanto para *TrypanoDEEPScreen*, como para el *baseline de similitud química*.

6.1. Cómputo y principales librerías utilizadas

Todo el trabajo de esta tesina se realizó utilizando el lenguaje de programación *Python* en versiones desde 3.8 hasta la 3.12. Se utilizó la librería *Pandas* (2.0.3), para todo el trabajo relacionado con el preprocesamiento, filtrado y curado de datos. El trabajo vinculado con métricas específicas de *machine learning* fue realizado con la librería *Sklearn* (1.3.2). Principalmente, se utilizó *RDkit* (2023.9.4) para el manejo de moléculas químicas. Se utilizó *Plot.ly* para todas las visualizaciones. El funcionamiento de las redes neuronales fue realizado con *PyTorch Lighting* (2.1.4), y el entrenamiento fue llevado a cabo con *RayTune* (2.10.0).

El cómputo del trabajo se realizó en servidores privados del laboratorio, específicamente en un servidor Dell PowerEdge R730 con dos procesadores Intel Xeon E5-2650 v4 de 2,2 GHz, cada uno con 12 núcleos físicos y 24 hilos lógicos (sumando un total de 24 núcleos y 48 hilos), y 512 GB de RAM. Además, se empleó el servidor CHE de la Universidad Federal de Río de Janeiro, equipado con una *GPUNVIDIA Tesla A100*.

6.2. Construcción de datasets

6.2.1. Filtrado y curado de datos de ChEMBL

Durante el proceso de filtrado y curado de datos, se obtuvieron los datos de la base de datos ChEMBL versión 34 a través de una *query* a su base de datos de MySQL (material suplementario S1). Los datos extraídos incluían información sobre ensayos de compuestos químicos, su estructura y bioactividad con sus correspondientes targets. La *query* extrae exclusivamente aquellos ensayos donde el *target* era una proteína única (*Single Protein*) y que correspondían a ensayos de tipo "*binding*" (B), conservando únicamente las moléculas de más de 3 átomos distintos al H (para descartar sales). También elimina los datos donde el organismo no está especificado y aquellos en los que la bioactividad no está reportada. Así se obtiene un conjunto de datos con 2.737.434 bioactividades.

A continuación todo el proceso de filtrado y curado de datos, se realizó en un notebook de Jupyter, usando Python y la librería pandas. Dicho notebook se encuentra disponible en el repositorio de la tesina.

Se excluyeron ensayos cuyas unidades de medida eran porcentajes, ya que no eran fácilmente interpretables o estandarizadas, para enfocarse en aquellos con unidades consistentes, como nM,

μM o equivalentes (obteniéndose 2.036.403 bioactividades). Luego se procedió a seleccionar únicamente los ensayos de tipo IC₅₀, EC₅₀, AC₅₀, Ki, Kd y Potency, ya que son fácilmente interpretables y correspondían a un gran porcentaje de las bioactividades disponibles. Se obtuvieron 1.875.897 bioactividades. Se realizó una conversión de unidades “no estándar” a una unidad común (nM) para homogeneizar las medidas de bioactividad, descartando aquellas unidades que no eran fácilmente convertibles o eran pocos casos aislados dentro del *dataset* que requerían una conversión manual dato a dato. Tras esta conversión se obtuvieron 1.874.312 bioactividades.

El siguiente paso fue la deduplicación de los datos, seleccionando la mediana de bioactividad cuando existían múltiples mediciones para el mismo par compuesto-target del mismo tipo de ensayo (Ki, Kd, IC₅₀, etc.). Se redujeron las bioactividades a 1.588.419. Luego, se realizó una binarización de las bioactividades en función de criterios establecidos para definir actividad e inactividad de Urán Landaburu *et al.* [100] siendo 20.000 nM la máxima concentración para considerar un compuesto activo y 100.000 nM la mínima concentración para considerar un compuesto inactivo. Se descartaron los datos que presentaban una bioactividad intermedia, ya que no podían clasificarse definitivamente como activos o inactivos, llegando a 1.339.104 bioactividades.

Ya obtenida la bioactividad binaria (actividad/inactividad) de cada par compuesto-target, se procedió a eliminar los duplicados remanentes, con un enfoque basado en los tipos de ensayos. Esto es, eliminar todos los duplicados compuesto-target que presentan discordancia respecto a la bioactividad binaria, excepto que alguno de los datos discordantes provenga de ensayos de disociación (Kd) o de inhibición (Ki). En dicho caso se priorizó estos ensayos como datos fidedignos, ya que son características intrínsecas de las proteínas independientes de las condiciones de ensayo. Aquellos pares compuesto-target duplicados que no eran discordantes en la bioactividad binaria, fueron unificados, conservando arbitrariamente el primero de los datos. Es así cómo se alcanzaron los 1.285.627 *datapoints* de bioactividades con los que se procedió a seleccionar targets.

6.2.2. Partición de datasets

Para el particionado de los datasets se utilizó el *Scaffold Splitter* de DeepChem [48], que realiza particiones de datos estipuladas de forma automatizada.

Se realizó una partición de 64%, 16% y 20% para los sets de entrenamiento, validación y testeо respectivamente. Esto es equivalente a realizar una partición tradicional de 80-20% para entrenamiento-testeo, y 80-20% al *set* de entrenamiento, para así obtener el *set* de validación. Al realizar 64-16-20 directamente con el scaffold splitter, se maximiza la independencia en similitud química entre los distintos conjuntos de datos.

