UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE CIENCIAS

ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACIÓN

Solución al problema de Rutas de Vehículos aplicando GNN y CNN

PROYECTO DE TESIS II

Autor: Jesús Miguel Yacolca Huamán

Asesor: Jaime Osorio Ubaldo

Mayo, 2022

Resumen

En esta tesis se realizará una comparación entre las Redes Neuronales basadas en Grafos y las Redes Neuronales Convolucionales, GNN y CNN respectivamente por sus siglas en inglès, en el problema de Rutas de Vehículos. Se tendrá en cuenta la rapidez en la convergencia para la obtención de parámetros que den resultados óptimos. A su vez se comparará la calidad de resultados de ambas arquitecturas de Redes Neuronales.

Índice general

Índice de figuras

Índice de Acrónimos

CVRP	Capacited Vehicle Routing Problem
GNN	Graph Neural Network
GCN	Graph Convolutional Network
CNN	Convolutional Neural Network
FNN	Feed-Forward Neural Network
FC	Fully-connected layer
MLP	Multi-layer Perceptron
NN	Neural Network
DNN	Deep Neural Network
DL	Deep Learning
ETC	Etcétera

Agradecimientos

Para la realización de esta tesis se agradece al asesor de tesis por la ayuda en la creación de la misma. A su vez debo agradecer a mi familia por haberme apoyado en el transcurso de mi estadía en la universidad. Al mismo tiempo dar las gracias a los profesores que me han impartido clases que han sido buenas y dieron lo mejor de sí para transmitir conocimientos a lo largo de mi carrera profesional.

Capítulo 1

Introducción

Las Redes Neuronales basadas en Grafos (GNN) son una arquitectura de Red Neuronal (NN) la cuál ha sido recientemente introducida en el campo del Deep Learning que está especializada en el tratamiento de datos con una estructura de Grafo. Un ejemplo de datos con esta estructura son las imágenes y el texto. La primera como una red interconectada donde cada arista puede ser la intensidad de cada pixel. En el caso del texto se puede estructurar como una serie de nodos consecutivos conectados uno tras otro. En este mismo sentido el problema de rutas de vehículos utiliza una estructura de grafo para representar los caminos que unen los nodos, siendo la distancia de estos caminos las aristas del grafo.

Por otro lado, las Redes Neuronales Convoluciones (CNN) han demostrado ser buenas en los problemas relacionados a imágenes. Las cuales son un tipo especial de grafo conocido como grid.

Estas dos arquitecturas pueden usarse para resolver el problema de rutas de vehículos, el cual es el problema a tratar en esta tesis.

1.1. Motivación

La motivación para este trabajo es la de dar una revisión e implementación de esta arquitectura que está creciendo en popularidad como son las GNN. Revisar para así conocer los alcances que pueda tener. Implementar para poder probar su eficacia. A su vez se comparara con un tipo de arquitectura más conocido y con una gran cantidad de estudios como son las CNN. Esta última

ha demostrado su valía para el procesamiento de imágenes. Sin embargo, este tipo de arquitectura no da buenos resultados sobre grafos que no euclideos. Es en estos últimos donde las GNN dan buenos resultados. Siendo así, el problema de Rutas de Vehículos, un problema sobre grafos no euclideos, es ideal para demostrar la capacidad de las GNN sobre este tipo de grafos.

1.2. Objetivos de la Investigación

1.2.1. Objetivo General

Evaluar las arquitecturas GNN y CNN en el problema de Rutas de Vehículos para de esta forma comprobar la efectividad de las GNNs sobre este problema.

1.2.2. Objetivos Específicos

- Determinar la eficiencia de ambas arquitecturas al tratar el problema de Rutas de Vehículos.
- Interpretar los resultados obtenidos por ambas arquitecturas sobre el problema de Rutas de Vehículos.
- Examinar los tiempos de entrenamiento para ambas arquitecturas.

Capítulo 2

Planteamiento del Problema

2.1. Descripción del Problema

El problema a tratar es el de Rutas de Vehículos. Este problema trata de hallar la ruta que optimiza el recorrido de las mismas. Este problema puede ser representado mediante un grafo, este grafo es del tipo no euclideo. Los pesos en las aristas suelen ser las distancias entre los nodos.

2.2. Formulación del Problema

Para poder abordar este problema se usará dos tipos de representaciones de las rutas. Estas serán la representación en forma de grafo y la representación en forma de matriz de adyacencia. La primera se usará para entrenar a la arquitectura de Red Neuronal GNN, mientras que la segunda, para la CNN. Esto se debe a que la CNN solo trabaja con grafos del tipo malla. Transformando el problema a uno de trazos sonre una ímagen. Después de entrenar estas arquitecturas se obtendrå la ruta que minimice la suma de los pesos en las aristas.

2.3. Justificación del Estudio

Las redes neuronales (NN) han demostrado ser uno de los métodos del Deep Learning que mejores resultados está dando. Un tipo particular de red

neuronal es la Red Neural Profunda (DNN). Esta se compone por capas, dos de las más conocidas son la capa convolucional y recurrente.

Además muchos de los tipos de datos en importantes componentes de la vida actual como las redes sociales o ramas de la ciencia como la biología tienen una estructura de grafo.

