Manuel Ruiz Botella

Desarrollo de una herramienta para la reconstrucción de metabolitos a partir del espectro de masas de los metabolitos.

TRABAJO DE FINAL DE GRADO

Tutor: Dra. Marta Sales Pardo. marta.sales@urv.cat

UNIVERSITAT ROVIRA i VIRGILI

Grado de Biotecnología.

Tarragona

2019

Índice

Datos del centro

Este trabajo de final de grado lo he realizado durante mi estancia de prácticas externas en el grupo de investigación Science and Engineering of Emerging Systems (SEESlab) que pertenece al Departamento de Ingenería Química de la Universidad Rovira i Virgili.

Los fundadores y jefes del grupo de investigación son el Dr. Roger Guimerà Manrique y la Dra. Marta Sales Pardo. El laboratorio del grupo de investigación se encuentra en el laboratorio 112 de la primera planta de la Escuela Técnica de Ingeniería Química del Campus Sescelades de la Universidad Rovira i Virgili.

Dirección: Laboratorio 112. Ingeniería Química, Av. Països Catalans 26 Tarragona, Tarragona 43007, España

Teléfono: +34-977-558-435

Resumen y palabras clave

Actualmente la metabolómica es la sección de la biología de sistemas que más problemas tiene a la hora de integrarse en modelos biológicos. Esto se debe a que no hay una manera de secuenciar un metabolito igual que ocurre en otras ciencias omicas. En la metabolómica se utiliza el acceso a las bases de datos para comparar el espectro de un metabolito con los espectros guardados. Gracias a la inteligencia artificial se puede integrar la informática y la biología de sistemas para resolver esta limitación.

Actualmente hay un modelo de inteligencia artificial que es capaz de reconstruir moléculas y generar nuevas moléculas. Nuestro trabajo se fundamenta en utilizar un modelo de inteligencia artificial que sea capaz de reconstruir y generar nuevos metabolitos. Además, hemos desarrollado una red neuronal convolucional que se integra con este primer modelo y que permite reconstruir y generar nuevos metabolitos a partir del espectro de masas de un metabolito. Nuestro modelo obtiene un 92,83 % de reconstrucción para los metabolitos que se usan para generarlo, y un 24,54 % para los metabolitos que nunca ha visto.

Nuestro modelo es capaz de determinar una molécula a partir de su espectro de masas sin que sea necesario que nadie haya asociado el espectro de masas de la molécula a la molécula previamente. Superando completamente a la metodología actual en la metabolómica para determinar un metabolito de una muestra biológica.

Palabras clave:

Biología de sistemas. Campo de investigación interdisciplinario basado en la biología, con el objetivo de entender las interacciones complejas en el interior de los sistemas biológicos.

Metabolómica. Disciplina científica que se centra en el estudio de los procesos químicos que ocurren con los metabolitos, moléculas intermedias y productos del metabolismo. Se centra en el estudio de los perfiles metabólicos y en los procesos que rodean a los metabolitos.

Junction Tree Variational Autoencoder. Modelo de inteligencia artificial que se utiliza para reconstruir y generar nuevas moléculas a partir de una codificación de estas. En nuestro caso metabolitos.

Red neuronal convolucional. Modelo de inteligencia artificial que permite extraer características de una información de entrada, para procesar la información extraída y utilizarla para resolver una tarea.

Espectro de masas. Representación química de una molécula en la que el eje x se corresponde con la relación masa/carga de los iones de esta molécula y el eje y la intensidad relativa en cada pico del eje x.

SMILES. Representación en forma de cadena de caracteres ASCII de una molécula que permite representar su forma, estructura y más características correctamente.

Vector de codificación. Vector que se obtiene a partir de procesar una información, y que un elemento de inteligencia artificial es capaz de decodificar para obtener la información inicial.

Metabolitos nunca vistos. En el entrenamiento de Inteligencia Artificial, siempre se deja un conjunto de todos los ejemplos de la tarea sin utilizar, para que, una vez acabada la generación del modelo, observar cuales son los resultados ante nuevos ejemplos de la tarea.

Metabolitos de entrenamiento. Metabolitos con los que se entrena una Inteligencia Artificial para que aprenda a realizar la tarea asignada.

1. INTRODUCCIÓN

La biología de sistemas estudia los sistemas biológicos de una manera global mediante una aproximación holística. Con ese objetivo se centra en las interacciones complejas dentro de los sistemas biológicos y en cómo estas interacciones dan sentido al funcionamiento y comportamiento del sistema biológico; como por ejemplo la interacción entre metabolitos y enzimas en una ruta metabólica. Esta biología de sistemas se basa en la realización de experimentos biológicos para caracterizar un sistema biológico, y a partir de la información extraída, realizar un análisis computacional y matemático de los sistemas complejos, para posteriormente modelar un sistema biológico que permita realizar investigación, hacer simulaciones y analizar datos del modelo, predecir comportamientos o encontrar relaciones entre componentes del sistema. (Tayassoly et al. 2018), (Snoep and Westerhoff, 2005)

La biología de sistemas está compuesta principalmente por la genómica, la transcriptómica, la proteómica y la metabolómica.

Idealmente para comprender los sistemas biológicos más complejos se deberían desarrollar las cuatro disciplinas a la vez, de tal manera que se puedan complementar entre genómica, proteómica, transcriptómica y metabolómica para estudiar sistemas más complejos, ya que todas tienen relación en los sistemas biológicos. (Chaston and Douglas, 2012)

En los últimos años ha aumentado el desarrollo de la genómica y la proteómica gracias a la existencia de técnicas de secuenciación. Estas técnicas permiten determinar con precisión, a partir de una muestra biológica, los genes y las proteínas que se expresan en un determinado momento en una célula o tejido.

La metabolómica se centra en los procesos químicos que envuelven a los metabolitos y a los productos del metabolismo. En ella se estudia de manera sistemática la huella que dejan estos metabolitos detrás, es decir, su perfil metabólico. El metaboloma representa el conjunto completo de metabolitos de un sistema biológico. El perfil metabólico puede dar una instantánea de la fisiología de una célula, con lo que la metabolómica puede proporcionar una lectura funcional fenotípica directa del estado fisiológico. (Hollywood et al., 2006)

A diferencia de la genómica o la proteómica, no hay técnicas comparables a la secuenciación. Esto significa que, dada una muestra biológica, es imposible identificar la mayoría de los metabolitos presentes. Es posible saber cuántos metabolitos hay, pero no identificarlos. Para la identificación de metabolitos se realizan 3 fases. Primero se separa la muestra biológica mediante diferentes técnicas, después se detectan los componentes de la muestra mediante espectrometría de masas o espectroscopia de resonancia magnética nuclear, y por último se realiza un análisis estadístico de los resultados obtenidos. Aunque la anotación a mano del espectro de masas del metabolito, y la identificación de los diferentes picos y su clasificación en diferentes componentes de una molécula, es posible, es un trabajo muy costoso y que supone una gran labor que se puede automatizar.(Domingo-Almenara et al., 2018) La técnica más utilizada actualmente, es la cromatografía, y posteriormente, obtener su espectro de fragmentación mediante espectrometría de masas, y volver a aplicar espectrometría de masas al ion que se quiere reconocer de la muestra (MS/MS). Una vez que se ha obtenido este espectro, si coincide con alguno de los existentes en las bases de datos para los metabolitos puros (por ejemplo, la HMDB), se puede identificar este metabolito y, por tanto, saber con qué metabolitos se trabaja. (Patti et al., 2012), (Rojas-Cherto et al., 2012) El problema ocurre cuando el espectro obtenido no se encuentra en las bases de datos, y, por tanto, resulta imposible reconstruir con certeza el metabolito que se ha obtenido en la muestra biológica. Esto supone un gran inconveniente en muchos casos, ya que, al obtener una muestra, solo se pueden identificar unos metabolitos, sin llegar a obtener toda la información posible de la muestra y, por tanto, se tiene que trabajar sin todo el contexto de la muestra posible, sin todo el metaboloma. Además, el retraso de la metabolómica frente a las otras ciencias ómicas produce que los sistemas biológicos complejos no puedan analizarse y modelarse correctamente. Esto obliga a realizar estudios dirigidos en que se miran unas concentraciones de metabolitos específicos que se pueden identificar, pero que al realizar un estudio completo del metaboloma se puede ver dificultada su detección por otros metabolitos no identificables.

Aproximadamente, solo el 10% de metabolitos conocidos se encuentra su espectro de masas en las bases de datos como HMDB (Wishart et al., 2012)

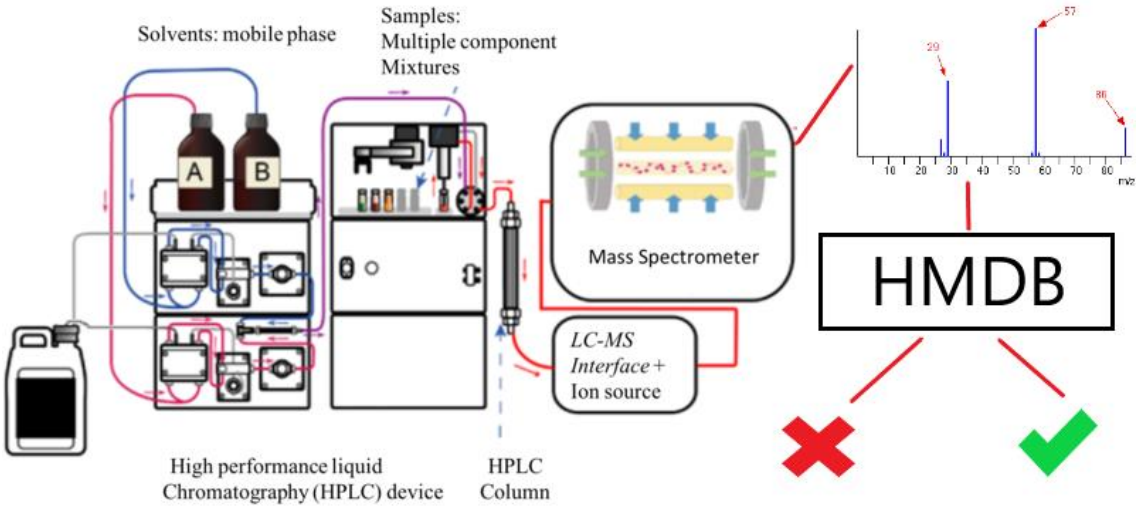

Figura 1. La metodología actual consiste en aplicar una separación (en este ejemplo cromatografía líquida de los analitos), posteriormente ionizar la muestra separada que se quiere identificar e introducirla en el espectrómetro de masas. Al final identificar en las bases de datos si se encuentra o no este espectro de masas.

El rol de la inteligencia artificial en el desarrollo de herramientas de bioinformática.

La inteligencia artificial es una de las ramas de las ciencias computacionales encargada de estudiar modelos de computación capaces de realizar actividades propias de los seres humanos en base a el razonamiento y la conducta (Lopez 2017), (Kaplan and Haenlein, 2019). Dentro de la inteligencia artificial hay diversos campos, que se dividen principalmente en la inteligencia artificial analítica, para resolver problemas mediante una representación del problema en base a un conocimiento aprendido, inteligencia artificial inspirada en humanos e inteligencia artificial humanizada, que se centran en comprender las emociones humanas y en tener inteligencia social y ser autoconsciente respectivamente.

El aprendizaje profundo, más comúnmente conocido como Deep Learning, es un conjunto de algoritmos de aprendizaje automático, que pertenecen al campo de inteligencia artificial analítica. El aprendizaje profundo se basa principalmente en redes neuronales artificiales, y presenta diferentes arquitecturas que permiten desarrollar diferentes modelos basados en matemáticas, que, a partir de una representación de datos, comprenden la tarea que se presenta en estos datos, y aprenden a resolver esta tarea. (Schmidhuber, 2015) Se han utilizado las redes neuronales en muchos campos de la ciencia como visión por computador, reconocimiento de voz, procesamiento de lenguaje natural, análisis de imágenes médico y bioinformática.

Uno de los objetivos de la biología de sistemas es obtener un modelo matemático de un sistema complejo, y las redes neuronales son capaces de generar modelos basados en las matemáticas. Las redes neuronales pueden incluso ser capaces de reconocer elementos que nunca han visto en la tarea que han aprendido a hacer. (Bengio et al., 2015), (Olshausen and Field., 1996), (Marblestone et al., 2016) Por tanto, creemos que una de las posibles soluciones al problema de la metabolómica, se puede encontrar en **generar modelos de aprendizaje profundo que sean capaces de determinar la identidad de un metabolito (fórmula y estructura molecular) a partir de su espectro de masas (MS/MS).** Haya visto este metabolito en el proceso de entrenamiento o no.

