Trabajo Final de Grado

Grado en Ingeniería de Sistemas Audiovisuales

Predicting magnetic interactions between atoms using graph convolutional networks

Autor: Tomás Navas Gutiérrez
Co-directores: Javier Ruiz Hidalgo y Albert Mosella Montoro
Convocatoria: Junio de 2020

Universidad Politécnica de Cataluña (UPC)

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Resumen

La constante de acoplamiento es el resultado de la interacción entre dos momentos magnéticos de dos átomos de una molécula. Usando métodos de vanguardia, en lo que a la mecánica cuántica se refiere, es posible calcular con precisión las constantes de acoplamiento escalar dada sólo una estructura molecular 3D como entrada. Sin embargo, estos cálculos de mecánica cuántica son extremadamente caros (conllevan días o semanas por molécula) y, por lo tanto, tienen una aplicabilidad limitada en los flujos de trabajo diarios.

La propuesta en este proyecto es utilizar diferentes metodologías con redes neuronales para predecir constantes de acoplamiento. En este trabajo, se emplean técnicas tales como la *Fully Connected Network* (FC), *Graph Convolutional Neural Network* (GCN) y la *Graph Attention Neural Network (GAT).* De esta forma, se quiere predecir la constante de acoplamiento obteniendo distintos tipos de datos de la estructura química 3D de la molécula.

Para poder llevar a cabo todas las pruebas realizadas en este trabajo, se ha utilizado la base de datos CHAMPS *Scalar Coupling dataset* [1] que contiene 135.000 moléculas clasificadas.

Con los resultados obtenidos en este proyecto se observa que las redes de grafos obtienen mejores resultados que el *baseline*. Sin embargo, la metodología propuesta no consigue superar el estado del arte en la predicción de la constante de acoplamiento. Debido a que durante el desarrollo de este proyecto solo se ha contemplado el uso de enlaces dobles y *J-Coupling*, para simplificar y reducir la carga computacional.

Abstract

The coupling constant is the result of the interaction between two magnetic moments of two atoms of a molecule. Using state-of-the-art methods, as far as quantum mechanics is concerned, it is possible to accurately calculate the scalar coupling constants given only one 3D molecular structure as input. However, these quantum mechanics calculations are extremely expensive (they take days or weeks per molecule) and therefore have limited applicability in daily workflows.

The purpose in this project will be the application of different methodologies with neural networks to predict coupling constants. In this work, techniques such as the Fully Connected Network (FC), Graph Convolutional Neural Network (GCN) and the Graph Attention Neural Network (GAT) will be used. In this way, it is wanted to predict the coupling constant by obtaining different types of data on the 3D chemical structure of the molecule.

In order to carry out all the tests, it will be used the CHAMPS Scalar Coupling dataset [1] that contains 135.000 classified molecules.

With the results obtained in this project, it is observed that graph networks obtain better results than the baseline. However, the proposed methodology fails to overcome the state of the art in predicting the coupling constant. Because during the development of this project only the use of double links has been contemplated, to simplify and reduce the computational load.

Agradecimientos

En primer lugar, me gustaría agradecer a Javier Ruiz y Albert Mosella por el constante seguimiento y la ayuda que me han dado en este proyecto.

Además, quiero agradecer a mis amigos y familiares por el apoyo incondicional que han demostrado incluso en los momentos más difíciles.

Declaración de Honor

I declare that, the work in this Degree Thesis is completely my own work, no part of this Degree Thesis is taken from other people's work without giving them credit, all references have been clearly cited.
I understand that an infringement of this declaration leaves me subject to the foreseen disciplinary actions by *The Universitat Politècnica de Catalunya - BarcelonaTECH*.

Tomás Navas Gutiérrez		30/06/20
Student Name	Signature	Date

Title of the Thesis: Predicting magnetic interactions between atoms using graph convolutional networks

Índice

Lista de ilustraciones

Lista de Tablas

Glosario

- IA Inteligencia Artificial
- CNN *Convolutional Neural Network*
- RMN Espectro de Resonancia Magnética Nuclear
- GT *Ground Truth*
- FC *Fully Connected*
- MAE *Mean Absolute Error*
- GNN *Graph Neural Network*
- GCN *Graph Convolutional Network*
- GAT *Graph Attention Network*

1. Introducción

Desde ya hace más de 70 años, el concepto de Inteligencia Artificial (IA) ha convivido con el ser humano. A medida que ha transcurrido el tiempo y los sistemas de computación han avanzado, el empleo de estas IAs se ha ido diversificando en todo tipo de campos. En el caso de este proyecto, trato con los campos de investigación de mecánica cuántica y química.

El problema fundamental que tienen estos campos de investigación es que muchas de sus prácticas, conllevan procesos muy costosos a nivel de tiempo y recursos. En mi caso, me centraré en predecir la constante de acoplamiento [2] de esta manera se busca recortar el tiempo requerido para realizar su estimación. La constante de acoplamiento se produce por la interacción magnética entre los átomos de una misma molécula, dicha interacción se mide en Hz. Esta constante se utiliza para determinar distintas fuerzas referentes a las moléculas. A pesar de la existencia de un gran número de tipos de constantes [3], en este caso solo se trabajará con 8 de ellos.

El principal objetivo de este proyecto es utilizar diferentes metodologías con redes neuronales para predecir constantes de acoplamiento. En este trabajo, se emplean metodologías tales como la *Fully Connected Network* (FC), *Graph Convolutional Neural Network* (GCN) y la *Attention Graph Neural Network* (GAT). La elección de la implementación de las redes grafo no es fortuita, debido a que una manera común de representar las moléculas es mediante grafos donde los nodos son átomos y sus enlaces se corresponden a aristas (enlaces de los grafos).

Para la realización de este trabajo ha sido utilizado el lenguaje de programación Python, juntamente con el *framework* de *Deep Learning* Pytorch [4] y la librería Pytorch Geometric [5], que nos da acceso a las funcionalidades necesarias para tratar con grafos.

En cuanto a la estructura del trabajo he seguido los apartados pautados en el diagrama de Gantt adjunto en el anexo 1.

En referencia a los objetivos personales, buscaba un proyecto en el cual me permitiera trabajar y aprender las técnicas básicas de *Deep Learning*. Por otro lado, buscaba entender las nuevas técnicas de *Deep Leaning* que usan grafos y permiten trabajar con estructura de datos no reticulares.

Este proyecto, tiene diversidad de objetivos a alcanzar, el primer objetivo ha sido entender los datos químicos que ofrecía el *dataset* con la finalidad de obtener el suficiente conocimiento como para desarrollar el proyecto. En segundo lugar, se ha buscado generar un *baseline* con técnicas de vanguardia no grafo para validar la viabilidad que tiene el uso de estas redes en datos no reticulares. En tercer lugar, se ha buscado diseñar redes grafos que fueran capaces de predecir la constante de acoplamiento a partir de la estructura 3D de las moléculas. En cuarto lugar, se quiere realizar una comparación de eficacia entre las redes grafos y las redes no grafos. Por último, se busca comparar la metodología grafo empleada en este proyecto con las metodologías existentes en el estado del arte.

2. Estado del arte

2.1 *Fundamentos teóricos*

2.1.1 *Fully Connected Layer*

La *Fully Connected Layer* [6] es una capa donde todos los nodos de entrada están conectados a todos los nodos que forman parte de la salida. Al estar completamente conectado se convierte en una herramienta muy costosa en términos de memoria debido a la gran cantidad de pesos que se emplean y a nivel computacional debido al número de conexiones.

La salida de una *Fully Connected Layer* se puede definir tal que:

$$y = xA^T + b$$

Donde A^T representa los pesos de la capa, b representa el *bias* y x la entrada de datos que tiene la capa.

2.1.2 *Graph Neural Network* (GNN)

Las *Graph Neural Networks* (GNN) son capaces de trabajar con datos que están en una estructura no reticular. Un ejemplo de estructura no reticular es el caso de las moléculas, las cuales tienen relación entre átomos gracias a los enlaces existentes. La función principal de dichas redes es describir datos relacionales. Dichas redes se basan en la teoría de grafos [7], que consiste en una rama de las matemáticas que define y estudia las propiedades de estos.

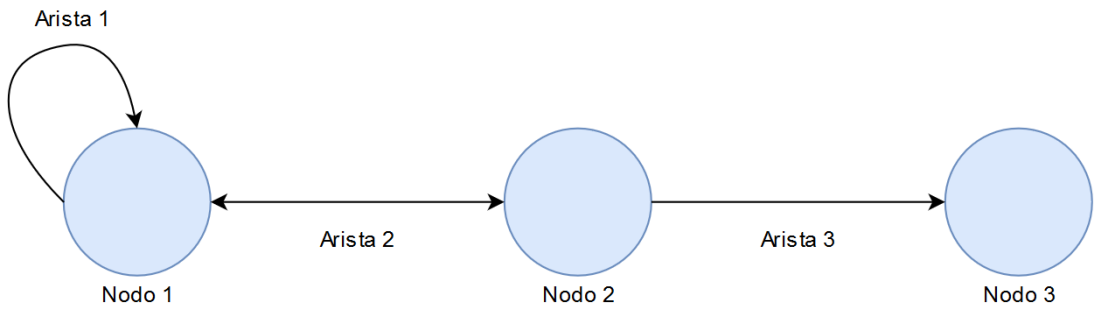

Ilustración 1 Grafo

Un grafo (G) es una estructura de datos que está compuesta por nodos (N) y aristas (A).

$$G = (N, A)$$

Tal y como vemos en la ilustración anterior, las aristas pueden tener o no dirección, dependiendo de si existen dependencias direccionales entre los nodos.