Esta estructura tambièn esta relacionada con muchos problemas de optimización. Problemas como el que se abordará a lo largo de esta tesis que viene a ser el de la Ruta de Vehículos.

En las imágenes las CNNs han demostrado tener gran performance gracias a características tales como la compartición de pesos y el uso de múltiples capas la localización de conexiones (LeCun, Yoshua Bengio, y G. Hinton, 2015). En contraste, suele tener problemas al momento de tratar con grafos no euclídeos.

Sin embargo, otro tipo de red neuronal conocido como GNN que está especializado en el tratamiento de datos con estructura de grafo, da mejores resultados en el tratamiento de datos representados como grafos.

El tratamiento de este tipo de datos como grafos suele ser complicado debido a la cantidad de aristas no homogenea que cada nodo en estas contienen. Aunque la importancia de su tratamiento es crucial pues está vinculado a importantes áreas de la matemática como es la optimización, incluso a otras ramas de la ciencia como lo son la medicina o la biología.

Capítulo 3

Hipótesis

3.1. Hipótesis General

La GNN obtendrá mejores resultados que la CNNZ en la predicción de la ruta que optimice el costo en un problema de CVRP.

3.2. Hipótesis Específica

- El tiempo de entrenamiento de las CNN es mayor al de las GNNs.
- La robustez de las predicciones no será tan alta debido a la alta complejidad del problema a tratar.

Capítulo 4

Estado del Arte

Se dará uun marco histórico que da el contexto en el cual se esta realizando esta tesis. A su vez se realizará una sumilla de los libros, artículos o tesis que sustentan este trabajo dando los resúmenes, los objetivos y las conclusiones de cada uno.

4.1. Marco Histórico

El área del Aprendizaje de Máquina, Machine Learning en inglés (ML), ha tenido un crecimiento pronunciado en la última década a partir del desarrollo de las Redes Neuronales (NN). Inicialmente se utilizaba el Perceptrón Multicapa (MLP) para los problemas de clasificación. Luego se dio un nuevo avance en este tipo de problemas gracias a las Redes Neuronales Convolucionales (CNN) que se especializan en los referidos a imágenes. En el presente un tipo de arquitectura nuevo especializado en los problemas con datos estructurados como grafos. Esta arquitectura se llama Graph Neural Network (GNN). Esta arquitectura fue presentada en 2009 en el paper titulado "The Graph Neural Network Model"[7]. Ahora las GNNs son potenciadas gracias al uso de la convolución tomando el nombre de Convolutional Graph Network (GCN).

Estas mismas podrían dar solución a uno de los problemas más importantes en lo que a optimización se refiere, como es el Problema de Rutas de Vehículos. Esta tradicionalmente es resuelto mediante heurísticas o métodos exactos,

ahora puede ser resuelto por este tipo de red neuronal que se especializa en este tipo de estructura como es el grafo.

4.2. An Overview of Vehicle Routing Problems [1]

4.2.1. Resumen

Este libro da una revisión a los problemas de rutas de vehículos, VRT por sus siglas en inglès, dando el marco teórico necesario para la elaboración de esta tesis. Escrito en 2014, da soluciones a estos problemas mediante algoritmos heurísticos o exactos. Sin embargo, no aborda su resolución mediante algoritmos del campo del Machine Learning o del Deep Learning.

4.3. An Investigation Into Graph Neural Networks [2]

4.3.1. Objetivos

- Dar una revisión de acerca de las GNNs.
- Describir las librerías y frameworks necesarios para crear una GNN y como implementarla.

4.3.2. Resumen

En esta tesis desarrollada por V. Kumar se da una revisión de las GNNs. Dando una sustentación teórica de las mismas. Además, se da un resumen de las librerías y frameworks que son necesarios para el desarrollo de las GNNs. También se dan las limitaciones de las mismas. Luego se implementan para resolver distintos tipos de tareas sobre datasets conocidos como lo son MNIST y Cora.

4.3.3. Conclusiones

- Es complicado encontrar la librería correcta para cada experimento.
- A pesar de que las GNNs no daban resultados satisfactorios en muchas condiciones, estas logran conseguir resultados destacables en otros.

4.4. An overview of convolutional neural network [3]

4.4.1. Objetivos

- Realizar una revisión de los fundamentos detrás de la arquitectura CNN.
- Describir las aplicaciones de las CNNs.

4.4.2. Resumen

En este paper se realiza una revisión de la arquitectura de las CNNs. Primero se presenta los fundamentos de la arquitectura. Luego se presenta algunas revisiones de la arquitectura más complejas. Finalmente se dan algunos casos de aplicaciones.

4.4.3. Conclusiones

- Se revisó la arquitectura de las CNNs y sus aplicaciones.
- Las CNNs han demostrado ser de las mejores arquitecturas para las tareas relacionadas al campo de la Visión Computacional.

4.5. A Gentle Introduction to Graph Neural Networks [4]

Objetivos

- Explorar y explicar la arquitectura de las modernas GNNs.

