UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA AGRONÓMICA, ALIMENTARIA Y DE BIOSISTEMAS

GRADO EN BIOTECNOLOGÍA

DEPARTAMENTO DE BIOTECNOLOGÍA-BIOLOGÍA VEGETAL

Aplicación de aprendizaje por transferencia para la predicción de tiempos de retención de metabolitos

TRABAJO FIN DE GRADO

Autor: Nicolás Lorenzo Fernández-Cordeiro

Tutor: Pablo Rodríguez Palenzuela

Julio de 2025

UNIVERSIDAD POLITÉCNICA DE MADRID

Escuela Técnica Superior De

Ingeniería Agronómica, Alimentaria y de Biosistemas

GRADO DE BIOTECNOLOGÍA

APLICACIÓN DE APRENDIZAJE POR TRANSFERENCIA PARA LA PREDICCIÓN DE TIEMPOS DE RETENCIÓN DE METABOLITOS

TRABAJO FIN DE GRADO

Nicolás Lorenzo Fernández-Cordeiro

MADRID, 2025

Tutor: Pablo Rodríguez Palenzuela
Dpto. de Biotecnología-Biología Vegetal

Cotutor: Abraham Otero Quintana
Grupo: Laboratorio de Ingeniería Biomédica Universidad CEU San Pablo

TITULO DEL TFG - APLICACIÓN DE APRENDIZAJE POR TRANSFERENCIA PARA LA PREDICCIÓN DE TIEMPOS DE PREDICCIÓN DE METABOLITOS

Memoria presentada por NICOLÁS LORENZO FERNÁNDEZ-CORDEIRO para la obtención del título de Graduado en Biotecnología por la Universidad Politécnica de Madrid

Fdo: Nicolás Lorenzo Fernández-Cordeiro

VºBº Tutor y Director del TFG

Dr. Pablo Rodríguez Palenzuela
Catedrático de Bioquímica y Biología Molecular
Dpto. de Biotecnología-Biología Vegetal
ETSIAAB Universidad Politécnica de Madrid

VºBº Cotutor

Dr. Abraham Otero Quintana
Catedrático de Ciencias de la Computación e Inteligencia Artificial
Grupo Laboratorio de Ingeniería Biomédica
Universidad CEU San Pablo

Madrid, 7 de Julio de 2025

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

LISTA DE SÍMBOLOS

β: tasa de decaimiento exponencial

LISTA DE ABREVIATURAS

HPLC: Cromatografía líquida de alto rendimiento, siglas del inglés (*High Performance Liquid Chromatography*)

DCM: diclorometano

ACN: acetonitrilo

MeOH: metanol

iPrOH: 2-propanolol

RT: Tiempo de retención, siglas del inglés (*Retention Time*)

RMN: Resonancia Magnética Nuclear

MS: Espectroscopía de masas, siglas del inglés (*Mass Spectroscopy*)

ESI: Ionización por electrospray, siglas del inglés (*Electrospray Ionization*)

EI: Ionización por bombardeo electrónico, siglas del inglés (*Electron Ionization*)

MALDI: Ionización láser asistida por matriz, siglas del inglés (*Matrix-Assisted Laser Desorption/Ionization*)

Q: Cuadrupolo, siglas del inglés (*Quadrupole*)

TOF: Analizador de tiempo de vuelo, siglas del inglés (*Time Of Flight*)

MS/MS: Espectrometría de masas en tándem

QqQ: Triple cuadrupolo

QTOF: Cuadrupolo-tiempo de vuelo

Q-Trap: Cuadrupolo-Orbitrap

TOF/TOF: Tiempo de vuelo-tiempo de vuelo

LC-MS/MS: Cromatografía líquida acoplada a espectrometría de masas en tándem, siglas del inglés (*Liquid Chromatography – Tandem Mass Spectrometry*)

ML: Aprendizaje automático, siglas del inglés (*Machine Learning*)

IA: Inteligencia Artificial

RNA: Redes neuronales artificiales

DNN: Redes neuronales profundas, siglas del inglés (*Deep Neural Network*)

DL: Aprendizaje profundo, siglas del inglés (*Deep Learning*)

MAE: Error medio absoluto, siglas del inglés (*Mean Average Error*)

MSE: Error cuadrático medio, siglas del inglés (*Mean Square Error*)

RMSE: Raíz del error cuadrático medio, siglas del inglés (*Root Mean Square Error*)

TL: Aprendizaje por transferencia, siglas del inglés (*Transfer Learning*)

AE: Autoencoder

SMILES: Notación simplificada de la entrada lineal de estructuras moleculares, siglas del inglés (*Simplified Molecular Input Line Entry Specification*)

InChi: Identificador químico internacional de la IUPAC, siglas del inglés (*IUPAC International Chemical Identifier*)

RESUMEN

La predicción de tiempos de retención (RT) en cromatografía líquida de alta resolución a través de métodos de aprendizaje automático es una aproximación en auge, fundamental para la identificación de metabolitos. Sin embargo, enfrenta una problemática de escasez de datos experimentales disponibles. Este trabajo investiga el empleo de redes neuronales profundas a través de una metodología de aprendizaje por transferencia no supervisado, basada en el preentrenamiento de autoencoders, con el objetivo de mejorar la predicción de los tiempos de retención de metabolitos en comparación con los métodos tradicionales de aprendizaje automático. El objetivo principal fue evaluar si el preentrenamiento de autoencoders mediante representaciones moleculares permitía la posterior construcción de modelos de predicción de RT con mejor rendimiento que aquellos no preentrenados.

Para ello, se entrenaron múltiples autoencoders mediante una labor de reconstrucción de representaciones moleculares (*fingerprints* y descriptores) de compuestos de la base de datos RepoRT. Se seleccionaron los autoencoders más eficaces en términos de calidad de reconstrucción y se transfirieron sus representaciones latentes como entrada a redes neuronales supervisadas encargadas de la predicción de RT. Se crearon modelos preentrenados con *fingerprints*, descriptores, y ambos a la vez. El rendimiento de estas redes con preentrenamiento se comparó con el de modelos equivalentes entrenados desde cero. Adicionalmente, se evaluó la calidad de diversas arquitecturas de las redes neuronales acopladas al espacio latente.

Los resultados mostraron que los autoencoders lograron reconstruir los datos moleculares con muy alta fidelidad, lo que da a entender que eran capaces de capturar las características relevantes de las moléculas. Sin embargo, en la tarea de predicción de RT, las redes preentrenadas ofrecieron peores resultados que las no preentrenadas. Solo al integrar las dos modalidades de datos (*fingerprints* y descriptores) se observó una eficacia equivalente. La discrepancia entre la alta eficacia lograda en el preentrenamiento de los autoencoders y los modestos resultados en la predicción de tiempos de retención podría sugerir que tanto la escasez de datos disponibles en metabolómica como la ausencia de un proceso de optimización de hiperparámetros han influido en el rendimiento de la aproximación basada en aprendizaje por transferencia no supervisado.

ABSTRACT

Prediction of retention time (RT) in high-performance liquid chromatography using machine learning methods is an emerging approach, key for metabolite identification. However, it faces significant challenges due to the scarcity of available experimental data. This project investigates the use of deep neural networks through an unsupervised transfer learning methodology, based on the pretraining of autoencoders, with the goal of improving the prediction of metabolite retention times compared to traditional machine learning methods. The main objective was to assess whether pretraining autoencoders on molecular representations would enable the subsequent construction of RT prediction models with better performance than those built from scratch.

To this end, multiple autoencoders were trained to reconstruct molecular representations (*fingerprints* and descriptors) of compounds from the RepoRT database. The most effective autoencoders, in terms of reconstruction quality, were selected, and their latent representations were transferred as inputs to supervised neural networks tasked with RT prediction. Pretrained models were created using *fingerprints*, descriptors, and both types of molecular representations combined. The performance of these pretrained networks was compared with that of equivalent models trained from scratch. Additionally, the quality of various neural network architectures coupled to the latent space was evaluated.

The results showed that the autoencoders successfully reconstructed the molecular data with very high fidelity, suggesting that they were able to capture relevant molecular features. However, in the RT prediction task, the pretrained networks performed worse than the non-pretrained models. Only when integrating both data modalities (fingerprints and descriptors) was comparable performance observed. The discrepancy between the high efficacy achieved during the autoencoder pretraining and the modest results in retention time prediction may suggest that both the limited amount of available metabolomics data and the absence of a hyperparameter optimization process have influenced the performance of the unsupervised transfer learning approach.

CAPÍTULO 1: INTRODUCCIÓN Y OBJETIVOS

1.1. Cromatografía líquida de alta resolución

La cromatografía líquida de alto rendimiento (HPLC: *High Performance Liquid Chromatography*) es un método de separación, identificación y cuantificación de compuestos ampliamente utilizado. Se trata de una tecnología empleada de forma extensa debido a su potente capacidad de separación molecular en muestras que pueden presentar una alta complejidad (Blum, 2014). En el equipo de HPLC los 2 componentes de mayor importancia son la fase móvil y la fase estacionaria. La naturaleza química de estas permitirá separar los compuestos de la mezcla en función de su polaridad, interacciones iónicas, o tamaño (Thomas et al., 2022).

La fase móvil es una mezcla de 2 o más solventes usados en un gradiente que permite aumentar o disminuir su polaridad total. Es inyectada en la columna, transportando los compuestos de la muestra. La fase estacionaria se encuentra empaquetada en el interior de la columna, como una matriz de microesferas de sílice recubiertas de partículas con diferente polaridad (Blum, 2014).

La polaridad de la fase móvil y de la fase estacionaria depende del tipo de cromatografía. La cromatografía en fase normal emplea una fase estacionaria de mayor polaridad y una fase móvil de menor polaridad que va aumentando en gradiente. Por ello, los analitos de menor polaridad eluyen primero al interaccionar en menor medida con la fase estacionaria. Entre las sustancias comúnmente empleadas en la fase estacionaria se encuentran el dióxido de silicio (SiO_2) y el óxido de aluminio (Al_2O_3), mientras que la fase estacionaria es un disolvente no acuoso como el diclorometano (DCM) o el cloroformo ($CHCl_3$) (Lu et al., 2008).

En la cromatografía de fase reversa, eluyen primero los compuestos polares. Se emplea una columna con fase estacionaria hidrofóbica, por lo general de carbohidratos con cadenas policarbonadas de 8 y 18 carbonos, y una fase móvil con mayor polaridad inicial, que va descendiendo gradualmente. Para la fase móvil, por lo general existen 2 mezclas empleadas. El primer método utiliza agua como solvente inicial y pasa progresivamente a un solvente apolar como el acetonitrilo (ACN) o el metanol (MeOH), y se emplea para analitos de hidrofobicidad media, mientras que el segundo utiliza gradientes de agua/ACN a 2-propanol (iPrOH) y se emplea típicamente para la separación de sustancias altamente apolares (Witting & Böcker, 2020a).