La primera GNN fue propuesta por Scarselly [8] y es un tipo de red que puede operar directamente en una estructura de datos como el grafo.

En los últimos años, han ido apareciendo diversas variantes de GNN [9], que tienen como propósito intentar suplir alguna carencia del método original. Un ejemplo de variante de GNN es el caso de la *Graph Convolutional Neural Network* (GCN) y la *Attention Graph Neural Network* (GAT). Estas redes han demostrado un alto rendimiento en diversas áreas, tales como, el procesado del lenguaje natural [10], [11], imágenes tridimensionales y nubes de puntos [12], predicción de propiedades químicas de moléculas [13], entre otros. No sorprende el uso de redes grafo en el campo de la química ya que las moléculas son naturalmente representables con grafos.

2.1.3 *Graph Convolutional Neural Network (GCN)*

GCN [14] es una variante eficiente de las redes neuronales convolucionales que operan directamente en grafos.

Se considera una GCN con la siguiente regla de propagación en sus capas:

$$x_i^{(k)} = \sum_{j \epsilon N(i)} \frac{1}{\sqrt{\deg(i)} \cdot \sqrt{\deg(j)}} \cdot (\Theta \cdot x_j^{(k-1)})$$

Donde, k se entiende como la capa inicial y $K-1$ como la anterior. El nodo del grafo viene representado por i, el nodo del vecindario se representa con j y N(i) es el vecindario de i. Θ representa los pesos y $x_j^{(k-1)}$ las características del vecindario. En la ecuación se ha obviado el termino bias para simplificar la formulación.

$$\frac{1}{\sqrt{\deg(i)} \cdot \sqrt{\deg(j)}}$$

Representa la normalización de las características. Para emplear dicha normalización, se emplea $\deg()$ que representa la operación *degree* de grafos. En el caso de la normalización, calcula el *degree* del nodo central y el del vecino. Al calcular el *degree* de un nodo, se obtiene el número de aristas incidente que tiene.

Para poder aplicar convoluciones, se debe de generalizar la convolución. Generalizar una convolución en este caso implica que, al no tener una estructura reticular, se tenga que crear de forma artificial con las aristas de los grafos.

Para realizar una convolución, se requiere de una retícula como en la que se representa en la siguiente imagen.

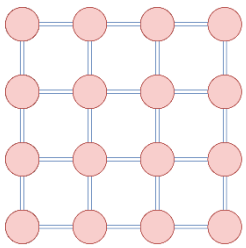

Ilustración 2 Estructura reticular

En la imagen anterior se puede observar una representación gráfica de una estructura reticular. En el caso de una imagen en cada uno de los nodos rojos representaría la información que contiene un píxel. Sin embargo, no todos los problemas poseen estructuras reticulares fijas.

Gracias a los grafos se pueden generar retículas artificiales como la que se observa en la imagen a continuación.

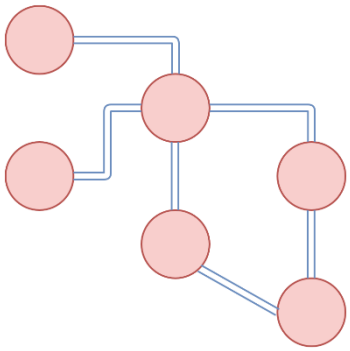

Ilustración 3 Estructura reticular artificial

Existen diversas formulaciones para realizar la convolución sobre un grafo, tal y como se indica en este estudio [15].

La formulación de la convolución discreta puede darse tal que:

$$x_i^k = \sum_{j \epsilon N(i)} \Theta_{ij} \cdot x^{(k-1)}(j)$$

Donde, k se entiende como la capa inicial y $K-1$ como la anterior. El nodo del grafo viene representado por i, el nodo del vecindario se representa con j y N(i) es el vecindario de i. Θ_{ij} representa los pesos y $x^{(k-1)}$ las características del vecindario. En la ecuación se ha obviado el termino bias para simplificar la formulación.

Las principales diferencias entre la convolución discreta y la GCN es que la GCN posee siempre un tamaño de *kernel* de 1x1 independientemente del tamaño del vecindario. Sin embargo, la convolución discreta emplea un tamaño KxK donde K es el tamaño de la ventana del vecindario (ventana de convolución). Los autores de la GCN [14] determinaron que el tamaño del *kernel* fuese 1x1 debido a que no se puede asegurar si los puntos en la retícula artificial están ordenado de forma cuadricular. De esta manera, se aplica el filtro con el *kernel* de tamaño 1x1 para cada uno de los nodos del vecindario.

Otra de las principales diferencias es la normalización que poseen la GCN y la convolución discreta. La normalización tal y como mencionan los autores de la GCN [14] sirve para estabilizar el entrenamiento.

2.1.4 *Graph Attention Neural Network (GAT)*

GAT [11], es una variante de la GNN posterior a la creación de la GCN. Aprovecha las capas de atención automática, para lograr abordar las deficiencias de métodos anteriores como el GCN.

Cuando se habla de atención, en este contexto, se refiere a la capacidad de concentrarse en las partes de datos relevantes para ayudar a la mejora del rendimiento del modelo. De esta manera, se estiman diferentes pesos para diferentes nodos en un vecindario, sin requerir ningún tipo de operación de matriz costosa.

Para poder definir una manera clara la arquitectura de la GAT se debe definir previamente la estructura de la entrada. La entrada se compone por un conjunto de característica de nodos,

$$h = \{\overrightarrow{h1}, \overrightarrow{h2}, \overrightarrow{h3} \dots, \overrightarrow{hN}\}, \overrightarrow{hi}\ \boldsymbol{\epsilon} |\ \mathrm{R^F}$$

donde se entiende que N es el número de nodos y F es el número de características que posee cada nodo.

Para que el modelo pueda aprender, necesita al menos una transformación lineal, parametrizada con unos pesos tal que $W\ \boldsymbol{\epsilon} |\boldsymbol{R^{F'xF}}$. Dicha capa se aplica a cada uno de los nodos. La salida se puede representar tal que:

$$h' = \{\overrightarrow{h'1}, \overrightarrow{h'2}, \overrightarrow{h'3} \dots, \overrightarrow{h'N}\}, \overrightarrow{h'i}\ \boldsymbol{\epsilon} |\ \mathrm{R^{F'}}.$$

Tras aplicar la transformación lineal, se le aplica un *self-attention* en cada uno de los nodos, que se puede definir tal que:

$$a : R^{F'} \times R^{F'} \rightarrow R$$

De esta manera se calcula la primera fase de los coeficientes de atención.

$$e_{ij} = a(W\vec{h}_i, W\vec{h}_j)$$

Dichos coeficientes indican la importancia del nodo $(j's)$ y las características del nodo $(i's)$. En términos generales, se le permite al modelo que cada nodo atienda a todos los demás, eliminando toda la formación estructural.

El tipo de GAT que se utiliza en este proyecto, aplica una función de normalización *softmax* con el objetivo de que los coeficientes sean más comparables entre los nodos. Los coeficientes quedarían representados de la siguiente manera:

$$\alpha_{ij} = softmaxj(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k\epsilon Ni} \exp(e_i k)}$$

Finalmente, se calculan las características aprendidas de los nodos. Se puede expresar de la siguiente manera:

$$\overrightarrow{h_i'} = \sigma(\sum_{j \epsilon N} \alpha_{ij} W \overrightarrow{h_j'})$$

Donde σ representa una transformación no lineal.

Para mejorar la estabilidad del proceso de aprendizaje, una de las opciones que ofrece la GAT es la opción de establecer distintos valores de *head attention*. El valor de *head attention* establece el número de cálculos simultáneos y diferentes que se realizan de las características de entrada. El cálculo de las características con diferentes *head attention* se puede representar tal que:

$$\overrightarrow{h_i}' = \sigma(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \epsilon N} \alpha_{ij}^k W^k \overrightarrow{h_j}')$$

Donde K representa el número de *head attention* que han sido aplicados. En la siguiente imagen podemos visualizar gráficamente como se comporta la expresión.