Actualmente ya se han desarrollado herramientas como iMet (Aguilar-Mogas et al., 2017), basadas en la identificación de metabolitos vecinos y las posibles transformaciones químicas necesarias a partir de metabolitos ya existentes. Además, herramientas como CFM-ID (Allen et al., 2016), utilizan técnicas de aprendizaje automático para predecir el espectro de masas a partir de una molécula.

Por otro lado, recientemente un equipo del MIT ha desarrollado un modelo de Inteligencia Artificial llamado Junction Tree Variational Autoencoder (JTVAE), capaz de generar nuevas moléculas, y reconstruir moléculas ya existentes a partir del SMILES de estas moléculas. Este algoritmo (también basado en técnicas de aprendizaje profundo) se centra en generar una representación de la molécula a partir del SMILES, en un grafo con forma de árbol, y codificar esta información en un vector de números, por último, a partir de este vector de números, es capaz de identificar el SMILES correcto de la molécula. (Jin et al., 2018)

Por tanto, la existencia de estas herramientas y el desarrollo de nuevas puede ayudar a cerrar la distancia entre la metabolómica y el resto de las secciones de la biología de sistemas.

2. OBJETIVOS

Nuestro objetivo es diseñar y desarrollar una herramienta capaz de identificar metabolitos a partir del espectro de masas del metabolito. Utilizando dos redes neuronales artificiales. Combinando el JTVAE desarrollado en (Jin et al., 2018), entrenado con metabolitos, y una red neuronal convolucional (CNN), que es capaz de producir el vector intermedio (que almacena características del metabolito) que se obtiene en el JTVAE, a partir del espectro de masas de un metabolito. (Tutorial redes neuroanles convolucionales, 2019), (Aghdam and Heravi, 2017) Con esto, el JTVAE es capaz de reconstruir el metabolito del que proviene el espectro de masas.

En este trabajo, se han seguido las siguientes etapas con tal de realizar la herramienta ya comentada.

1) Entender y comprobar el funcionamiento del JTVAE del estudio (Jin et al., 2018).
2) Entrenar el JTVAE con los metabolitos de la base de datos NIST. Encontrando los parámetros correctos para obtener el mejor rendimiento posible.
3) Obtener el vector intermedio del JTVAE entrenado por nosotros con metabolitos.
4) Generar ficheros que contengan: el SMILES, el vector intermedio y el espectro de masas del metabolito.
5) Diseñar y entrenar una red neuronal convolucional que permita obtener el vector intermedio a partir del espectro de masas.
6) Obtener los vectores intermedios generados a partir de la red convolucional e introducirlos en el JTVAE para obtener la molécula a partir del espectro de masas.

3. METODOLOGÍA

En este apartado explico la metodología del trabajo utilizada. Primero, voy a explicar: a) en qué consiste y en que está basado el aprendizaje profundo, b) los fundamentos y cómo funciona el Junction Tree Variational Autoencoder, y c) cómo funciona una red neuronal convolucional. Por último, haré una breve explicación sobre la obtención de los espectros de masas de metabolitos y la base de datos que utilizamos.

3.1. Aprendizaje profundo

El aprendizaje profundo es conjunto de algoritmos de inteligencia artificial que tiene como objetivo resolver una tarea asignada a partir de conocimiento extraído al realizar esta tarea con ejemplos muchas veces. Dentro del aprendizaje profundo, hay una gran cantidad de diferentes arquitecturas que se pueden organizar para conformar una red neuronal. Estas arquitecturas van desde un simple perceptrón, que simularía una única neurona, a una red neuronal multicapa, que simularía un cerebro humano con muchas neuronas conectadas, a arquitecturas más complejas como una red generativa antagónica, utilizadas recientemente para dar vida a la Gioconda. (Zakharov, 2019).

Cualquier arquitectura de aprendizaje profundo es una colección de unidades matemáticas artificiales llamadas neuronas. Este aprendizaje profundo intenta simular el cerebro humano y las diferentes conexiones entre neuronas. (Marblestone et al., 2016) Como podemos ver en la figura 2, una vez que se capta la imagen en la retina, nuestro cerebro tiene un "camino" establecido que sigue la información. Esta información llega a nuestro cerebro, a un grupo de neuronas que tienen la función de identificar formas visuales simples, ejes y esquinas, una vez procesada la información, mediante la sinapsis, se comparte esta información procesada con un segundo grupo de neuronas que procesan de nuevo la información recibida. Esto se replica muchas veces, puesto que cada neurona tiene una gran cantidad de conexiones con otras neuronas. Una vez que la información es procesada por muchas capas de neuronas, acaba siendo interpretada y actuamos en consecuencia a la información extraída.

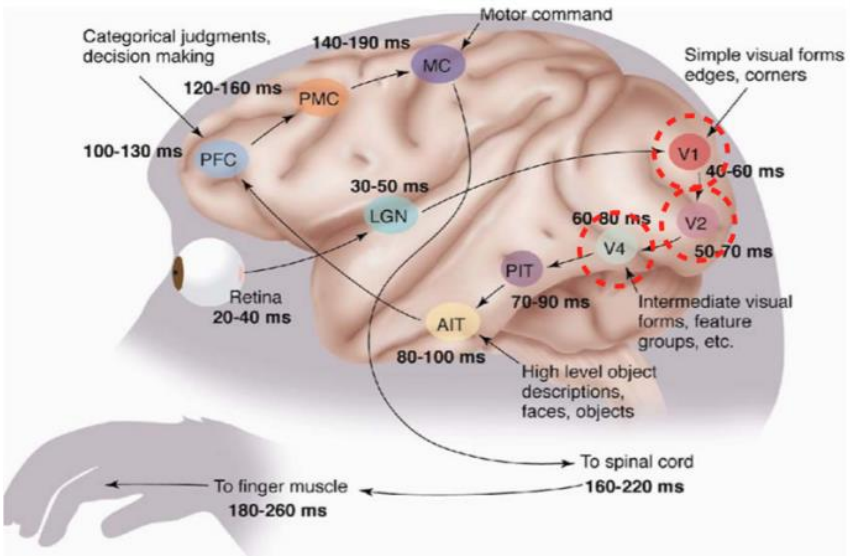

Figura 2. Funcionamiento de un cerebro para procesar la información recibida.

Así pues, en el aprendizaje profundo la red neuronal tiene múltiples capas de neuronas artificiales conectadas entre sí. Cada una de las neuronas artificiales tiene varias conexiones con la siguiente capa. Cada una de éstas tiene asociado un peso que modifica el valor (la información) que se pasa de una neurona a otra. Además, cada neurona tiene un peso propio, que se añade siempre a su valor, y una función de activación, simulando el potencial eléctrico que necesita una neurona en el cerebro para activarse. Con estos atributos de cada neurona, se puede obtener una red neuronal, que, al intentar aprender una tarea asignada, sea capaz de modificar estos valores de pesos entre conexiones y peso de cada neurona para ajustar el resultado final. Un ejemplo de una neurona sería la figura 3:

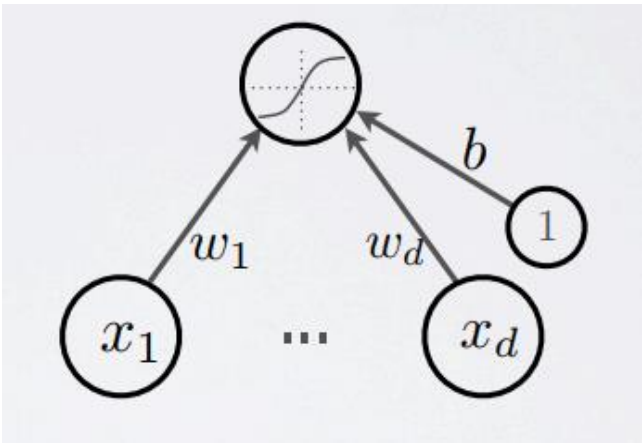

Figura 3. Cada esfera representa una neurona, siendo la neurona superior una neurona que recibe información de las neuronas x1 hasta xd. Esta información se ve modificada por los pesos de las conexiones de esa neurona con las que le pasan información. Además, la neurona tiene un peso propio, que es la esfera pequeña, que se indica con una b. Al recibir toda esta información numérica la neurona aplica una función de activación que es la curva que se representa en su esfera.

En conjunto, una red neuronal utilizada para resolver tareas, independientemente de su arquitectura, presenta:

- Una capa de entrada, que recibe los datos a analizar para resolver la tarea. Normalmente se tienen tantas neuronas como datos de entrada.
- N capas ocultas, que reciben la salida de la capa anterior (la primera capa oculta recibe la salida de la capa de entrada) y producen una salida. Cada capa puede tener un conjunto diferente de neuronas artificiales.
- Una capa de salida, que recibe la salida de la última capa de entrada y la procesa. Normalmente tiene tantas neuronas como datos de salida se quieren tener.

Una posible red neuronal artificial sencilla, que pertenece al aprendizaje profundo, sería la de la figura 4, con 3 capas (una de entrada, una oculta y una de salida), donde cada neurona se representa con una esfera y las conexiones entre ellas en gris. Al final se obtendría un único número de información, que podría ser usado para clasificación de los datos de entrada.

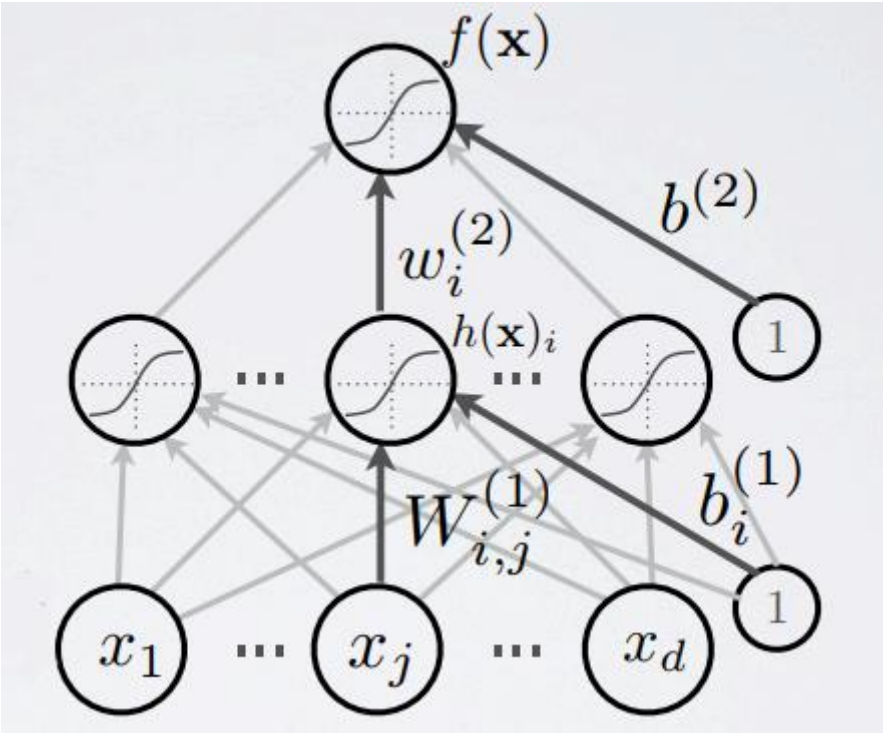

Figura 4. Arquitectura de una red neuronal convolucional. La capa inferior es la capa de entrada, la capa del medio es la capa oculta y la capa superior es la capa de salida.

Para que una arquitectura de aprendizaje profundo sea capaz de resolver una tarea, se ha de entrenar, esto consiste en dividir todos los datos en un conjunto de datos de

entrenamiento y un conjunto de datos de testeo. Se ha de hacer que el modelo de aprendizaje profundo vaya intentando resolver la tarea con los datos de entrenamiento, y en función de lo acertado que está en su predicción, se modifican los pesos de las conexiones y el peso de la neurona mediante unos algoritmos de optimización y de regularización, un ratio de aprendizaje y una función de pérdida, basada en la diferencia entre la predicción y el resultado real. Conforme se van modificando estos valores, el modelo será capaz de realizar mejor la tarea. Para asegurar que el modelo de aprendizaje profundo resuelve correctamente la tarea, se ha probar a que resuelva los datos de un conjunto de ejemplos que no ha tratado nunca. Si resuelve correctamente, es que el modelo de aprendizaje profundo ha conseguido aprender a resolver la tarea, si no, es que ocurre overfitting. El overfitting ocurre cuando un modelo de aprendizaje profundo aprende a resolver correctamente solo el conjunto de datos de entrenamiento, esto quiere decir que, aunque ha aprendido a resolver la tarea, al enfrentarse a nuevos retos de la tarea, puede ser que se equivoque. Para evitarlo, se intenta utilizar diversas técnicas:

- Parar el entrenamiento antes de tiempo, mejor que no sepa resolver la tarea perfecta pero que no se equivoque cuando podría funcionar.
- Definir un ratio de aprendizaje a la tarea y al conjunto de datos.
- Desactivar aleatoriamente unas neuronas en las capas de la red neuronal, para que no siempre dependa de unas pocas neuronas artificiales que han aprendido a resolver el problema.