Tras pasar por la columna cromatográfica, las moléculas son detectadas por un detector UV que registra el tiempo que les ha tomado recorrer la columna. Esta medida recibe el nombre de tiempo de retención (RT: *Retention Time*), y es un valor característico de cada compuesto, permitiendo su identificación y cuantificación (Blum, 2014). Sin embargo, no existe un tiempo de retención único para un compuesto, pues es altamente dependiente de la columna empleada, su longitud, las proporciones y gradientes de la fase móvil, la temperatura o la presión (Baker & Regg, 2018). Esto conlleva que el RT sea una propiedad con una reproducibilidad relativa, pues depende de una amplia lista de condiciones.

El HPLC constituye una herramienta de análisis metabolómico muy valiosa debido a su potente capacidad de separación molecular. Permite el fraccionamiento de muestras complejas de muy diversa naturaleza mediante la manipulación de variables como la temperatura, los tiempos de elución, la proporción de disolventes en la fase móvil, las dimensiones de la columna, la concentración de los gradientes o el tipo de fase estacionaria y fase móvil.

1.2. Metabolómica y Anotación de metabolitos

La metabolómica es la ciencia que lleva a cabo el análisis sistemático del conjunto de moléculas pequeñas de un sistema biológico u organismo. Se trata de la disciplina de las ómicas de más reciente surgimiento y a su vez aquella que presenta un mayor nivel de complejidad (Novoa-Del-Toro & Witting, 2024). La caracterización e identificación de moléculas pequeñas es más compleja en comparación con la información tratada en otros campos como la genómica o la transcriptómica, debido a la diversidad fisicoquímica y amplios rangos de concentraciones de los diversos metabolitos, así como por el gran tamaño del metaboloma y las numerosas rutas metabólicas que lo componen (Godzien et al., 2018). Sin embargo, de la misma manera que la metabolómica estudia información biológica compleja, también aporta información de mayor nivel. El metaboloma constituye una instantánea de la fisiología de una célula u organismo bajo determinadas condiciones fisiológicas; permitiendo la extracción de información valiosa sobre el estado en el que se encuentra dicho sistema biológico (Witting & Böcker, 2020b).

La metabolómica no dirigida sigue un flujo de trabajo ampliamente establecido. Por lo general, este contiene diseño experimental, preparación de muestras, análisis, pretratamiento de datos y análisis estadístico, identificación y análisis de rutas, y

generación final de hipótesis (Godzien et al., 2018). Para obtener un fraccionamiento óptimo de la muestra separando sus componentes en función de diversas características fisicoquímicas se emplean métodos de separación basados en principios de cromatografía o electroforesis, seguidos de una detección mediante resonancia magnética nuclear (RMN: *Resonancia Magnética Nuclear*) o espectroscopía de masas (MS: *Mass Spectroscopy*).

La espectroscopía de masas comienza con la ionización de la muestra en un ionizador, para posteriormente atravesar un filtro de masas que separa los iones en relación a su ratio masa-carga (m/z) y finalmente ser registrados por un detector. Los métodos de ionización más empleados son la ionización por electrospray (ESI: *Electrospray ionization*), la ionización por bombardeo electrónico (EI: *Electron Ionization*) o la ionización láser asistida por matriz (MALDI: *Matrix-Assisted Laser Desorption/Ionization*). Los filtros de masas más comunes son el cuadrupolo (Q: *Quadrupole*), los analizadores de tiempo de vuelo (TOF: *Time Of Flight*) y los analizadores Orbitrap. (Thomas et al., 2022).

Un método común es el acoplamiento de dos espectrómetros de masas, consiguiendo un espectrómetro de masas en tándem (MS/MS). La espectrometría de masas en tándem se basa en la fragmentación de los metabolitos para facilitar su identificación y emplea aparatos de triple cuadrupolo (QqQ), cuadrupolo-tiempo de vuelo (QTOF), cuadrupolo-Orbitrap (Q-Trap) o tiempo de vuelo-tiempo de vuelo (TOF/TOF) (Novoa-Del-Toro & Witting, 2024). El primer analizador de masas filtra los iones entrantes, permitiendo que solo los iones precursores avancen hacia la celda de colisión. En esta celda, los iones precursores se fragmentan en iones producto, los cuales son medidos en el segundo analizador de masas. Este sistema se muestra esquematizado en la *Figura 1*.

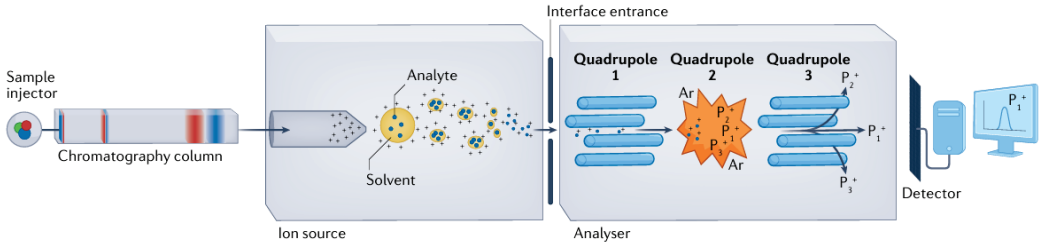

Figura 1. Funcionamiento básico de LC-MS (Thomas et al., 2022)

El espectro de masas resultante refleja únicamente los iones producto seleccionados y es de gran utilidad para la anotación de metabolitos. En la actualidad, la cromatografía líquida acoplada a espectrometría de masas en tándem (LC-MS/MS: *Liquid*

Chromatography – Tandem Mass Spectrometry) se ha consolidado como la tecnología por excelencia en los ensayos metabolómicos, gracias a la alta precisión y resolución que es capaz de ofrecer (Godzien et al., 2018) (Thomas et al., 2022).

La anotación de metabolitos se basa en la caracterización de datos de LC-MS/MS, identificando los picos con fórmulas moleculares o compuestos conocidos (Domingo-Almenara et al., 2018). Actualmente la anotación de metabolitos supone uno de los cuellos de botella más importantes en la disciplina. A pesar de los avances en la instrumentación, los niveles de precisión que se pueden alcanzar en la actualidad no permiten determinar fórmulas de manera inequívoca. Incluso una precisión de 1 ppm o menos sigue generando docenas de posibles fórmulas. Por esta razón, es importante ordenar o filtrar las fórmulas más probables y químicamente correctas. Solo el 1,8 % de los espectros en un experimento de metabolómica no dirigida consiguen ser anotados (Godzien et al., 2018) (Witting & Böcker, 2020a). Este cuello de botella provoca que las bases de datos de metabolómica pasen a tener una enorme importancia.

La dificultad de identificar un único compuesto a partir de información espectral ha llevado a la utilización de otros tipos de datos para complementar la búsqueda en bibliotecas espectrales como KEGG, HMDB, Lipid Maps o METLIN con el objetivo de obtener coincidencias (Godzien et al., 2018) (Nguyen et al., 2024). Entre estos datos complementarios encontramos el espectro de fragmentación, datos bioquímicos y rutas metabólicas (accesibles mediante bases de datos como MI-Pack o ProbMetab (Domingo-Almenara et al., 2018)) o el RT. Métodos de predicción de RT mediante diferentes aproximaciones están despertando un mayor interés en la actualidad debido a su gran utilidad como información complementaria a datos MS/MS para identificación de metabolitos (Witting & Böcker, 2020a).

1.3. Aprendizaje automático y redes neuronales profundas

El aprendizaje automático (ML: *Machine Learning*) es una rama de la inteligencia artificial (IA: *Inteligencia Artificial*) que se ocupa de la implementación de algoritmos computacionales que mejoran su rendimiento a partir de la experiencia; es decir, un sistema de machine learning aprende a partir de los datos (Pomyen et al., 2020). Busca establecer relaciones entre unos datos de input y unos datos de output, cuando la relación subyacente entre ambos no se conoce a partir de principios fundamentales (Meyer, 2021). Mientras que en la programación tradicional se aplican conjuntos de

reglas predefinidas (es decir, algoritmos) a los datos para producir un resultado deseado, en ML se utilizan los propios datos para entrenar un modelo (es decir, para derivar las reglas a partir de los datos), que luego puede aplicarse para hacer predicciones sobre datos similares (Jovel & Greiner, 2021). El aprendizaje automático puede a su vez dividirse en 3 tipos:

- Aprendizaje supervisado: Se dice que los datos están "etiquetados" ya que se conoce el valor de output que debe ser obtenido (etiqueta) a partir de los datos de entrada. El algoritmo busca las relaciones subyacentes entre las características de entrada y un output conocido. Este tipo de algoritmos aprende a encontrar patrones en las parejas atributos-etiqueta, lo que permite posteriormente realizar predicciones sobre datos no etiquetados. A su vez encontramos problemas de regresión, en los que se busca predecir un valor en un rango de valores continuos, y problemas de clasificación, en los que se busca predecir a qué clase pertenece una instancia dentro de un número finito de clases.
- Aprendizaje no supervisado: Los datos no se encuentran etiquetados. El algoritmo busca detectar patrones en un conjunto de datos y categorizar las instancias individuales según dichos patrones, sin necesidad de etiquetas. Se permite al algoritmo descubrir por sí mismo las relaciones subyacentes entre los datos, sin que dicha búsqueda esté guiada por una etiqueta.
- Aprendizaje por refuerzo: El algoritmo aprende mediante un proceso de prueba y error, recibiendo recompensas o penalizaciones en función de cómo sus actos han colaborado a alcanzar el objetivo. Busca maximizar las recompensas y minimizar las penalizaciones (Choi et al., 2020) (Jovel & Greiner, 2021).

Las redes neuronales artificiales (RNA) son algoritmos de ML inspirados en las interconexiones de las neuronas biológicas. Una RNA está compuesta por nodos (equivalentes al soma celular de una neurona) comunicados a través de conexiones (similares a las dendritas y axones neuronales) (Géron, 2019). De la misma manera que la intensidad de la sinapsis neuronal aumenta entre neuronas que se activan a la vez, a las conexiones entre los nodos de la red neuronal artificial se les asignan pesos en función de cuánto han contribuido a generar un valor de salida deseado. Los nodos de una RNA se organizan en capas: una capa de entrada, una capa de salida, y una o más capas ocultas intermedias. El valor que una neurona transmite a la siguiente capa en función de los valores recibidos de la capa previa se calcula mediante la función de

pérdida. Esta función lleva a cabo una transformación no lineal de los valores recibidos, permitiendo que la red pueda modelar relaciones no lineales presentes en los datos.

Las redes neuronales de aprendizaje profundo, también denominadas redes neuronales profundas (DNN: *Deep Neural Network*) pueden contener cientos, o incluso miles de capas ocultas entre la capa que recibe el vector con los datos de entrada y la capa que genera los datos de salida (Choi et al., 2020). Forman parte del campo del aprendizaje automático conocido como aprendizaje profundo (DL: *Deep Learning*).