6.3. Baseline basado en similitud química

El algoritmo *baseline* desarrollado para este trabajo se basa en la similitud química entre compuestos, bajo la premisa de que moléculas estructuralmente similares tienen mayor probabilidad de compartir bioactividad. Aunque este algoritmo desarrollado tiene puntos en común con los conceptos generales del trabajo de Safizadeh *et al.* [60], el diseño y ajuste final fueron desarrollados independientemente.

6.3.1. Desarrollo y selección del modelo

El desarrollo del modelo involucró la implementación de un esquema de búsqueda sistemática (*grid search*) para optimizar los componentes principales: el método de generación de *fingerprints*, el coeficiente de similitud y el método de cálculo del resultado. Se evaluaron múltiples combinaciones para identificar aquella que ofreciera el mejor desempeño en la predicción de bioactividad en los conjuntos de validación.

1. **Generación de *fingerprints*:** Se probaron diferentes tipos de *fingerprints*, incluyendo ASP (*All-Shortest Path*), LSTAR, RAD2D y RDKit. ASP fue seleccionado como el más adecuado debido a su desempeño consistente en todos los targets evaluados (Figura S4). Los *fingerprints* ASP, LSTAR y RAD2D fueron generados con el *wrapper* de python del software *jCompoundMapper* [101], con las configuraciones de *fingerprints* por defecto. Las *fingerprints* de RDKit [102] fueron generadas con su versión 2023.9.4.
2. **Coeficiente de similitud:** Se evaluaron los coeficientes de similitud Tanimoto, Cosine y Braun-Blanquet. Todos los coeficientes de similitud fueron calculados con RDKit [102] en su versión 2023.9.4.
3. **Método de cálculo del resultado:** Se probaron diferentes estrategias para integrar las similitudes calculadas, incluyendo valores máximos, promedios y percentiles superiores. También se evaluaron metodologías basadas en el AUROC y AUPR de la similitud de cada compuesto contra el *set* de entrenamiento. El método basado en la similitud máxima con compuestos activos e inactivos por separado mostró el mejor desempeño, especialmente cuando los resultados se procesaron mediante una función *softmax* para generar probabilidades de bioactividad.

6.3.2. Selección final del modelo

El algoritmo final combina los *fingerprints* ASP con el coeficiente de Braun-Blanquet y utiliza el método de similitud máxima, integrando los resultados de la similitud del *set* positivo y negativo con la función *softmax*. Esta configuración fue seleccionada tras demostrar un desempeño robusto y consistente en la predicción de bioactividad en todos los targets, evaluado mediante AUROC en el conjunto de validación (Figura S4). Anecdóticamente, la elección de ASP y el coeficiente Braun-Blanquet coincide con los resultados presentados en Safizadeh *et al.* [60].

6.4. Modelos TrypanoDEEPscreen

Se desarrollaron modelos de redes neuronales convolucionales profundas (*CNN*), basados en el sistema planteado en Rifaioglu et al. [39], que entrena un modelo por cada *target*. Los modelos consistieron en *CNN* entrenadas con una lista de compuestos (en forma de *SMILES*), activos e inactivos contra cada uno de los targets. A partir de dichas *SMILES*, el sistema genera automáticamente imágenes de las estructuras de Lewis de 200x200px utilizando la librería RDKit [102], que luego son procesadas individualmente por la *CNN* retornando un *score* entre 0 y 1, asociado a la bioactividad predicha por la red. Estos modelos fueron entrenados con el *set* de datos de entrenamiento y evaluadas contra el *set* de testeо, el cual no fue expuesto a los modelos en ningún momento del entrenamiento. El *set* de datos de validación, se utilizó internamente para seleccionar los mejores modelos entrenados, esto se explica en la sección [entrenamiento](#).

6.4.1. Arquitectura

Se reimplementó la arquitectura tomando de referencia el código provisto en el repositorio de *github DEEPscreen* [103]. Para ello, se utilizó la librería PyTorch Lighting [92] conservando parte de la red, reescribiendo todo el código respecto al procesamiento de las *SMILES*, el loop de entrenamiento, la obtención de métricas de *performance* y el procesamiento del *output* final de la red.

En la Figura 32 se presenta un diagrama simplificado de la arquitectura, que consiste en un total de 13 capas, contando la capa de salida. Las primeras 10 capas constan de la red convolucional, que intercala capas de convoluciones con capas de *pooling* (MaxPooling) que reducen la dimensionalidad a la mitad en cada instancia. Además, a la salida de las capas convolucionales se aplica normalización en *batch* de 2D, seguido de la función de activación ReLU. Luego de la última convolución, se aplana el pool de salida a un vector en 1D de tamaño 800 que ingresa a una MLP de dos capas completamente conectadas de dimensiones variables, ajustables como hiperparámetros, y la capa de salida de dos dimensiones, a la que se le aplica la función *softmax*.

Los filtros convolucionales y las capas de *pooling* tienen un tamaño de 2x2. Todos los pesos utilizan la inicialización por defecto de *PyTorch* y las capas completamente conectadas emplean *dropout*, con una frecuencia de *dropout* ajustable por hiperparámetros. También se ajustaron el tamaño de los lotes y la tasa de aprendizaje. Para la optimización, se utilizó el algoritmo *ADAM* (*Adaptive Moment Estimation*) y la función de *loss Cross Entropy*.