A veces, el overfitting se puede solucionar, otras el conjunto de datos no es suficientemente extenso, y la separación entre el conjunto de entrenamiento y el de ejemplos nunca vistos no es suficientemente correcta, y, por tanto, aprender a resolver el conjunto de entrenamiento no garantiza resolver correctamente el conjunto de ejemplos nunca vistos.

Todo este conocimiento de redes neuronales lo he aprendido a partir de un tutorial de Hugo Larochelle en (Larochelle, 2019)

3.1.1. Junction Tree Variational Autoencoder (JTVAE)

El JTVAE utilizado y desarrollado en (Jin et al., 2018), asegura que se puede codificar una molécula y posteriormente volver a obtener el SMILES de esta molécula a partir de la codificación.

La entrada de este modelo es el SMILES de una molécula, esta es una especificación para describir sin ambigüedades la estructura de la molécula a partir de cadenas ASCII cortas. Posee la ventaja que es inteligible para las personas, y está fundamentado en la teoría de grafos. (Weininger, 1998)

Esta notación permite la especificación de SMILES isoméricos, que permite determinar entre dos moléculas que sean isómeros. Además, el SMILES se puede canonizar, y es que hay algoritmos y módulos de PYTHON que permite obtener la misma representación de la molécula, a partir de una representación diferente en SMILES. **La codificación en SMILES de una molécula permite determinar los átomos, los enlaces, la aromaticidad, las ramificaciones, la estereoquímica y los isótopos.** Por tanto, podemos concluir que el SMILES es una representación acertada de una molécula y que podemos utilizar para introducir los metabolitos en el JTVAE. Un ejemplo de cómo se codifica una molécula de Glucosa en formato SMILES está en la figura 5.

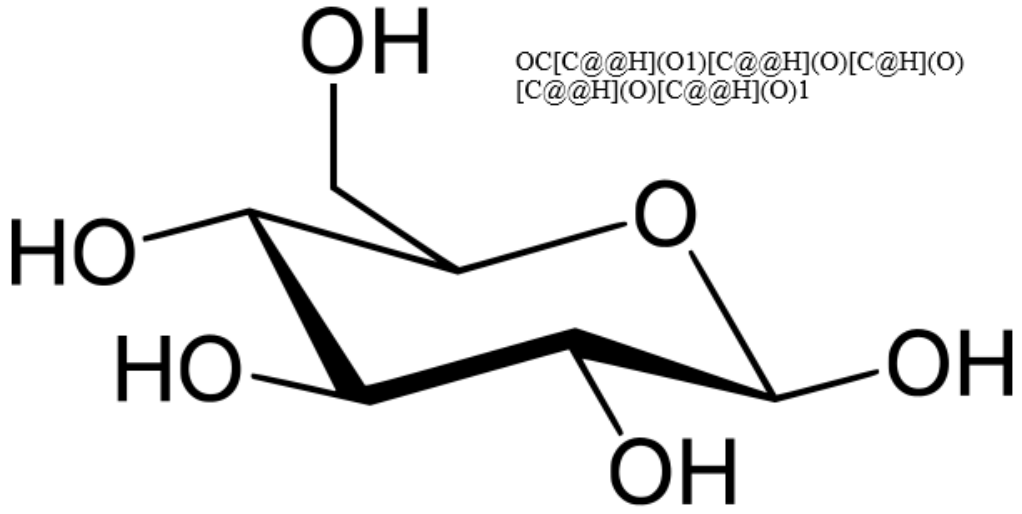

Figura 5. Representación de una molécula de glucosa, y su SMILES correspondiente.

El JTVAE tiene dos partes:

En la primera parte, el Junction Tree, se genera un grafo en forma de árbol estructurado a partir de la representación de la molécula y un vocabulario de SMILES de diferentes elementos y grupos que pueden conformar una molécula. Por un lado, esto permite representar a la molécula en su totalidad (incluyendo enlaces). Por otro lado, se evita que se obtengan moléculas que no cumplan las reglas químicas, como por ejemplo enlaces aromáticos en la molécula sin sentido, sino que sólo ocurren cuando todo un anillo aromático está presente.

Por tanto, a partir de la estructura de la molécula, y del vocabulario, se puede obtener un grafo con forma de árbol que representa a la molécula, y donde cada nodo representa un elemento del vocabulario de SMILES.

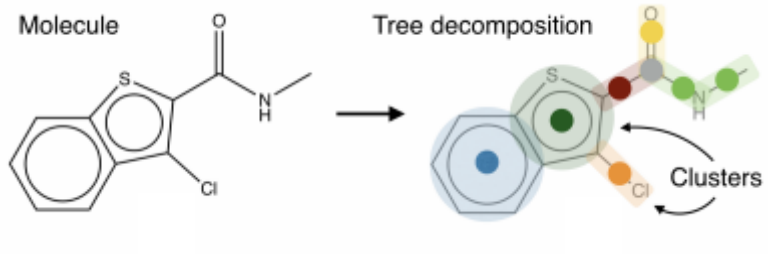

Figura 6. Representación en forma de grafo de árbol de una molécula en el JTVAE.

3.1.1.1. Autoencoder (AE)

La segunda parte del JTVAE es un Variational Autoencoder (VAE), que es un tipo de arquitectura de las redes neuronales de aprendizaje profundo. Primero, voy a explicar las características y cómo trabaja un Autoencoder, para posteriormente explicar en qué se diferencia de un Variational Autoencoder.

Un Autoencoder es una arquitectura de aprendizaje profundo que se utiliza para ser capaz de codificar una información de entrada, a un vector que representa esta información y que se puede decodificar para obtener la información de entrada. (Boesen et al., 2016)

Un ejemplo, podría ser un Autoencoder, que es capaz de determinar, a partir de una imagen donde hay un gato o un perro, que animal es el que se encuentra en la imagen. La arquitectura sería la de la figura 7.

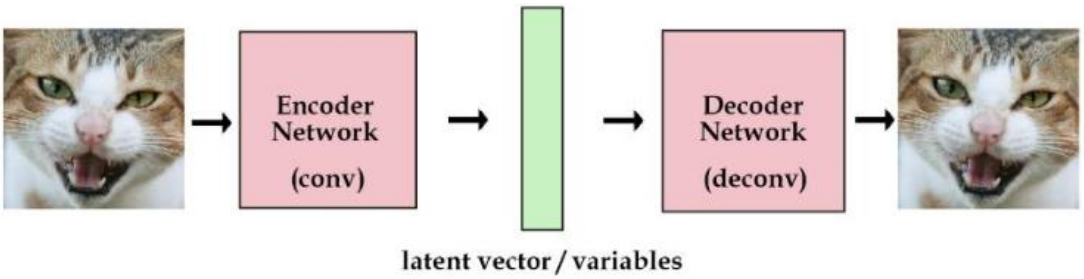

Figura 7. Arquitectura de un Autoencoder.

Se basa en 3 capas:

- Una capa de entrada de la información, de x neuronas artificiales, donde se codifica la información de entrada.

- Una capa oculta, de y neuronas artificiales. Siempre siendo y menor que x. Por tanto, tiene que haber una compresión de los datos.
- Una capa de salida, de x neuronas artificiales. Es de la misma dimensión que la entrada, puesto que se intenta reconstruir la información de entrada.

La capa de entrada es el encoder, que codifica la información de entrada, generalmente un vector de números, a la dimensión de la capa intermedia, conocido como vector latente o vector de codificación. **La capa de salida es conocida como decoder**, que decodifica a partir del vector latente el vector de salida, que se espera que sea igual que el de entrada. La manera en la que se codifica y se trata esta información de entrada hasta obtener el vector latente, y lo mismo para decodificar el vector latente, es lo que cambia entre diferentes AEs, ya que puede ser simples conexiones de neuronas artificiales o varias, según el problema que se quiere resolver.

3.1.1.2. Variational Autoencoder

Un VAE, es una arquitectura de aprendizaje profundo que se utiliza para obtener modelos generativos de datos, es decir, modelos que comprenden la tarea, y son capaces de generar nuevos ejemplos de la tarea, en este caso, **nuestro VAE sería capaz de generar metabolitos que nunca ha visto.** Un VAE comprende su definición de lo importante de los datos, no necesita etiquetas para saber si ha trabajado correctamente. (Boesen et al., 2016), (Explicación VAE., 2019) Hasta aquí, hemos explicado un Autoencoder (sin el variational), puesto que, para la misma entrada de datos, se busca obtener el mismo vector latente, esto podría provocar que el Autoencoder aprenda a codificar para el caso exacto de entrada, y no para el conjunto de todos los casos de la tarea, y, por tanto, no sería capaz de resolver nuevos casos de la tarea. La arquitectura de un VAE, para el mismo problema que antes, sería:

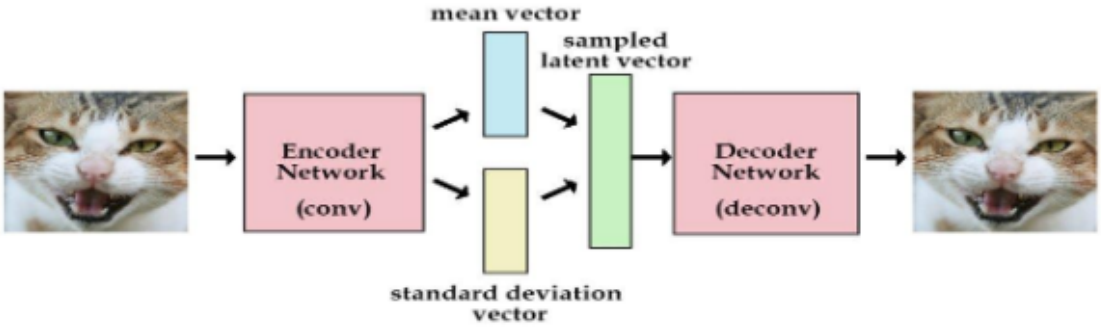

Figura 8. Arquitectura de un Variatonal Autoencoder.

Un Variational Autoencoder, es la misma arquitectura de un Autoencoder, **pero el encoder genera, un vector de valores, y un vector de desviación media de igual dimensión**, de tal manera que cada elemento del vector de valores se modifica un valor aleatorio entre 0 y el elemento correspondiente del vector de desviación media. **Generando así diferente vector latente para el mismo output y evitando el problema antes mencionado.**

De tal manera, mientras que, en el Autoencoder, una misma imagen se representaba siempre igual, ahora una misma imagen de un gato, tiene un espacio latente en el que se representa, y el decoder tiene que comprender y diferenciar dentro de este espacio, que animal había en la imagen. Si tuviéramos un espacio de 2 dimensiones, en el que representar las imágenes para distinguir si es un perro o un gato, la diferencia entre Autoencoder y VAE se ve en la figura 9. En el autoencoder, cada imagen de un tipo de gato o perro diferente se representa con una cruz, si recibe una imagen de un gato con una mancha que nunca ha visto en el pelaje de un gato, el Autoencoder no sabrá determinar si es gato o perro, mientras que el VAE, al tener un espacio latente grande para los gatos con manchas, identificará correctamente que es un gato.

