UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

EXPLORANDO EL POTENCIAL DE LAS REDES NEURONALES DE GRAFOS PARA EXTRAER INFORMACIÓN DE BGP

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN COMPUTACIÓN

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO EN COMPUTACIÓN

VALENTINA FRANCISCA ESTEBAN LAGOS

PROFESORA GUÍA:
IVANA BACHMANN ESPINOZA

PROFESOR CO-GUÍA:
SEBASTIÁN FERRADA

MIEMBROS DE LA COMISIÓN:
NOMBRE UNO
NOMBRE DOS
NOMBRE TRES

SANTIAGO DE CHILE
2024

Resumen

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini erudito in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicatissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Quaerimus igitur, quid sit extremum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diuturnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum.

dedicatoria.

Agradecimientos

a mi por soportar?

Índice

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

En la sociedad actual el Internet juega un papel esencial en la vida cotidiana, facilitando la comunicación, la colaboración y el intercambio de información. En los últimos años, su uso y desarrollo ha crecido exponencialmente, convirtiéndose en una herramienta indispensable. Esta creciente interconexión global ha hecho que las redes sean fundamentales para el funcionamiento de la sociedad moderna. En este contexto, es crucial entender cómo está formado el Internet, ya que conocer su funcionamiento y estructura permite preservar su integridad y eficiencia. Además, garantizar un funcionamiento continuo y óptimo.\

El Internet está conformado por miles de Sistemas Autónomos (SA) interconectados entre sí. Cada SA consiste en un conjunto de IPs que comparten un mismo protocolo de enrutamiento y están administradas por una misma entidad, como proveedores de servicios de Internet (ISP), empresas comerciales, universidades, entre otros. A cada uno de estos Sistemas Autónomos se le asigna un número y prefijos de direcciones IP, los cuales anuncia a sus vecinos a través del Border Gateway Protocol (BGP). BGP es un protocolo dinámico de enrutamiento externo en el que los SA anuncian sus tablas de ruteo y cambios en sus AS-Paths para alcanzar direcciones IP específicas. De esta manera, cada SA recibe estos anuncios de todos sus vecinos BGP y toma decisiones sobre la mejor forma de direccionar sus paquetes. \

Dentro del grafo de Sistemas Autónomos que conforma Internet, el camino que un paquete recorre de un nodo a otro no suele ser el más corto debido a los acuerdos comerciales que cada SA, como entidad independiente, establece con sus vecinos. Estos acuerdos se clasifican en tres tipos de relaciones: 1) Provider-to-customer (P2C), en el cual el cliente paga al SA proveedor para que este enlace permita el tráfico de sus paquetes hacia el resto de Internet. 2) Peer-to-peer (P2P), donde los SA intercambian tráfico entre sí y con sus clientes, pero no con sus proveedores u otros pares. 3) Sibling-to-sibling (S2S), cuando dos

SA pertenecen al mismo dominio. Gao [???] propuso reglas para modelar estas relaciones entre SA, que reflejan cómo suelen configurarse en BGP, lo que permite inferir las posibles rutas seleccionadas por este protocolo. Sin embargo, estas soluciones se basan principalmente en el cálculo de heurísticas. Por otro lado, estudios más recientes como el de Shapira y Shavitt[1] proponen técnicas de Deep Learning, creando representaciones de los SA que luego son utilizadas en una Red Neuronal.\

Aquí entra en juego un nuevo enfoque que podría cambiar nuestra forma de analizar datos: el uso de Redes Neuronales de Grafos (GNNs). Las GNNs están diseñadas específicamente para trabajar con datos organizados en forma de grafos. Al ser Redes Neuronales, tienen la capacidad de encontrar representaciones efectivas de la información y descubrir patrones en datos estructurados de esta manera. A diferencia de las Redes Neuronales convencionales, las GNNs pueden aprovechar la información de los nodos vecinos, lo que les permite entender mejor la estructura del grafo y realizar análisis más detallados.\

Así nace esta tesis, con el objetivo de explorar el comportamiento de las Redes Neuronales de Grafos (GNNs) utilizando datos de BGP para representar Internet y las relaciones entre los Sistemas Autónomos que lo componen. Esta área es aún poco explorada y presenta varias dificultades, principalmente debido a la necesidad de aplicar conocimiento tanto en redes como en Deep Learning. En particular, obtener una representación precisa de la topología de Internet requiere comprender a fondo el funcionamiento del protocolo BGP y saber cómo y dónde obtener datos, que no siempre están disponibles públicamente. Así como también tener un conocimiento de teoría de grafos y Deep Learning necesario al momento de la implementación de un modelo de GNNs y diferentes técnicas para la tarea requerida.