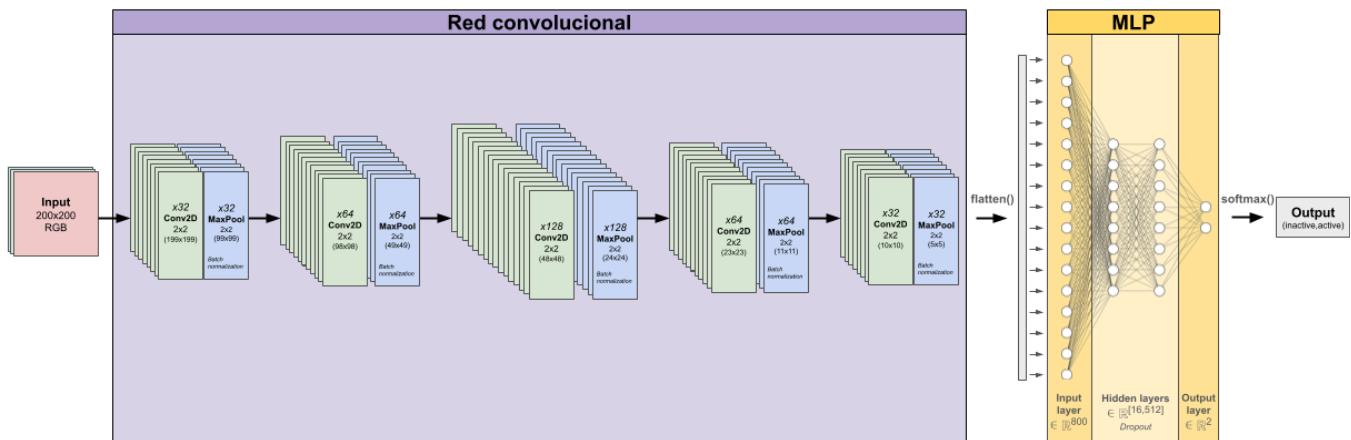


Figura 32. Diagrama de bloques de la arquitectura de la red neuronal convolucional utilizada. Consta de 5 convoluciones, que utilizan un *kernel* de 2x2, *padding* de 0 y stride de 1. Luego se aplana el vector de salida, que ingresa a una red multicapa de perceptrones, con *dropout*, que retorna un vector de dos dimensiones, el cual ingresa a la función *softmax* para retornar un *score* entre 0 y 1.

6.4.2. Entrenamiento y optimización por hiperparámetros

La optimización de hiperparámetros se realizó con *RayTune*, utilizando el "*Asynchronous Successive Halving Algorithm*". Este método entrena múltiples configuraciones en paralelo, descartando iterativamente las menos prometedoras. Se evaluaron 1000 combinaciones de hiperparámetros (Tabla 3), de las cuales se seleccionaron 350 mediante muestreo aleatorio. Cada modelo (con una configuración de hiperparámetros única) fue entrenado durante un máximo de 150 *epochs*, reteniendo los cinco mejores checkpoints basados en la métrica AUROC en el conjunto de validación.

Tabla 3. Hiperparámetros ajustados en el entrenamiento de los modelos.

hiperparámetros	Valores evaluados
<i>Learning rate</i>	0.0005, 0.0001, 0.005, 0.001, 0.01
<i>Dropout rate</i>	0.3, 0.5, 0.6, 0.8
<i>Batch Size</i>	32, 64
<i>Fully connected layer 1</i>	16, 32, 128, 256, 512
<i>Fully connected layer 2</i>	16, 32, 128, 256, 512

6.4.3. Ensamblado

El ensamblado de los modelos consistió en utilizar 5 modelos por conjunto de hiperparámetros entrenados y distintas configuraciones de hiperparámetros. Es decir, que se seleccionan automáticamente los 5 mejores checkpoints de los entrenamientos en base al AUROC del *set* de validación y se ordenan todos los modelos entrenados en el ajuste de hiperparámetros, nuevamente en base a la *performance* en el *set* de validación. En ese orden, se seleccionaron 26 modelos que se ensamblan utilizando el promedio como método de agregación.