Una DNN lleva a cabo múltiples iteraciones en las que busca patrones latentes entre los datos de entrada para generar una salida. Estas iteraciones son llamadas épocas, y en cada una de ellas el algoritmo busca obtener una salida más acertada que la anterior. El error en la predicción es medido por una función de pérdida. El algoritmo busca minimizar la función de pérdida ajustando los valores internos que controlan cómo se combinan las señales en cada neurona. Estos parámetros, llamados pesos, determinan la influencia que tiene cada neurona de la capa anterior sobre la salida de cada neurona de la capa siguiente. Durante el entrenamiento, la red calcula el vector gradiente, que contiene el valor de la derivada parcial de la función de pérdida con respecto al peso de cada neurona de la capa anterior. Variando los pesos en la dirección opuesta al gradiente se minimiza la función de pérdida. La tasa de aprendizaje determina la magnitud de la variación de los pesos. Todos aquellos parámetros no minimizables, como la función de activación o el número de capas, reciben el nombre de hiperparámetros. El algoritmo deja de iterar este proceso cuando llega a un número predeterminado de épocas o hasta que la función de pérdida deja de disminuir (Géron, 2019). La calidad de los modelos de regresión comúnmente es cuantificada por métricas como los errores MAE (*Mean Average Error*), MSE (*Mean Square Error*) o RMSE (*Root Mean Square Error*) que permiten cuantificar la diferencia promedio entre los valores predichos y los reales.

Para que un algoritmo generalice de manera óptima ante datos no vistos previamente se emplea una división de los datos, creándose conjuntos de datos de entrenamiento y de prueba (generalmente 80%-20%). El algoritmo solo será entrenado con el conjunto de entrenamiento y evaluado con el conjunto de prueba. En ocasiones puede emplearse un conjunto de validación para llevar a cabo optimización de hiperparámetros. Esta división nos permite identificar 2 problemas en nuestro algoritmo:

- Sobreajuste (*overfitting*): Se dice que el modelo está sobreajustado cuando presenta un rendimiento considerablemente superior sobre los datos de entrenamiento que sobre los de prueba. Se produce cuando el algoritmo ha encontrado patrones específicos del conjunto de entrenamiento, provocando pérdida de capacidad de generalizar a otros conjuntos de datos.
- Subajuste (*underfitting*): Un modelo subajustado tiene un mal rendimiento tanto en el conjunto de entrenamiento como en el de prueba. Se debe a una pobre extracción de relaciones latentes entre los datos (Choi et al., 2020; Géron, 2019).

1.4. Aprendizaje por transferencia y reducción de dimensionalidad

El éxito de los algoritmos supervisados de ML, especialmente de aprendizaje profundo, es altamente dependiente de la disponibilidad de gran cantidad de datos etiquetados, uniformes y de calidad. En muchas disciplinas la obtención de datos puede ser difícil, lenta o económicamente exigente. La capacidad de extracción de características latentes que son capaces de llevar a cabo las DNNs se ve considerablemente mermada en situaciones de falta de datos (Tan et al., 2018). El éxito de las técnicas tradicionales de aprendizaje automático depende en gran medida de contar con muchos ejemplos de entrenamiento y está limitado por la suposición de que los datos de entrenamiento y de prueba pertenecen al mismo dominio y tienen la misma distribución (Farahani et al., 2021). Este problema consigue verse en cierta medida solucionado por el aprendizaje semi-supervisado (Zhuang et al., 2020), una modalidad híbrida entre el aprendizaje supervisado y el no supervisado que emplea una pequeña cantidad de datos etiquetados y una gran cantidad de datos no etiquetados para desarrollar la capacidad predictiva. Sin embargo, los propios datos no etiquetados pueden llegar a ser difíciles de obtener, haciendo que estos modelos no remedien el problema.

El aprendizaje por transferencia (TL: *Transfer Learning*) es una disciplina del aprendizaje automático que surgió con el objetivo de eliminar el desafío de la falta de datos. Se basa en la reutilización de modelos preexistentes para mejorar el rendimiento de nuevos modelos. Soluciona problemas de falta de datos en un modelo nuevo utilizando uno o más modelos previos dedicados a tareas similares sin problemas de escasez de datos (Zhuang et al., 2020).

En aprendizaje por transferencia, se denomina dominio al tipo de datos que usa el modelo y la manera en la que se encuentran distribuidos. Se denomina tarea a la labor u objetivo de predicción que el modelo lleva a cabo. Existen 3 tipos de estrategias de TL:

- TL inductivo: Los modelos difieren en su tarea a pesar de tener dominios similares.
- TL transductivo: Los modelos tienen la misma tarea, pero presentan diferentes dominios.
- TL no supervisado: Difiere de las otras 2 estrategias al no utilizar ninguno de los modelos datos etiquetados. Se emplea en tareas no supervisadas, como el agrupamiento (*clustering*) o la reducción dimensional de conjuntos de datos.

Una práctica común en TL transductivo consiste en reutilizar una red neuronal profunda previamente entrenada, o una parte sustancial de sus capas. Habitualmente se descartan la capa de salida y las capas ocultas superiores, ya que las características de alto nivel más relevantes para la nueva tarea pueden diferir significativamente de las que fueron útiles en la tarea original. Cuanto mayor sea la similitud entre ambas tareas mayor será el número de capas que podrán reutilizarse. Se añaden nuevas capas y se entrena la red manteniendo congelados los pesos de las capas reusadas (Géron, 2019). Finalmente, es posible descongelar progresivamente algunas de estas capas, habitualmente acompañado de una reducción de la tasa de aprendizaje, con el fin de afinar el modelo y mejorar su rendimiento en la tarea objetivo.

La reducción de dimensionalidad es una metodología del transfer learning no supervisado que busca la minimización de dimensiones de un conjunto de datos, preservando la mayor cantidad posible de la información original. Lo consigue conservando las características más relevantes en la descripción del conjunto de datos (Kim et al., 2024), llamada información de alto nivel. Se diferencian métodos lineales y no lineales. Los métodos no lineales son menos interpretables, pero presentan mayor rendimiento al extraer características no lineales en el conjunto de datos, creando compresiones más informativas. La reducción de dimensionalidad tiene aplicaciones en numerosos ámbitos, como la selección de características, la eficiencia computacional, el almacenamiento de datos y la reducción de ruido en la visualización de datos.

El autoencoder (AE) es una estructura empleada en reducción de dimensionalidad no lineal mediante aprendizaje por transferencia. Los autoencoders son una DNN compuesta por un encoder y un decoder. En cada capa del encoder se reduce gradualmente el tamaño

de la información, comprimiéndola hasta llegar a la capa que conecta al encoder con el decoder, que recibe el nombre de espacio latente. Posteriormente, el decoder pretende reconstruir los datos originales a partir de la salida del encoder (Fong & Xu, 2021).

1.5. Aprendizaje automático en la anotación de metabolitos

El aprendizaje profundo es uno de los métodos mediante los que se ha pretendido abordar el problema de la falta de datos en metabolómica. Su uso en cromatografía líquida para predicción de RTs es relativamente reciente, y aún no ha alcanzado la precisión necesaria para tener una adopción generalizada (Yang, Ji, Lu, et al., 2021). Un método de predicción de RTs con bajo error supondría un gran avance en el campo de la metabolómica al evitar mediciones experimentales complejas, costosas y en ocasionas de baja reproducibilidad (Sen et al., 2021).

Cuando se emplea aprendizaje automático como método de predicción de RTs, se deben representar las características y naturaleza del analito en forma de datos útiles. Se diferencian 2 tipos de datos: descriptores moleculares y *fingerprints*.

Los *fingerprints* son datos binarios que reflejan la ausencia o presencia de una característica en la molécula. Los descriptores moleculares son un valor cuantitativo que refleja una propiedad fisicoquímica del analito. Se obtienen mediante la aplicación de una función matemática sobre una representación bien definida de la estructura del analito (Liapikos et al., 2022). Pueden ser obtenidos mediante numerosos softwares (Dragon software, VolSurf+, RDKit, ChemoPy). Dependiendo de la representación elegida diferenciamos 4 tipos de descriptores moleculares:

- Descriptores de 0 dimensiones: Reflejan propiedades dependientes del número de átomos y el peso molecular.
- Descriptores de 1 dimensión: Reflejan propiedades relacionadas con el número y tipo de grupos funcionales presentes.
- Descriptores de 2 dimensiones: Reflejan propiedades de la molécula completa representada en forma de grafo; composición atómica y conexiones entre átomos.
- Descriptores de 3 dimensiones: Reflejan propiedades de la posición y conexiones de los átomos en el espacio tridimensional (Mauri, 2020).

En la actualidad el cuello de botella en el uso de aprendizaje automático para predicción de RTs sigue siendo la falta de datos. Varios autores han hecho esfuerzos por crear bases

de datos de RTs con relativo éxito. RepoRT es la base de datos de mayor dimensión, con más de 160.000 moléculas con sus respectivos RTs (Kretschmer et al., 2024), sin embargo, su uso presenta la dificultad de que los experimentos que contiene emplean diferentes columnas y procedimientos cromatográficos. SMRT (*Small Molecule Retention Time*) es el conjunto de datos correspondientes a un mismo método cromatográfico de mayor tamaño, con más de 80.000 moléculas. Pertenece a la biblioteca METLIN (Domingo-Almenara et al., 2018), pero sus datos corresponden a moléculas mayoritariamente sintéticas. Esto puede dificultar la predicción de modelos entrenados en SMRT sobre moléculas naturales. SMRT se encuentra contenido dentro de RepoRT.

Recientemente, integrantes del grupo de investigación en el que se ha desarrollado este proyecto llevaron a cabo avances que contribuyeron de forma sustancial al estado del arte en la disciplina. En el trabajo de Amil Manjón (2024), se modeló como vector de características el método cromatográfico de todas las moléculas de RepoRT. Esto permite integrar en el vector de entrada a la red neuronal la información de las condiciones cromatográficas de cada uno de los 375 experimentos diferentes que componen RepoRT. Además, se modelaron *fingerprints* y descriptores moleculares para todas las moléculas.

Las estrategias de proyección constituyen un enfoque utilizado para entrenar modelos capaces de predecir los tiempos de retención en una determinada condición cromatográfica siendo entrenados con datos de una condición cromatográfica distinta. Utilizan procedimientos matemáticos para relacionar los RT entre ambos métodos cromatográficos. Empleando diferentes separaciones de los datos generados, Amil Manjón (2024) consiguió un error MAE mínimo de 27.47 segundos, que mejoró los obtenidos en otros experimentos (García et al., 2022; Osipenko et al., 2020; Yang, Ji, Lu, et al., 2021) que empleaban estrategias de proyección.

Por otra parte, el aprendizaje por transferencia ha sido empleado con anterioridad en la predicción de tiempos de retención (Fedorova et al., 2022; Yang, Ji, Fan, et al., 2021; Yang, Ji, Lu, et al., 2021). Sin embargo, únicamente se han empleado procedimientos de TL transductivo, en los que se entrena un modelo de predicción con una base de datos para posteriormente congelar las capas y predecir sobre otra base de datos de moléculas medidas con condiciones cromatográficas diferentes. El uso de modelos de aprendizaje por transferencia transductivo ha demostrado una capacidad predictiva superior en comparación con los modelos entrenados desde cero. Esta metodología, aplicada al

diseño de redes de aprendizaje profundo, demuestra poder adquirir una notable capacidad de generalización.