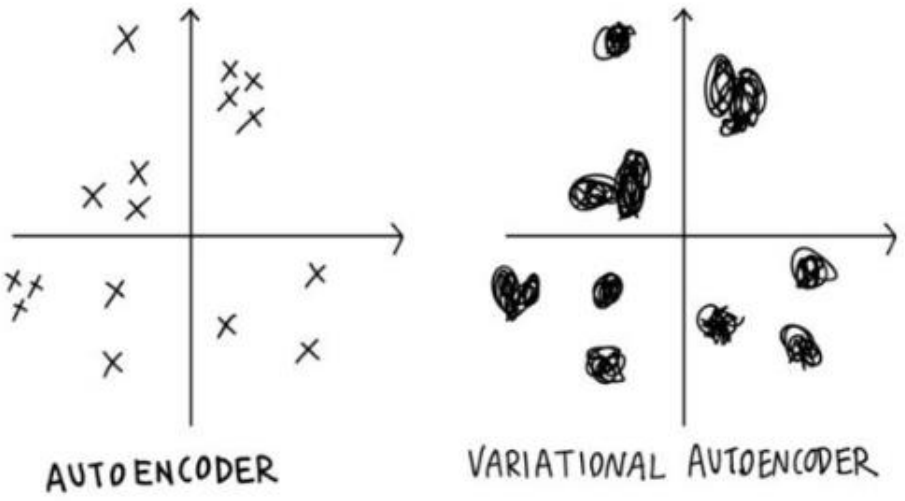

Figura 9. Comparación del espacio de codificación entre un Autoencoder y Variational Autoencoder.

3.1.1.3. VAE aplicado a moléculas

Esta diferenciación entre VAE y Autoencoder es muy importante, ya que es la razón por la que hemos escogido utilizar el VAE. Ya que este permite que, aunque no se haya recibido nunca un metabolito, se puede generar un vector de codificación parecido al de un metabolito que si se ha recibido. Por tanto, se puede predecir que es parecido a este

metabolito, y si está bien ajustado el VAE, determinar correctamente la estructura del metabolito que nunca se ha visto.

El VAE que utilizamos, utiliza toda la estructura del encoder y decoder que se define en el artículo (Jin et al., 2018). Sigue la estructura de la figura 10. **Tiene 2 VAE.**

1) El primero. **A partir de la estructura de árbol de la molécula que se generó anteriormente,** que tiene una dimensión máxima de X niveles y N nodos finales, **se obtiene el vector latente intermedio de M nodos finales,** gracias a unos modelos matemáticos y paso de mensajes entre los nodos del grafo. Este vector latente intermedio, es determinista y representa la estructura de la molécula, es decir, a partir de este vector intermedio se puede obtener el árbol que representa a la molécula. **El decoder, tiene que aprender** a generar los mismos modelos matemáticos y cambios en los pesos de los enlaces de las neuronas artificiales, **para ser capaz de reconstruir el árbol de X niveles y N nodos, y a partir de aquí obtener la estructura de la molécula.**
2) El segundo VAE. **Simplemente se basa en todos los átomos que se encuentran en el SMILES, y a partir de aquí genera una representación numérica de tamaño M** (como el primer VAE). Esta representación no es determinista, no sirve para extraer directamente los átomos del metabolito, sino que a partir de la estructura obtenida con el primer VAE, se aplican unos modelos matemáticos y se regulan los pesos de las neuronas, para llegar a obtener la probabilidad de cada átomo (o agrupación, como un benceno) en cada uno de los nodos del grafo que representa la estructura.

Una vez aplicado este segundo VAE sobre la estructura que se obtiene del primero, se obtiene un metabolito, que se busca que sea el mismo SMILES, que el SMILES que se introdujo al JTVAE. El conjunto de la arquitectura que sigue este JTVAE se observa en la figura 10. Se observa la ramificación de la descomposición de la molécula en un árbol, y su codificación y decodificación en la parte derecha. Mientras que se observa también en la parte izquierda la codificación a partir del grafo molecular, y la decodificación basada en esta codificación y el árbol de la estructura de la molécula obtenido.

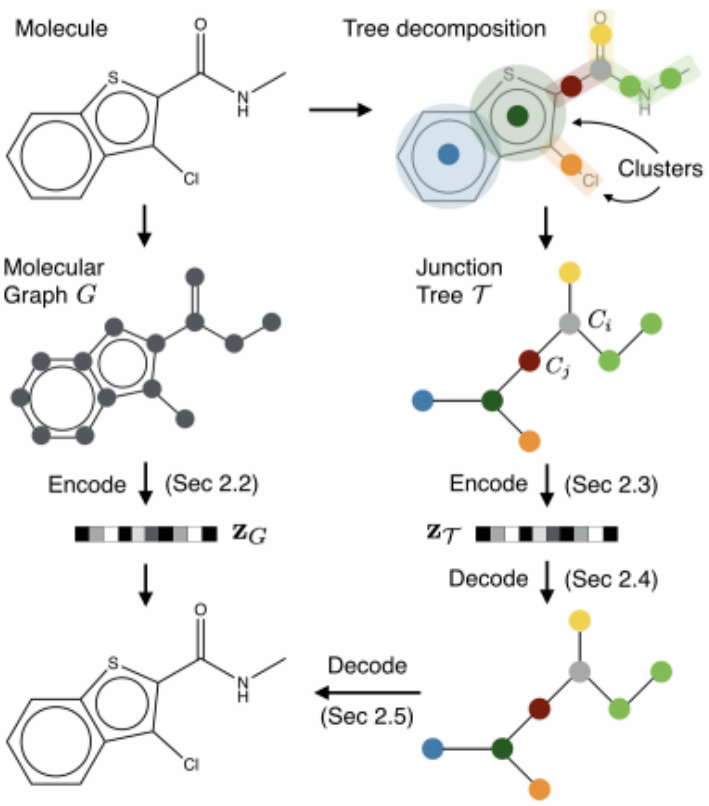

Figura 10. Arquitectura del JTVAE para codificar y decodificar una molécula.

3.1.2. Red neuronal convolucional

Una red neuronal convolucional es un tipo de aprendizaje profundo que se fundamenta en las operaciones de convolución utilizadas principalmente en la visión por computador, clasificación de imágenes, análisis de imágenes médico, procesamiento de lenguaje y reconocimiento de video. (Collobert and Weston., 2008), (Van Den Oord et al., 2013) **La operación de convolución se utiliza para reducir la información en una imagen, para que esta sea más fácil de procesar, y obtener las características importantes de esta imagen.** (Tutorial redes neuroanles convolucionales, 2019), (Aghdam and Heravi, 2017) Las redes neuronales convolucionales se utilizan para extraer estas características y posteriormente entenderlas como una red neuronal artificial típica. En la figura 11, se puede encontrar una estructura de red neuronal convolucional, esta red se basa en reconocer imágenes que contienen números que han sido dibujados a mano e interpretar qué número hay en la imagen. (Explacacion CNNs reconocimiento, 2019)

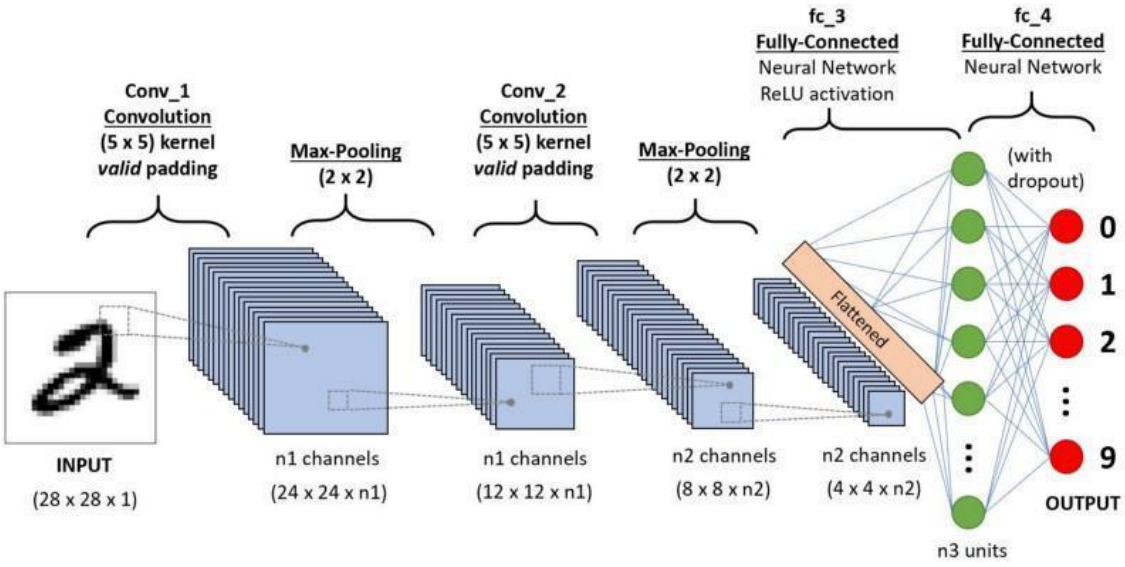

Figura 11. Ejemplo de red neuronal convolucional. A partir de una imagen de entrada se clasifica la imagen en función del número que contiene.

Una red neuronal convolucional se compone de:

1) **Capas de convolución**. En estas capas se realiza la operación de convolución el objetivo es extraer las características más importantes de los datos de entrada (imagen, señal temporal...). Las primeras capas de convolución se encargan de detectar menos características y más sencillas, mientras que las capas de convolución más profundas se encargan de características más complejas. Para las imágenes, que es donde se usan principalmente estas redes neuronales**, se realiza con un núcleo convolucional,** de 2 dimensiones sobre una imagen de 2 dimensiones, donde la dimensión del núcleo es menor que la de la imagen (se aplica un núcleo de dimensiones 3*3 píxeles a una imagen de 100*100 pixeles). En lugar de enlaces entre las diferentes capas de neuronas, el valor que le llega a una neurona en la capa n+1, se obtiene al aplicar el núcleo convolucional en la capa n. **De tal manera que en la capa n+1, se obtiene la información anterior con una característica extraída.** Se pueden aplicar tantos núcleos convolucionales como sean necesarios y cada uno extrae una característica importante de la información de entrada. Por ejemplo, si se aplican 8 nucleos convolucionales a la información de entrada, en la primera capa convolucional se obtienen 8 características diferentes de esta información de entrada. En la figura 11 (arriba), el primer paso es una capa de convolución donde se aplican varios

filtros de convolución y se obtienen varias imágenes con las características del dígito extraídas en la capa 1.

2) **Capas de reducción de características**. Esta capa siempre va después de una capa convolucional, y en estas se aplica una reducción de la imagen donde se ha obtenido la característica. **Con esto se pretende añadir simplicidad y resumir las características de la información de entrada, para que dos entradas de información similares obtengan resultados similares ante un mismo núcleo de convolución.** Normalmente, se basan en coger el valor máximo de un grupo de neuronas, por ejemplo, en la figura 11, se aplica una reducción de (2,2) a la imagen y, por tanto, de los valores que tienen las 4 neuronas de esta reducción, sólo se escoge el máximo, esto se realiza para toda la imagen saltando de 4 en 4 neuronas. Debido a esto, se reduce la dimensión de cada una de las imágenes de entrada a la mitad y se obtiene sólo los valores más importantes para cada filtro. Esta capa y la anterior se pueden introducir juntas tantas veces como se quiera en la estructura de la red neuronal convolucional, para extraer tantas características como sea necesario y para simplificar estas características.
3) **Capas totalmente conectadas.** Una vez obtenidas todas las características de la información de entrada se obtiene una única capa neuronal típica, que procesa la información de todas estas características y la pasa a la siguiente capa. Hasta que se llega a la capa de salida, que da información de salida clasificando la información de entrada si es un problema de clasificación, o prediciendo unos valores de salida si es un problema de regresión.

3.2. Obtención del espectro de masas de los metabolitos y nuestra base de datos de espectro de masas

Los espectros de masas con los que trabajamos en nuestro proyecto son los espectros de masas en tándem. (Niessen, 1999) La espectrometría de masas en tándem consiste en:

1) **Separación de los analitos.** Normalmente es mediante cromatografía líquida (la más común), cromatografía de gases o electroforesis capilar. El método varía en función de la muestra biológica y del tipo de metabolitos que se quieren separar.
2) Una vez los metabolitos están separados se seleccionan. Posteriormente se ionizan los metabolitos y **se separan en función de su relación carga/masa en un espectrómetro de masas.** Esto sería obtener el espectro de masas 1.

3) Después, los iones de una relación masa/carga concreta se seleccionan y se crean iones fragmentados por diversas técnicas como fotodisociación o disociación inducida por colisión. **Los iones resultantes se separan y detectan en un espectrómetro de masas. Produciendo el espectro de masas en tándem o espectro de masas 2.**

Ambas fases de separación de formación de los iones y separación en función de la relación masa/carga pueden tener diferentes arquitecturas de proceso. Un ejemplo de arquitectura sería tándem en el espacio como muestra la figura 12.