4.5.1. Resumen

En este trabajo se detalla de una manera simple y entendible los conceptos principales concernientes a las GNNs. Se divide en 4 partes. En la primera se da una explicación y ejemplos de los tipos de datos que son mejor representados por grafos. En la segunda se da una explicación de que hace diferente a los grafos de otros tipos de datos. Para la tercera se construye una GNN explicando en este proceso los conceptos de la misma. En la última parte se provee una representación animada de una GNN en la cual se puede ajustar sus parámetros para un mayor entendimiento de su funcionamiento.

4.5.2. Conclusiones

- Los grafos son una estructura de datos poderosa que genera retos diferentes a los vistos con imágenes y texto.
- Se revisó en el artículo algunas de las decisiones de diseño importantes que se deben tener en cuenta al momento de diseñar una GNN.

4.6. Understanding Convolutions on Graphs [5]

4.6.1. Objetivos

- Ilustrar los retos de la computación relacionada a grafos.
- Explorar las variantes más recientes de las GNNs.

4.6.2. Resumen

En este artículo se da una revisión de la arquitectura de las GNN. Además, se da una explicación de la problemática del uso de la convolución convencional en el tratamiento de grafos. En particular se trata el problema de la falta de orden en los nodos. A esto se muestra una solución mediante la extensión de la noción de laplaciano a grafos. También se muestran variantes de las GNNs como lo son las GCNs.

4.6.3. Conclusiones

- En el artículo se da una muestra del campo amplio de las GNNs a la vez de mostrar varias técnicas e ideas relacionadas a este campo.
- Se comunicaron las ideas principales para entender las GNNs y algunas de sus variantes.

4.7. Finding Solutions to the Vehicle Routing Problem using a Graph Neural Network [6]

4.7.1. Objetivos

- Explorar GNN para resolver el VRP.
- Uso de GNN, en especifico, Recurrent Relational Network, para obtener las probabilidades de cada arista y luego usar Beam Search para obtener el grafo.

4.7.2. Resumen

En esta tesis se utilizó un subtipo de GNN, conocido como RRN o Red Recurrente Relacional, para obtener las probabilidades de elección de cada arista y así mediante el uso de Beam Search obtener el grafo que de solución

al Problema de Rutas de Vehículos sin restricciones, con 20 y 50 nodos, y al Problema del Vendedor Viajero con 20 nodos.

4.7.3. Conclusiones

- El modelo tiene mejor performance en VRP 20 y VRP 50, pero no en TSP 20 que OR-Tools. Esto puede deberse a que esta herramienta maneja eficientemente TSP, pero para VRP con muchos nodos tarda en alcanzar una solución.
- Aunque RRN no es bueno prediciendo datos de test, solo de entrenamiento, es un buen punto de partida para Beam Search.
- Los mejores resultados son dados en las iteraciones iniciales, esto junto a la irregularidad progresión sugiere que el carácter relacional de RRN no aporta mejoras significativas, por lo que, otras arquitecturas de GNN, pueden dar mejores resultados.
- Mejor data ayudaría a obtener mejores resultados, pero esto es complicado, siendo la obtención de datasets una de las debilidades del aprendizaje supervisado.

Capítulo 5

Marco Teórico

5.1. Conceptos Previos

5.1.1. Problema de Rutas de Vehículos

La definición para este problema dada por el libro An Overview of Vehicle Routing Problem [1] es la siguiente: "Dado un conjunto de requerimientos de transporte y una flota de vehículos se pide determinar un conjunto de Rutas de vehículos que satisfaga todos (o la mayoría) de requerimientos de transporte con la flota de vehículos dada con el mínimo coste; en particular, decidir que vehículos manejaran cuales requerimientos y en que orden de tal forma que todas las rutas de vehículos puedan ser factibles."

Así como se menciona en el libro citado anteriormente, este problema tiene implicaciones directas en el mundo real, tales como la planificación de rutas para vehículos auto-guiados.

Existen muchas versiones de este problema como el Problema de Rutas de Vehículos con Ventanas de Tiempo (Vehicle Routing Problem with Time Windows VRPTW en inglés) o la versión estocástica conocida como Problema de Rutas de Vehículos Estocástico (Stochastic Vehicle Routing Problem SVRP en inglès). Sin embargo la versión que se abordará en esta tesis será la conocida como Problema de Rutas de Vehículos con limitaciones de capacidad, CVRP por sus siglas en inglès.

5.1.2. Problema de Rutas de Vehículos con limitaciones de capacidad (CVRP)

Esta es la variante más estudiada de las que existen en la familia de problemas de Rutas de Vehículos. Además de ser fácil de entender

Antes de definir el problema formalmente, es necesario conocer la terminología que se usa. En este modo se define un "*depot*" o centro de distribución, usualmente denotado como "0", desde donde se parte a realizar las entregas a los "*consumidores*" denotados como una lista $N = \{1, 2, 3, ...\}$ de los nodos que los representa y lo que solicita cada consumidor $i \in N$ se denomina "*demanda del consumidor*" el cual es un número $q_i \geq 0$ que podría ser el peso de las entregas solicitadas. La "*flota de vehículos*" de la que se dispone en el *depot* para realizar las entregas es representada por la lista $K = \{1, 2, ..., |K|\}$, cada vehículo puede llevar una carga $Q > 0$. Un vehículo que responda a la *demanda* de un subconjunto $S \in N$ empieza su recorrido en el *depot* recorre cada consumidor en S una vez y retorna a donde partió inicialmente. Un vehículo que recorra el tramo entre el nodo i y j conlleva un "*costo de viaje*" c_{ij}.