Sin embargo, no hay precedentes en la literatura sobre el uso de TL no supervisado para predicción de RTs. La única referencia que emplea una metodología parecida ha aparecido en un artículo de reciente publicación. Bittremieux & Noble (2025) desarrollaron un modelo de aprendizaje autosupervisado capaz de generar un espacio latente de 1024 componentes a partir de un conjunto de 24 millones de espectros MS/MS mediante un autoencoder de preentrenamiento. Esa representación latente se usó acoplada a decoders de distintos tipos que permitieron llevar a cabo con gran éxito tareas de diversa índole, como la generación de *fingerprints*, la predicción de presencia de flúor o el cálculo de similitud espectral. Este ensayo ha demostrado el inmenso potencial que tiene el TL no supervisado en la predicción de propiedades moleculares cuando el conjunto de datos es lo suficientemente grande como para crear una representación latente rica, generalizable y químicamente informativa.

Este proyecto tiene como objetivo estudiar la viabilidad y calidad predictiva de modelos de DNNs basados en aprendizaje por transferencia no supervisado para la predicción de RTs. Esta meta precisará de varios pasos:

- Preentrenamiento del modelo a través de una tarea relacionada: Se empleó la reconstrucción de los datos de entrada (*fingerprints* y descriptores moleculares, por separado) mediante autoencoders como tarea de preentrenamiento. Se plantea la hipótesis de que una reducción dimensional cercana al 10 % del tamaño original permitiría generar un espacio latente capaz de capturar representaciones de alto nivel.
- Acoplamiento de *redes de predicción* con diversas arquitecturas al espacio latente de los autoencoders con mejor desempeño en la tarea de reconstrucción, buscando un modelo exitoso en la predicción de RTs.
- Generar un modelo que emplee como datos de entrada tanto *fingerprints* como descriptores moleculares. Para ello se integrarán en un modelo único el modelo de mayor éxito para cada uno de los tipos de datos.
- Comparación de los resultados obtenidos por los modelos preentrenados con los aquellos obtenidos por las *arquitecturas símil* (modelos con arquitectura homóloga a las *redes de predicción,* pero carentes del encoder preentrenado).

CAPÍTULO 2: MATERIALES Y MÉTODOS

2.1. Repositorio de GitHub y datos de RepoRT

El código empleado para la realización de este proyecto puede ser encontrado en un repositorio público de GitHub (Lorenzo, 2025).

Para este proyecto fue de gran relevancia el uso de datos del repositorio público RepoRT (Kretschmer et al., 2024). Este repositorio contiene 375 conjuntos de datos con información sobre los tiempos de retención (RT) de moléculas obtenidos en diversos experimentos cromatográficos. Todas las moléculas de una misma carpeta fueron medidas bajo las mismas condiciones cromatográficas (columna, eluyentes, gradientes de elución, longitud, temperatura, fase estacionaria…), que también aparecen como datos. RepoRT es el repositorio de mayor tamaño de datos de moléculas y sus tiempos de retención; con más de 160.000 moléculas, y uno de los pocos que integran información metodológica procedente de múltiples experimentos cromatográficos. A su vez, RepoRT contiene el conjunto de datos METLIN SMRT (Amil Manjón, 2024a). SMRT es el conjunto de moléculas medidas bajo las mismas condiciones cromatográficas de mayor tamaño. La mayoría de los compuestos de SMRT son moléculas pequeñas similares a fármacos, pero también contiene moléculas naturales.

El repositorio creado por Amil Manjón (2024) fue clonado y empleado como punto de partida, asentando el material base a partir del cual se produjo la evolución de esta línea de investigación. Este repositorio contiene el código a través del cual se lleva a cabo un preprocesamiento de la base de datos de RepoRT, modelando un vector de características del método cromatográfico, así como uno de *fingerprints* y uno de descriptores moleculares para cada molécula. Tras este preprocesamiento, las más de 160.000 moléculas de RepoRT se vieron reducidas a 70.029. Los compuestos que se eliminaron no contaban con la información suficiente para su representación computacional. De estas 70.029 moléculas, 52.641 corresponden al conjunto de datos SMRT.

El código del repositorio también lleva a cabo una identificación de cada molécula por su código SMILES (Simplified Molecular Input Line Entry Specification), InChi (IUPAC International Chemical Identifier) y un identificador propio de RepoRT. También se encuentran clasificadas taxonómicamente a través del servidor web ClassyFire (Djoumbou Feunang et al., 2016).

Tras ser preprocesados se obtuvieron 2212 *fingerprints* y 1977 descriptores moleculares por molécula. Para este proyecto no se emplearon los datos del método cromatográfico.

2.2. Preentrenamiento de la red en una tarea relacionada

El uso de autoencoders para la reducción de dimensionalidad de un conjunto de datos es una práctica común que permite reducir el tamaño de los datos manteniendo información sobre las características más relevantes (Fong & Xu, 2021). Se llevó a cabo una aproximación basada en el preentrenamiento de diversos modelos de autoencoders para la reconstrucción de los datos de entrada. Se diseñaron múltiples modelos de autoencoder buscando reducir la dimensión del conjunto de datos hasta generar un espacio latente equivalente a aproximadamente el 10% de su tamaño original, con el objetivo de reconstruir posteriormente los datos de entrada a partir de dicho espacio latente con la mayor fidelidad posible. Para el entrenamiento de todos los modelos los datos fueron divididos en conjuntos de entrenamiento y validación de manera aleatoria en una relación 80%-20%.

2.2.1. Creación de un espacio latente para los *fingerprints*

El desarrollo de un modelo predictor de RTs entrenado con *fingerprints* comenzó con pruebas de distintos autoencoders sobre los que se fueron modificando hiperparámetros, arquitectura, número de capas o tamaño del espacio latente, entre otros.

Para el entrenamiento de autoencoders se utilizó el conjunto de datos de RepoRT preprocesado. El vector de datos de entrada de *fingerprints* contaba con 2.212 características. Por ello, todas las arquitecturas emplearon una capa de entrada de 2212 neuronas para asegurar una óptima extracción de características de los datos de entrada. Denominamos *estructura básica* a la siguiente arquitectura: 2212/2200/550/280/140/280/550/1100/2212. El espacio latente de 140 neuronas de la arquitectura básica representa el 6,3% del tamaño original de los datos de entrada.

Se comenzó empleando *Adam* como optimizador con una tasa de aprendizaje de 0.001, $\beta1 = 0.9$ y $\beta2 = 0.999$. Se usó la entropía cruzada (*cross_entropy*) como función de pérdida, *glorot_normal* como inicializador del kernel y *selu* como función de activación. También se introdujo el callback *EarlyStopping*, que finaliza el entrenamiento si la función de pérdida no disminuye durante 10 épocas.

Adam solo fue empleado para los 2 primeros modelos. De ese punto en adelante se comenzó a utilizar *RMSprop* debido a que, a diferencia de *Adam*, no utiliza momento. El momento es un método para suavizar el gradiente en el tiempo, pues para cada iteración, un optimizador con momento tiene en cuenta el valor de los gradientes en pasos previos (Géron, 2019). Sin embargo, si el conjunto de datos es poco uniforme, como era el caso al haber *fingerprints* que eran mayoritariamente 0 o mayoritariamente 1, el uso de un optimizador con momento empeora la minimización de la función de pérdida y por ello la predicción. El paso a *RMSprop* redujo el error con respecto a *Adam*.

Se buscó mejora mediante un nivel de reducción dimensional menor, por lo que se testeó un espacio latente de 280 neuronas; un 12,65% del tamaño del input original. También se probaron modelos en los que se duplicó, triplicó e incluso cuadruplicó cada capa o algunas de las capas. El objetivo de ello era conseguir que las capas extrajesen relaciones no lineales de los datos antes de enviar esa información a la siguiente capa del autoencoder, reduciendo el tamaño. Se pretendía dar a la red pasos intermedios para procesar y refinar las representaciones, creando datos con mayor nivel de abstracción y buscando un espacio latente más complejo; con información de mayor nivel.

También se realizó el cambio de *glorot_normal* a *lecun_normal* en el inicializador del kernel al estar más indicado para *selu*. Se empleó la activación *selu* junto con la inicialización *lecun_normal* con el objetivo de aprovechar sus propiedades de autonormalización, especialmente útiles en redes sin técnicas de normalización explícitas. Sin embargo, se exploró también el uso de *relu* en combinación con *he_normal*, ya que esta activación es más robusta frente al problema del desvanecimiento del gradiente y suele ofrecer una mayor velocidad de convergencia en muchos casos. Del autoencoder 11 en adelante se introdujo una métrica llamada *metric_common_zeros_ones* que devuelve el porcentaje de *fingerprints* reconstruidos exitosamente. También se introdujo el callback *ReduceLROnPlateau* de *Keras* para reducir la tasa de aprendizaje a la mitad si no había mejora por 3 épocas seguidas.

El último factor que se modificó fue la simetría de la arquitectura, utilizando modelos con decoder asimétrico: 2212/1100/550/280/140/2212/2212. Las características de los 23 autoencoders diferentes que se emplearon quedan reflejadas en la *Tabla 1*.

Tabla 1. Características de los modelos de Autoencoder de fingerprints

Autoencoder	Modificación de arquitectura	Capas repetidas	Algoritmo	Kernel + Activación
1	Estructura básica	No	*Adam*	*glorot_normal + selu*
2	Estructura básica	550 (X2)	*Adam*	*glorot_normal + selu*
3	Estructura básica	No	*RMSprop*	*glorot_normal + selu*
4	Espacio latente de 280 neuronas	No	*RMSprop*	*glorot_normal + selu*
5	Estructura básica	X2	*RMSprop*	*glorot_normal + selu*
6	Estructura básica	X3	*RMSprop*	*glorot_normal + selu*
7	Estructura básica	2212, 1100 (X2)	*RMSprop*	*glorot_normal + selu*
8	Estructura básica	X4	*RMSprop*	*glorot_normal + selu*
9	Estructura básica	2212, 550 (X2)	*RMSprop*	*glorot_normal + selu*
10	Estructura básica	2212, 1100 (X2)	*RMSprop*	*glorot_normal + selu*
11	Estructura básica	No	*RMSprop*	*lecun_normal + selu*
12	Decoder 2212/2212	No	*RMSprop*	*lecun_normal + selu*
13	Estructura básica	X2	*RMSprop*	*lecun_normal + selu*
14	Estructura básica	X3	*RMSprop*	*lecun_normal + selu*
15	Decoder 2212/2212	X2	*RMSprop*	*lecun_normal + selu*
16	Decoder 2212/2212	X3	*RMSprop*	*lecun_normal + selu*
17	Estructura básica	No	*RMSprop*	*he_normal + relu*
18	Decoder 2212/2212	No	*RMSprop*	*he_normal + relu*
19	Estructura básica	No	*RMSprop*	*he_normal + selu*
20	Decoder 2212/2212	No	*RMSprop*	*he_normal + selu*
21	Espacio latente de 550 neuronas	No	*RMSprop*	*he_normal + selu*
22	Espacio latente de 280 neuronas	No	*RMSprop*	*lecun_normal + selu*
23	Espacio latente de 280 neuronas	X2	*RMSprop*	*glorot_normal + selu*

2.2.2. Creación de un espacio latente para los descriptores moleculares

Para la creación de redes predictoras de RT empleando descriptores moleculares como datos de entrada se llevó a cabo el mismo proceso experimental. Los descriptores se procesaron con la función *DescriptorsPreprocessor* del repositorio de GitHub de Amil Manjón (2024). Mediante esta función los descriptores fueron normalizados con la clase de sklearn *StandardScaler* para mantener todos los valores en la misma escala. Posteriormente se realizó la imputación de los valores faltantes con la mediana mediante

otra de las clases de sklearn; *SimpleImputer*. Se eliminaron las características que poseían una varianza igual a 0 con la clase *VarianceThreshold* y también se eliminaron las características que poseían un grado de correlación por encima de un umbral de 0.9, es decir, características altamente correlacionadas entre ellas.