7. Referencias

1. *Machine Learning Glossary*. Google Dev. n.d.
<https://developers.google.com/machine-learning/glossary>. Accessed February 11, 2025.
2. Wermuth CG, Ganellin CR, Lindberg P, Mitscher LA. Glossary of terms used in medicinal chemistry (IUPAC Recommendations 1998). *Pure Appl Chem*. 1998;70:1129–43.
<https://doi.org/10.1351/pac199870051129>.
3. Truchon J-F, Bayly CI. Evaluating Virtual *Screening* Methods: Good and Bad Metrics for the “Early Recognition” Problem. *J Chem Inf Model*. 2007;47:488–508.
<https://doi.org/10.1021/ci600426e>.
4. de Sousa DS, da Silva AP, de Angelo RM, Chiari LPA, Honorio KM, da Silva ABF. *Machine Learning and Neural Network Methods Applied to Drug Discovery*. In: Maltarollo VG, editor. Comput.-Aided Mach. Learn.-Driven Drug Des. Theory Appl. Cham: Springer Nature Switzerland; 2024. pp. 65–107. https://doi.org/10.1007/978-3-031-76718-0_4.
5. Maltarollo VG, editor. Computer-Aided and *Machine Learning*-Driven Drug Design: From Theory to Applications. vol. 3. Cham: Springer Nature Switzerland; 2024.
<https://doi.org/10.1007/978-3-031-76718-0>.
6. Urán Landaburu L, Didier Garnham M, Agüero F. Targeting trypanosomes: how chemogenomics and artificial intelligence can guide *drug discovery*. *Biochem Soc Trans*. 2023;51:195–206. <https://doi.org/10.1042/BST20220618>.
7. Géron A. Hands-on *machine learning* with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems. Third edition. Beijing Boston Farnham Sebastopol Tokyo: O'Reilly; 2023.
8. LeCun Y, Bengio Y, Hinton G. *Deep learning*. *Nature*. 2015;521:436–44.
<https://doi.org/10.1038/nature14539>.
9. Szyc K. Determining the Minimal Number of Images Required to Effectively Train Convolutional Neural Networks. In: Zamojski W, Mazurkiewicz J, Sugier J, Walkowiak T, Kacprzyk J, editors. *Theory Appl. Dependable Comput. Syst.*, vol. 1173. Cham: Springer International Publishing; 2020. pp. 652–61. https://doi.org/10.1007/978-3-030-48256-5_64.
10. Walters P. Why Don't *Machine Learning* Models Extrapolate? *Pract Cheminformatics*. 2025.
<https://patwalters.github.io/Why-Dont-Machine-Learning-Models-Extrapolate/>. Accessed May 10, 2025.
11. Goodfellow I, Bengio Y, Courville A. *Deep Learning*. MIT Press; 2016.
12. Llanos MA. Descubrimiento de nuevos anticonvulsivos que actúan mediante interacciones con canales iónicos. Tesis. Universidad Nacional de La Plata, 2022.
<https://doi.org/10.35537/10915/140503>.
13. Tropsha A. Best Practices for QSAR Model Development, Validation, and Exploitation. *Mol Inform*. 2010;29:476–88. <https://doi.org/10.1002/minf.201000061>.
14. Imrie F, Bradley AR, Deane CM. Virtual *Screening* with Convolutional Neural Networks. n.d.
15. Ho YC, Pepyne DL. Simple Explanation of the No-Free-Lunch Theorem and Its Implications. *J Optim Theory Appl*. 2002;115:549–70. <https://doi.org/10.1023/A:1021251113462>.
16. James G, Witten D, Hastie T, Tibshirani R, Taylor JE. An introduction to statistical learning:

- with applications in Python. Cham: Springer; 2023.
17. Kuhn M, Johnson K. Over-Fitting and Model Tuning. In: Kuhn M, Johnson K, editors. *Appl. Predict. Model.* New York, NY: Springer; 2013. pp. 61–92.
https://doi.org/10.1007/978-1-4614-6849-3_4.
 18. Jabbar HK, Khan RZ. Methods to Avoid Over-Fitting and Under-Fitting in Supervised *Machine Learning* (Comparative Study). *Comput. Sci. Commun. Instrum. Devices. Research Publishing Services*; 2014. pp. 163–72. https://doi.org/10.3850/978-981-09-5247-1_017.
 19. *Cross-validation* (statistics). Wikipedia. 2025.
 20. Regularization (mathematics). Wikipedia. 2025.
 21. Bonaccorso G. *Machine Learning* Algorithms: Popular Algorithms for Data Science and *Machine Learning*. Second Edition. Birmingham Mumbai: Packt; 2018.
 22. Rectifier (neural networks). Wikipedia. 2025.
 23. Descenso del gradiente. Wikipedia Encicl Libre. 2025.
 24. Hidayat T, Astuti IA, Yaqin A, Tjilen AP, Arifianto T. Grouping of Image Patterns Using Inceptionv3 For Face Shape Classification. *JOIV Int J Inform Vis.* 2023.
 25. What Is a Convolution? How To Teach Machines To See Images. 8th Light. n.d.
<https://8thlight.com/insights/what-is-a-convolution-how-to-teach-machines-to-see-images>. Accessed June 22, 2025.
 26. Katzung BG, editor. Basic & clinical pharmacology. Fourteenth edition. New York Chicago San Francisco Athens London Madrid Mexico City Milan New Delhi Singapore Sydney Toronto: McGraw-Hill Education; 2018.
 27. Myers S, Baker A. *Drug discovery*—an operating model for a new era. *Nat Biotechnol.* 2001;19:727–30. <https://doi.org/10.1038/90765>.
 28. Morgan S, Grootendorst P, Lexchin J, Cunningham C, Greyson D. The cost of drug development: A systematic review. *Health Policy.* 2011;100:4–17.
<https://doi.org/10.1016/j.healthpol.2010.12.002>.
 29. Hughes JP, Rees S, Kalindjian SB, Philpott KL. Principles of early *drug discovery*. *Br J Pharmacol.* 2011;162:1239–49. <https://doi.org/10.1111/j.1476-5381.2010.01127.x>.
 30. Sinha S, Vohora D. Chapter 2 - *Drug Discovery* and Development: An Overview. In: Vohora D, Singh G, editors. *Pharm. Med. Transl. Clin. Res.* Boston: Academic Press; 2018. pp. 19–32.
<https://doi.org/10.1016/B978-0-12-802103-3.00002-X>.
 31. Gad SC. *Drug Discovery* Handbook. 1st ed. Hoboken: John Wiley & Sons, Incorporated; 2005.
 32. Berdigaliyev N, Aljofan M. An Overview of *Drug Discovery* and Development. *Future Med Chem.* 2020;12:939–47. <https://doi.org/10.4155/fmc-2019-0307>.
 33. Phatak SS, Stephan CC, Cavasotto CN. High-throughput and *in silico* screenings in *drug discovery*. *Expert Opin Drug Discov.* 2009. <https://doi.org/10.1517/17460440903190961>.
 34. Sliwoski G, Kothiwale S, Meiler J, Lowe EW. Computational Methods in *Drug Discovery*. *Pharmacol Rev.* 2014;66:334–95. <https://doi.org/10.1124/pr.112.007336>.
 35. Johnson MA, Maggiora GM. Concepts and applications of molecular similarity. 196th : 1988 : Los Angeles, Calif: Wiley-interscience; 1990.
 36. Fernandes PO, Matarollo VG. QSAR and *Machine Learning* Predictors. In: Matarollo VG, editor. *Comput.-Aided Mach. Learn.-Driven Drug Des. Theory Appl.* Cham: Springer Nature