1.1 Motivación

(está copiado el problema, es similar pero revisar de cambiar la forma en cómo se plantea)

1.2 Hipótesis

Las Redes Neuronales de Grafos (GNNs) pueden ofrecer un rendimiento superior en comparación con las metodologías del estado del arte [1] para la inferencia del tipo de relación entre Sistemas Autónomos.

1.3 Objetivos

1.3.1 Objetivo general

El objetivo principal de este estudio es evaluar diversas arquitecturas de Redes Neuronales de Grafos (GNNs) para determinar su viabilidad en la inferencia del tipo de relación de tráfico entre dos Sistemas Autónomos. Esto se logrará mediante el análisis de características específicas de cada Sistema Autónomo, la información de actualizaciones BGP, la topología y los cambios en esta.

1.3.2 Objetivos específicos

1. Obtención de datos: Recopilar datos de fuentes confiables como

que correspondan a Sistemas Autónomos representativos de la Red de Internet. Esto implica obtener datos sobre nodos, características y relaciones entre ellos. Asimismo, obtener información relevante sobre flujos de paquetes BGP.

1. Preparación de datos: Mejorar la calidad de los datos mediante el uso de técnicas de normalización, conversión de atributos categóricos a numéricos, manejo de desequilibrio de clases, entre otros.

Además, construir el grafo y definir cómo se proporcionarán los datos de entrada a nuestros modelos GNNs.

1. Diseño e implementación de modelos: Diseñar e implementar modelos GNN y framework específicos que permita la inferencia del tipo de relación que dos Sistemas Autónomos comparten.
2. Evaluación de performance: Comparar el desempeño de diferentes arquitecturas de GNNs en las inferencias, identificando los parámetros de mayor relevancia.
3. Análisis de resultados: Comprender los resultados obtenidos mediante el estudio y la comparación con los valores esperados y estado del arte [1].

1.4 Metodología

El plan de trabajo que se espera llevar a cabo durante esta investigación consta de cuatro etapas:

Investigación y familiarización En esta primera etapa, se llevará a cabo la lectura de artículos académicos relacionados con el uso de GNNs, además de artículos relevantes en la representación de datos de internet, con el objetivo de adquirir conocimiento sobre el problema en cuestión. Al mismo tiempo, se realizará un estudio detallado de datasets representativos de internet y, más importante aún, de la topología de BGP, junto con actualizaciones de estos e información adicional que se puede obtener tan-

to de sistemas autónomos como de los paquetes que intercambian. En paralelo a la investigación, se procederá al desarrollo de modelos básicos de GNNs con el propósito de familiarizarse con las herramientas que se utilizarán a lo largo del proyecto.

Preparación de datos Una vez se tenga información sobre la topología BGP, los Sistemas Autónomos que la componen y los tipos de relaciones de entre ellos, se procederá a convertir los datos a la representación de entrada que nuestro modelo recibirá. Esto también implica el uso de diversas técnicas destinadas a mejorar la calidad de los datos. El enfoque de esta etapa dependerá del estado inicial de los datos, lo que podría implicar acciones como la limpieza de datos, normalización y reducción de la variabilidad, entre otros procesos que se consideren necesarios.

Construcción de modelos y entrenamiento Una vez finalizada la investigación y la familiarización con el problema y las herramientas pertinentes, se dará inicio a la implementación de diversos frameworks y metodologías, utilizando diferentes modelos de GNNs con el conjunto de datos. Posteriormente, se procederá a entrenar los modelos y a ajustar los hiperparámetros o realizar cambios según sea necesario. Se realizará un seguimiento de los resultados, comparándolos con los hallazgos de los artículos académicos previamente revisados. Esto permitirá un proceso de mejora continua, aprendizaje y adaptación en la creación de estos modelos.

Análisis de resultados Una vez terminada la construcción de los modelos, se procederá a analizar los resultados obtenidos para finalmente empezar a escribir el informe de esta tesis.