El vector de datos de entrada de descriptores moleculares contaba con 1.977 características. Siguiendo la lógica de los autoencoders de *fingerprints*, se creó una *estructura básica* en la que se reducía aproximadamente a la mitad el número de neuronas en cada capa: 1977/985/493/246/123/246/493/985/1977. El espacio latente de 123 neuronas representaba el 6,22% de los datos de entrada. Se empleó el error MAE como función de pérdida.

Siguiendo el procedimiento empleado con *fingerprints*, también se buscó el mejor modelo empleando *Adam* y *RMSprop*, así como con *relu* + *he_normal* y *selu* + *lecun_normal*. A su vez se buscó mejora mediante un aumento del espacio latente a 246 neuronas (12,44% del tamaño de los datos de entrada). Se crearon únicamente 8 modelos en comparación a los 23 del ensayo con *fingerprints*. Este menor número de autoencoders se debe a que se descartó la búsqueda de mejora mediante arquitecturas asimétricas y se indagó en menor medida en la hipótesis de duplicar capas debido a que no ofrecieron un desempeño superior en el ensayo con *fingerprints* (*Tabla 2*).

Tabla 2. Características de los modelos de Autoencoder de descriptores moleculares

Autoencoder	Modificación de arquitectura	Capas repetidas	Algoritmo	Kernel + Activación
1	Estructura básica	No	*RMSprop*	*lecun_normal + selu*
2	Estructura básica	No	*RMSprop*	*he_normal + relu*
3	Estructura básica	No	*Adam*	*lecun_normal + selu*
4	Estructura básica	No	*Adam*	*he_normal + relu*
5	Estructura básica	X2	*RMSprop*	*lecun_normal + selu*
6	Espacio latente de 246 neuronas	No	*RMSprop*	*lecun_normal + selu*
7	Espacio latente de 246 neuronas	X2	*RMSprop*	*lecun_normal + selu*
8	Estructura básica	No	*RMSprop*	*glorot_normal + selu*

2.3. Predicción de RT a partir de las redes preentrenadas

Se realizaron pruebas de predicción de RT empleando el encoder de aquellos autoencoders que ofrecieron buenos resultados en la reconstrucción de los datos. Para

ello, se congelaron los pesos de las neuronas del encoder y a su salida se añadió una *red de predicción* que generaba como output la predicción de RT. Se llevó a cabo la predicción sobre el conjunto SMRT y se estableció como función de pérdida el error MAE. Se llevó a cabo una división aleatoria 80%-20% en entrenamiento y validación tanto para los datos de entrada como para los RTs. En todas las pruebas se normalizaron los RTs con la clase *StandardScaler*. La *red de predicción* más simple empleada fue una única neurona de activación lineal, que permitió analizar la calidad predictiva del encoder en comparación con los datos por sí solos. Además de la arquitectura de una sola neurona con activación lineal, en todos los ensayos de predicción se evaluaron cuatro arquitecturas más complejas. La *Figura 2* refleja estas cuatro arquitecturas de *red de predicción* empleadas en este trabajo. En todas ellas, los datos de entrada son transformados previamente mediante el encoder con pesos congelados, cuya salida es un espacio latente de tamaño "A", como aparece representado en la *Figura 2*.

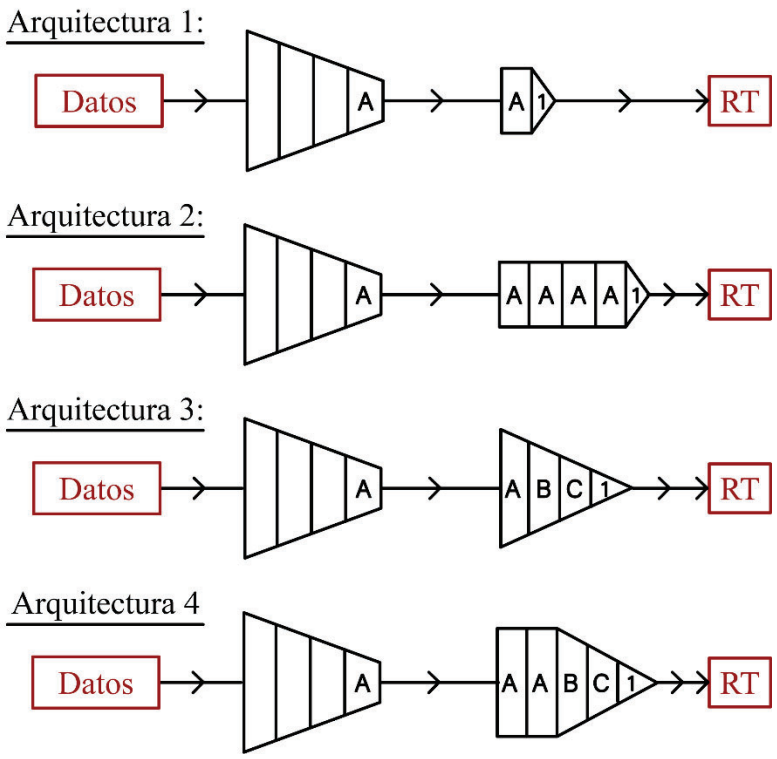

Figura 2. Diagramas de las 4 arquitecturas de red de predicción empleadas en este trabajo.

En particular, en las Arquitecturas 3 y 4, la *red de predicción* presenta una estructura con reducción de tamaño progresiva. En ellas, la capa de tamaño B representa la mitad del tamaño de la capa A, y la capa de tamaño C representa la mitad del tamaño de la capa B.

2.3.1. Predicción de RT en base al espacio latente de *fingerprints*

Los encoders que fueron seleccionados fueron el 4, el 11, el 12, el 18 y el 22. Los tiempos de retención se normalizaron con la clase de sklearn *StandardScaler*. También se buscó la función con mejor resultado entre *Adam* y *RMSprop*. Las características de los modelos vienen reflejadas en la *Tabla 3*.

Tabla 3. Características de los modelos de predicción empleando fingerprints.

Modelo	Encoder	Red de predicción	Algoritmo	Kernel + Activación
1	12	1 (Neurona lineal)	*RMSprop*	*lecun_normal + selu*
2	12	1 (Neurona lineal)	*Adam*	*lecun_normal + selu*
3	18	1 (Neurona lineal)	*RMSprop*	*lecun_normal + selu*
4	4	1 (Neurona lineal)	*Adam*	*lecun_normal + selu*
5	4	280/1	*Adam*	*lecun_normal + selu*
6	4	280 (X4) /1	*Adam*	*lecun_normal + selu*
7	4	280/140/70/1	*Adam*	*lecun_normal + selu*
8	4	280/140/70/1	*RMSprop*	*lecun_normal + selu*
9	4	280/280/140/70/1	*Adam*	*lecun_normal + selu*
10	11	140 (X4)/1	*Adam*	*lecun_normal + selu*
11	11	140/140/70/35/1	*Adam*	*lecun_normal + selu*
12	12	140 (X4)/1	*Adam*	*lecun_normal + selu*
13	12	140/140/70/35/1	*Adam*	*lecun_normal + selu*
14	22	280 (X4)/1	*Adam*	*lecun_normal + selu*
15	22	280/140/70/1	*Adam*	*lecun_normal + selu*
16	22	280/280/140/70/1	*Adam*	*lecun_normal + selu*
17	4	280 (X2) /140 (X2) /70/1	*Adam*	*lecun_normal + selu*
18	11	140 (X2) /70 (X2) /35/1	*Adam*	*lecun_normal + selu*

2.3.2. Predicción de RT en base al espacio latente de descriptores moleculares

Se seleccionaron los encoders de los autoencoders 1 y 6 por haber ofrecido los mejores resultados. Los descriptores fueron preprocesados mediante la función *DescriptorsPreprocessor* y los RTs fueron normalizados con la clase de sklearn *StandardScaler*. Las particularidades de cada modelo vienen reflejadas en la *Tabla 4*.

Tabla 4. Características de los modelos de predicción empleando descriptores moleculares.

Modelo	Encoder	Red de predicción	Algoritmo	Kernel + Activacion
1	1	1	*RMSprop*	*lecun_normal + selu*
2	6	1	*RMSprop*	*lecun_normal + selu*
3	1	123/1	*RMSprop*	*lecun_normal + selu*
4	6	246/1	*RMSprop*	*lecun_normal + selu*
5	1	123 (X4) /1	*RMSprop*	*lecun_normal + selu*
6	6	246 (X4) /1	*RMSprop*	*lecun_normal + selu*
7	1	123/61/30/1	*RMSprop*	*lecun_normal + selu*
8	6	246/123/61/1	*RMSprop*	*lecun_normal + selu*
9	1	123/123/61/30/1	*RMSprop*	*lecun_normal + selu*
10	6	246/246/123/61/1	*RMSprop*	*lecun_normal + selu*

2.3.3. Predicción de RT en base a los espacios latentes combinados

El objetivo final fue la creación de un modelo que emplease tanto los *fingerprints* como los descriptores moleculares para predecir tiempos de retención, y analizar mediante los resultados su viabilidad con respecto al uso de estos por separado. Se crearon así modelos híbridos empleando 2 encoders; uno de *fingerprints* y otro de descriptores. Todos ellos emplearon *RMSprop*, *selu* y *lecun_normal*. Se normalizaron los tiempos de retención con la clase de sklearn *StandardScaler* y se procesaron los descriptores moleculares con la función *DescriptorsPreprocessor.*

Al ser el ensayo con mayor cantidad de datos, se buscó no únicamente analizar la viabilidad del uso de TL no supervisado, sino optimizar la minimización y mejorar los resultados en mayor medida. Para ello, se aumentó la paciencia de *EarlyStopping* a 5 y la de *ReduceLROnPlateau* a 15. También se utilizó *RMSprop* con una tasa de aprendizaje más reducida (0.0001), *rho* = 0.7 y *momentum* = 0.7.