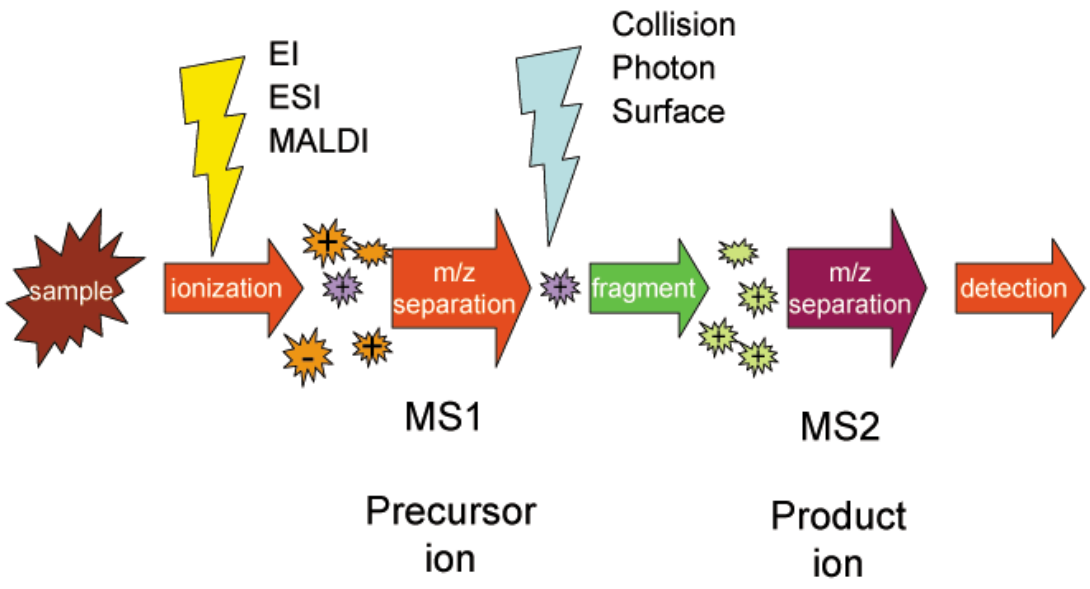

Figura 12. Ejemplo de arquitectura de espectrometría de masas en tándem. A partir de una muestra se ioniza y se separa en relación a la masa/carga (MS1). Se selecciona una molécula ionizada y se crean los iones fragmentados para el espectro de masas 2 (MS2).

Gracias a esta técnica podemos obtener el espectro de masas 2 de un metabolito, que representa las diferentes partes del metabolito separadas gracias a la ionización. Cada una de las partes del metabolito representa un pico en el espectro. La altura del pico representa la intensidad relativa del pico y el eje x representa su relación masa/carga. (McLafferty and Turecek, 1993)

La disposición de los picos en un espectro de masas tiene un sentido químico, y por tanto se puede determinar inequívocamente el metabolito a partir de estos picos. Actualmente se hace comparando con las bases de datos de espectro de masas de los metabolitos. El problema es que apenas hay espectros de masas para los metabolitos (10% de los metabolitos conocidos), y por tanto no se puede determinar a qué metabolito corresponde un espectro de masas a no ser que ya se haya obtenido. Otra solución es mediante un gran

conocimiento de química y bioquímica estudiar la disposición de los picos y estimar con qué metabolito se está trabajando, pero los métodos existentes que se basan en esta premisa son poco precisos (Wolf et al., 2010).

Nosotros trabajamos con espectros de masas 2 de metabolitos. Estos espectros han sido obtenidos con métodos distintos, y en la base de datos está indicada la técnica que se ha utilizado. Normalmente se utiliza como ión para separar por iones el ión de H+ debido a que su representación en las bases de datos es mayor, pero para nuestro trabajo no hacemos distinción de ion de ionización. Además, el espectro de masas 2 se obtiene a una energía de ionización, que varía en función de los métodos usados para la obtención de espectro de masas. Nosotros nos hemos centrado en trabajar solo con aquellos espectros que son de entre 15 y 25 eV para que haya consistencia en nuestro trabajo.

4. RESULTADOS Y DISCUSIÓN

En este apartado voy a explicar el desarrollo de todo el proceso siguiendo los pasos que hemos realizado para el desarrollo final del modelo de reconstrucción de metabolitos a partir del espectro de masas de estos. Y voy a comentar los diferentes resultados que vamos obteniendo.

4.1.Reproducción de resultados del JTVAE

A pesar de la precisión que reporta el artículo (Jin et al., 2018) con la reconstrucción de moléculas, nuestra primera tarea ha sido cerciorarnos del correcto funcionamiento y del poder de reconstrucción del modelo. Por tanto, hemos intentado recrear el modelo que ellos proponen, que obtiene un 77% de reconstrucción de fármacos nunca vistos por el modelo. El entrenamiento de este JTVAE, se compone de dos partes, al principio empieza sin regularización de **Kullback–Leibler divergence** (Kullback and Leibler., 1951), lo que facilita un rápido aprendizaje del modelo, posteriormente, se añade esta regularización para que las neuronas que componen el JTVAE se refinen y funcionen mejor ante ejemplos que nunca ha visto el modelo. Los fármacos que se utilizan en este modelo proceden de la base de datos de moléculas Zinc, contiene 249455 moléculas, que se reparten en 5000 moléculas para el test, 24445 para la validación de parámetros y 220011 para el entrenamiento.

El código del JTVAE se encuentra en github (Codigo github JTVAE, 2019) y también una recomendación de cómo lo han entrenado con sus moléculas para obtener sus resultados. Además, para esta primera prueba, seguimos los parámetros que indican con el que crearon el modelo, donde el vector de codificación de dimensión mayor es de 900 y el vector de codificación menor de 56. No obtuvimos correctamente los resultados que se indica el artículo. **Como máximo obtuvimos un 52% de reconstrucción de moléculas.** Por tanto, decidimos probar nuevas configuraciones de entrenamiento para obtener los mejores resultados posibles de reconstrucción.

Nuestro mejor intento de reconstrucción se compone de:

1) 12 iteraciones de preentrenamiento sin regularización Kullback–Leibler sobre todas las moléculas.
2) 0 iteraciones de entrenamiento sin regularización Kullback–Leibler, ya que vimos que al aplicarla siempre disminuían los resultados obtenidos.

Los resultados obtenidos con nuestra configuración fueron de un 72% de reconstrucción de moléculas que nunca ha visto el modelo. Esto demuestra que hemos sido capaces de recrear un JTVAE para todo tipo de moléculas y, por tanto, podemos crear un JTVAE para metabolitos.

4.2. Reconstrucción de metabolitos en el JTVAE

Posteriormente, comprobamos que los metabolitos se pueden reconstruir a partir de la representación del modelo de JTVAE generado por el MIT para todo tipo de moléculas. Para esto, realizamos una reconstrucción de los metabolitos que tenemos en nuestra base de datos. Estos metabolitos provienen de la base de datos Nist donde tenemos 8256 metabolitos. Para todos estos metabolitos hicimos dos pruebas de reconstrucción.

- La primera que consiste en que la predicción de la molécula sea exactamente igual que la real.
- La segunda prueba, utilizando un coeficiente de Tanimoto de 0.95, que introducimos como mejora más adelante. En este caso, solo si las moléculas son 95% iguales o más se aceptan como reconstrucción.

De los 8256 metabolitos, 1799 fueron reconstruidos perfectamente con un SMILES exactamente igual. Esto constituye el 21,79% de todos los metabolitos. En la segunda prueba, con un coeficiente de Tanimoto de 0.95, **se obtuvo una puntuación de reconstrucción de 30,40%, es decir, se reconstruyeron 2510 metabolitos en total casi iguales que el metabolito objetivo.** Esto quiere decir, que un modelo que no se ha entrenado específicamente para los metabolitos, es capaz de reconstruirlos, y por tanto, un JTVAE es capaz de entender cómo funciona la reconstrucción de moléculas, y concretamente, la reconstrucción de metabolitos.

4.3. Desarrollo de un JTVAE con metabolitos

Una vez comprobada la manera de entrenar un JTVAE, y ver que permite la reconstrucción de metabolitos, hemos desarrollado un JTVAE entrenado con los metabolitos de la base de datos Nist, concretamente hemos hecho una división del 75% para el entrenamiento y el 25% para comprobar que ha aprendido a realizar la tarea. Hemos hecho unas modificaciones al conjunto del JTVAE para que funcionara con metabolitos.

- Hemos corregido pequeños bugs que hemos encontrado en el código, por ejemplo, en la detección de anillos aromáticos enlazados.
- Hemos generado un nuevo vocabulario de elementos y agrupaciones de elementos que se corresponda con nuestros metabolitos.
- Hemos cambiado diferentes parámetros de entrenamiento y del modelo. El cambio principal realizado es que el vector de codificación de dimensión mayor que utiliza nuestro JTVAE es de 1260 valores (compuesto por dos subvectores de 630 valores) y el vector de codificación menor es de 78 (compuesto por dos subvectores de 39 valores). Este cambio lo hemos realizado debido a que la mayoría de los metabolitos son más grandes que las moléculas utilizadas por el MIT.
- Para reconstruir y probar nuestro modelo, aunque consideramos importante la diferencia que puede haber entre un metabolito y su isómero, puesto que puede llevar a que el comportamiento de la molécula en una ruta metabólica sea completamente diferente. Consideramos que disminuye mucho la eficiencia de los modelos de JTVAE de metabolitos que obtenemos y, por tanto, no convertimos los SMILES a canónicos e isoméricos, simplemente dejamos que el modelo procese los SMILES tal y como los recibe en la entrada. Esto puede llevar a que la reconstrucción de una molécula se identifique como correcta, pero posteriormente sea el isómero, o una molécula muy similar.
- A la hora de reconstruir moléculas que no se han visto nunca, trabajamos con el coeficiente de Tanimoto en vez de comparar los SMILES obtenidos y que sean exactamente iguales. El coeficiente de Tanimoto es una métrica utilizada para comparar la similitud y diversidad de una prueba frente al objetivo, permite categorizar dos moléculas prácticamente iguales como iguales. Si dos moléculas tienen un coeficiente de Tanimoto de 1, es que son iguales. (Real and Vargas, 1996)

Estos dos últimos puntos nos permiten ser menos severos con la reconstrucción del metabolito, ya que no buscamos que sea siempre perfecto. Sino que nuestro objetivo final es que al juntar el JTVAE con la CNN, a partir del espectro de masas, se obtenga el metabolito correcto por lo menos una vez cada diez intentos, aunque nunca haya lo haya visto nuestro modelo.

La manera de entrenamiento que hemos utilizado para la reconstrucción de nuestro modelo final es:

1) 14 iteraciones de preentrenamiento sin Kullback–Leibler sobre todos los metabolitos de entrenamiento.
2) 36 iteraciones de entrenamiento con Kullback–Leibler sobre todos los metabolitos de entrenamiento.

Al final, hemos generado un JTVAE para reconstruir metabolitos a partir del SMILES de los metabolitos. Los resultados obtenidos los hemos hecho con las dos pruebas mencionadas anteriormente. Primero dando como correctos aquellos metabolitos que la predicción de representación de la molécula (SMILES) es exactamente igual que el SMILES real. Segundo, dando como correctas aquellas predicciones que son un 95% iguales que la molécula objetivo. Estos resultados los comparamos con el JTVAE generado por el MIT, y que hemos obtenido los resultados en el punto 3.2. Los resultados se encuentran en la figura 13.

Creador	Método de Reconstrucción	Score
MIT	Iguales	21.79
MIT	0.95 Tanimoto	30.40
seesLab	Iguales	23.95
seesLab	0.95 Tanimoto	47.84

Figura 13. Resultados del JTVAE generado por nosotros con metabolitos y comparado con el JTVAE generado por el MIT.

Como vemos, nuestro modelo tiene un porcentaje de acierto de 23.95% en el primer tipo de reconstrucción y **un 47,84% en el segundo tipo de reconstrucción, para los metabolitos que no ha visto nunca**. Además, tiene un 89,45% para los metabolitos con los que ha sido entrenado. Estos resultados, son peores que el JTVAE generado por el MIT para todo el conjunto moléculas, pero mejores para los metabolitos en concreto. **Es decir, hemos generado un JTVAE que entiende mejor la tarea de reconstrucción de metabolitos que el generado en (Jin et al., 2018).**

En la reconstrucción de metabolitos, hemos guardado el SMILES de los metabolitos reconstruidos correctamente y el vector latente que se ha obtenido en el encoder para la reconstrucción, en total son 6501 metabolitos reconstruidos de los 8256 metabolitos iniciales. Hemos generado dos ficheros, uno con el vector de codificación de dimensión

mayor que genera el autoencoder y otro con el vector latente de dimensión baja que recibe el decoder, ya con la media y desviación estándar aplicadas a este vector.