Definiendo el conjunto de vértices de un grafo que represente este problema como $V = \{0\} \cup N$ y $q_0 = 0$. A partir de esto se puede construir un grafo dirigido o no dirigido dependiendo de como se defina c_{ij} y c_{ji},)el camino contrario entre estos dos nodos.

Si se toma como iguales; es decir, la ida cuesta igual que la vuelta, entonces se tiene un grafo no dirigido. En este el conjunto de aristas es $E = \{e = \{i, j\} = \{j, i\} : i, j \in V, i \neq j\}$. Así se tiene el grafo no dirigido $G = (V, E)$.

Si existe $c_{ij} \neq c_{ji}$ entonces un camino tiene un costo diferente al ir y volver. Debido a esto se tiene un grafo dirigido donde el conjunto de arcos es $A = \{(i, j) \in V \times V : i \neq j\}$

Ahora el grafo para el CVRP esta completamente definido, teniendo una versión no dirigida $G = (V, E, c\{ij\}, q_i)$ y una dirigida $G = (V, A, c\{ij\}, q_i)$. Junto a la cantidad de vehículos disponible $|K|$ y su capacidad Q.

Ahora se define el concepto de "*ruta*" como $r = \{i_0, i_1, i_2, ..., i_s, i_{s+1}\}$ con

$i_0 = i_{s+1} = 0$ donde el subconjunto $S = \{i_1, ..., i_s\} \subseteq N$ es visitado. El costo de esta ruta es $c(r) = \sum_{p=0}^{s} c_{i_p i_{p+1}}$. Esta ruta es factible si cumple las restricciones de capacidad del vehículo que la realiza, $q(S) = \sum_{i \in S} q_i \leq Q$ y además un nodo es visitado solo una vez. De esta forma $S \subseteq N$ se denomina como un conjunto factible.

Una solución al CVRP consta de $|K|$ rutas factibles, cada una por cada vehículo $k \in K$, teniendo así las rutas $r_1, r_2, ..., r_{|K|}$ y sus conjuntos $S_1, S_2, ..., S_{|K|}$ proveyendo una "*solución factible*" si todas las rutas son factibles y los conjuntos forman una partición en N.

Para definir el problema como uno de programación entera mixta se tiene la siguiente notación. Sea $S \subseteq N$ arbitrario. Para la forma de grafo no dirigido se tiene el "*conjunto de corte*" $\delta(S) = \{\{i, j\} \in E : i \in S, j \notin S\}$ el conjunto de aristas con uno o ambos puntos terminales en S. Para la forma como grafo dirigido se tiene $\delta^-(S) = \{(i, j) \in A : i \notin S, j \in S\}$ y $\delta^+(S) = \{(i, j) \in A : i \in S, j \notin S\}$.

Para un conjunto de compradores $S \subseteq N$, $r(S)$ es el número mínimo de rutas necesario para cubrir S. A la vez, se denota como x_{ij} la frecuencia con la cual un vehículo cruza la arista i, j.

Ahora se puede modelar para la forma como grafo dirigido como sigue:

$$
\begin{aligned}
minimizar \sum_{(i,j) \in A} c_{ij} x_{ij} \\
\sum_{j \in \delta^+(i)} x_{ij} = 1 \forall i \in N, \\
\sum_{i \in \delta^-(j)} x_{ij} = 1 \forall j \in N, \\
\sum_{j \in \delta^+(0)} x_{0j} = |K|, \\
\sum_{(i,j) \in \delta^+(S)} x_{ij} \geq r(S) \forall S \subseteq N, S \neq \emptyset, \\
x_{ij} \in \{0, 1\} \forall (i, j) \in A
\end{aligned}
$$

Para la versión no dirigida se tiene:

$$
\begin{aligned}
minimizar \sum_{\{i,j\}\in E} c_{ij}x_{ij} \\
\sum_{j\in\delta(i)} x_{ij} = 2 \forall i \in N, \forall j \in N \\
\sum_{j\in\delta(0)} x_{0j} = 2|K|, \\
\sum_{(i,j)\in\delta(S)} x_{ij} \geq 2r(S) \forall S \subseteq N, S \neq \\
x_{ij} \in \{0,1,2\} \forall (i,j) \in \delta(0) \\
x_{ij} \in \{0,1\} \forall (i,j) \in E \setminus \delta(0)
\end{aligned}
$$

5.1.3. Redes Neuronales (NN)

Una Red Neuronal es un algoritmo del Aprendizaje Supervisado que, en su forma más simple, es una serie de capas compuestas por neuronas que están conectadas entre sí. Esto viene inspirado de la biología donde las neuronas transmiten información entre si mediante pulsos eléctricos en sus conexiones. Esta transmisión de información en las Redes Neuronales se da de manera similar asignando un peso a cada conexión, que en una Red Neuronal simple se conecta cada neurona en una capa con todas las neuronas en la capa siguiente. Los pesos se pueden colocar de manera simplificada en un vector para cada neurona. La unión de estos pesos en forma de vector de cada neurona en una capa forma una matriz conocida como "Matriz de pesos" representada con la letra W.