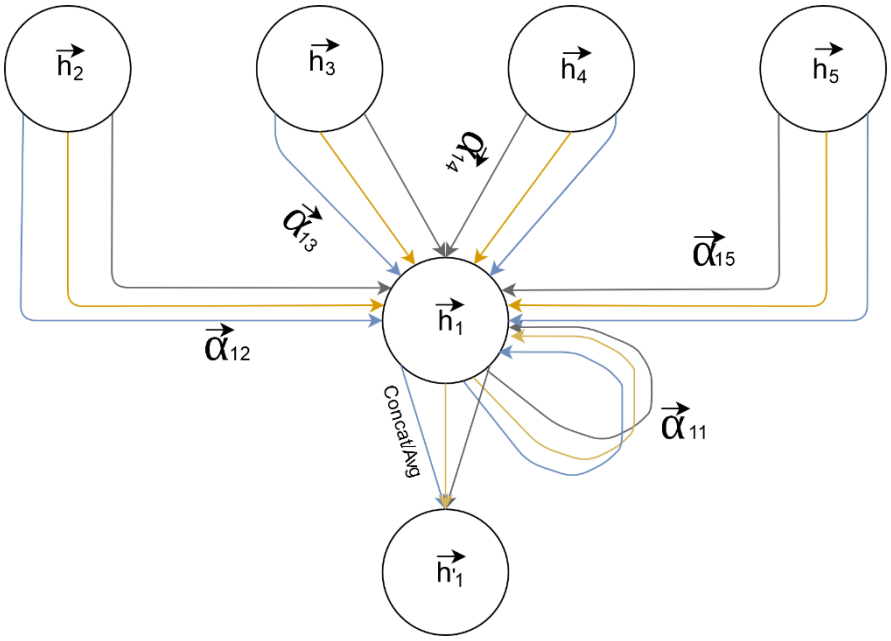

Ilustración 4 Predicción multi-head

En la imagen anterior podemos visualizar como trabaja una GAT con diversas *heads.* Cada una de las flechas de colores distintos que van hacia el nodo central representan una computación independiente de los valores de los coeficientes de atención. Por último, la salida de la capa se puede tratar de diferentes maneras. En la imagen se pueden observar la concatenación de las predicciones y la media.

2.2 Estado del arte

Como he comentado con anterioridad, la tecnología basada en IAs está en constante evolución. Es un hecho que cada vez se amplía la diversidad de metodologías en el campo de las redes neuronales. No sorprende que su radio de acción sea cada vez más grande.

La implementación de las redes grafo no es fortuita, la estructura principal de las moléculas y las redes grafo están compuestas por los mismos elementos; nodos (átomos) y aristas (enlaces). Actualmente se pueden divisar trabajos de investigación muy diversos como la generación de operadores de productos de matriz para representar el Hamiltoniano de una molécula [11], o el uso de redes grafo para el descubrimiento de drogas o diseños de moléculas [12].

A pesar de que la eficiencia de las redes grafo en el entorno molecular ha sido demostrada en diversas ocasiones, existen metodologías muy variadas para afrontar con mayor eficiencia el cálculo de la constante de acoplamiento.

A continuación, se explicarán algunas de las metodologías que complementan a las redes grafo para llevar a cabo la estimación de las constantes de acoplamiento.

2.2.1 Uso de *Bidirectional Encoder Representations from Transformers* [BERT]

El grupo de ingeniería química del Instituto de Energía Atómica de Corea del Sur (KAERI) [18] realizó predicciones de la constante de acoplamiento con la implementación de BERT [19]. A pesar de ser un modelo especializado en la representación del lenguaje, está diseñado para entrenar con anterioridad las representaciones bidireccionales profundas. Su implementación a la hora del cálculo de la constante de acoplamiento se puede tomar como un extractor de características previas a un modelo GNN. Tras la implementación de BERT, los resultados de la predicción que ofrecía la red GNN se veían mejorados.

2.2.2 Uso de Pointnet

Otro enfoque para abordar la estimación de la constante de acoplamiento, que ha sido implementado por el grupo llamado Quantum Uncertainty [20], consiste en implementar el modelo ya existente Pointnet [21]. Dicho modelo, se trata de un modelo ya creado que proporciona una arquitectura unificada para aplicaciones que van desde la clasificación de objetos, la segmentación de piezas, hasta el análisis semántico de escenas. Finalmente, se complementa con un mecanismo de atención [22] que añade codificaciones posicionales que son necesarias en este caso, ya que la entrada depende muy estrechamente de la posición.

2.2.3 Bosch Center for AI (BCAI)

La metodología que ha funcionado con mayor eficacia y ha obtenido los mejores resultados en lo que a la predicción de la constante de acoplamiento se refiere, ha sido la empleada por el grupo de BCAI [23], [24]. La premisa principal que han seguido ha sido el uso de transformadores grafo de redes neuronales, con una arquitectura de meta-grafo [25]. A diferencia de la mayoría de los métodos grafo que se emplean para moléculas, donde los átomos resultan ser los nodos de la red, optan por asignar a los nodos los valores de los enlaces o *J-Coupling* (químicos y no químicos), estructuras de 3 átomos (*Triplets*) y estructuras de 4 átomos (*Quadruplets*).

BCAI optó por aumentar la cantidad de datos añadiendo información de la molécula gracias a diversas librerías de Python como RDKit [26], [27] y OpenBabel [28]. Los datos que se añaden a la base de datos son:

- Enlaces químicos sin constante de acoplamiento.
- Cargas parciales de átomos.
- *Closed angle*.

Finalmente complementaron su estrategia añadiendo distintos grados de clasificación a los datos. Cuanto más grande es el grado, más información respecto a la molécula le añaden.

En el primer grado de clasificación tienen en cuenta los 8 tipos diferentes de *J-Coupling* y los diferentes tipos de enlaces químicos que aparecen dentro de la base de datos procesada.

El segundo grado de clasificación se divide en dos sectores. El primero es el formado por los enlaces químicos que pertenecen a la molécula, pero no generan constantes de acoplamiento. Dentro de este sector, se clasifican gracias a los átomos que forman dicho enlace químico y, además, teniendo en cuenta el orden del enlace.

Por otro lado, tenemos los *J-Coupling*. Recordemos, que anteriormente se mencionó que hay una totalidad de 8 tipos diferentes de constantes de acoplamiento dentro de la base de datos. Para realizar la subclasificación de estos tipos se tienen en cuenta las características del átomo que no es un hidrógeno en la interacción. Los datos que se tienen en cuenta del átomo son los siguientes.

- Tipo de átomo.
- Número total de enlaces (valencia), contando el enlace doble como 2
- Número total de vecinos unidos, contando el enlace doble como 1.
- Orden más grande de enlaces.
- Segundo orden más largo de enlaces.

El número total de subclasificaciones resultantes es de 68. Esta subclasificación es muy importante en el desarrollo de este proyecto ya que diversifica en grupos más pequeños los 8 tipos principales.

En el proyecto, se utilizan las subclasificaciones debido a que la desviación de los valores de cada subclasificación es mucho más pequeña respecto a la desviación de los 8 tipos principales.

De esta manera las redes neuronales predicen por cada enlace o *J-Coupling* 68 valores diferentes, uno por cada tipo de enlace o *J-Coupling* existente.

3. Metodología

3.1 *Dataset*

3.1.1 Datos originales

Para poder llevar a cabo todas las pruebas realizadas en este trabajo, se ha utilizado la base de datos *CHAMPS Scalar Coupling dataset* [1] generado por el grupo de Química y Matemáticas en el Espacio de Fase (CHAMPS) en la Universidad de Bristol, la Universidad de Cardiff, el Imperial College y la Universidad de Leeds.

La base de datos está formada por 135.000 moléculas. Cada una de estas moléculas tiene un número de interacciones magnéticas a predecir diferente. En total, se disponen de 7.164.265 de casos donde se producen constantes de acoplamiento a predecir (*J-Coupling*).

Por cada uno de los casos en los cuales se genera una constante de acoplamiento, la base de datos nos ofrece la siguiente información:

1. Posición de cada átomo en el espacio.
2. La molécula la cual estos pertenecen.
3. Tipo de cada átomo.
4. Tipo de la constante de acoplamiento.
5. Valor de la constante de acoplamiento.

Cabe clarificar, que cuando se habla de *J-Coupling* nos referimos a un par de átomos que generan una constante de acoplamiento. Debido a que no siempre este par de átomos son enlaces químicos de la molécula en este proyecto se nombrará a los pares de átomos que generen constantes de acoplamiento como *J-Coupling*.

En el apartado 3.1.2 se añaden enlaces químicos de la molécula para aportar información estructural de la molécula, dichos enlaces se nombrarán enlaces químicos.

3.1.1.1 Tipos diferentes de constantes de acoplamiento

A pesar de la existencia de una gran diversidad de tipos de constantes de acoplamiento, dentro de nuestra base de datos solamente poseemos 8 tipos diferentes.