1.5 Contribuciones

1.6 Estructura del trabajo

la tesis está organizada de la siguiente manera:

- capítuulo 2 blabla
- capítulo 3 blablabla
- capítulo 4 no existe todavía

Capítulo 2

Marco Teórico

2.1 Grafos

Un grafo (Figura 2.1) es una estructura discreta formada a partir del conjunto de vértices (también conocido como nodos) y aristas las cuales son las uniones entre estas [2]. De forma más sencilla un grafo es una representación visual de una relación binaria. Un gráfo G se caracteriza mediante la pareja de conjuntos (V, E) donde V es el conjunto no vacío de vértices y E denota el conjunto de aristas, este último es, a su vez, es un conjunto de pares de nodos. Así, la definición de un grafo está dada por $G = (V, E)$. Usamos $v_i \in V$para denotar que un nodo forma parte del grafo y $e_{ij} = (v_i, v_j) \in E$ para indicar que existe una arista entre el nodo v_i y v_j . Cada nodo v_i tiene vecinos con los cuales comparte una arista, estos se representan de la forma $N(v_i) = \{v_j \in V : (i, j) \in E\}$. El número de vertices y aristas en un grafo se representan mediante $n = |V|$ y $m = |E|$.

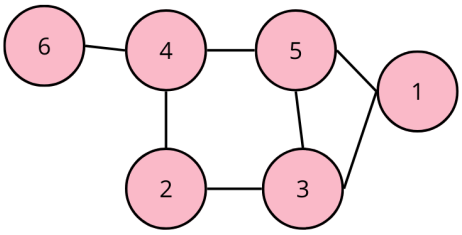

Figura 2.1: Representación de un grafo no dirigido de 6 nodos numerados.

Una forma de representar un grafo es mediante una matriz de adyacencia denotada $A \in \mathbb{R}^{n \times n}$, donde un el valor $A_{ij} = 1$ si $e_{ij} \in E$ y $A_{ij} = 0$ si $e_{ij} \notin E$. Si la matriz es simétrica, el grafo es no dirigido; de lo contrario, se trata de un grafo dirigido.

Nodos y Aristas de un grafo pueden contener atributos. De esta manera, los atributos de los nodos pueden ser representados mediante una matriz $H_n \in \mathbb{R}^{n \times d}$ donde cada fila representa un vector de características de un nodo. En el caso de los atributos de las aristas, estos pueden ser representados por la matriz de adyacencia, en la cual, en lugar de contener 1 y 0, contiene dichos atributos.

Además, los grafos pueden clasificarse en diferentes categorías que ofrecen información adicional y características distintivas. A continuación, se presentan algunas categorías comunes:

- Grafos dirigidos/no dirigidos: En un grafo dirigido, cada arista tiene una dirección específica, indicando un flujo unidireccional entre los nodos conectados. A diferencia de un grafo no dirigido, donde las aristas no tienen una orientación definida, lo que representa conexiones bidireccionales entre nodos.
- Grafos homogéneos/heterogéneos: En un grafo homogéneo, tanto nodos como aristas son del mismo tipo, en contraste de grafos heterogéneos donde los nodos y aristas pueden ser diferentes y por tanto representar cosas diferentes.
- Grafos estáticos/dinámicos: Un grafo dinámico experimenta cambios en su estructura a medida que transcurre el tiempo, a diferencia de un grafo estático, que mantiene una topología constante en función del tiempo.

2.2 Inteligencia Artificial

Inteligencia Artifial (IA) es un campo de la informatica que busca simular el comportamiento de la inteligencia humanana, es decir, intenta replicar y automatizar la capacidad del ser humano para tomar desiciones.

Dentro del área de la Inteligencia Artificial, nos encontramos con el Machine Learning, diciplina que a través del desarrollo de algoritmos y modelos busca que las máquinas aprendan patrones por medio de la experiencia, la cual incluye datos de entrenamiento y retroalimentación. El objetivo es entrenar una máquina para una tarea especifica sin la necesidad de programar explícitamente un algoritmo.