- Switzerland; 2024. pp. 131–61. https://doi.org/10.1007/978-3-031-76718-0_6.
37. Stokes JM, Yang K, Swanson K, Jin W, Cubillos-Ruiz A, Donghia NM, et al. A *Deep Learning* Approach to Antibiotic Discovery. *Cell*. 2020;180:688-702.e13. <https://doi.org/10.1016/j.cell.2020.01.021>.
38. Barbosa H, Espinoza GZ, Amaral M, de Castro Levatti EV, Abiuzi MB, Veríssimo GC, et al. Andrographolide: A Diterpenoid from *Cymbopogon schoenanthus* Identified as a New *Hit* Compound against *Trypanosoma cruzi* Using *Machine Learning* and Experimental Approaches. *J Chem Inf Model*. 2024;64:2565–76. <https://doi.org/10.1021/acs.jcim.3c01410>.
39. Rifaioglu AS, Nalbat E, Atalay V, Martin MJ, Cetin-Atalay R, Doğan T. *DEEPScreen*: high performance drug–target interaction prediction with convolutional neural networks using 2-D structural compound representations. *Chem Sci*. 2020;11:2531–57. <https://doi.org/10.1039/C9SC03414E>.
40. Wallach I, Dzamba M, Heifets A. AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based *Drug Discovery*. 2015.
41. Fourches D, Muratov E, Tropsha A. Trust, but verify: On the importance of chemical structure curation in cheminformatics and QSAR modeling research. *J Chem Inf Model*. 2010;50:1189–204. <https://doi.org/10.1021/ci100176x>.
42. Zdrazil B, Felix E, Hunter F, Manners EJ, Blackshaw J, Corbett S, et al. The ChEMBL Database in 2023: a *drug discovery* platform spanning multiple bioactivity data types and time periods. *Nucleic Acids Res*. 2024;52:D1180–92. <https://doi.org/10.1093/nar/gkad1004>.
43. Caldwell GW, Yan Z, Lang W, Masucci JA. The IC50 Concept Revisited. *Curr Top Med Chem*. n.d.;12:1282–90. <https://doi.org/10.2174/156802612800672844>.
44. Dealing with a data dilemma. *Nat Rev Drug Discov*. 2008;7:632–3. <https://doi.org/10.1038/nrd2649>.
45. Cáceres EL, Mew NC, Keiser MJ. Adding Stochastic Negative Examples into *Machine Learning* Improves Molecular Bioactivity Prediction. *J Chem Inf Model*. 2020;60:5957–70. <https://doi.org/10.1021/acs.jcim.0c00565>.
46. Guo Q, Hernandez-Hernandez S, Ballester PJ. Scaffold Splits Overestimate Virtual *Screening Performance*. 2024. <https://doi.org/10.48550/arXiv.2406.00873>.
47. Wu Z, Ramsundar B, Feinberg EN, Gomes J, Geniesse C, Pappu AS, et al. MoleculeNet: a benchmark for molecular *machine learning*. *Chem Sci*. 2018;9:513–30. <https://doi.org/10.1039/C7SC02664A>.
48. Ramsundar B, Eastman P, Walters P, Pande V. *Deep learning* for the life sciences: applying *deep learning* to genomics, microscopy, *drug discovery*, and more. First edition, revision, second release. Beijing Boston Farnham Sebastopol Tokyo: O'Reilly; 2021.
49. Bemis GW, Murcko MA. The properties of known drugs. 1. Molecular frameworks. *J Med Chem*. 1996;39:2887–93. <https://doi.org/10.1021/jm9602928>.
50. Weininger D. *SMILES*, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J Chem Inf Comput Sci*. 1988;28:31–6. <https://doi.org/10.1021/ci00057a005>.
51. Daylight Theory: *SMILES*. n.d. <https://www.daylight.com/dayhtml/doc/theory/theory.smiles.html>. Accessed May 19, 2025.