Con lo cual creemos que la representación latente de una molécula que se genera en el encoder y que se interpreta en el decoder puede utilizarse para reconstruir nuevas moléculas que nunca ha visto el modelo. Para ello en los siguientes puntos hemos buscado una manera de generar esta representación latente a partir del espectro de masas de un metabolito.

4.4. Desarrollo de una CNN para el espectro de masas de metabolitos

Para nuestra red neuronal convolucional, tratamos el espectro del metabolito como si fuera un vector de una dimensión al que se le pueden aplicar operaciones de convolución para extraer características interesantes.

Nosotros transformamos los espectros de metabolitos que tienen como máximo 420 m/z en el eje x, a un **vector de información de entrada de la red neuronal de tamaño 4200.** Por tanto, cada 0.1 del eje x del espectro de masas, se corresponde con un valor de entrada de la red neuronal, este valor de entrada es la intensidad relativa del pico. Cada 0.1 que no tiene ningún pico en el espectro de masas, simplemente se indica con un 0.

Para todos los metabolitos que ha reconstruido el JTVAE generado por nosotros y que ha obtenido correctamente su reconstrucción, hemos guardado en un archivo los vectores de codificación de estos y su SMILES. Además, a estos archivos (uno para el vector de codificación de dimensión mayor y otro para el vector de codificación de dimensión menor), se les ha añadido: i) la Inchi-Key, que sirve como identificador al igual que el SMILES y ii) los espectros de masas obtenidos con una energía de ionización entre 15 y 25 eV que se encuentran en la base de datos. Un metabolito en el archivo que contiene la codificación latente de dimensión menor se representa en este archivo como se aprecia en la figura 14.

```
#SMILES:OC1=NC2=C(C=C(Cl)C=C2)C(C2=CC=CC=C2)=NC1
#INCHIKEY:AKPLHCDWDRPJGD-UHFFFAOYSA-N
#ESPECTRO:
    91     1013.0
   105      236.0
   116      128.0
   140     3269.0
   158      239.0
   165     2089.0
   166      244.0
   168      324.0
   193      467.0
   207      217.0
   208     1459.0
   214      116.0
   226      488.0
   242      198.0
   243     1073.0
   271     9999.0
#LABEL: [-1.3882606, -0.9089582, -0.6138637, -1.0128647, 0.025252074, -1.001285, 1.3721867, -0.624197, -1.9271256, 0.15692875,
0.94248646, 0.98777497, 3.979269, 1.8566954, -0.95830667, -3.143151, 1.5297697, -0.24411547, 5.541855, 0.49772432, 1.8827755,
-1.6843252, 0.07987944, 1.5359628, -3.0779846, -0.95625746, -2.1040988, -0.28204605, 2.9470553, 1.5414604, -1.1679413, 0.6472247,
-1.4911536, 2.7717059, -1.3708091, 1.7562816, -2.9226148, 2.3855472, -3.3449774]  [-0.85598326, 0.12136239, -0.6236068,
0.31375653, 0.041722655, -1.2066479, -0.23541185, 1.3065804, 0.09046663, 1.0096129, 0.08546713, -0.01694423, 0.7387039,
-0.8125488, 0.031346515, -0.28418413, 0.8081554, 0.10718234, 0.4682376, 0.1860568, 1.7676907, -0.93215835, -0.2369164,
0.64434457, 0.19280368, 0.18022856, -0.17266306, -0.48351285, 0.2515006, 0.2230373, -0.25714442, -0.2734939, -1.3068135,
1.5257738, -0.43389437, -0.20784388, 0.1959379, 0.0186418, 2.8274179]
```

Figura 14. Representación de un metabolito en los archivos para trabajar con la red neuronal convolucional. Se presenta el SMILES y la INCHI-KEY, que identifican al metabolito, el espectro de masas 2 del metabolito y el vector de codificación del metabolito.

Como se observa en la figura, tenemos dos vectores de codificación de cada dimensión, uno que se ha generado para el árbol que representa la molécula y otro para el grafo molecular de la molécula. En nuestro caso el vector de dimensión baja tiene 78 valores, 39 para el árbol y 39 para el grafo molecular. **Por tanto, tenemos que desarrollar 2 redes neuronales convolucionales donde a partir del espectro de masas se obtiene en cada una un vector de codificación, que juntos forman el vector de codificación que usa el decoder del JTVAE.**

Para el entrenamiento hemos usado la función de perdida llamada error cuadrático medio ECM. Para calcular el ECM, sumamos para cada uno de los valores de salida el cuadrado de la diferencia entre la predicción y el valor real. Esta suma se divide por el número de valores *n*. La fórmula es:

$$\mathrm{ECM} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2.$$

Figura 15. Fórmula del error cuadrático medio. $\hat{\boldsymbol{Y}}$ es el vector de predicciones. $\boldsymbol{Y}$ es el vector de valores reales.

La red neuronal convolucional ha sido entrenada con el 80% de todos los metabolitos que teníamos espectro y que nuestro JTVAE ha funcionado. Mientras que el 20% restante se ha utilizado como prueba para comprobar si funciona con espectro de masas de metabolitos que nunca ha visto la red neuronal convolucional.

Hemos trabajado con los vectores de codificación de dimensión alta (1260 valores) y con los vectores de dimensión baja (78 valores). Para comprobar cual proporciona mejores

resultados en la red neuronal convolucional y para comprobar cual obtiene mejores resultados al reconstruir la salida de la red neuronal convolucional con el decoder y obtener el metabolito reconstruido. En la figura 16 se observan los resultados del primer metabolito del entrenamiento. En la representación de la salida de la CNN se aprecian dos líneas, mientras que la azul son los valores que se busca predecir, la línea naranja es la predicción realizada.

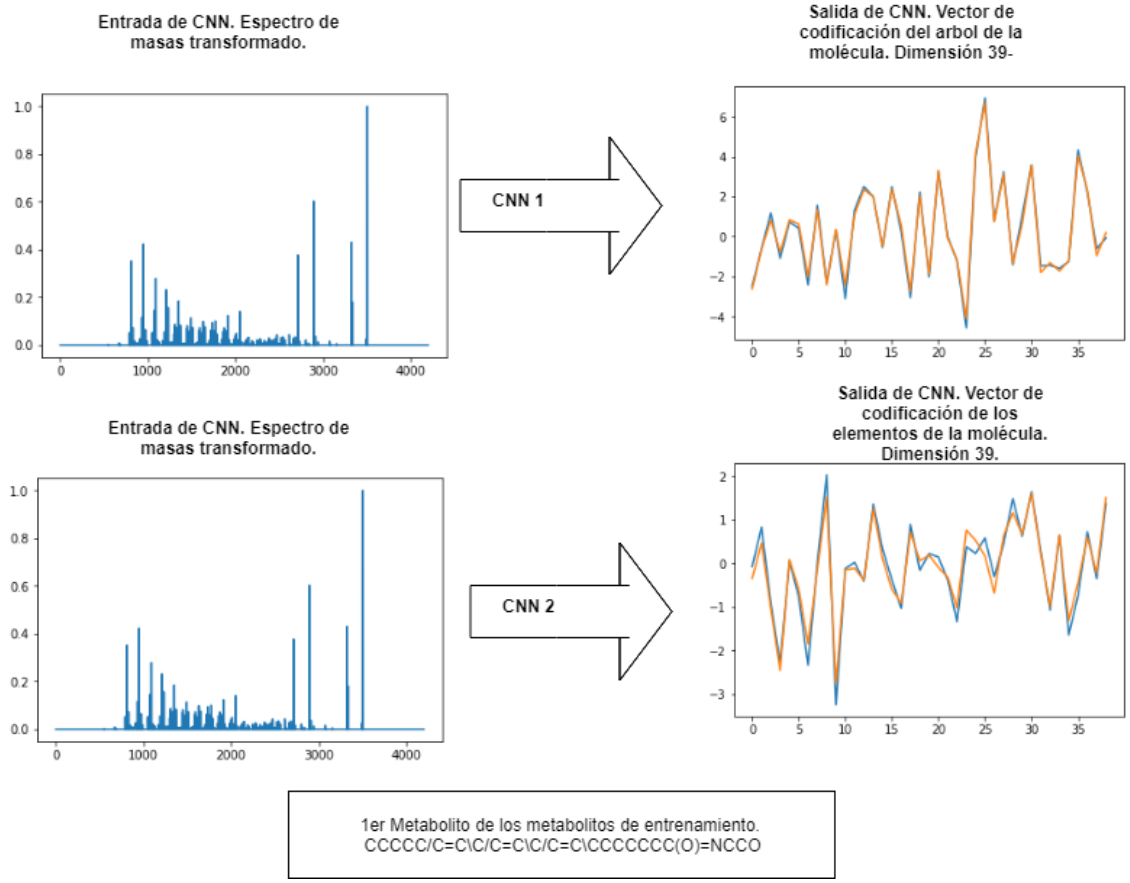

Figura 16. Representación de la obtención del vector de codificación a partir del espectro de masas de un metabolito que se ha usado para el entrenamiento. Mediante ambas redes neuronales.

En la figura 17 se observan los resultados del primer metabolito de los metabolitos nunca vistos.

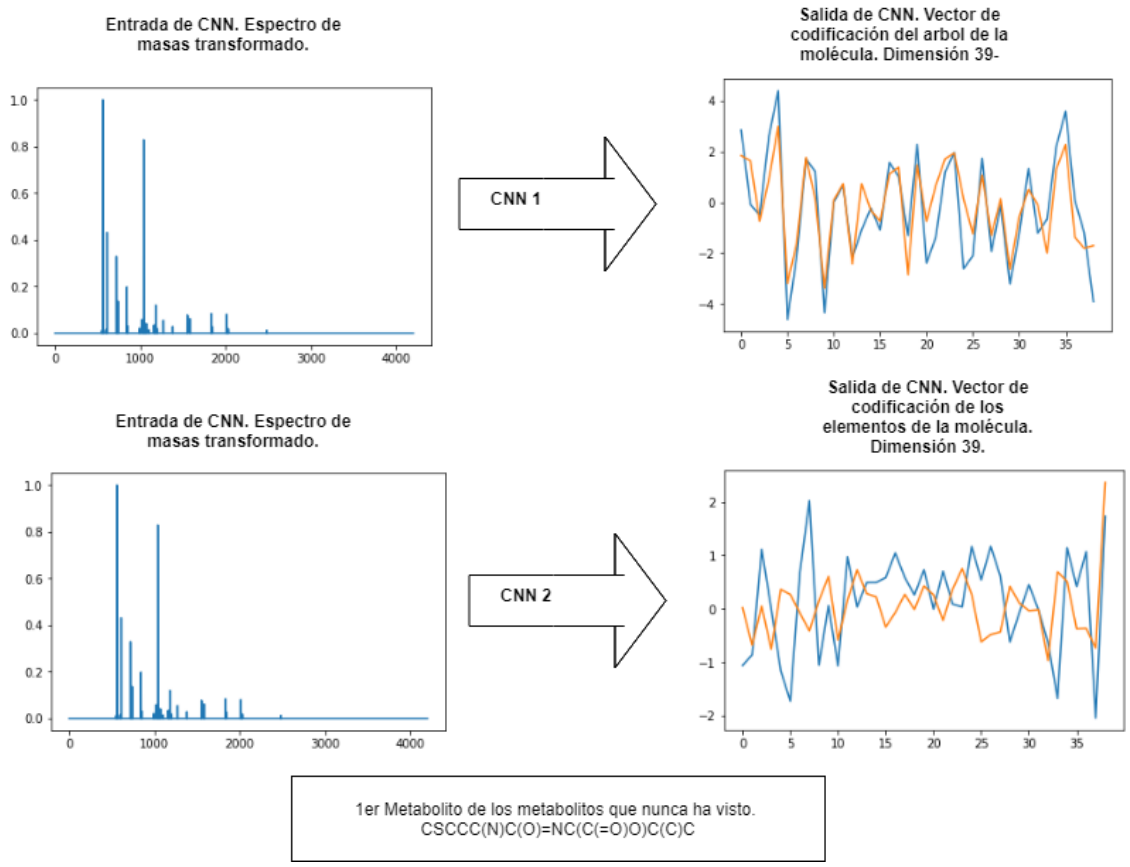

Figura 17. Representación de la obtención del vector de codificación a partir del espectro de masas de un metabolito que no se ha visto nunca. Mediante ambas redes neuronales.