Finalmente, dentro de Machine Learning se encuentra el campo de Deep Learning, un área que se centra en el uso de arquitecturas de Redes Neuronales profundas para aprender representaciones de datos de manera jerárquica. A diferencia de Machine Learning convencional, donde las características se extraen manualmente de los datos y se proporcionan al modelo, en Deep Learning, estas representaciones se aprenden de forma automática mientras el modelo lleva a cabo la tarea asignada. Una característica distintiva de esta disciplina es el uso de Redes Neuronales, estructuras compuestas por múltiples capas entre la entrada y la salida. Cada capa procesa la información y extrae características cada vez más abstractas a medida que se profundiza en la Red. Permitiéndole al modelo así capturar patrones y características complejas en los datos.

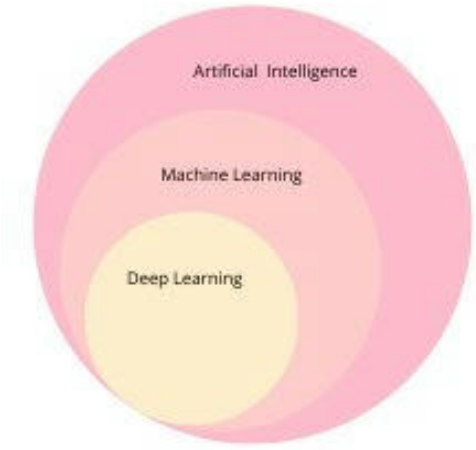

Figura 2.2: jerarquía conceptual entre Inteligencia Artifial, Machine Learning y Deep Learning.

1. Machine learning es una rama de la inteligencia artificial, que ya no depende de unas reglas y u programador, sino que la computadora puede establecer sus propias reglas y aprender por si misma
2. Aprendizaje Supervisado:

2.3 Redes Neuronales

Una Red Neuronal es un modelo computacional compuesto de neuronas (perceptrones), dispuestas en capas y conectadas entre si con el fin de aprender patrones mediante el intercambio de información ponderada por pesos. Estos pesos se ajustan en base a los datos de entrada, asignando valores en función del reconocimiento de patrones, que permiten una salida esperada.

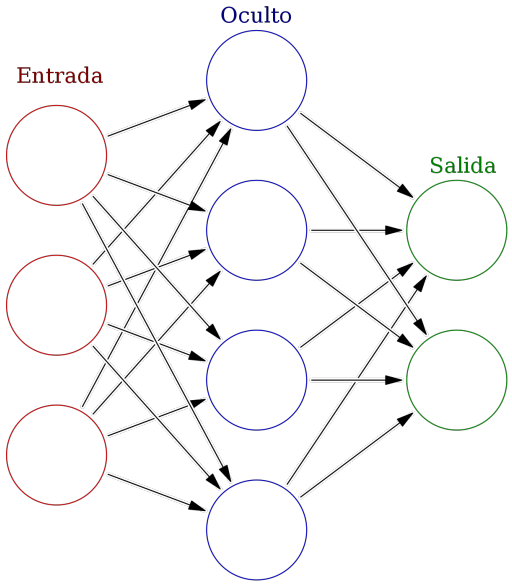

Figura 2.3: Arquitectura básica de una Red Neuronal Fully Connected de una capa.

En una Red Neuronal, cada unidad toma entradas, las pondera por separado, suma los valores y pasa esta suma a través de una función para producir una salida, la cual es compartida con otras neuronas a las cuales está conectada. El perceptrón, que funciona como una representación matemática de una unidad básica en la Red, realiza cálculos para determinar tendencias en los datos de entrada, asignándole diferentes pesos a cada valor de entrada en base a patrones entre los datos para realizar tareas específicas.

Un perceptrón está compuesto por cuatro elementos distintivos (Figura 2.4), i) los valores de entrada definidos como $x_i, x_{i+1}, ..., x_{n-1}, x_n$ donde cada x_j corresponde a un vector de tamaño d, ii) los pesos definidos como $w_j \in \mathbb{W}^{n \times d}$ donde $\mathbb{W}$ corresponde a la matriz de pesos los cuales son ajustados durante la etapa de entrenamiento de la Red, iii) la suma $z = \sum_{j=1}^{d} w_j x_j + b$ y iv) la función de activación, la cual establece un umbral de salida para evitar que los valores de salida se disparen. Esta función de activación permite incluir más capas y, por ende, mayor complejidad en las arquitecturas de redes que se construyan. Las funciones de activación tienen la capacidad de mejorar el aprendizaje de patrones en los datos[3]. Algunas de las funciones de activación comúnmente empleadas incluyen la Sigmoide, la Tangente Hiperbólica (tanh), la Rectified Linear Unit (ReLU) y la Leaky ReLU.