Este algoritmo realiza un proceso de aprendizaje el cual le permite ajustar sus parámetros en base a los datos que se le proporcionen. Estos parámetros son las matrices de peso de cada capa. De esta forma el proceso de aprendizaje consiste en actualizar estas matrices de pesos. Existen varios algoritmos que realizan estas tareas. Sin embargo, el más usado es el algoritmo de "Back-propagation". Este realiza un proceso de optimización sobre los parámetros para minimizar una función de coste. Esta función mide la

diferencia entre las salidas que se desean obtener y las que se obtienen al culminar el proceso del cálculo hacia adelante o "Forward calculation". El proceso de optimización conocido como "Backward propagation" o propagación hacia atrás pues se construye una función en base a las derivadas parciales de las capas anteriores.

Las capas de las Redes Neuronales se pueden dividir en tres grupos de acuerdo a su función y relación con los datos que se le proporcionen:

- **Capa de entrada:** Es la que se relaciona con los datos de los que se busca una predicción.
- **Capas ocultas:** Estas dan la robustez a la red para procesar los datos y la cantidad de neuronas y capas se determina de tal forma que se alcance un nivel de predicción aceptable.
- **Capa de salida:** De esta capa salen los valores predichos por la Red Neuronal.

Aunque con el pasar del tiempo y la realización de más estudios se han creado tipos de capas especializadas en ciertas tareas como la predicción de texto (RNNs) o clasificación de imágenes (CNNs). A continuación, se presenta tres tipos que son de vital importancia para este estudio:

- **Feed-Forward Neural Network (FNN):** Es el tipo más simple de Red Neuronal. Consiste en las 3 capas ya mencionadas. Además, posee la restricción de que una neurona de una capa solo puede conectarse a las neuronas de las capas vecinas. Un ejemplo de esto se puede ver en la Figura 5.1

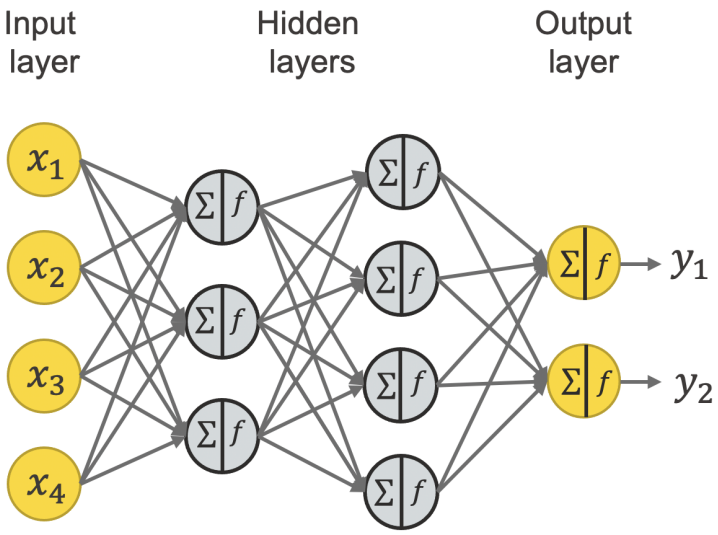

FIGURA 5.1: Feed-Forward Neural Network

- **Red Neuronal Convolucional (CNN):** Esta es un tipo particular de la anteriormente mencionada. Pero a diferencia de esta la conectividad local se preserva. [2]. Este tipo de capa también está compuesta por subcapas como:

  - Capa Convolucional
  - Capa de reducción de muestra o Pooling Layer
  - Capa simple donde todas las neuronas están conectadas con las de la capa inmediatamente anterior y posterior.

- **Red Neuronal basada en Grafos (GNN):** Esta capa esta especialmente diseñada para tratar con tipos de datos gráficos o con una estructura no euclídea. Tiene subtipos que pueden aprovechar la operación de convolución conocidos como GCN.

5.1.4. Red Neuronal Convolucional (CNN)

Una Red Neuronal Convolucional está compuesta, como se mencionó en la sección anterior, por tres tipos diferentes de capas: una capa convolucional, una capa Pool y una capa completamente conectada o Fully-conncected (FC).

Las CNNs se centran en la idea primaria que la entrada contiene una imagen que enfocan la arquitectura a construir en un método que ajuste con precisión la necesidad de lidiar con una forma particular de la información. Sin embargo, una de las características de las CNNs es que las capas intermedias están compuestas por neuronas organizadas en 3 partes conocidas como la dimensión espacial de la entrada. [3]

A continuación se realizará una revisión de los 3 tipos de capas que componen esta arquitectura especializada en el procesamiento de imágenes y las cuales son cruciales para el entendimiento de esta.