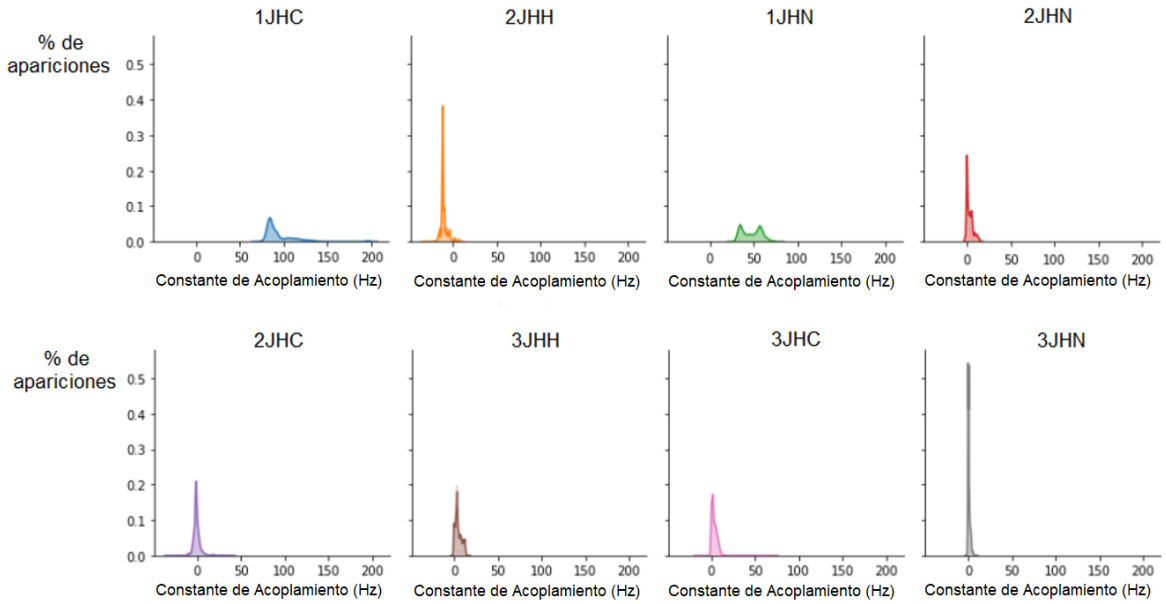

Ilustración 5 Valores de las constantes de acoplamiento por tipo

En la ilustración anterior, se puede observar el porcentaje de aparición que tienen los valores de la constante de acoplamiento en cada clase. Tal y como se observa en la imagen, cada uno de los tipos de constantes de acoplamiento, posee una media y una desviación estándar diferentes.

La nomenclatura usada para nombrar cada uno de los diferentes tipos de las constantes de acoplamiento muestra información relevante respecto a su composición.

Tipo
3JHC
3JHH
3JHN
2JHC
2JHH
2JHN
1JHC
1JHN

Tabla 1 Tipos de constante de acoplamiento en la base de datos

El primer número que aparece en la clasificación determina el número de enlaces químicos entre los átomos involucrados en esta interacción.

La letra J determina que estamos tratando con un *J-Coupling* [29].

Finalmente, las dos últimas letras que componen el nombre del tipo determinan los tipos de átomos involucrados en esta interacción.

Cabe mencionar, que los tipos de constantes de acoplamientos que se emplean en este proyecto siempre poseen un átomo de hidrógeno. La diversidad de átomos que forma las constantes de acoplamiento junto al hidrógeno se muestra en la tabla a continuación.

Letra	Nombre
H	Hidrógeno
N	Nitrógeno
C	Carbono

Tabla 2 Tipos de átomos

3.1.2 Datos procesados

Para la implementación de este proyecto me he basado en la técnica de procesado de datos del *dataset* del grupo BCAI [23]. Dicha técnica está explicada con detalle en el apartado 2.2.3.

En este proyecto, se utilizan diversos tipos de estructura de datos de entrada. A continuación, se muestra la unidad mínima que representa un enlace o *J-Coupling*.

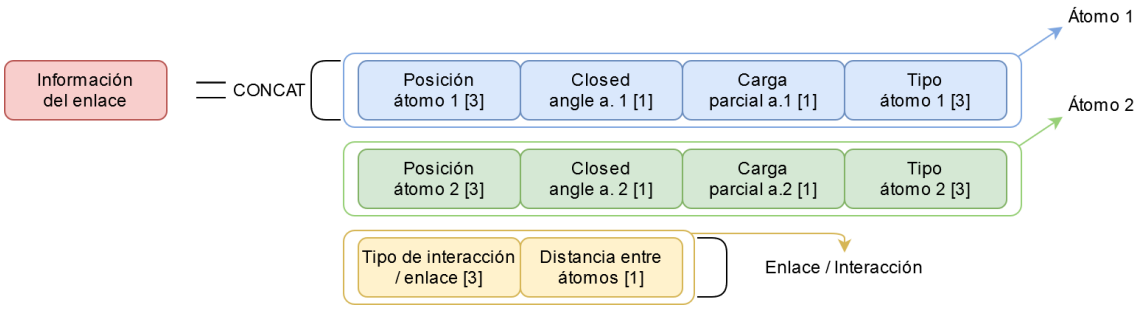

Ilustración 6 Entrada de los modelos

Entendiendo que en la imagen anterior los números entre [ ] indican el tamaño de cada subelemento, la dimensión total es de 20.

Como más adelante se verá en el apartado 3.3, se usa una red FC para estimar la constante de acoplamiento directamente a partir de un único *J-Coupling.* Sin embargo, posteriormente, se realiza la implementación de un modelo FC que requiere la información de todos los enlaces o *J-Coupling* de la molécula para realizar la estimación. En este caso, se necesita como unidad mínima una matriz de 406 filas por 20 columnas, donde cada una de las filas posee el contenido de un enlace o *J-Coupling*. El número 406 es el número de enlaces y *J-Coupling* máximos que se encuentra en una molécula dentro de la base de datos. Establecer un tamaño estático de las filas de la matriz

conlleva que si las moléculas no alcanzan el número de enlaces y *J-Coupling* máximo, se queden filas con valor 0.

En cuanto hablamos de la entrada de datos de los modelos grafo, los procedimientos para la creación de la estructura mínima de entrada, se ven alterados. Recordemos que un grafo está compuesto de dos elementos, los nodos y las aristas.

A nivel conceptual, si la unidad de entrada es un grafo, se entiende que cada grafo tiene información única y exclusivamente de una molécula. Cada uno de los nodos representa un enlace o un *J-Coupling*.

Finalmente, queda por explicar la obtención de las aristas de cada uno de los grafos. Recordemos, que a diferencia de los nodos que poseen la información de los enlaces o *J-Coupling* de la molécula, las aristas del grafo indican las relaciones de información entre los nodos de cada grafo. Esto quiere decir que determinan la influencia que tiene cada uno de los nodos sobre el resto.

En este proyecto se han empleado dos tipos de técnicas en lo que a las aristas de los grafos se refiere, todos con todos o relación nodos bajo un criterio aplicado.

La técnica todos con todos, se basa en relacionar todos los nodos del grafo entre sí. La principal desventaja de emplear esta técnica es que puede llegar a requerir un coste computacional muy elevado, ya que en elementos grandes el número de relaciones se dispara desmesuradamente. En el caso de este proyecto, se podía observar un máximo de 82.621 aristas.

La segunda opción, es utilizar un criterio para determinar que nodos están relacionados. En el caso de este proyecto, se aplica un criterio que solo relaciona nodos que compartan átomos dentro de su composición. La principal ventaja es la obtención de grafos, con una cantidad de aristas mucho menor que en la primera técnica.

Veamos un ejemplo de cómo son los datos resultantes de una molécula de agua y, posteriormente, un ejemplo de creación de su grafo.

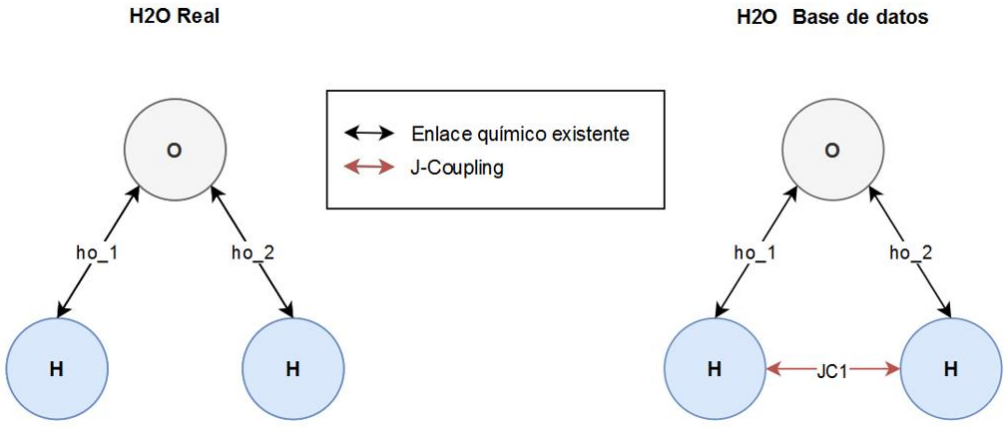

Ilustración 7 Molécula de agua Real VS Base de datos

En la imagen anterior se muestra una molécula de agua que está compuesta por dos átomos de hidrógeno y uno de oxígeno. Inicialmente la molécula solo tiene 2 enlaces químicos, que son los que se generan entre los átomos de hidrógeno y el átomo de oxígeno. Sin embargo, en cuanto tenemos en cuenta las interacciones magnéticas entre los átomos aparece un nuevo componente, el *J-Coupling*. Este componente, de ahora en adelante será adherido a la composición de la molécula del agua. De esta manera en la base de datos, la molécula H_2O tiene una cantidad de enlaces y *J-Coupling* de 3.