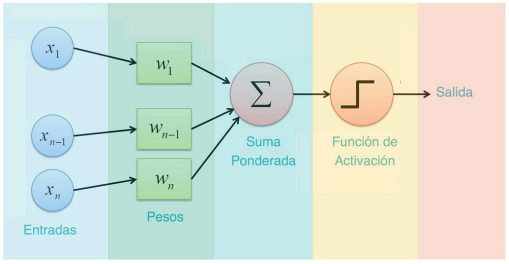

Figura 2.4: Estructura general de un perceptrón.

La técnica comúnmente utilizada para el entrenamiento de Redes Neuronales es el *back-propagation*, que tiene como objetivo ajustar los pesos de los parámetros de la Red para minimizar la función de pérdida[4]. Esta función cuantifica la diferencia entre las predicciones hechas y los valores reales. Una vez que se ha calculado la pérdida, el proceso de optimización se centra en modificar los pesos para mejorar la precisión general de la Red.

Durante el entrenamiento de las Redes Neuronales, se emplea el descenso de gradiente, un método que implica el cálculo de la derivada de la función de perdida con respecto a los pesos de la Red. Este cálculo determina la dirección y magnitud en la que los parámetros de un modelo deben ser ajustados para minimizar la función de perdida. Por ende, es fundamental que esta función sea continua y derivable. En problemas de regresión, se suele utilizar funciones como el Mean Squared Error y Mean Absolute Error, mientras que en problemas de clasificación, destaca la Cross-Entropy Loss.

2.3.1 Funciones de Activación

2.3.2 Función de Pérdida

2.3.3 Tipos de Redes Neuronales

2.3.4 Redes Neuronales Feed Forward (FFNN)

También conocida como *multilayer perceptrons*, esta arquitectura representa la forma más simple y fundamental de una Red Neuronal, sirviendo como la base de la mayoría de los modelos de Deep Learning. En esta arquitectura la información fluye exclusivamente hacia «adelante», sin bucles o conexiones hacia atrás.

El flujo de información comienza en la capa de entrada, donde se reciben los datos, seguida de las capas ocultas (*hidden layers* en ingles), siendo las Fully Connected las más comunes (Figura ???fully-connected), donde cada neurona está conectado a cada neurona

de la capa anterior. De esta manera, las salidas de cada perceptrón generan una salida que, al estar conectada con otros nodos, funcionan como entrada para la siguiente capa, continuando así hasta llegar a la capa de salida.

El objetivo principal de una Red Feed Forward es aproximar alguna función $f(x)$. Por ejemplo, en un problema de regresión, se busca modelar la relación $y = f(x)$.

2.3.5 Redes Neuronales Recursivas (RNN)

Las Redes Neuronales Recurrentes (RNN) son una variante de las Redes Neuronales Feed Forward, diferenciándose por su capacidad para retener y utilizar información previa, es decir, poseen «memoria».

A diferencia de las Redes Neuronales Feedforward convencionales, que asumen que los datos de entrada en cada capa son independientes entre sí, las Redes Neuronales Recurrentes (RNN) introducen conexiones entre las salidas previas y la salida actual, generando así un proceso de retroalimentación.

Esta característica en las RNN las hace particularmente eficientes para trabajar con datos secuenciales, como en aplicaciones de procesamiento del lenguaje natural incluyendo traducción, generación de texto y la predicción de series temporales.

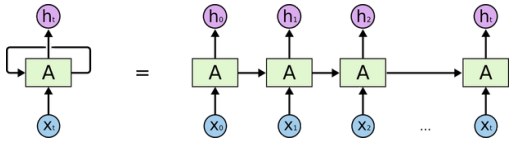

Figura 2.5: Arquitectura básica de las Redes Neuronales Recurrentes.

La imagen de arriba es una representación simple de una Red Neuronal Recurrente (Figura 2.5). En el lado izquierdo se encuentra la notación abreviada y, en el lado derecho, la notación desplegada para representar RNNs. Donde x_t es un vector que representa la entrada en el instante de tiempo t. A el estado oculto con el paso del tiempo t y actúa como la «memoria» de la Red, calculando en función del estado oculto anterior y la entrada en el paso actual.