52. McGibbon M, Shave S, Dong J, Gao Y, Houston DR, Xie J, et al. From intuition to AI: evolution of small molecule representations in *drug discovery*. *Brief Bioinform.* 2023;25:bbad422. <https://doi.org/10.1093/bib/bbad422>.
53. Heid E, Greenman KP, Chung Y, Li S-C, Graff DE, Vermeire FH, et al. Chemprop: A *Machine Learning* Package for Chemical Property Prediction. *J Chem Inf Model.* 2024;64:9–17. <https://doi.org/10.1021/acs.jcim.3c01250>.
54. Precision and *recall*. Wikipedia. 2025.
55. Harmonic Mean Explained: A Guide to Rates and Ratios. n.d. <https://www.datacamp.com/tutorial/harmonic-mean>. Accessed July 21, 2025.
56. Huijzer R. The history of the ROC curve. Riks Weblog. 2024. <https://huijzer.xyz/posts/57>. Accessed July 22, 2025.
57. Cai C, Gong J, Liu X, Gao D, Li H. Molecular Similarity: Methods and *Performance*. *Chin J Chem.* 2013;31:1123–32. <https://doi.org/10.1002/cjoc.201300390>.
58. N. Muratov E, Bajorath J, P. Sheridan R, V. Tetko I, Filimonov D, Poroikov V, et al. QSAR without borders. *Chem Soc Rev.* 2020;49:3525–64. <https://doi.org/10.1039/DQCS00098A>.
59. Bajorath J, Peltason L, Wawer M, Guha R, Lajiness MS, Van Drie JH. Navigating structure–activity landscapes. *Drug Discov Today.* 2009;14:698–705. <https://doi.org/10.1016/j.drudis.2009.04.003>.
60. Safizadeh H, Simpkins SW, Nelson J, Li SC, Piotrowski JS, Yoshimura M, et al. Improving Measures of Chemical Structural Similarity Using *Machine Learning* on Chemical–Genetic Interactions. *J Chem Inf Model.* 2021;61:4156–72. <https://doi.org/10.1021/acs.jcim.0c00993>.
61. Gimeno A, Ojeda-Montes MJ, Tomás-Hernández S, Cereto-Massagué A, Beltrán-Debón R, Mulero M, et al. The Light and Dark Sides of Virtual *Screening*: What Is There to Know? *Int J Mol Sci.* 2019;20:1375. <https://doi.org/10.3390/ijms20061375>.
62. WHO. Ending the neglect to attain the sustainable development goals: a sustainability framework for action against neglected tropical diseases 2021–2030. World Health Organization; 2021.
63. Ferreira LLG, Andricopulo AD. Drugs and vaccines in the 21st century for neglected diseases. *Lancet Infect Dis.* 2019;19:125–7. [https://doi.org/10.1016/S1473-3099\(19\)30005-2](https://doi.org/10.1016/S1473-3099(19)30005-2).
64. Yamey G, Batson A, Kilmarx PH, Yotebieng M. Funding innovation in neglected diseases. *The BMJ.* 2018;360:k1182. <https://doi.org/10.1136/bmj.k1182>.
65. Ferreira LLG, de Moraes J, Andricopulo AD. Approaches to advance *drug discovery* for neglected tropical diseases. *Drug Discov Today.* 2022;27:2278–87. <https://doi.org/10.1016/j.drudis.2022.04.004>.
66. Pedrique B, Strub-Wourgaft N, Some C, Olliari P, Trouiller P, Ford N, et al. The drug and vaccine landscape for neglected diseases (2000–11): a systematic assessment. *Lancet Glob Health.* 2013;1:e371–9. [https://doi.org/10.1016/S2214-109X\(13\)70078-0](https://doi.org/10.1016/S2214-109X(13)70078-0).
67. Peña I, Pilar Manzano M, Cantizani J, Kessler A, Alonso-Padilla J, Bardera AI, et al. New Compound Sets Identified from High Throughput Phenotypic *Screening* Against Three Kinetoplastid Parasites: An Open Resource. *Sci Rep.* 2015;5:8771. <https://doi.org/10.1038/srep08771>.
68. Chagas disease. n.d.