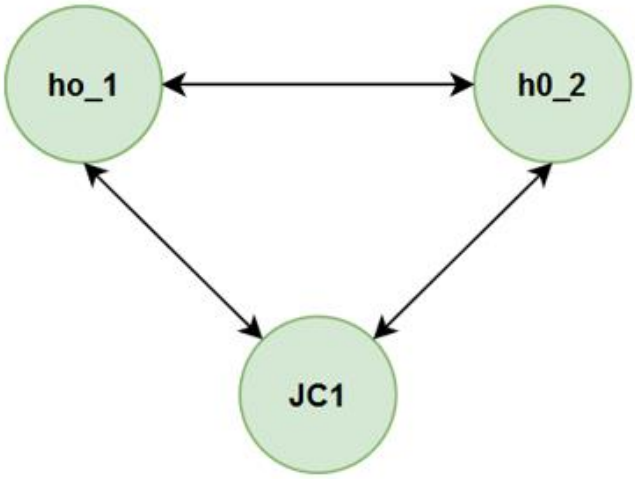

Ilustración 8 Grafo molécula H_2O

La imagen anterior representa la estructura grafo de entrada que llega a los modelos. Los nodos están compuestos por enlaces químicos o *J-Coupling* y las aristas del grafo indican la relación que tienen dichos nodos.

3.3 Arquitectura

Durante el desarrollo de todo este proyecto se han diseñado 4 arquitecturas principales de redes neuronales diferentes, con tal de poder comprobar su eficiencia y contrastar todos los experimentos realizados con cada uno de ellos.

Cabe mencionar que todos los modelos son regresivos. Esto quiere decir que su función es la de la estimación de un valor determinado, en este caso, la constante de acoplamiento.

3.3.1 Arquitecturas FC

3.3.1.1 Fully Connected Network

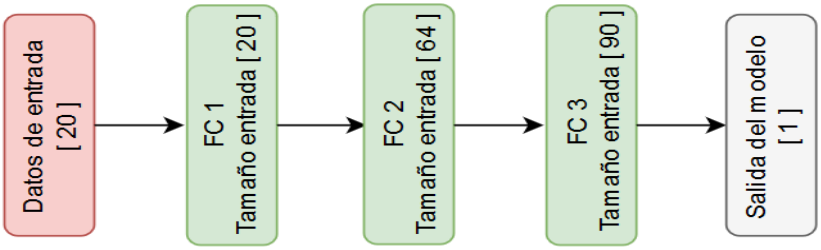

Ilustración 9 Fully Connected Network

El primer modelo implementado en este proyecto es una *Fully Connected Network*. Se compone de los elementos más básicos de las redes neuronales, las capas FC. Tras la salida de cada una de las capas FC, exceptuando la última, se aplica una de función de activación llamada ReLu [30].

Esta red neuronal, está diseñada para poder estimar los valores de las constantes de acoplamiento mediante únicamente información individual de cada *J-Coupling*.

Una de las variantes de este modelo es el cambio de tamaño de la salida, de 1 a 68. Este cambio se basó en la clasificación de subtipos que se explica en el apartado 2.2.3.

3.3.1.2 Modelo Compuesto Fully Connected

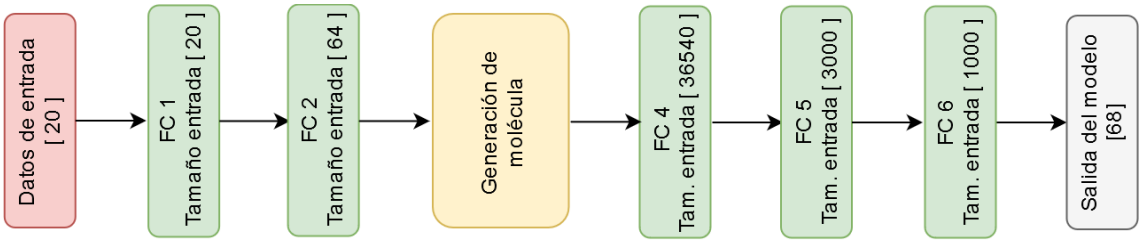

Ilustración 10 Modelo Compuesto Fully Connected

El segundo modelo creado es el nombrado como Compuesto *Fully Connected*, su composición parte de una herencia del primer modelo. Además, se le adhiere una segunda parte compuesta por una capa generadora de moléculas y capas FC. Tras la salida de cada una de las capas FC, exceptuando la última, se aplica una de función de activación llamada ReLu. Finalmente se puede observar que la capa de salida cambia de tamaño de 1 a 68. Para comprender este modelo debe ser seccionado en dos partes.

La primera parte de este modelo se corresponde con la herencia del modelo *Fully Connected Network*, genera *embeddings* de forma individual para cada uno de los enlaces o *J-Coupling* que forman la molécula.

La segunda parte del modelo está orientada a poder trabajar en el ámbito de la molécula. El generador de la molécula genera una matriz que contiene los *embeddings* de los enlaces o *J-Coupling* que conforman la molécula, de esta manera permite al modelo trabajar a nivel molecular. El tamaño de la matriz es fijo, ya que se establece según la molécula que tiene mayor número de enlaces dentro de la base de datos, tal y como se

explica en el apartado 3.1.2. El hecho de que este modelo trabaje con una matriz de tamaño fijo representa una limitación del modelo, ya que puede contener filas con valores 0. Además, este modelo depende del orden de los enlaces y *J-Coupling* de dentro de la molécula debido a que en esta red el orden posee importancia.

Por último, el tamaño de la salida se debe a la subclasificación de constantes de acoplamiento. Al aumentar el tamaño de la salida y crear agrupaciones pequeñas se reduce la desviación de los valores que puede tomar cada elemento de la salida. Los elementos que componen la subclasificación son explicados en el apartado 2.2.3.

3.3.2 Arquitecturas grafo

3.3.2.1 Modelo GCNConv

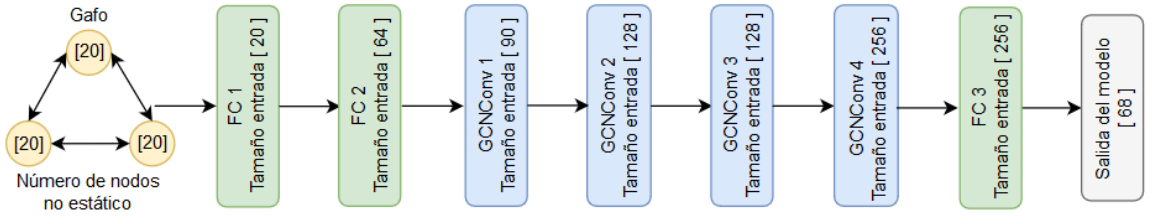

Ilustración 11 Modelo GCNConv

El tercer modelo creado es el GCNConv. Su nombre proviene del uso de las capas GCN dentro de este. Dichas capas son explicadas previamente en el apartado 2.1.3. La composición del modelo GCNConv, al igual que se da en el modelo Compuesto *Fully Connected*, parte de la composición del modelo *Fully Connected Network*. Esta primera parte del modelo trabaja de manera independiente para cada uno de los nodos. Posteriormente, se le adhieren capas GCNConv. Tras la salida de cada una de las capas GCNConv, exceptuando la última, se aplica una de función de activación llamada ReLu.

El primer aspecto que se puede observar en la imagen anterior que difiere de los modelos previos es la entrada. La entrada se compone de grafos en los cuales sus nodos representan los enlaces o *J-Coupling* que generan las constantes de acoplamiento. Este modelo tiene dos configuraciones de relaciones de aristas del grafo de entrada. La primera se basa en relacionar todos los nodos con todos y la segunda se basa en relacionar los nodos que tengan al menos 1 átomo en común. Ambas configuraciones son explicadas con mayor detalle en el apartado 3.1.2.

Las capas GCNConv son la primera implementación de redes grafo en este proyecto. Estas capas permiten al modelo trabajar a nivel molecular a la hora de realizar las predicciones de las constantes.

Las ventajas principales del uso de las redes grafo es que se adaptan a moléculas de diferente tamaño y además son independientes del orden de los enlaces o *J-Coupling* dentro de la molécula. Dichas ventajas permiten evitar las carencias que se presentaban en el modelo Compuesto *Fully Connected*. Además, permite estimar todas las constantes de acoplamiento de la molécula de manera simultánea.

3.3.2.2 Modelo GATConv

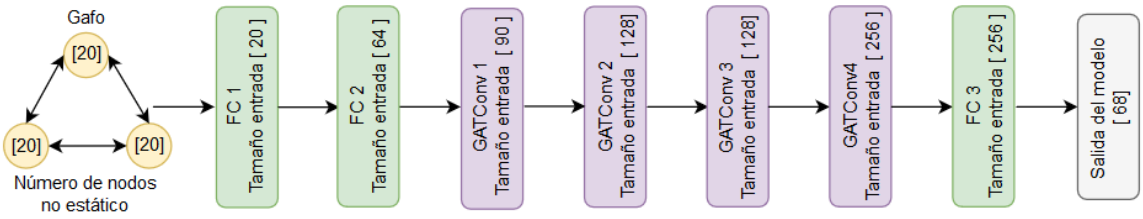

Ilustración 12 Modelo GATConv

El cuarto y último modelo creado es el GATConv. Su composición es una variante del GCNConv, ya que se reemplazan las capas GCNConv por GATConv. Las capas GATConv son explicadas previamente en el apartado 2.1.4.

La motivación inicial para el empleo de esta metodología se debe a que gracias a las capas atencionales se obtiene la capacidad de concentrarse en las partes de datos relevantes para ayudar a la mejora del rendimiento del modelo. De esta manera, se estiman diferentes pesos para diferentes nodos en un vecindario.