2.3.6 Redes Neuronales Convolucionales (CNN)

Las Redes Neuronales Convolucionales (Figura 2.6) son un tipo especializado de modelo de Red Neuronal diseñado especialmente para procesar información en forma de grilla [5].

Su aplicación principal se encuentra en el análisis de imagenes, en el reconocimineto de objetos, clases y categorías.

Las CNN se componen de una capa de entrada, una capa de salida y varias capas ocultas intermedias. Estas capas ocultas llevan a cabo operaciones de convolución, lo que les permite aprender características específicas de las imágenes. En el proceso de convolución, se aplican filtros, a través de matrices de pesos. Estos filtros aprenden a detectar diversas características como bordes, patrones, colores, entre otros. Así a medida que se avanza en las capas de la CNN, la red es capaz de reconocer elementos más complejos.

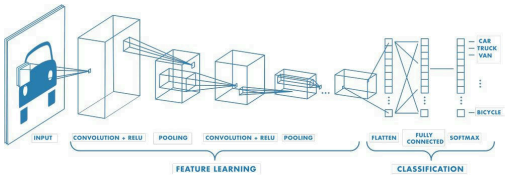

Figura 2.6: Estructura de un modelo de Convolucional con tres capas.

2.3.7 Graph Neuronla Network (GNN)

Las GNN son una arquitectura de Redes Neuronales especialmente diseñada para realizar predicciones basadas en datos representativos de grafos. A diferencia de las Redes Neuronales convencionales, las GNNs reciben datos en forma de tensores que pueden representar nodos, atributos de nodos, aristas y atributos de aristas.

Existen diferenets enfoques, dependiendo de la tarea de aprendizaje que se quiere llevar a cabo, estos son:

- Nivel de nodo: Incluye tareas como clasificación, regresión y clustering de nodos. Se realizan inferencias a partir de las conexiones con otros nodos.
- Nivel de aristas: Se abordan tareas de clasificación y predicción de aristas. Por ejemplo, determinar la existencia de una relación entre dos nodos.
- Nivel de grafo: Se encuentran tareas de clasificación, regresión y matching de grafos para las cuales el modelo debe ser capaz de aprender una representación para el grafo completo.

Las GNN tienen una serie de ventajas sobre las Redes Neuronales convencionales cuando se trabaja con datos de grafos. En contraste con los modelos tradicionales, las GNN aprovechan las relaciones entre las entidades que conforman los datos de entrada a el modelo. Estas relaciones pueden incluir aspectos como orden, jerarquía, dependencias o relaciones

de otro tipo que son comunes en grafos y se representan a través de las aristas que conectan los nodos.

En cuanto a la adaptabilidad a variaciones en el tamaño de entrada, las Redes Neuronales convencionales requieren que los datos de entrada mantengan un mismo tamaño. Para ello, recurren a técnicas como padding o broadcast, los cuales no tienen efectos significativos en el desempeño de los modelos. Las GNNs, por su parte, ofrecen flexibilidad para adaptarse a distintos tamaños de entrada[6].

Otro motivo para optar por GNNs es su capacidad para manejar el isomorfismo de los grafos, es decir dos grafos pueden lucir diferentes, pero ser estructuralmente iguales. Un modelo tradicional trataría ambos grafos como si fuesen datos diferentes, sin embargo, no lo son. Esto es comparable a lo que sucedería si se le presenta como entrada dos imágenes donde una se encuentra invertida. Es por esta razón que no se puede trabajar directamente con una matriz de adyacencia en una Red Feed Forward, ya que es sensible a estos cambios. Las GNNs utilizan técnicas que son invariantes ante permutaciones, lo que permite trabajar con el isomorfismo en grafos.

Finalmente, el último desafío radica en que la estructura de un grafo no puede ser reducida a un espacio euclidiano, y su conceptualización no puede limitarse a una distancia euclidiana[7]. A diferencia de Redes Neuronales que trabajan, por ejemplo, con imágenes, las cuales pueden ser interpretadas como un grafo, la representación de la información se puede entender en términos de píxeles en un espacio bidimensional.