- [https://doi.org/10.1056/NEJMra064142](https://www.who.int/news-room/fact-sheets/detail/chagas-disease-(american-trypanosomiasis). Accessed August 8, 2024.</p><p>69. Hotez PJ, Molyneux DH, Fenwick A, Kumaresan J, Sachs SE, Sachs JD, et al. Control of Neglected Tropical Diseases. <i>N Engl J Med.</i> 2007;357:1018–27.
<a href=).
70. Gascon J, Bern C, Pinazo M-J. Chagas disease in Spain, the United States and other non-endemic countries. *Acta Trop.* 2010;115:22–7.
<https://doi.org/10.1016/j.actatropica.2009.07.019>.
71. Antinori S, Galimberti L, Bianco R, Grande R, Galli M, Corbellino M. Chagas disease in Europe: A review for the internist in the globalized world. *Eur J Intern Med.* 2017;43:6–15.
<https://doi.org/10.1016/j.ejim.2017.05.001>.
72. Hotez PJ. Neglected Infections of Poverty in the United States of America. *PLoS Negl Trop Dis.* 2008;2:e256. <https://doi.org/10.1371/journal.pntd.0000256>.
73. Teixeira AR, Gomes C, Lozzi SP, Hecht MM, Rosa A de C, Monteiro PS, et al. Environment, interactions between *Trypanosoma cruzi* and its host, and health. *Cad Saude Publica.* 2009;25:S32–44.
74. Pérez-Molina JA, Molina I. Chagas disease. *The Lancet.* 2018;391:82–94.
[https://doi.org/10.1016/S0140-6736\(17\)31612-4](https://doi.org/10.1016/S0140-6736(17)31612-4).
75. Tanowitz HB, Kirchhoff LV, Simon D, Morris SA, Weiss LM, Wittner M. Chagas' disease. *Clin Microbiol Rev.* 1992;5:400–19. <https://doi.org/10.1128/CMR.5.4.400>.
76. Pinazo M-J, Muñoz J, Posada E, López-Chejade P, Gállego M, Ayala E, et al. Tolerance of Benznidazole in Treatment of Chagas' Disease in Adults. *Antimicrob Agents Chemother.* 2010;54:4896–9. <https://doi.org/10.1128/aac.00537-10>.
77. Fabbro DL, Streiger ML, Arias ED, Bizai ML, del Barco M, Amicone NA. Trypanocide treatment among adults with chronic Chagas disease living in Santa Fe city (Argentina), over a mean follow-up of 21 years: parasitological, serological and clinical evolution. *Rev Soc Bras Med Trop.* 2007;40:1–10. <https://doi.org/10.1590/s0037-86822007000100001>.
78. Lemke TL, Williams DA. *Foye's Principles of Medicinal Chemistry.* Lippincott Williams & Wilkins; 2008.
79. Smirlis D, Soares MBP. Selection of Molecular Targets for Drug Development Against Trypanosomatids. In: Santos ALS, Branquinha MH, d'Avila-Levy CM, Kneipp LF, Sodré CL, editors. *Proteins Proteomics Leishmania Trypanos.* Dordrecht: Springer Netherlands; 2014. pp. 43–76. https://doi.org/10.1007/978-94-007-7305-9_2.
80. Hall BS, Wilkinson SR. Activation of Benznidazole by Trypanosomal Type I Nitroreductases Results in Glyoxal Formation. *Antimicrob Agents Chemother.* 2012;56:115–23.
<https://doi.org/10.1128/aac.05135-11>.
81. Dattani A, Drammeh I, Mahmood A, Rahman M, Szular J, Wilkinson SR. Unraveling the antitypanosomal mechanism of benznidazole and related 2-nitroimidazoles: From prodrug activation to DNA damage. *Mol Microbiol.* 2021;116:674–89.
<https://doi.org/10.1111/mmi.14763>.
82. Nwaka S, Hudson A. Innovative *lead* discovery strategies for tropical diseases. *Nat Rev Drug Discov.* 2006;5:941–55. <https://doi.org/10.1038/nrd2144>.
83. Klug DM, Gelb MH, Pollastri MP. Repurposing strategies for tropical disease *drug discovery.*

- Bioorg Med Chem Lett. 2016;26:2569–76. <https://doi.org/10.1016/j.bmcl.2016.03.103>.
84. Osborne CK. Tamoxifen in the Treatment of Breast Cancer. N Engl J Med. 1998;339:1609–18. <https://doi.org/10.1056/NEJM199811263392207>.
85. Mendez D, Gaulton A, Bento AP, Chambers J, De Veij M, Félix E, et al. ChEMBL: towards direct deposition of bioassay data. Nucleic Acids Res. 2019;47:D930–40. <https://doi.org/10.1093/nar/gky1075>.
86. Assay and Activity Questions | ChEMBL Interface Documentation. 2024. <https://chembl.gitbook.io/chembl-interface-documentation/frequently-asked-questions/chembl-data-questions>. Accessed October 3, 2024.
87. Nikolova N, Jaworska J. Approaches to Measure Chemical Similarity – a Review. QSAR Comb Sci. 2003;22:1006–26. <https://doi.org/10.1002/qsar.200330831>.
88. Guo Q, Hernandez-Hernandez S, Ballester PJ. Scaffold Splits Overestimate Virtual Screening Performance. In: Wand M, Malinovská K, Schmidhuber J, Tetko IV, editors. Artif. Neural Netw. Mach. Learn. – ICANN 2024. Cham: Springer Nature Switzerland; 2024. pp. 58–72. https://doi.org/10.1007/978-3-031-72359-9_5.
89. Why Don't Machine Learning Models Extrapolate? Pract Cheminformatics. 2025. <https://patwalters.github.io/Why-Dont-Machine-Learning-Models-Extrapolate/>. Accessed May 29, 2025.
90. Meslamani J, Li J, Sutter J, Stevens A, Bertrand H-O, Rognan D. Protein–Ligand-Based Pharmacophores: Generation and Utility Assessment in Computational Ligand Profiling. J Chem Inf Model. 2012;52:943–55. <https://doi.org/10.1021/ci300083r>.
91. Gfeller D, Michelin O, Zoete V. Shaping the interaction landscape of bioactive molecules. Bioinformatics. 2013;29:3073–9. <https://doi.org/10.1093/bioinformatics/btt540>.
92. Falcon W, Borovec J, Wälchli A, Eggert N, Schock J, Jordan J, et al. PyTorchLightning/pytorch-lightning: 0.7.6 release. 2020. <https://doi.org/10.5281/ZENODO.3828935>.
93. Liaw R, Liang E, Nishihara R, Moritz P, Gonzalez JE, Stoica I. Tune: A Research Platform for Distributed Model Selection and Training. 2018. <https://doi.org/10.48550/arXiv.1807.05118>.
94. Li L, Jamieson K, Rostamizadeh A, Gonina E, Hardt M, Recht B, et al. A System for Massively Parallel Hyperparameter Tuning. 2020. <https://doi.org/10.48550/arXiv.1810.05934>.
95. Wang Y (Marcia). Statistical Methods for High Throughput Screening Drug Discovery Data. 2005.
96. Ren J, Wang H. Chapter 3 - Calculus and optimization. In: Ren J, Wang H, editors. Math. Methods Data Sci. Elsevier; 2023. pp. 51–89. <https://doi.org/10.1016/B978-0-44-318679-0.00009-0>.
97. Jo J, Kwak B, Choi H-S, Yoon S. The message passing neural networks for chemical property prediction on SMILES. Methods. 2020;179:65–72. <https://doi.org/10.1016/j.ymeth.2020.05.009>.
98. Malhotra P, Biswas S, Sharma GD. Directed Message Passing Neural Network for Predicting Power Conversion Efficiency in Organic Solar Cells. ACS Appl Mater Interfaces. 2023;15:37741–7. <https://doi.org/10.1021/acsami.3c08068>.
99. Swanson KW. Message passing neural networks for molecular property prediction. Thesis. Massachusetts Institute of Technology, 2019.
100. Urán Landaburu L, Berenstein AJ, Videla S, Maru P, Shanmugam D, Chernomoretz A, et al.