La GATConv tiene una configuración aplicada final de 4 *heads attention*, esto quiere decir que realiza los cálculos de predicción de 4 maneras diferentes simultáneamente. Tal y como gestiona la salida, esta capa concatena las predicciones simultáneas en un mismo vector. Dicha concatenación hace que la salida de las capas se vea multiplicado por el número de *head attention* que tiene configurado cada capa. Por ese motivo, para poder mantener el mismo tamaño de entrada para cada una de las capas, todas las salidas de las GATConv han sido divididas por un factor de 4.

4. Experimentos realizados

4.1 *Training*

En este apartado se explica todos los componentes que se han empleado en todos los entrenamientos.

4.1.1 Métrica de evaluación

Los resultados se evalúan con una extensión logarítmica del error absoluto medio (MAE). Esta métrica tiene en cuenta el tipo de constante de acoplamiento debido a que hace un promedio de todas las clases. Esta métrica fue propuesta en el *challenge* de *Predicting Molecular Properties* [1].

La fórmula empleada se puede expresar tal y como se muestra a continuación.

$$Score = \frac{1}{T} * \sum_{t=1}^{T} \log(\frac{1}{nt} * \sum_{i=1}^{nt} |yi - \hat{yi}|)$$

Donde:

- T es el número de tipos de constantes de acoplamiento.
- nt es el número de predicciones realizadas.
- yi es el valor de la constante de acoplamiento.
- $\hat{yi}$ es el valor de la constante de acoplamiento predicha.

Cabe resaltar que, debido a la existencia de un logaritmo, el sistema de evaluación retorna números pequeños cuanto mayor sea la precisión, incluyendo números negativos.

4.1.2 Optimizador

El optimizador escogido para ser empleado durante todo el proyecto es Adam [31], que es un algoritmo de optimización de velocidad de aprendizaje adaptativo diseñado específicamente para entrenar redes neuronales profundas.

Los parámetros de configuración de Adam han permanecido en su estado por defecto y sus valores son los siguientes:

- *Learning Rate*: 1e-3.
- *Betas*: [0.9, 0.999].
- *Weight decay*: 0.

4.1.3 Loss

Aun teniendo una métrica de evaluación, no es posible su uso para el entrenamiento de un modelo debido a que tiene valores negativos en su salida. En su lugar, todos los modelos han sido entrenados con la MAE, ya que posee un comportamiento similar a la

métrica de evaluación. Sin embargo, difiere en aspectos tales como que no realiza agrupaciones, no posee un logaritmo y su mínimo es 0.

4.1.4 Partición del *Dataset*

Con anterioridad, se mencionó que el *Dataset* contiene 135.000 moléculas. La partición de este *Dataset* para poder realizar el entrenamiento, la validación y el test es la siguiente.

Para realizar el test y adquirir la evaluación de los resultados se destinaron 45.000 moléculas. En el *dataset* original se establecía el número y el orden que se destinaba a la sección de test. El resto del *dataset* se destinó al entrenamiento. Se generó una partición de manera aleatoria de 80%-20% para definir la partición de validación (80%) con la cual evaluar los *hyperparameters* de la red.

4.1.5 Implementación

Toda la implementación de este proyecto se ha llevado a cabo única y exclusivamente con el lenguaje de programación Python. Además, he empleado el *framework* Pytorch [4] y la librería PyTorch Geometric [5].

La implementación de este proyecto se puede encontrar en: https://github.com/imatge-upc/coupling_constant

4.2 Experimentos y resultados

En este apartado se explicará con más detalle cada una de las pruebas que se han realizado en este proyecto y sus resultados. Desde un inicio se ha dividido los experimentos en modelos no grafo y modelos grafo.

Los *scores* reportados para la partición de test se obtuvieron al aplicar el modelo que reporto una *loss* más baja en la partición de validación.

4.2.1 Experimentos con redes *Fully Connected*

4.2.1.1 Baseline con Fully Connected Network

Como primer experimento, se ha empleado el modelo *Fully Connected Network* con tamaño de salida 1. En este punto se busca validar que el elemento más básico del aprendizaje profundo, las capas FC, sean capaces de minimizar la *loss* y aprender de la entrada de datos seleccionada.

El entrenamiento se llevó a cabo con un *batch size* de 64, durante 80 épocas. El *score* reportado para la partición de test con la métrica es de 0,42612.

4.2.1.2 Cambio de tamaño de salida de Fully Connected Network

Tras comprobar que las capas FC son capaces de disminuir la *loss*, se ha procedido a modificar el modelo *Fully Connected Network*, cambiando la última capa de tamaño 1 a tamaño 68.

En este experimento se quiere evaluar si al ampliar el tamaño de la salida y reservando cada posición a un subtipo de constante de acoplamiento determinado, se facilita el aprendizaje del modelo. Realizando este tipo de asignación, debería producirse que las desviaciones de los valores en cada una de las posiciones de la salida fueran más pequeñas.

El entrenamiento se realizó en las mismas condiciones que en el experimento anterior. El *score* reportado para la partición de test con la métrica es de 0,25597.

4.2.1.3 Evaluación del modelo Compuesto Fully Connected

En este experimento se pone a prueba el modelo Compuesto *Fully Connected*. Este modelo, a diferencia del anterior, puede trabajar con toda la información de la molécula para realizar la estimación de la constante de acoplamiento.

Se quiere demostrar si las predicciones son más precisas con toda la información de la molécula. Si hay mejoría en los resultados, significa que la constante de acoplamiento se ve influenciada por el conjunto total de la molécula, lo que conlleva descartar el poder tratar a la constante de acoplamiento como un evento aislado en una molécula.

El entrenamiento se ha llevado a cabo con un *batch size* de tamaño 256 y un total de 150 épocas. El *score* reportado para la partición de test con la métrica es de 0,21587.

4.2.1.4 Normalización del Ground Truth (GT)

En este experimento, se ha realizado la normalización de los valores de la constante de acoplamiento.

Se busca evaluar si dicha normalización de los valores puede facilitar al modelo la predicción de la constante.

El entrenamiento se realizó con las mismas condiciones que en el experimento anterior. El *score* reportado para la partición de test con la métrica es de 0,29929. Normalizar la constante de acoplamiento no supone una mejora.

4.2.1.5 Substitución de 0 por 1

El modelo Compuesto *Fully Connected*, como ya mencioné anteriormente en el apartado 3.3.1.2, posee una primera parte compuesta por el modelo *Fully Connected Network*. Esta primera parte puede tener entradas que solo contengan 0. Esto se debe a que todas las moléculas tienen una cantidad diferente de enlaces y *J-Coupling* e, inicialmente, se estipula una matriz de un tamaño fijo igual a la molécula con mayor número de enlaces y *J-Coupling* dentro de la base de datos.

Este experimento quiere comprobar si al reemplazar los 0 por 1, se le facilita el entrenamiento a la red.

El entrenamiento se realizó con las mismas condiciones que en el experimento anterior. El *score* reportado para la partición de test con la métrica es de 0,37022. Es un rendimiento muy inferior a los experimentos anteriores.

4.2.1.6 Resultados de experimentos FC

Nombre de experimento	*Score*
Baseline con *Fully connected Network*	0,42612
Cambio de tamaño de salida de *Fully Connected Network*	0,25597
Evaluación del modelo Compuesto *Fully Connected*	0,21587
Normalización del Ground Truth	0,29929
Substitución de 0 por 1	0,37022

Tabla 3 Puntuación de experimentos FC

Tal y como se muestra en la tabla anterior, el experimento que mejor resultado ha obtenido es el "*Evaluación del modelo Compuesto Fully Connected*". Debido a ello, es la estructura que se emplea como referencia en el inicio de los experimentos grafo.

Al finalizar los experimentos del apartado de las redes *Fully Connected*, se entrenó durante 240 épocas el modelo Compuesto *Fully Connected*. En la ilustración a continuación se puede observar como la *loss* de entrenamiento comienza a estabilizarse en un nivel. Esto indica que el modelo no puede aprender a ser más preciso.

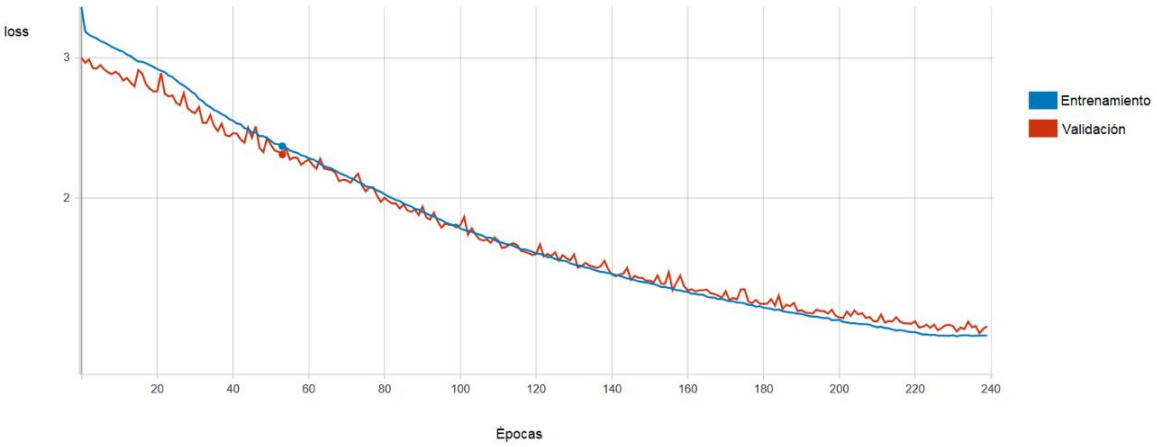

Ilustración 13 Gráfico loss modelo Compuesto Fully Connected

4.2.2 Experimentos con redes grafo

4.2.2.1 Modelo GCNConv todos los nodos relacionados

En este experimento se ha empleado el modelo GCNConv. Cuando se trabaja con estructuras grafo, se debe indicar que información porta cada nodo y que nodos están relacionados. En esta ocasión, la información que posee cada nodo es la salida de la primera parte del modelo, es decir, el resultado del extractor de características a nivel de enlace. Respecto a la relación entre nodos, en este caso es una relación de todos con todos.