La salida (espacio latente) de cada uno de los encoders se concatenó empleando la API funcional de Keras, creando un espacio latente conjunto que sería tomado como datos de entrada por la *red de predicción*. Se emplearon 2 encoders diferentes de *fingerprints* y 2 de descriptores, con el objetivo de experimentar con diferentes tamaños de espacio latente. Las diferentes combinaciones de encoders y de arquitecturas de la red neuronal de predicción dieron lugar a 20 modelos diferentes (*Tabla 5*). La *Figura 3* muestra la arquitectura del Modelo 15, con espacio latente conjunto de 403 características.

Tabla 5. Características de los modelos de predicción empleando fingerprints y descriptores moleculares.

Modelo	Encoder de *fingerprints*	Encoder de descriptores	Tamaño del espacio latente conjunto	Arquitectura de predicción
1	4	6	526	1 (Neurona lineal)
2	11	6	386	1 (Neurona lineal)
3	4	1	403	1 (Neurona lineal)
4	11	1	263	1 (Neurona lineal)
5	4	6	526	526/1
6	11	6	386	386/1
7	4	1	403	403/1
8	11	1	263	263/1
9	4	6	526	526(X4) /1
10	11	6	386	386(X4) /1
11	4	1	403	403(X4) /1
12	11	1	263	263(X4) /1
13	4	6	526	526/263/131/1
14	11	6	386	386/193/96/1
15	4	1	403	403/201/100/1
16	11	1	263	263/131/65/1
17	4	6	526	526/526/263/131/1
18	11	6	386	386/386/193/96/1
19	4	1	403	403/403/201/100/1
20	11	1	263	263/263/131/65/1

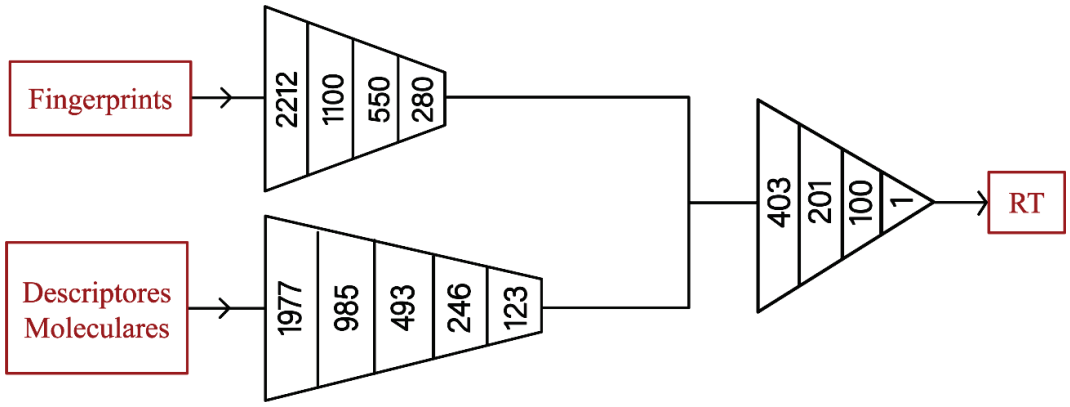

Figura 3. Arquitectura del Modelo 15. El espacio latente de salida de cada uno de los encoders se concatena para dar lugar al espacio latente conjunto; que será el input de la red de predicción.

2.4. Construcción de arquitecturas equivalentes

Se buscó evaluar el impacto real del encoder preentrenado sobre el rendimiento predictivo. Para averiguar si la aproximación a través de TL no supervisado era verdaderamente útil en la tarea de predicción, se construyeron modelos equivalentes que sirvieran como referencia comparativa. Para ello, se diseñó una arquitectura de red

neuronal idéntica a cada una de las utilizadas en combinación con el encoder, pero sin incluir este último, de modo que los resultados obtenidos pudieran atribuirse de forma específica a la contribución del espacio latente generado. Esta aproximación permitió comparar de manera controlada ambas configuraciones y analizar si la inclusión del encoder mejoraba la predicción. Con el objetivo de facilitar la comprensión de este proyecto se ha otorgado el nombre de *arquitectura símil* a cada uno de estos modelos. Se llevó a cabo este proceso para modelos de *fingerprints*, de descriptores moleculares, y de espacios latentes combinados. La *Figura 4* muestra una ejemplificación de un modelo predictor y su *arquitectura símil*.

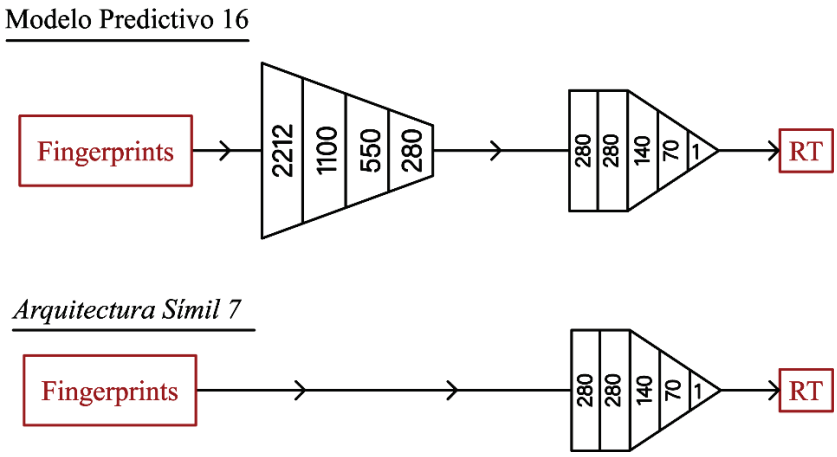

Figura 4. Modelo 16 (fingerprints) y su arquitectura símil.

CAPÍTULO 3: RESULTADOS Y DISCUSIÓN

3.1. Preentrenamiento: Reconstrucción de datos con Autoencoders

La reconstrucción de los *fingerprints* fue evaluada a través de la entropía cruzada (*cross_entropy*), el error MSE y la métrica *metric_common_zeros_ones*, que indica el % de datos correctamente reconstruidos. Este proceso experimental obtuvo resultados de considerable éxito. Todos los modelos de autoencoder reconstruyeron un porcentaje superior al 99,1% de los *fingerprints*, siendo la media 99,872% y el mejor valor 99,999%, obtenido por el autoencoder 21, de espacio latente de 550 neuronas, que representaría un único error entre las 2212 características. Dentro del éxito que presentaron todos los modelos, los autoencoders de espacio de latente de 280 neuronas (99,994%, *cross_entropy* = 0,0167 y MSE = 0,0017) y los de espacio latente de 140 neuronas asimétricos (99,928%, *cross_entropy* = 0,0160 y MSE = 0,0017) presentaron rendimientos ligeramente superiores a los de espacio latente de 140 neuronas simétricos (99,775%, *cross_entropy* = 0,0394 y MSE = 0,002). La *Figura 5* muestra la evolución de las métricas de entrenamiento y validación del autoencoder 4.

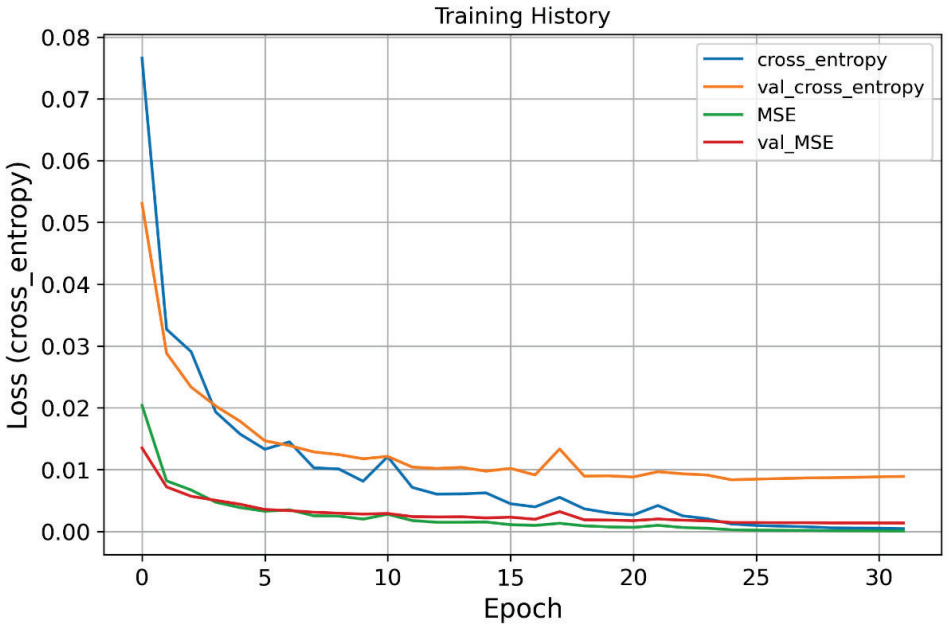

Figura 5. Rendimiento del autoencoder 4. Representación de la función de pérdida (cross_entropy) y MSE a lo largo de las distintas épocas de entrenamiento.

Con respecto a la hipótesis de que repetir varias o todas las capas pudiera permitir capturar relaciones no lineales más profundas en los datos antes de proceder a la reducción dimensional en las capas siguientes del autoencoder, esta no tuvo éxito. Ningún autoencoder con capas duplicadas consiguió un mejor rendimiento que su homólogo sin

ellas. En concreto, el error aumentaba a medida que se repetía más veces una misma capa. La *Tabla 6* ejemplifica estos resultados mediante los autoencoders 12, 15 y 16. Estos autoencoders eran idénticos en hiperparámetros y arquitectura, a excepción de la repetición de capas.

Tabla 6. Variación de las métricas de error en función de las capas repetidas

Autoencoder	Arquitectura	Capas repetidas	*cross entropy*	*metric_common_zeros_ones*	*MSE*
12	Decoder 2212/2212	No	0.01	99,998	0.001
15	Decoder 2212/2212	X2	0.015	99,942	0.002
16	Decoder 2212/2212	X3	0.031	99,723	0.005

En el preentrenamiento de reconstrucción de descriptores moleculares se emplearon como métricas de error MAE y MSE. Los autoencoders ofrecieron valores de MSE mayores que con respecto al uso de *fingerprints.* La repetición de capas también provocó desmejora de los resultados. De entre los múltiples autoencoders que se evaluaron, los que mejores métricas ofrecieron fueron el 1 y el 6 (*Tabla 7*). Al igual que en los autoencoders de *fingerprints*, un espacio latente de mayor tamaño redujo el error.