Capa convolucional

Esta capa es la más importante que posee esta arquitectura. Además, es la que mayor coste computacional genera. Esta realiza el proceso de la convolución sobre la imagen siguiendo unos parámetros que se centran en el empleo de lo que se conoce como kernels de aprendizaje. Estos suelen ser de una dimensionalidad reducida, aunque de todas formas recorriendo completamente la entrada. La capa convolucional opera de la siguiente forma sobre las entradas. A cada entrada se le aplica una multiplicación entre los pesos, que son los componentes del kernel, y una región asociada de la entrada para luego sumarlos dando un mapa en forma de matriz de la misma. Una imagen que ilustra este procedimiento es la Figura 5.2. Esta capa posee tres hiperparámetros que ayudan a controlar a la misma los cuales son el padding (P), el stride (S) y las dimensiones de la entrada (V). A partir de estos es posible calcular las dimensiones del kernel necesario para generar una salida de tamaño determinado (R) usando la siguiente fórmula:

$$\frac{(V - R) + 2P}{S + 1} \tag{5.1}$$

El proceso de entrenamiento de este tipo de red también hace uso del algoritmo de Backpropagation.

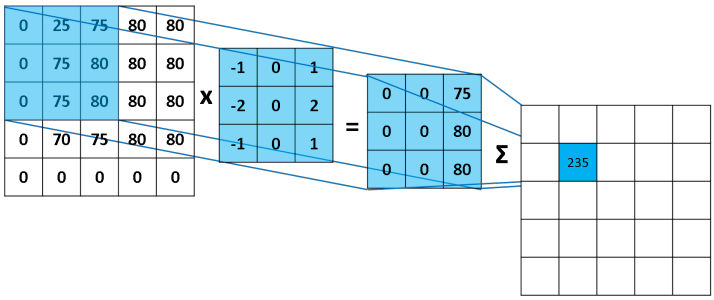

FIGURA 5.2: Proceso de una Capa Convolucional sobre una entrada

Capa de reducción de muestra o Pooling Estas capas también hacen parte de la arquitectura de una CNN y normalmente se le puede encontrar después de una convolucional. Siendo así la salida de la capa convolucional es la entrada de la capa de Pooling. En efecto este tipo de capas se usan para realizar una reducción de la dimensión de sus entradas mediante el uso de funciones como calcular el valor más común o el mayor en una región donde se aplique. La importancia de esta capa radica en el hecho que estas pueden reducir la complejidad del modelo. Existen varios tipos de estas capas como son Max pooling basado en hallar el valor máximo de la región u otro como uno llamado Average pooling basado en hallar la media de la región. La imagen que se puede ver en la Figura 5.3 ofrece una explicación visual de cómo funciona una capa de Pooling, en concreto una Max Pooling.

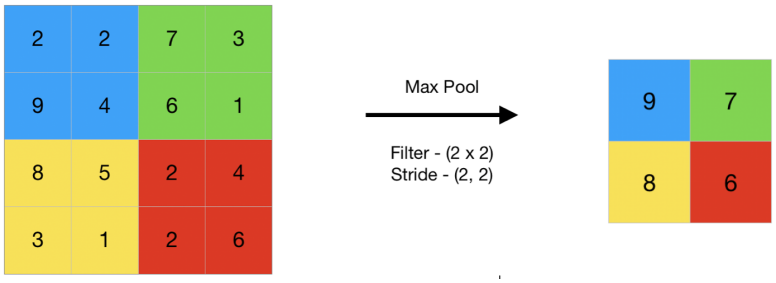

FIGURA 5.3: Proceso de una Capa de tipo Max Pooling sobre una entrada

Capa Fully-Connected Esta capa ya se vio en la sección anterior. Esta capa como parte de la arquitectura de las CNNs se coloca al final pues es esta capa no

encapsula bien la información espacial. También es usada para aplanar la salida de las capas de Pooling que llegan en una dimensión superior a 1. Sin embargo, esta no es indispensable para la creación de la arquitectura de las CNNs pues recientemente algunos diseños reemplazan esta con otra capa llamada General Average Pooling [8].

Una visualización de cómo funcionan estas capas conjuntamente se puede ver en la imagen de la Figura 5.4

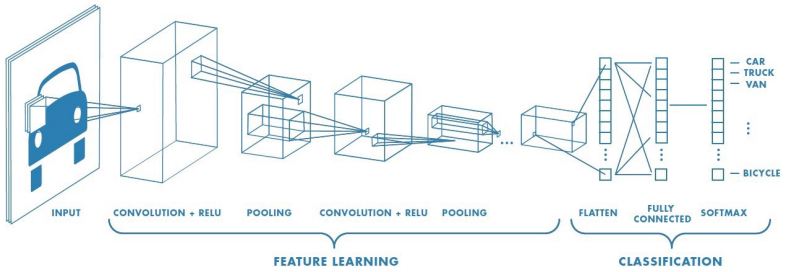

FIGURA 5.4: Funcionamiento de una CNN

5.1.5. Red Neuronal basada en Grafos (GNN)

Los grafos son una estructura de datos que está muy presente en la vida real pues esta sirve para representar la conexión entre grupos de objetos. Y es por esto que se han realizado estudios sobre redes neuronales que operan sobre grafos. Este tipo de red neuronal recibe el nombre de GNN. Estos estudios llevan realizándose más de una década. Sin embargo, son los avances recientes los que han aumentado sus capacidades en el poder de representación de la data. Gracias a esto se está empezando a ver utilidades en campos como la simulación de sistemas físicos como en la detección de noticias falsas. [4]

Un grafo puede ser representado mediante un conjunto de nodos y las aristas que los unen. Esto a su vez puede representarse con una matriz conocida como Matriz de adyacencia. En este sentido las imágenes también pueden ser representadas como grafos. En esta representación cada pixel es un nodo y se coloca una arista entre este y los vecinos que posee. Aunque esta representación puede ser muy redundante pues solo se crea una banda en la matriz de adyacencia.