En este experimento se busca comprobar si la red GCNConv es capaz de realizar predicciones cuando tiene una relación entre nodos total.

El entrenamiento se llevó a cabo con un *batch size* de tamaño 32 durante 240 épocas. El *score* reportado para la partición de test con la métrica es de 0,74661. Es el peor resultado obtenido hasta entonces.

4.2.2.2 Reducción del número de relaciones entre nodos

Tras observar que la red neuronal GCNConv no era capaz de disminuir el valor de *loss* y a su vez obtener un mal *score*, se procedió a cambiar la relación entre nodos.

En este experimento se establece una nueva regla de relación entre nodos la cual solamente relaciona enlaces o *J-Coupling* que contengan al menos un átomo en común. De esta manera, se busca evaluar si al menguar la cantidad de relaciones entre nodos la red es capaz de disminuir la *loss* y mejorar las estimaciones.

El entrenamiento se llevó a cabo con un *batch size* de 128 durante 240 épocas. El *score* reportado para la partición de test con la métrica es de -0,03879.

Dada la mejoría, a partir de este experimento, todos los realizados a posteriori mantendrán una relación entre nodos tal y como se ha implementado en este experimento.

4.2.2.3 Normalización de las capas de la red GCNConv

En este experimento se optó por aplicar capas de normalización, en concreto la *Batch Normalization* [32].

Se buscaba evaluar si la *Batch Normalization* ayuda a converger a la red y facilita el aprendizaje de esta.

El entrenamiento se llevó a cabo con un *batch size* de 128, durante 240 épocas. El *score* reportado para la partición de test con la métrica es de -0,38936.

4.2.2.4 Comprobación de compatibilidad de normalizaciones

Debido a que, en el experimento anterior, donde se aplicaba la normalización a través de la *Batch Normalization*, se obtuvo una mejoría de los resultados, se optó por normalizar los valores de las constantes de acoplamiento durante el entrenamiento.

Se busca comprobar si ambas normalizaciones son compatibles para que el modelo tenga mayor facilidad para aprender.

El entrenamiento se llevó a cabo con las mismas condiciones que el experimento anterior. El *score* reportado para la partición de test con la métrica es de -0,46089.

Debido a la compatibilidad entre normalizaciones y la mejoría que conlleva su empleo, serán utilizadas en el resto de los experimentos.

4.2.2.5 Introducción de la GATConv

Visto el rendimiento de las redes grafo se optó por la creación del último modelo, el GATConv. Este modelo, tal y como indica su nombre, posee la capa GATConv. Dicha capa tiene una configuración aplicada de 1 *head attention*.

El objetivo principal de este experimento es contrastar la eficiencia entre capas GATConv y GCNConv.

El entrenamiento se llevó a cabo con un *batch size* de tamaño 16 durante 120 épocas. El *score* reportado para la partición de test con la métrica es de -0,51561.

Como resultado, podemos ver mejoría cuando empleamos las capas GATConv en lugar de las GCNConv. Queda demostrada que las capas atencionales mejoran el rendimiento de los modelos grafos respecto a los modelos GCN.

4.2.2.6 Cambio número de head attention

Para finalizar, se modifica la configuración de las capas GATConv del modelo anterior. En este caso se aumenta el número de *head attention* a 4. El incrementar el valor de este número a 4 supone que las capas calculan de manera diferente y simultánea la estimación de las características.

Se busca demostrar que la implementación de una capa atencional con una configuración de *head attention* 4 obtiene mejores resultados que una configuración de *head attention* 1.

El entrenamiento mantuvo las mismas condiciones que el experimento anterior. El *score* reportado para la partición de test con la métrica es de -0,72281. Finalmente, obtenemos el modelo con mayor puntuación de este proyecto. Este resultado induciría a pensar que cuanto mayor es el número de *head attention* mejor es el resultado; en paralelo se realizo el mismo experimento con un valor de *head attention* igual a 6 y el modelo obtiene un *score* de -0,43031. Además, en el estudio original de la capa GATConv [11] se realizan experimentos que corroboran que el valor óptimo de *head attention* es 4.

4.2.2.7 Resultados de experimentos grafo

Nombre de experimento	*Score*
Modelo GCNConv todos los nodos relacionados	0,74661
Reducción del número de relaciones entre nodos	-0,03879
Normalización de las capas de la red GCNConv	-0,38936
Comprobación de compatibilidad de normalizaciones	-0,46089
Introducción de la GATConv, *head attention (1)*	-0,51561
Cambio número de *head attention* (6)	-0,43031
Cambio número de *head attention* (4)	-0,72281

Tabla 4 Puntuación de experimentos grafo

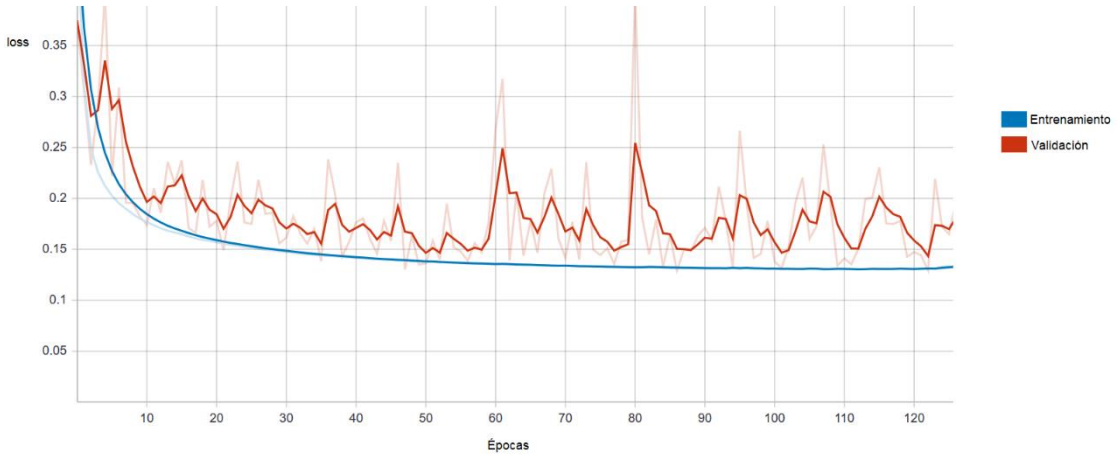

Ilustración 14 Gráfico loss modelo GATConv head attention 4

Tal y como se puede apreciar en la imagen anterior, el modelo GATConv alcanza el mínimo de *loss* alrededor de la época 120. Tras pasar las 120 épocas, el modelo no logra obtener menor *loss*.

4.2.3 Contrastación entre experimentos grafo y FC

Tal y como demuestran los experimentos, la aplicación de las metodologías que emplean una estructura grafo dan resultados altamente superiores. Esto se debe a que son invariables al orden y al número de elementos. Eso convierte a las redes grafos en una herramienta muy potente para el análisis de moléculas.

4.3 Comparativa con el estado del arte

Nombre del grupo	*Score*
BCAI	-3,23968
Quantum Uncertainty	-3,22349
KA.KR	-3,19498
Método Propuesto	-0,72281

Tabla 5 Comparativa con estado del arte

Pese a la buena progresión que ha tomado este proyecto, tal y como indica la tabla anterior, los resultados aún están lejos de los resultados que han obtenido los equipos que forman parte del estado del arte, previamente mencionado en el apartado 2.2.

La diferencia entre resultados respecto al estado del arte indica que existe un margen de mejora. La explicación de esta diferencia es que en el estado del arte se usan *ensembles* de varias arquitecturas y, además, utilizan mayor cantidad de datos como las estructuras triples y cuádruples. El hecho de no tener en cuenta estos datos, puede llegar a generar que no se tengan en cuenta posibles interacciones magnéticas dentro de la molécula. La decisión de no incluir mayor cantidad de combinatorias se debió a acotar el problema y reducir la carga computacional que conlleva su uso.

Sin embargo, en el apartado 6.1 se proponen varias vías de mejora de este proyecto. La finalidad es acotar la infinidad de vías para solucionar este problema y mejorar los resultados de este proyecto.

5. Presupuesto

En esta sección se analiza cual sería el presupuesto teórico para llevar a cabo todo el desarrollo del proyecto.

La ejecución de todos los programas empleados en este proyecto se ha realizado en el servidor del Grupo de Procesamiento de Imágenes (GPI). Las especificaciones de la máquina que se ha utilizado son las siguientes:

1. RAM: 20 GB.
2. CPUs: 3.
3. Almacenamiento físico: 100GB.
4. GPUs: 1 unidad con 11GB

Para poder realizar una aproximación del coste del servidor, me he basado en los precios que ofrece la plataforma Google Cloud Platform [33], que son alrededor de los 0,8€/h.