Tabla 7. Resultados de los modelos de Autoencoder empleando descriptores

Autoencoder	Modificación de arquitectura	Capas repetidas	Algoritmo	Kernel + Activación	MAE	MSE
1	Estructura básica	No	*RMSprop*	*lecun + selu*	0.066	0.296
2	Estructura básica	No	*RMSprop*	*he_normal + relu*	0.096	0.363
3	Estructura básica	No	*Adam*	*lecun + selu*	0.098	0.371
4	Estructura básica	No	*Adam*	*he_normal + relu*	0.126	0.417
5	Estructura básica	X2	*RMSprop*	*lecun + selu*	0.074	0.328
6	Espacio latente de 246 neuronas	No	*RMSprop*	*lecun + selu*	0.057	0.274
7	Espacio latente de 246 neuronas	X2	*RMSprop*	*lecun + selu*	0.065	0.315
8	Estructura básica	No	*RMSprop*	*glorot + selu*	0.070	0.313

La obtención de mejores resultados con *RMSprop* que con *Adam* tanto empleando *fingerprints* como descriptores refuerza la hipótesis de que, en problemas con datos altamente dispersos o con distribuciones poco homogéneas, resulta más efectivo utilizar optimizadores sin momento, que se adapten localmente al gradiente sin depender de acumulaciones pasadas, ya que esto evita que el ruido o la predominancia de ciertos valores distorsione el proceso de aprendizaje y conduzca a mínimos locales menos óptimos.

Como era de esperar, tanto en los autoencoders de *fingerprints* como en los de descriptores moleculares, un aumento del tamaño del espacio latente condujo a una mejor reconstrucción de los datos. Este equilibrio resulta especialmente relevante, ya que, según la tarea secundaria que se pretenda abordar mediante transfer learning, será necesario valorar si conviene incrementar el tamaño del espacio latente en función de la reducción que se obtenga en el error. Otro factor de peso es la cantidad de datos, pues a mayor cantidad de información de entrada, más se podrá reducir el tamaño del espacio latente. Este trabajo ha empleado espacios latentes que suponían entre un 6% - 12% del vector de datos de entrada. El único otro ensayo de TL autosupervisado basado en el uso de

encoders preentrenados utilizó una base de datos de 24 millones de espectros MS/MS que permitió generar un espacio latente de únicamente 1024 neuronas (0,0042% de los datos de entrada). Esto evidencia la enorme importancia de la cantidad de datos a la hora de llevar a cabo reducción dimensional basada en autoencoders.

Los resultados muestran además que emplear los mismos hiperparámetros y arquitecturas genera peores resultados en los modelos de descriptores moleculares que en los de *fingerprints*, a pesar de que ambos tipos de datos cuentan con un número de características de entrada muy similar. Esto evidencia que la reconstrucción de *fingerprints* es menos costosa al tratarse de datos binarios.

3.2. Predicción de RT a partir de las redes preentrenadas

3.2.1. Comparativa de predicción entre *red de predicción* y *arquitectura símil*

Se emplearon los errores MAE y RMSE para analizar la viabilidad del TL no supervisado en la predicción de tiempos de retención. Comparar los errores de los modelos preentrenados frente a sus arquitecturas símiles permitió evaluar si el preentrenamiento de autoencoders resulta una estrategia útil para facilitar la predicción.

Ninguno de los modelos preentrenados mediante *fingerprints* o descriptores por separado produjo una mejora en la capacidad de predicción con respecto a su *arquitectura símil* homóloga. Los modelos preentrenados con *fingerprints* presentaron un MAE promedio de 79,26s y un RMSE de 155,2s, mientras que sus *arquitecturas símil* correspondientes arrojaron un MAE medio de 74,41s y un RMSE de 152,38s. Similarmente, los modelos preentrenados con descriptores presentaron un MAE medio de 81,10s y un RMSE de 156,14s, mientras que las *arquitecturas símil* homólogas obtuvieron un MAE medio de 76,88s y un RMSE de 152,38s.

Por otra parte, los modelos preentrenados con *fingerprints* y descriptores de manera conjunta presentaron un rendimiento más cercano a sus *arquitecturas símil*. Los modelos predictores presentaron un MAE medio de 73,59s y un RMSE medio de 153,46s frente al MAE de 74,32s y RMSE de 152,44s promedio de las *arquitecturas símil*. Es decir, el uso de ambos tipos de datos conjuntamente no solo produjo la menor diferencia de RMSE, sino también un menor MAE medio en los modelos preentrenados que en las arquitecturas símil, algo que no ocurrió cuando se emplearon los datos por separado (*Figura 6*).

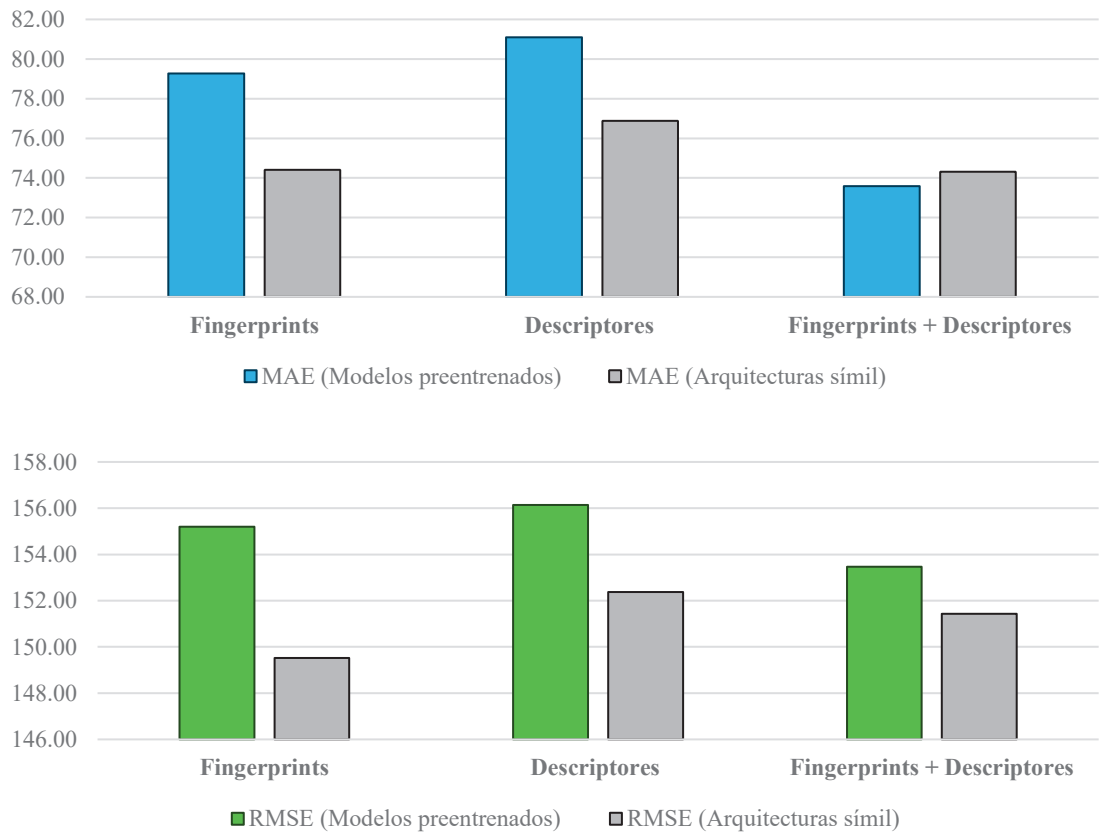

Figura 6. Errores MAE y RMSE promedio obtenidos para modelos preentrenados y arquitecturas símil en función del tipo de datos empleados.

Si comparamos los errores obtenidos en este trabajo con los de otros ensayos de predicción de RT mediante aprendizaje automático es notable que el error obtenido ha sido considerablemente más elevado. Sin embargo, estos experimentos tenían como objetivo la minimización del error predictivo, empleando optimización de hiperparámetros, modelizado de columna cromatográfica (Amil Manjón, 2024a) o técnicas de proyección (Domingo-Almenara et al., 2018; García et al., 2022). Por el contrario, este ensayo buscaba analizar la viabilidad del uso de aprendizaje por transferencia no supervisado, comparando los resultados para una misma arquitectura y analizando el uso de dos tipos de datos diferentes.

3.2.2. Comparativa de predicción entre las diferentes arquitecturas de *red de predicción*

Las distintas arquitecturas de *red de predicción* se diseñaron para poner a prueba varias hipótesis. Se estableció una comparación entre redes simples y redes con mayor número de capas, así como entre modelos que incorporaban capas repetidas en mayor o menor medida. La hipótesis de que las capas repetidas podían facilitar la extracción de relaciones no lineales entre los datos no se confirmó durante el preentrenamiento con autoencoders.

No obstante, se intentó demostrar su utilidad en la transformación de las características del espacio latente para la predicción del tiempo de retención.

Los resultados no mostraron una superioridad predictiva considerable por parte de ninguna de las arquitecturas. Los valores de MAE y RMSE no evidenciaron diferencias considerables en las predicciones de los diversos modelos. En cuanto a la arquitectura de neurona lineal, esta presentó errores ciertamente mayores a los de las arquitecturas con varias capas, algo a esperar dado que su función era meramente comparativa con el uso de datos desnudos. La *Figura 7* muestra la uniformidad de resultados entre las diversas arquitecturas y tipos de datos empleados.

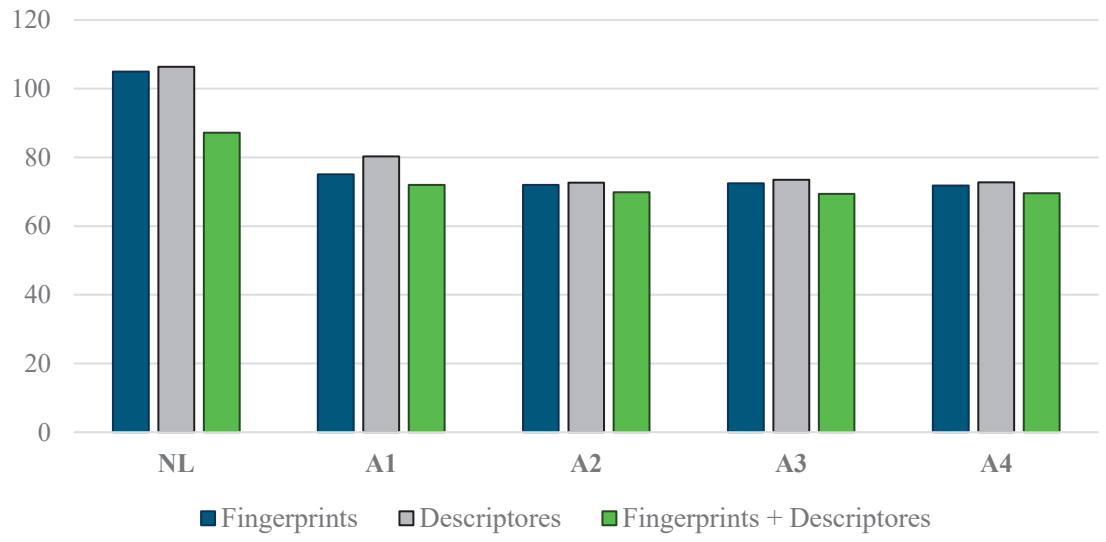

Figura 7. Error MAE de las diversas arquitecturas de red neuronal en función de los datos empleados. NL representa la arquitectura de neurona de activación lineal. A1-A4 representan las 4 arquitecturas complejas.