Como se observa, tanto los resultados del entrenamiento como de los metabolitos que nunca ha visto son correctos y predicen correctamente el vector de codificación que recibe el VAE de metabolitos. También se observa que los resultados del entrenamiento son mucho más correctos que los resultados de los metabolitos nunca vistos. Esto indica que obtendremos mucha mejor reconstrucción en el entrenamiento que en los metabolitos nunca vistos.

4.5. Reconstrucción de los metabolitos a partir del vector de codificación generado

Hemos utilizado el conocimiento de redes neuronales convolucionales y hemos obtenido la codificación que da el JTVAE a partir del espectro de masas de un metabolito. Al final, la codificación que nos proporciona nuestra red neuronal convolucional la introducimos en el decoder del JTVAE y, por tanto, se puede reproducir el metabolito a partir del espectro, independientemente de si este metabolito se ha utilizado para entrenar alguna

de las arquitecturas de Inteligencia Artificial o si no se ha visto nunca el espectro del metabolito.

La arquitectura final de nuestro trabajo es la que se encuentra en la figura 18.

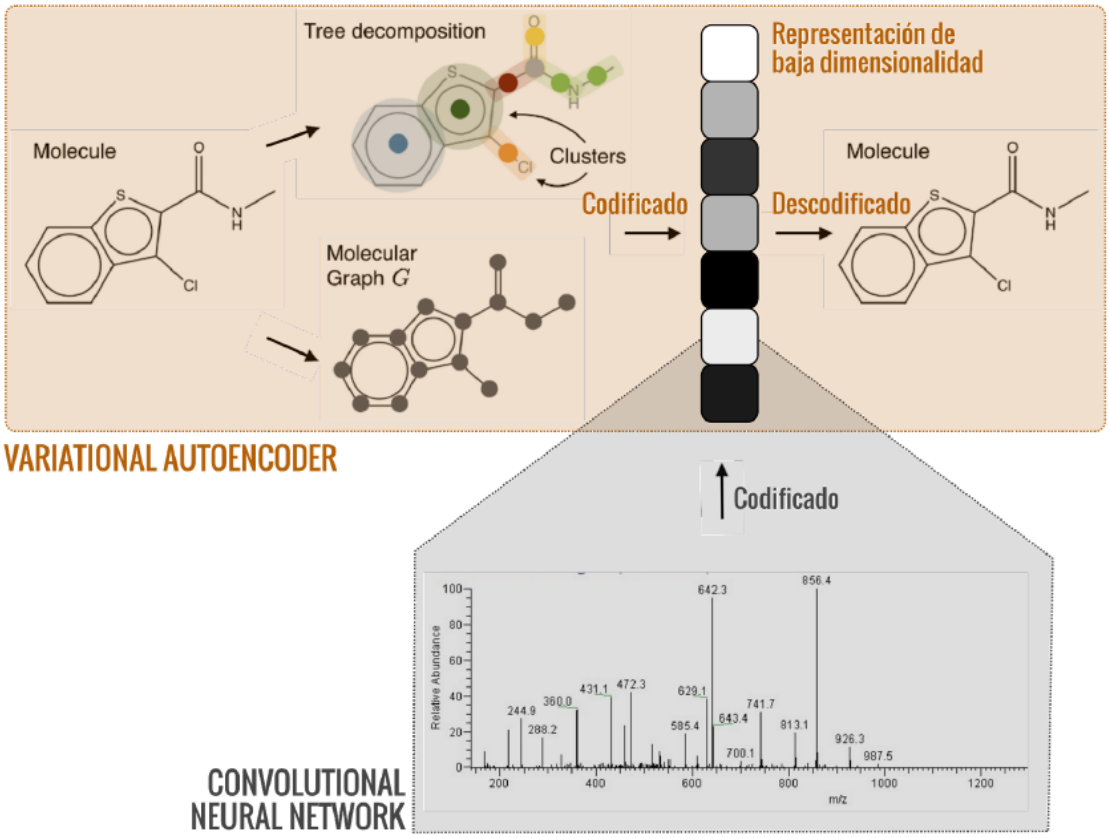

Figura 18. Arquitectura de nuestro trabajo que integra tanto el JTVAE como la red neuronal convolucional para reconstruir metabolitos a partir del espectro de masas.

Como he mencionado anteriormente, he generado dos redes neuronales para generar las dos partes del vector de codificación. Y esto lo hemos hecho tanto para el vector de codificación que genera el encoder, de 1260 valores, como para el vector de codificación que recibe el decoder, de 78 valores. Los resultados de reconstrucción de las moléculas a partir de estos dos vectores de codificación son los que se encuentran en la figura 19.

Vector de codificación	Metabolitos	Reconstrucción
78	Entrenamiento	93.21
78	Nunca vistos	24.54
1260	Entrenamiento	90.83
1260	Nunca vistos	18.34

Figura 19. Resultados de reconstrucción de metabolitos a partir del espectro de masas 2 de los metabolitos.

Nuestra herramienta reconstruye mucho mejor aquellos metabolitos con los que las redes neuronales se han entrenado. El mejor modelo es el que se utiliza el vector de codificación de 78 valores que recibe el decoder y a partir del que se reconstruye directamente la molécula. Este vector está compuesto por 2 subvectores de 39 valores que se generan a partir del espectro de masas por dos redes neuronales diferentes. Para los metabolitos de entrenamiento tiene una reconstrucción del 93.21%. Aunque no funciona igual de bien que con los metabolitos de entrenamiento, nuestro modelo es capaz de reconstruir el 24,54% de los metabolitos que no ha visto nunca.

Estos resultados suponen un gran avance ya que permite determinar la molécula a la que pertenece un espectro de masas, aunque no se encuentre en las bases de datos, un 24,54% de las veces**. Por tanto, al utilizar esta herramienta de Inteligencia Artificial se puede acercar la metabolómica a las otras ciencias ómicas, ya que no es necesario que se haya catalogado en las bases de datos el espectro de masas de un metabolito,** a diferencia de CFM-ID (Allen et al., 2016) y MetFrag (Wolf et al., 2010) que se identifican los compuestos en las bases de datos, o iMet. (Aguilar-Mogas et al., 2017), que necesita el posterior trabajo de un experto bioquímico. Como limitación de estos resultados, encontramos que se ha producido overfit, hemos trabajado con muchas arquitecturas de redes neuronales y probado muchos modelos hasta obtener estos resultados. Creemos que esta diferencia tan grande en reconstrucción de metabolitos nunca vistos y metabolitos de entrenamiento se debe principalmente a que nuestra base de datos es pequeña y no permite representar todas las posibles maneras de codificar el espectro de masas de un metabolito

Por último, vamos a enseñar unos ejemplos de reconstrucción de diferentes moléculas, donde a partir del espectro, se genera un vector de codificación (en naranja) que se compara con el espectro de codificación real del metabolito (en azul), y a partir de este vector de codificación se obtiene la molécula, y la comparamos con la molécula esperada. Hay casos donde nuestro modelo ha acertado y casos donde nuestro modelo se ha equivoca. Todo se observa en la figura 20. Es importante mencionar que, en la primera molécula, aunque la disposición espacial en la representación sea diferente, nuestro modelo considera que son moléculas iguales ya que el objetivo es reconstruir la fórmula, no encontrar un isómero concreto. Además, en la tercera molécula, nuestro modelo considera que acierta la molécula, pero con un coeficiente de Tanimoto de 0.95, debido a que la molécula es muy parecida a la esperada. Destacamos también, como una predicción claramente errónea la quinta molécula que la declara errónea correctamente, y que las

moléculas segunda, cuarta y sexta, las acierta totalmente sin ningún problema o inconveniencia.

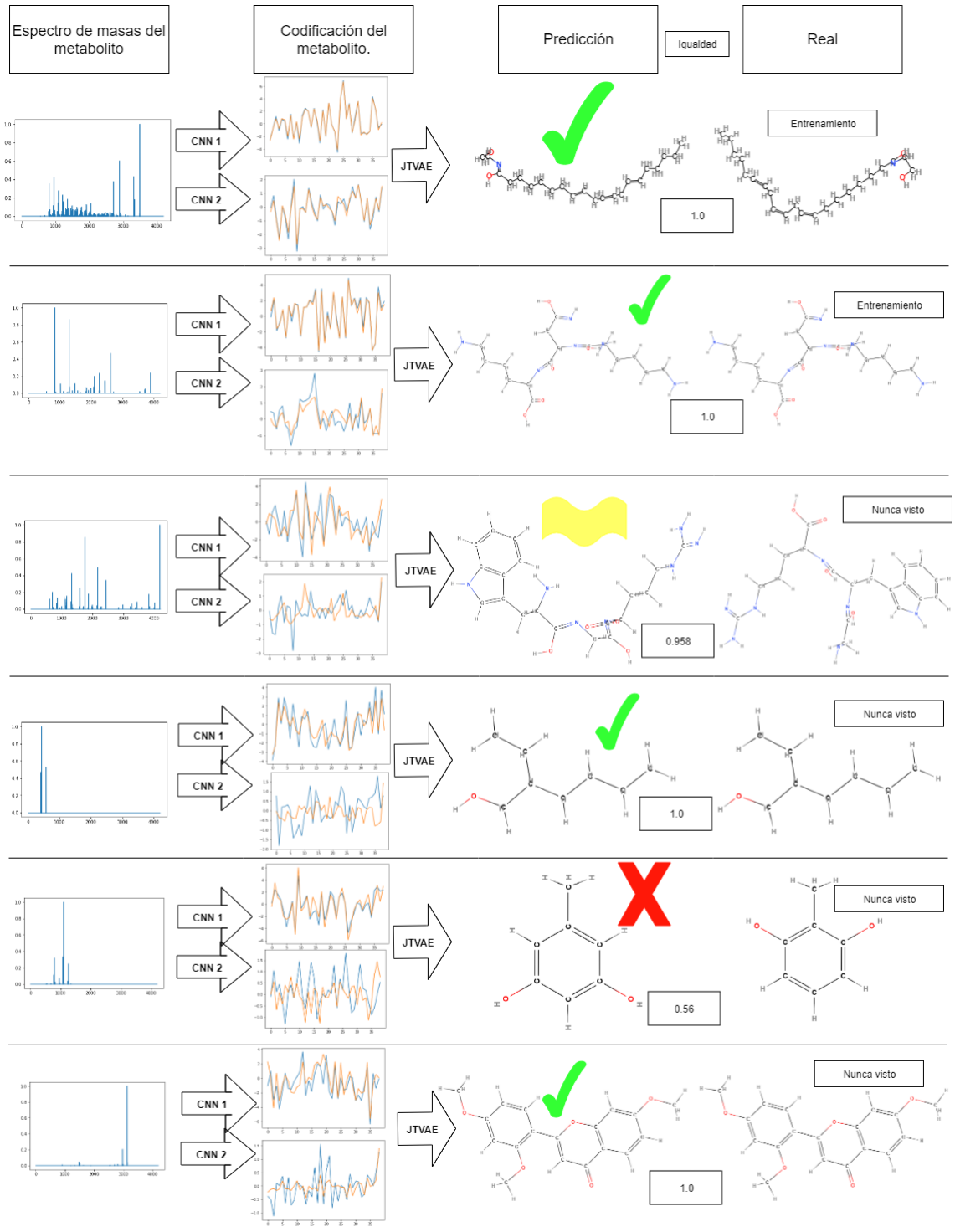

Figura 20. Representación de la reconstrucción de varios metabolitos. De izquierda a derecha se observan el espectro de masas del metabolito, los vectores de codificación predichos comparados con los objetivos, el metabolito predicho, el porcentaje de parecido entre el metabolito predicho y el objetivo, y el metabolito objetivo.

5. CONCLUSIONES

La reconstrucción de los metabolitos a partir del espectro de masas del metabolito es un grave problema actual que afecta a la metabolómica y a todas las ciencias de la biología de sistemas, ya que impide el desarrollo de modelos biológicos que comprendan completamente un sistema biológico y permitan realizar análisis y nuevas experimentaciones.

Actualmente no hay mejor manera que obtener el espectro de masas del metabolito y acceder a una base de datos para comprobar que metabolito es. Si el espectro de masas no se encuentra en la base de datos, no se puede determinar que metabolito es.