Cabe mencionar que tanto las herramientas de programación como el código empleados son de código libre, por lo que no suponen ningún gasto adicional al proyecto.

El personal necesario para el desarrollo del proyecto está constituido por un Ingeniero Junior, un Ingeniero Senior y un consultor. En último lugar, el coste de los salarios de los ingenieros ha sido calculado en función del tiempo dedicado al proyecto.

	€/hora	Tiempo dedicado (h)	€
Ingeniero Junior	8	400	3200
Ingeniero Senior	30	50	1500
Servidor	0,8	1320	1056
Consultor	50	30	1500
Coste total			7256

Tabla 6 Presupuesto

6. Conclusión

A nivel personal considero cumplidos mis objetivos debido a que, para llevar a cabo el desarrollo de este proyecto, he aprendido a utilizar muchas de las técnicas básicas de *Deep Learning* y, además, he tenido la oportunidad de aprender a trabajar con redes grafo.

A lo que se refiere a los objetivos del proyecto, se podría decir que mayoritariamente todos los objetivos han sido cumplidos. Si se revisan los objetivos principales de este proyecto, se puede observar que se ha obtenido el conocimiento sobre aspectos químicos suficiente para poder desarrollar un proyecto como este.

Por otro lado, se ha demostrado que se pueden realizar predicciones con técnicas de vanguardia no grafo y se observa claramente como estas técnicas no están diseñadas para trabajar con estructuras de datos no reticulares. En contraposición, se ha implementado y demostrado la eficacia de las redes grafo respecto a las técnicas de vanguardia cuando el problema a solucionar tiene una estructura de datos no reticular.

Finalmente, el único objetivo que no se ha podido lograr ha sido alcanzar un nivel de precisión igual al que se puede encontrar en el estado del arte. Sin embargo, en el apartado 6.1 se proponen varias vías de mejora de este proyecto con la finalidad de acotar la infinidad de vías para solucionar este problema y mejorar los resultados de este proyecto.

6.1 Propuestas de futuro

Con el propósito de mejorar los resultados obtenidos y hacerlos comparables con el estado del arte se proponen dos posibles opciones de mejora:

La primera opción es la agregación de los *triplets* como datos de entrada. Dichos datos se encuentran en la base de datos y representan la combinación estructural de tres átomos. Estos datos contienen información estructural de la molécula, y basándome en el desarrollo de este proyecto, deberían repercutir positivamente en los resultados.

En este proyecto cada nodo contiene un enlace o una *J-Coupling*, ya que dichos elementos generan contantes de acoplamiento. La segunda opción de mejora que propongo es cambiar el contenido de los nodos por átomos y que las aristas del grafo tengan las características de los enlaces químicos o *J-Coupling*. De esta manera se debería de predecir el valor de la arista en vez de predecir el valor del nodo, tal y como he implementado en este proyecto. Al cambiar conceptualmente la manera en la cual se representa cada molécula con un grafo, debería de ofrecer unos mejores resultados ya que se respetaría la configuración total de la molécula.

Bibliografía

[1] "Predicting Molecular Properties | Kaggle." https://www.kaggle.com/c/champs-scalar-coupling (accessed Jun. 07, 2020).

[2] Michael. E. Peskin and Daniel V. Schroeder, *An introduction to quantum field theory*. 1995.

[3] "Coupling constants - SSPPS NMR Facility - UC San Diego." http://sopnmr.ucsd.edu/coupling.htm (accessed Jun. 11, 2020).

[4] A. Paszke *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[5] M. Fey and J. E. Lenssen, "Fast Graph Representation Learning with PyTorch Geometric," *ICLR Work. Represent. Learn. Graphs Manifolds*, Mar. 2019, Accessed: Jun. 20, 2020. [Online]. Available: http://arxiv.org/abs/1903.02428.

[6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. 2016 , pp. 164 [Online]. Available: http://www.deeplearningbook.org.

[7] A. Czumaj, K. Jansen, F. Meyer auf der Heide, and I. Schiermeyer, "Algorithmic Graph Theory," *Oberwolfach Reports*, pp. 379–460, 2006, doi: 10.4171/OWR/2006/07.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Dec. 2016, vol. 2016-December, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[9] J. Zhou *et al.*, "Graph Neural Networks: A Review of Methods and Applications," *arXiv e-prints*, Dec. 2018, Accessed: Jun. 23, 2020. [Online]. Available: http://arxiv.org/abs/1812.08434.

[10] J. A. and D. Towsley, "Diffusion-convolutional neural networks," pp. 1993–2001, 2016, [Online]. Available: https://arxiv.org/abs/1511.02136.

[11] P. Veličković, A. Casanova, P. Liò, G. Cucurull, A. Romero, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, Oct. 2018.

[12] V. Garcia and J. Bruna, "Few-Shot Learning with Graph Neural Networks," *arXiv e-prints*, Nov. 2017, Accessed: Jun. 24, 2020. [Online]. Available: http://arxiv.org/abs/1711.04043.

[13] S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley, "Molecular graph convolutions: moving beyond fingerprints," *J. Comput. Aided. Mol. Des.*, vol. 30, no. 8, pp. 595–608, Aug. 2016, doi: 10.1007/s10822-016-9938-8.

[14] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc.*, Sep. 2016, Accessed: Jun. 07, 2020. [Online]. Available: http://arxiv.org/abs/1609.02907.

[15] A. Mosella-Montoro and J. Ruiz-Hidalgo, "Residual Attention Graph Convolutional Network for Geometric 3D Scene Classification," *Proc. - 2019 Int. Conf. Comput. Vis. Work. ICCVW 2019*, pp. 4123–4132, Sep. 2019, doi: 10.1109/ICCVW.2019.00507.

[16] J. Ren, W. Li, T. Jiang, and Z. Shuai, "A General Automatic Method for Optimal Construction of Matrix Product Operators Using Bipartite Graph Theory," *arXiv e-prints*, Jun. 2020, Accessed: Jun. 09, 2020. [Online]. Available: http://arxiv.org/abs/2006.02056.

[17] J. Stier and M. Granitzer, "Deep Graph Generators," *arXiv e-prints*, Jun. 2020, Accessed: Jun. 09, 2020. [Online]. Available: http://arxiv.org/abs/2006.04159.

[18] "3rd solution - BERT in chemistry - End to End is all you need | Kaggle." https://www.kaggle.com/c/champs-scalar-coupling/discussion/106572 (accessed

Jun. 09, 2020).
[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv e-prints*, Oct. 2018, Accessed: Jun. 09, 2020. [Online]. Available: http://arxiv.org/abs/1810.04805.
[20] “#2 solution Quantum Uncertainty | Kaggle.” https://www.kaggle.com/c/champs-scalar-coupling/discussion/106468 (accessed Jun. 09, 2020).
[21] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Nov. 2017, vol. 2017-January, pp. 77–85, doi: 10.1109/CVPR.2017.16.
[22] A. Vaswani *et al.*, “Attention Is All You Need,” *arXiv e-prints*, Jun. 2017, Accessed: Jun. 09, 2020. [Online]. Available: https://arxiv.org/abs/1706.03762.
[23] “#1 Solution - hybrid | Kaggle.” https://www.kaggle.com/c/champs-scalar-coupling/discussion/106575#634894 (accessed Jun. 09, 2020).
[24] “boschresearch/BCAI_kaggle_CHAMPS: Bosch solution to CHAMPS Kaggle competition.” https://github.com/boschresearch/BCAI_kaggle_CHAMPS (accessed Jun. 09, 2020).
[25] A. S. and R. B. D. Gaur, “Metagraph: A New Model of Data Structure,” *2008 Int. Conf. Comput. Sci. Inf. Technol.*, pp. 729–733, Dec. 2008, Accessed: Jun. 09, 2020. [Online]. Available: http://arxiv.org/abs/1912.09867.
[26] R. Todeschini and V. Consonni, “Descriptors from Molecular Geometry,” in *Handbook of Chemoinformatics*, vol. 3, Wiley Blackwell, 2008, pp. 1004–1033.
[27] “RDKit: Open-source cheminformatics.” http://www.rdkit.org (accessed Jun. 09, 2020).
[28] “Open Babel: An open chemical toolbox,” 2011, Accessed: Jun. 09, 2020. [Online]. Available: https://doi.org/10.1186/1758-2946-3-33.
[29] “J-Coupling (Scalar) - Chemistry LibreTexts.” https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Spectroscopy/Magnetic_Resonance_Spectroscopies/Nuclear_Magnetic_Resonance/NMR_-_Theory/NMR_Interactions/J-Coupling_(Scalar) (accessed Jun. 12, 2020).
[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016 , pp. 189 [Online]. Available: http://www.deeplearningbook.org.
[31] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Dec. 2015.
[32] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *32nd International Conference on Machine Learning, ICML 2015*, Feb. 2015, vol. 1, pp. 448–456, Accessed: Jun. 19, 2020. [Online]. Available: https://arxiv.org/abs/1502.03167v3.
[33] “Precios de GCP | Google Cloud.” https://cloud.google.com/pricing (accessed Jun. 13, 2020).