Existen 3 tipos de problemas a resolver en grafos. Estos son a nivel de grafo, nodo o arista. Un ejemplo de un problema a resolver a nivel de grafo relacionado a imágenes es la clasificación de estas. En cuanto a una tarea a nivel de nodo es la segmentación de imágenes. Y un ejemplo a nivel de arista es establecer relaciones entre los objetos de una imagen.

Para representar un grafo se puede utilizar la matriz de adyacencia, pero esta no es eficiente pues ocupa mucha memoria y no es única para un mismo grafo llegando a tener varias de estas. Una alternativa a esto es una lista de adyacencia donde el elemento $K - esimo$ es la representación de la arista que conecta un nodo i con uno J de la forma (i, j). Un ejemplo de esto se puede ver en la Figura 5.5.

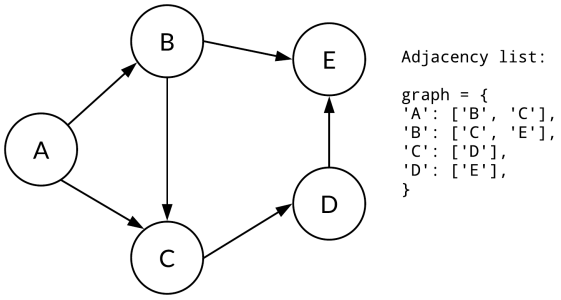

FIGURA 5.5: Representación de la lista de adyacencia de un grafo

Un modelo simple de GNN es pasar los nodos, las aristas o el contexto global del grafo a través del MLP. De esta forma se puede resolver las distintas problemáticas que se mostraron anteriormente y así realizar predicciones. Por ejemplo, si queremos realizar predicciones en el problema a nivel de nodo, solo es necesario aplicar el MLP entrenado a este. Una forma de visualizar esto se puede ver en la Figura 5.6.

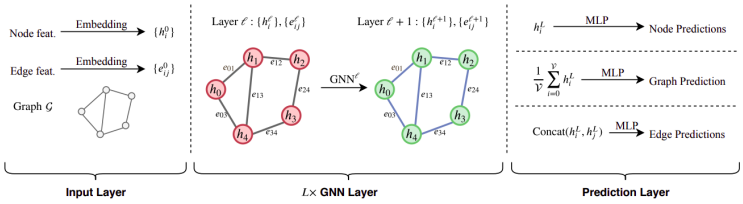

FIGURA 5.6: Representación del funcionamiento de una GNN

A veces no se disponen de la información de todas las partes que se utilizan para representar el grafo, como la de los nodos o aristas, afortunadamente este problema puede solventarse gracias a una técnica conocida como la transferencia de mensajes. Esta consiste en, generalmente, sumar la información contenida en los objetos adyacentes al que necesitamos suministrar información. Por ejemplo, si necesitamos información de los nodos, pero solo tenemos la de los vértices se puede obtener esta información como la suma de los vectores de las aristas de las cuales este nodo es parte.

El proceso de transferir mensajes también se puede realizar entre distintas capas de la GNN. Por ejemplo, se puede transferir información de los nodos de la capa anterior sumando a los vectores de los nodos de la nueva capa los vectores de los nodos vecinos a este en la capa anterior por cada nodo en la nueva capa.

Existen diversos subtipos de GNNs. Sin embargo, esta tesis se centrará en las GCNs.

5.1.6. Red Convolucional en Grafos (GCN)

Esta red es parte de las GNNs. Esta hace uso de la generalización de la operación de convolución de las CNNs, que opera sobre un grafo en forma de rejilla que es como se puede representar a una imagen como grafo, a un tipo más general de grafo. Esto puede resultar complicado pues un grafo, en general, son invariantes bajo el orden de los nodos; es decir, no importa el orden en el cual estos se elijan. Esta última propiedad otorga flexibilidad en la representación de datos, pero la convolución depende de la posición absoluta de los pixeles. Además de esto la estructura de un nodo a otro cambia sustancialmente pues puede tener diferente número de nodos vecinos. [5]

Una forma de conseguir utilizar la operación de convolución en grafos es mediante la definición de filtros sobre estos como se hace en las CNNs.

Primero se mostrará el Laplaciano de un grafo que se puede definir como sigue:

$$L = D - A \tag{5.2}$$

Donde A es la matriz de Adyacencia del grafo y D es una matriz diagonal que se define como sigue:

$$D_v = \sum_u A_{vu} \tag{5.3}$$

Esto esencialmente viene a ser que para el nodo v el valor es la suma de esa fila en la matriz de adyacencia. Este Laplaciano representa la misma información que A. Además, a partir de A se puede hallar L y viceversa.