Sin embargo, sí podemos apreciar un menor error cuando se emplearon ambos tipos de datos que cuando se emplearon *fingerprints* o descriptores únicamente. Este resultado sugiere una vez más la problemática de la falta de datos y cómo un aumento de la cantidad de características reduce el error. Estos resultados se ven reforzados por el hecho de que la arquitectura de neurona lineal preentrenada con ambos tipos de datos fue la única que presentó un rendimiento mayor al de su *arquitectura símil*, demostrando que la adición del encoder preentrenado favorece la predicción del tiempo de retención. Consideramos que la vía experimental basada en redes neuronales preentrenadas mediante autoencoders de reducción dimensional merece seguir siendo explorada. Es probable que con una optimización exhaustiva de hiperparámetros y/o el empleo de conjuntos de datos de mayor tamaño puedan alcanzarse resultados significativamente mejores, lo que permitiría

evidenciar con mayor solidez la utilidad del preentrenamiento mediante autoencoders en este contexto.

3.2.3. Comparativa de predicción entre los diferentes tamaños de espacio latente

Como se puso en evidencia en el análisis de los resultados de preentrenamiento, un aumento del tamaño del espacio latente provocó mejoras en la capacidad de reconstrucción de los datos. Si se analizan las diferencias en las métricas de error en función del tamaño del espacio latente en los modelos preentrenados de predicción, obtenemos resultados mixtos. En los modelos que emplearon *fingerprints*, no hubo una diferencia de peso en la calidad predictiva debida al tamaño del espacio latente. Los modelos con 140 neuronas en la representación latente lograron, de hecho, un error promedio mínimamente inferior al de los modelos equivalentes con 280 neuronas. Concretamente, obtuvieron un MAE de 72,22s y un RMSE de 147,3s, frente a un MAE de 72,25s y un RMSE de 147,61s en los modelos de 280 neuronas. Por el contrario, en los modelos preentrenados con descriptores, las redes con un espacio latente de 246 neuronas mostraron mejores resultados, alcanzando un MAE de 76,37s y un RMSE de 150,20s, en comparación con los modelos de 123 neuronas en la representación latente, que obtuvieron un MAE de 73,17s y un RMSE de 147,78s.

En los modelos preentrenados con ambos tipos de datos tampoco se obtuvo una mejora relacionada con la dimensión del espacio latente. Los resultados mostraron una eficacia uniforme e independiente del tamaño del espacio latente. Los modelos con espacio latente de 526, 403, 386 y 263 neuronas obtuvieron errores MAE de 70,25s, 69,92s, 70,47s y 70,14s, y errores RMSE de 155,76s, 147,90s, 154,16s y 147,27s, respectivamente.

CAPÍTULO 4: CONCLUSIONES

1. Los autoencoders lograron reconstruir con gran éxito los datos de entrada en el preentrenamiento, capturando características generales y reduciendo la dimensionalidad hacia un espacio latente de alto nivel. Sin embargo, al usar esas representaciones latentes en modelos supervisados para predecir RT, no se observó mejora en la capacidad predictiva en comparación con las *arquitecturas símil* no preentrenadas. Este conflicto podría evidenciar una diferencia demasiado amplia entre las tareas que se ha pretendido ligar mediante *transfer learning*.

2. A diferencia de los modelos preentrenados con un único tipo de datos, aquellos preentrenados con todos los datos replicaron los resultados de las *arquitecturas símil*. Este hecho recalca la problemática de la escasez de datos en metabolómica y sugiere que, con un mayor volumen de datos, esta aproximación basada en TL no supervisado podría haber sido exitosa.

3. No se apreciaron diferencias significativas en las predicciones al comparar las distintas arquitecturas ni los diversos tamaños de espacio latente. Este resultado podría ser, igualmente, un reflejo de la limitada cantidad de datos disponible.

4. Este trabajo no centró sus esfuerzos en la optimización de hiperparámetros, lo que podría haber sido decisivo para el éxito de la aproximación basada en TL no supervisado. Se propone que futuras investigaciones orientadas en esta línea incorporen estrategias de optimización de hiperparámetros como parte fundamental de su diseño experimental, para explorar plenamente el potencial de esta metodología.

CAPÍTULO 5: BIBLIOGRAFÍA

Amil Manjón, A. (2024a). Predicción de tiempos de retención para varios métodos cromatográficos mediante aprendizaje profundo.

Amil Manjón, A. (2024b). Anaamil/Regressor: Great Simplification of Tino's DNN (garcia2022probabilistic), GitHub: https://Github.Com/Anaamil/Regressor.

Baker, T. R., & Regg, B. T. (2018). A multi-detector chromatographic approach for characterization and quantitation of botanical constituents to enable in silico safety assessments. Analytical and Bioanalytical Chemistry, 410(21), 5143.

Bittremieux, W., & Noble, W. S. (2025). Self-supervised learning from small-molecule mass spectrometry data: Metabolomics. Nature Biotechnology.

Blum, F. (2014). High performance liquid chromatography. British Journal of Hospital Medicine, 75(SUPPL. 2) C18 – C21

Choi, R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Peter Campbell, J. (2020). Introduction to machine learning, neural networks, and deep learning. Translational Vision Science and Technology, 9(2)-14

Djoumbou Feunang, Y., Eisner, R., Knox, C., Chepelev, L., Hastings, J., Owen, G., Fahy, E., Steinbeck, C., Subramanian, S., Bolton, E., Greiner, R., & Wishart, D. S. (2016). ClassyFire: automated chemical classification with a comprehensive, computable taxonomy. Journal of Cheminformatics, 4;8-61

Domingo-Almenara, X., Montenegro-Burke, J. R., Benton, H. P., & Siuzdak, G. (2018). Annotation: A Computational Solution for Streamlining Metabolomics Analysis. Analytical Chemistry, 90(1), 480–489.

Farahani, A., Rasheed, K., Arabnia, H., Pourshojae, B., & Arabnia, H. R. (2021). A Concise Review of Transfer Learning.

Fedorova, E. S., Matyushin, D. D., Plyushchenko, I. V., Stavrianidi, A. N., & Buryak, A. K. (2022). Deep learning for retention time prediction in reversed-phase liquid chromatography. Journal of Chromatography A, 1664.

Fong, Y., & Xu, J. (2021). Forward Stepwise Deep Autoencoder-Based Monotone Nonlinear Dimensionality Reduction Methods. Journal of Computational and Graphical Statistics, 30(3), 519–529.

García, C. A., Gil-de-la-Fuente, A., Barbas, C., & Otero, A. (2022). Probabilistic metabolite annotation using retention time prediction and meta-learned projections. Journal of Cheminformatics, 14(1).

Géron, A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2ª Edición, O'Reilly, Canadá, 2019; 21-543.

Godzien, J., Gil de la Fuente, A., Otero, A., & Barbas, C. (2018). Metabolite Annotation and Identification. Comprehensive Analytical Chemistry, 82, 415–445.

Jovel, J., & Greiner, R. (2021). An Introduction to Machine Learning Approaches for Biomedical Research. Frontiers in Medicine, 8.

Kim, S., Chu, S. H., Park, Y. J., & Lee, C. Y. (2024). A tied-weight autoencoder for the linear dimensionality reduction of sample data. Scientific Reports, 14(1)

Kretschmer, F., Harrieder, E. M., Hoffmann, M. A., Böcker, S., & Witting, M. (2024). RepoRT: a comprehensive repository for small molecule retention times. Nature Methods, 21(2), 153–155.

Liapikos, T., Zisi, C., Kodra, D., Kademoglou, K., Diamantidou, D., Begou, O., Pappa-Louisi, A., & Theodoridis, G. (2022). Quantitative structure retention relationship (QSRR) modelling for Analytes' retention prediction in LC-HRMS by applying different Machine Learning algorithms and evaluating their performance. Journal of Chromatography B: Analytical Technologies in the Biomedical and Life Sciences, 1191.

Lorenzo, N. (2025, July 7). TFG-Nicolas-Lorenzo. GitHub: https://github.Com/Nicoolorenzo/TFG-Nicolas-Lorenzo.

Lu, X., Zhao, X., Bai, C., Zhao, C., Lu, G., & Xu, G. (2008). LC-MS-based metabonomics analysis. Journal of Chromatography B: Analytical Technologies in the Biomedical and Life Sciences, 866(1–2), 64–76.

Mauri, A. (2020). alvaDesc: A Tool to Calculate and Analyze Molecular Descriptors and Fingerprints. Methods in Pharmacology and Toxicology, 801–820.

Meyer, J. G. (2021). Deep learning neural network tools for proteomics. Cell Reports Methods, 1(2).

Nguyen, Q. H., Nguyen, H., Oh, E. C., & Nguyen, T. (2024). Current approaches and outstanding challenges of functional annotation of metabolites: a comprehensive review. Briefings in Bioinformatics, 25(6).

Novoa-Del-Toro, E. M., & Witting, M. (2024). Navigating common pitfalls in metabolite identification and metabolomics bioinformatics. Metabolomics : Official Journal of the Metabolomic Society, 20(5), 103.

Osipenko, S., Bashkirova, I., Sosnin, S., Kovaleva, O., Fedorov, M., Nikolaev, E., & Kostyukevich, Y. (2020). Machine learning to predict retention time of small molecules in nano-HPLC. Analytical and Bioanalytical Chemistry, 412(28), 7767–7776.

Pomyen, Y., Wanichthanarak, K., Poungsombat, P., Fahrmann, J., Grapov, D., & Khoomrung, S. (2020). Deep metabolome: Applications of deep learning in metabolomics. Computational and Structural Biotechnology Journal, 18, 2818–2825.

Sen, P., Lamichhane, S., Mathema, V. B., McGlinchey, A., Dickens, A. M., Khoomrung, S., & Orešič, M. (2021). Deep learning meets metabolomics: A methodological perspective. Briefings in Bioinformatics, 22(2), 1531–1542.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A Survey on Deep Transfer Learning.

Thomas, S. N., French, D., Jannetto, P. J., Rappold, B. A., & Clarke, W. A. (2022). Liquid chromatography–tandem mass spectrometry for clinical diagnostics. Nature Reviews Methods Primers, 2(1).

Witting, M., & Böcker, S. (2020a). Current status of retention time prediction in metabolite identification. Journal of Separation Science, 43(9–10), 1746–1754.

Yang, Q., Ji, H., Fan, X., Zhang, Z., & Lu, H. (2021). Retention time prediction in hydrophilic interaction liquid chromatography with graph neural network and transfer learning. Journal of Chromatography A, 1656.

Yang, Q., Ji, H., Lu, H., & Zhang, Z. (2021). Prediction of Liquid Chromatographic Retention Time with Graph Neural Networks to Assist in Small Molecule Identification. Analytical Chemistry, 93(4), 2200–2206.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Member, S., Xiong, H., & He, Q. (2020). A Comprehensive Survey on Transfer Learning.