- TDR Targets 6: driving *drug discovery* for human pathogens through intensive chemogenomic data integration. Nucleic Acids Res. 2019;gkz999.
<https://doi.org/10.1093/nar/gkz999>.
101. OlivierBeq. OlivierBeq/jCompoundMapper_pywrapper. 2024.
 102. Greg Landrum, Paolo Tosco, Brian Kelley, Ric, David Cosgrove, sriniker, et al. RDKit: Open-source cheminformatics. 2024. <https://doi.org/10.5281/ZENODO.10633624>.
 103. Tunca D. cansyl/DEEPScreen. 2024. <https://github.com/cansyl/DEEPScreen>

8. Reconocimientos

Deseo expresar mi reconocimiento al *Consejo Interuniversitario Nacional* por el otorgamiento de la beca *Estímulo a las Vocaciones Científicas (EVC-CIN)*, que brindó el apoyo económico útil para llevar adelante este trabajo.

Agradezco también a Martín Makler (International Center for Advanced Studies, UNSAM) y a la Universidade Federal do Rio de Janeiro por haber facilitado el acceso a recursos de cómputo con *GPU*, fundamentales para el entrenamiento y desarrollo de las redes neuronales utilizadas en esta tesis.

Finalmente, extiendo mi agradecimiento a la *Universidad Nacional de San Martín* y al *Instituto de Investigaciones Biotecnológicas (IIB)* por la formación académica y profesional recibida durante el trayecto de la carrera.

Agradecimientos

A continuación, los únicos párrafos de mi tesina que no fueron retocados con una de las tantas LLM utilizadas para escribir el manuscrito:

Siempre me gustó saber cosas. Mientras más cosas mejor. Así que supongo tiene sentido haber estudiado una carrera científica, siendo la ciencia el arte de descubrir cosas, para que todos sepamos más cosas y a veces también para hacer algo con esas cosas. Desde que arranqué la carrera, algo que siempre me fascinó es que a toda la gente que conocía ahí, también le gustaba saber cosas. Por sobre todo me encantó encontrarme que la mayoría de personas sabía muchas más cosas que yo. Es más, en comparación, yo casi que no sabía ninguna cosa. Estos son unos agradecimientos a todas esas personas que tanto admiro, por haberme enseñado muchas cosas, por haberme acompañado mientras aprendíamos cosas juntos y por haberme ayudado a construir toda la cosa feliz que soy hoy en día.

Arrancando por las personas que conocí que más saben cosas, y que más les gusta compartir esas cosas, agradezco inmensamente a todo el laboratorio de bioinformática. Gracias a ellos orienté mi carrera profesional a algo que me gusta, me enseñaron demasiadas cosas y me acompañaron en todo el proceso.

Mientras cursaba la carrera, fui dándome cuenta que me resultaba casi imposible aprender cosas sin amigos aprendecosas al lado. Por eso, no sé si hubiera podido terminar sin los amigos que me fui haciendo durante la carrera, así que para todos ellos también van unas gracias bastante importantes.

Por último, creo que si bien dejé en claro que me encanta saber cosas, para estar feliz me parece que se necesita mucho más que solo saber cosas, y ahí entran a jugar mis viejos, mi hermano y mi novia, a quienes creo que ni hace falta aclarar que estoy absolutísimamente agradecido por todo el apoyo que siempre me dieron.

Por todo eso, escribo las últimas palabras de mi carrera diciendo:

¡Aguante aprender cosas! ¡Aguante la Biotecnología! ¡Aguante la Ciencia! ¡Aguante el CONICET!

¡Aguante todas las personas que me bancaron todos estos años!