Nosotros, hemos cumplido con el objetivo principal de este trabajo de final de grado, y hemos desarrollado un modelo de inteligencia artificial que es capaz de:

- Reconstruir un metabolito que ya ha utilizado para aprender la tarea, con una seguridad del 93,21 %.
- Reconstruir un metabolito que nunca ha visto, con una seguridad del 24,54 %.

Por tanto, hemos conseguido desarrollar un modelo que mejora totalmente el método actual de determinar la estructura de un metabolito a partir del espectro de masas, ya que no es necesario que alguien haya encontrado el espectro de masas del metabolito que se tiene en una muestra biológica. Siendo este proyecto de investigación una solución a la problemática de la metabolómica.

Como conclusiones derivadas del objetivo principal del proyecto tenemos:

- Hemos generado un JTVAE, que comprende la tarea de reconstrucción de metabolitos, demostrando que este modelo de Inteligencia Artificial puede ser útil en la bioinformática.
- Además, este modelo puede ser utilizado de igual manera que nosotros para generación de nuevos metabolitos a partir de los vectores de codificación que presenta.
- Hemos demostrado que la lógica detrás de las redes neuronales convolucionales que generalmente se usan para estudios con análisis de imagen, también sirven para el espectro de masas de un metabolito. Esto se debe a que el espectro de masas se basa en los componentes del metabolito, y esto son características extraíbles del espectro.

Dentro de este proyecto, hay varios frentes abiertos para proyectos futuros que podrían ser muy interesantes y que refinarían la solución al problema de la metabolómica y mejorarían resultados:

- Estudiar otras maneras con las que se puede representar un metabolito además del espectro de masas.
- Trabajar con bases de datos más grandes que contengan más metabolitos como la HMDB, que permita mejorar la manera de comprender la tarea de la Inteligencia Artificial.
- Implementar a la CNN el peso molecular de la molécula objetivo, ya que, para la reconstrucción, muchas de las representaciones erróneas de moléculas tienen un peso molecular diferente. Con esto creemos que se podría aumentar bastante la puntuación de reconstrucción. Esto ya se utiliza en (Aguilar-Mogas et al., 2017)

6. AUTOEVALUACIÓN.

Personalmente, considero que este proyecto de investigación ha sido la mejor experiencia de toda la carrera. He sido capaz de integrarme totalmente en un grupo de investigación con un proyecto de investigación realmente importante y necesario. Este proyecto une ambas disciplinas en las que he sido formado a lo largo de 5 años, como son la Biotecnología y la Ingeniería Informática, y me ha permitido encontrar como ambos campos de conocimiento se pueden complementar para mejorar el entendimiento de la biología a partir de los conocimientos informáticos.

Respecto a mis expectativas, desde un principio eran muy altas ya que el proyecto era perfecto para mis estudios y para mis intereses. Aunque, a decir verdad, también tenía dudas de si sería capaz de desarrollar el modelo tan complejo que hemos desarrollado. Aun así, como ya conocía a mis jefes, tenía plena confianza en ellos y en que este modelo se podía obtener.

Durante el desarrollo de este trabajo de final de grado he aprendido cómo funciona la investigación científica de manera real, y cómo se organiza un grupo de investigación y las tareas que en él realizan sus componentes. Obviamente, también he aumentado mis conocimientos en la biología de sistemas y concretamente en la metabolómica, y he sido capaz de comprender realmente la limitación actual, como se puede resolver y aportar mi propio trabajo para resolver esta limitación.

Bibliografía

Aghdam, H. H., & Heravi, E. J. (2017). *Guide to convolutional neural networks : a practical application to traffic-sign detection and classification*. Retrieved from https://urv.on.worldcat.org/search?queryString=no%3A+987790957&scope=#/oclc/987790957

Aguilar-Mogas, A., Sales-Pardo, M., Navarro, M., Guimerà, R., & Yanes, O. (2017). iMet: A Network-Based Computational Tool To Assist in the Annotation of Metabolites from Tandem Mass Spectra. *Anal. Chem*, *89*, 3482. https://doi.org/10.1021/acs.analchem.6b04512

Allen, F., Pon, A., Wilson, M., Greiner, R., & Wishart, D. (2014). CFM-ID: a web server for annotation, spectrum prediction and metabolite identification from tandem mass spectra. *Nucleic Acids Research*, *42*(Web Server issue), W94-9. https://doi.org/10.1093/nar/gku436

Bengio, Y., Lee, D.-H., Bornschein, J., Mesnard, T., & Lin, Z. (2015). Towards Biologically Plausible Deep Learning. Retrieved from http://arxiv.org/abs/1502.04156

Boesen, A., Larsen, L., Sønderby, S. K., Larochelle, H., Winther, O., & Dk, O. (2016). *Autoencoding beyond pixels using a learned similarity metric*. Retrieved from https://arxiv.org/pdf/1512.09300.pdf

Bruno López Takeyas. (2007). *INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL*. Nuevo Laredo, Tamps. Retrieved from http://www.cs.bham.ac.uk/~rmp/slide_book/slide

Chaston, J., & Douglas, A. E. (2012). Making the most of omics for symbiosis research. *The Biological Bulletin*, *223*(1), 21–29. https://doi.org/10.1086/BBLv223n1p21

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing. In Proceedings of the 25th international conference on Machine learning - ICML '08 (pp. 160–167). New York, New York, USA: ACM Press. https://doi.org/10.1145/1390156.1390177

CS231n Convolutional Neural Networks for Visual Recognition. (n.d.). Retrieved June 14, 2019, from https://cs231n.github.io/convolutional-networks/

Curso deep learning Hugo Larochelle.

http://www.dmi.usherb.ca/~larocheh/neural_networks/content.html 14/06/2019

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. https://doi.org/10.1021/acs.analchem.7b03929

Explicación VAE. https://jaan.io/what-is-variational-autoencoder-vae-tutorial/ 13/06/2019

Hollywood, K., Brison, D. R., & Goodacre, R. (2006). Metabolomics: Current technologies and future trends. *PROTEOMICS*, *6*(17), 4716–4723. https://doi.org/10.1002/pmic.200600106

Jin, W., Barzilay, R., & Jaakkola, T. (2018). *Junction Tree Variational Autoencoder for Molecular Graph Generation*. Retrieved from https://github.com/wengong-jin/icml18-jtnn

Kaplan, A., & Haenlein, M. (2019). Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence. *Business Horizons*, *62*(1), 15–25. https://doi.org/10.1016/j.bushor.2018.08.004

Kullback, S., & Leibler, R. A. (1951). On Information and Sufficiency. *The Annals of Mathematical Statistics*, *22*(1), 79–86. https://doi.org/10.1214/aoms/1177729694

Marblestone, A. H., Wayne, G., & Kording, K. P. (2016). Toward an Integration of Deep Learning and Neuroscience. Frontiers in Computational Neuroscience, 10, 94. https://doi.org/10.3389/fncom.2016.00094

McLafferty, F. W., & Tureček, F. (1993). *Interpretation of mass spectra*. University Science Books.

Niessen, W. M. A. (1999). MS–MS and MSn. *Encyclopedia of Spectroscopy and Spectrometry*, 1675–1681. https://doi.org/10.1016/B978-0-12-374413-5.00215-3

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature, 381(6583), 607–609. https://doi.org/10.1038/381607a0

Patti, G. J., Yanes, O., & Siuzdak, G. (2012). Metabolomics: the apogee of the omics trilogy. *Nature Reviews Molecular Cell Biology*, *13*(4), 263–269. https://doi.org/10.1038/nrm3314

Real, R., & Vargas, J. M. (1996). The Probabilistic Basis of Jaccard's Index of Similarity. *Systematic Biology*, *45*(3), 380–385. https://doi.org/10.1093/sysbio/45.3.380

Rojas-Cherto, M., Peironcely, J. E., Kasper, P. T., van der Hooft, J. J. J., de Vos, R. C. H., Vreeken, R., … Reijmers, T. (2012). Metabolite Identification Using Automated Comparison of High-Resolution Multistage Mass Spectral Trees. *Analytical Chemistry*, *84*(13), 5524–5534. https://doi.org/10.1021/ac2034216

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural Networks, 61, 85–117. https://doi.org/10.1016/J.NEUNET.2014.09.003

Snoep, J. L., & Westerhoff, H. V. (2005). From isolation to integration, a systems biology approach for building the Silicon Cell. In *Systems Biology* (pp. 13–30). Berlin/Heidelberg: Springer-Verlag. https://doi.org/10.1007/b106456

Tavassoly, I., Goldfarb, J., & Iyengar, R. (2018). Systems biology primer: the basic methods and approaches. *Essays In Biochemistry*, *62*(4), 487–500. https://doi.org/10.1042/EBC20180003

Tutorial redes neuronales convolucionales. http://deeplearning.net/tutorial/lenet.html 14/06/2019

Van Den Oord, A., Dieleman, S., & Schrauwen, B. (2013). Deep content-based music recommendation. Retrieved from http://papers.nips.cc/paper/5004-deep-content-based-music-recommendation.pdf

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

wengong-jin/icml18-jtnn: Junction Tree Variational Autoencoder for Molecular Graph Generation (ICML 2018). (n.d.). Retrieved June 14, 2019, from https://github.com/wengong-jin/icml18-jtnn

Wishart, D. S., Jewison, T., Guo, A. C., Wilson, M., Knox, C., Liu, Y., … Scalbert, A. (2012). HMDB 3.0—The Human Metabolome Database in 2013. *Nucleic Acids Research*, *41*(D1), D801–D807. https://doi.org/10.1093/nar/gks1065

Wolf, S., Schmidt, S., Müller-Hannemann, M., & Neumann, S. (2010). In silico fragmentation for computer assisted identification of metabolite mass spectra. *BMC Bioinformatics*, *11*(1), 148. https://doi.org/10.1186/1471-2105-11-148

Zakharov, E., Shysheya, A., Burkov, E., & Lempitsky, V. (2019). Few-Shot Adversarial Learning of Realistic Neural Talking Head Models. Retrieved from http://arxiv.org/abs/1905.08233

Anexo 1. Base de datos NIST.

En la base de datos, la información de un metabolito se representa como se observa en la figura 21.

El nombre del compuesto está indicado con la etiqueta ##CAS_NAME. El ión utilizado con un $:03, en este caso sería [M+H]+, posteriormente vienen indicadas las diferentes técnicas y procesos que se han usado para la obtención del espectro de masas 2.

A continuación, se indica la forma molecular del compuesto con la etiqueta ##MOLFORM y el peso molecular con ##MW.

Por último, se indica el espectro de masas 2 de metabolito con la etiqueta ##XYDATA, donde la columna de la izquierda es el eje x (la relación masa/carga) y la columna de la derecha es el eje y, la intensidad relativa del ión en ese pico, en este caso el máximo es 9999.

```
4944994  ##TITLE=Library Entry 72173
4944995  ##JCAMPDX=Revision 5.00
4944996  ##DATA TYPE=MASS SPECTRUM
4944997  ##SAMPLE DESCRIPTION=NIST Mass Spectrometry Data Center
4944998  ##CAS NAME=Tri-2-ethylhexyl trimellitate
4944999  ##NAMES=$:03[M+H]+
4945000   $:00MS2
4945001   $:04547.3993
4945002   $:06HCD
4945003   $:07Thermo Finnigan Elite Orbitrap
4945004   $:09direct flow injection
4945005   $:10ESI
4945006   $:12N2
4945007   $:0527
4945008   $:11P
4945009   $:17Consensus spectrum; Acetonitrile/Water/Formic acid; Vial_ID=5930; mz_d=
4945010  iff=0.0011
4945011   1,2,4-Benzenetricarboxylic acid, 1,2,4-tris(2-ethylhexyl) ester
4945012   $:28KRADHMIOFJQKEZ-UHFFFAOYSA-N
4945013  ##MOLFORM=C33H5406
4945014  ##CAS REGISTRY NO=003319-31-1
4945015  ##MP= -300
4945016  ##BP= -300
4945017  ##MW= 546
4945018  ##$RETENTION INDEX=0
4945019  ##$CONDENSED SPECTRUM=NO
4945020  ##NPOINTS= 10
4945021  ##XYDATA=(XY..XY)
4945022  57.0697      731.0
4945023  71.0853     1217.9
4945024  72.0886       28.0
4945025  113.1325      195.0
4945026  193.0131      515.0
4945027  194.0169       20.0
4945028  211.0240       19.0
4945029  305.1385     9999.0
4945030  306.1418      506.0
4945031  323.1492      176.0
```

Figura 21. Información de un metabolito en la base de datos Nist.