Ahora se puede formar polinomios con estos de la siguiente forma:

$$p_w(L) = \sum_{i=0}^{d} w_i L^i \tag{5.4}$$

Ahora este polinomio de grado d puede ser un filtro tal y como se tienen en una CNN con los coeficientes w como los pesos. Ahora si consideramos a la información asociada a un nodo como un valor único real; entonces, podemos concatenar estos para cada nodo obteniendo un vector x. De esta forma aplicar el filtro obteniendo un vector x' es equivalente a:

$$x' = p_w(L)x \tag{5.5}$$

Un dato importante a considerar sobre el grado d del polinomio es que este afecta a como se realiza el filtro en un nodo v, esto es solo se realiza entre nodos que no estén separados más de d pasos.

Se dice que un algoritmo f es equivariante si dada una permutación P sobre la entrada x se cumple que:

$$f(Px) = Pf(x) \tag{5.6}$$

Se puede probar que este polinomio es equivariante del orden de los nodos. Por tanto, esta forma de definir la convolución es equivariante. Además, se puede ver que con un polinomio de grado d un nodo v solo afecta otro nodo u como ya se mencionó antes. Esto es similar a una transferencia de información entre los nodos. Si este paso se repitiese k veces, entonces el mensaje se propagará a

todos los nodos del grafo que estén a una distancia k entre sí.

Una manera de diferenciar a los diferentes tipos de GNNs es mediante el cálculo del embedding. Como esta tesis se centrará en el uso de las GCN este será el que se presentará a continuación [5]:

$$h_v^{(0)} = x_v \qquad \forall\, v \in V \tag{5.7}$$

Aquí $h_v^{(0)}$ es el embedding inicial del nodo v. Para K iteraciones se pueden calcular los enbeddings como sigue:

$$h_v^{(k)} = f^{(k)} \left(W^{(k)} \cdot \frac{\sum\limits_{u \in \mathcal{N}(v)} h_u^{(k-1)}}{|\mathcal{N}(v)|} + B^{(k)} \cdot h_v^{(k-1)} \right) \qquad \forall\, v \in V \tag{5.8}$$

Nótese que tanto $f^{(K)}$ como $W^{(K)}$ y $B^{(K)}$ son los mismos para todos los nodos. Las predicciones pueden obtenerse como sigue:

$$y_v = PREDICT\left(h_v^{(K)}\right) \tag{5.9}$$

Aquí PREDICT suele ser otra red neuronal que trabaja junto a la GCN. Esta formulación de la normalización del embedding es diferente a la dada en el paper original donde se presentaron las GCN.

Capítulo 6

Metodología de Trabajo

En esta sección se mostrará la forma en la que se procedió a resolvió el problema de Rutas de Vehículos con las CNNs y GNNs. En este sentido se dividirá este capítulo en 4 secciones. En la primera se detallará la forma de abordar el problema de obtener el grafo que de solución al problema. Seguidamente se especificará como se obtuvieron los datasets a usar. Luego se detallará la elección de los modelos. Finalmente se mostrará la forma de comparar los resultados de las implementaciones, tanto con CNN como GNN, en base a determinadas mètricas.

6.1. Problema

Como se ha ido mencionando en secciones anteriores, el problema a tratar es el de Rutas de Vehículos. El motivo de esto se debe a la gran importancia de este en la optimización y con aplicaciones varias en la industria. Además de dar una solución que implique el uso de Redes Neuronales y sea eficiente. El resultado que se busca es el grafo que de solución al mismo.

6.2. Datasets

Se generaran 10000 instancias de VRP, mediante el uso de una librería de conocida como OR-Tools, con su respectivo grafo solución.

6.3. Elección del modelo

Primero se tendrá como base solución brindada por MLP, tambièn conocido como Feed-Forward Neural Network. Este se comparará al modelo más simple de CNN con un subtipo de GNN. Para la elección del subtipo de modelo de GNN se tendrá en cuenta la complejidad del modelo y el que más parecido sea a las CNNs.

6.4. Métricas

Para la comparación se obtendrá la precisión y la perdida en cada época para ambos modelos. A su vez se usará una mètrica conocida como Average Optimality Gap y se calcula como sigue:

$$\frac{1}{m}\sum_{i=1}^{m}(\frac{l_i}{t_i}-1) \tag{6.1}$$

Donde l_i es el valor obtenido del tour y t_i es el esperado.

Luego estas se graficarán para la observación de cual obtiene mejores resultados en los datos de evaluación y la rapidez con la cual los obtiene.

Bibliografía

[1] Paolo Toth and Daniele Vigo. *1. An Overview of Vehicle Routing Problems*, pages 1–26.

[2] Vishal Kumar. *An Investigation Into Graph Neural Networks*. PhD thesis, Trinity College Dublin, Ireland, 2020.

[3] Shadman Sakib, Nazib Ahmed, Ahmed Jawad Kabir, and Hridon Ahmed. An overview of convolutional neural network: its architecture and applications. 2019.

[4] Benjamin Sanchez-Lengeling, Emily Reif, Adam Pearce, and Alexander B Wiltschko. A gentle introduction to graph neural networks. *Distill*, 6(9):e33, 2021.

[5] Ameya Daigavane, Balaraman Ravindran, and Gaurav Aggarwal. Understanding convolutions on graphs. *Distill*, 6(9):e32, 2021.

[6] Fredrik Hagström et al. Finding solutions to the vehicle routing problem using a graph neural network. 2022.

[7] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[8] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.