Así, la esencia detrás del uso de GNNs radica en su capacidad de entrenar un modelo que pueda procesar un grafo, sus nodos y relaciones, logrando identificar patrones relevantes en la topología para lograr de forma efectiva la tarea asignada. Por ejemplo, en el ámbito de las redes sociales, las GNNs pueden ser utilizadas para clasificar usuarios según sus interacciones, identificando así grupos afines. Otra aplicación puede ser la recomendación de contenido de interés de un usuario, basándose en sus conexiones y preferencias históricas. En el campo de la biología, es posible predecir el tipo de moléculas basándose en sus características estructurales y propiedades.

El diseño de una GNN se hace por medio de la combinación de diferentes módulos:

- Módulo de propagación: Este módulo se utiliza para propagar información entre los nodos capturando tanto la topología como los atributos de los nodos. Esto se logra combinando los datos de cada nodo con los de sus vecinos.

- Módulo de muestreo: Cuando los grafos son muy grandes, se utiliza generalmente un módulo de muestreo con el fin de seleccionar un subconjunto del grafo, aportando de este modo en la capacidad de generalización de un modelo y reducción de complejidad. Se combina generalmente con un módulo de propagación.
- Módulo de pooling: Cuando se necesita representaciones de subgrafos su utiliza este módulo para extraer información de los nodos. Se utiliza para reducir la dimensionalidad de las representaciones de nodos.

Un modelo GNN se construye generalmente combinando estos módulos. A continuación (Figura 2.7), se ilustra el pipeline de una arquitectura GNN. El modelo recibe como entrada un grafo, y en la capa GNN, se emplea un operador convolucional, un módulo de muestras y una operación skip-connection que se fusionan para propagar la información y extraer detalles de alto nivel mediante el módulo de pooling. Después de pasar por todas las capas intermedias, se obtiene una salida en forma de embeddings, a los cuales se les aplica una función de pérdida para obtener los resultados del ajuste del modelo en base a la tarea asignada.

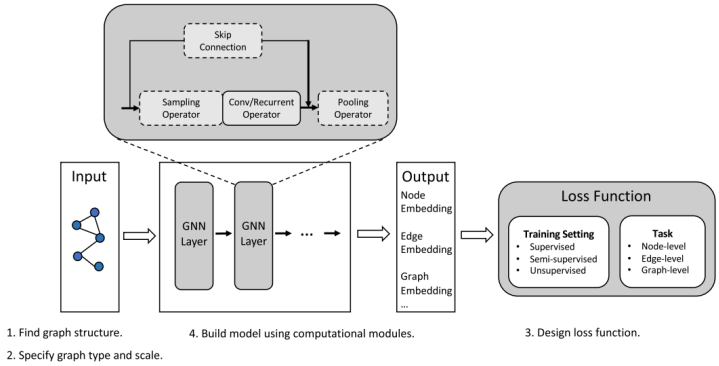

Figura 2.7: Pipeline básico de una arquitectura GNN.

2.3.7.1. Message Passing Neural Networks (MPNN)

Es la arquitectura de Red Neuronal para grafos más utilizada. Su funcionamiento radica en la idea de que cada nodo en un grafo puede intercambiar información con sus vecinos de manera que cada nodo podrá actualizar su representación en base a la información acumulada por su entorno.

La información se propaga entre nodos a través de mensajes. Cada nodo envía mensajes a sus nodos vecinos y recibe mensajes de ellos. Estos mensajes pueden contener información sobre el nodo emisor y se utilizan para actualizar la representación del nodo receptor[8].

Se emplea un mecanismo denominado *message passing*, el cual consta de tres pasos:

1. Propagacion de mensajes entre nodos: Cada nodo envía un mensaje que contine su representación actual a sus nodos vecinos.
2. Aplicación de una función de agregación: Luego de la propagación de mensajes, se aplica una función de agregación para combinar la información recibida de los nodos vecinos.

Esta función puede adoptar diversas formas como la suma o la media.

1. Actualización de la representación: La representación de cada nodo se actualiza mediante la información agregada proveniente de sus nodos vecinos, así como a partir de su representación previa.

A continuación (Figura 2.8), se presenta el comportamiento de una capa de MPNN para un nodo. El proceso inicia con el envío de un mensaje M por parte de cada nodo vecino de B. B recibe estos mensajes y los agrega mediante una operación generando una representación A. Finalmente, el nuevo estado del nodo B se calcula mediante una última función que toma el valor A y su propia representación para crear su nueva descripción U.

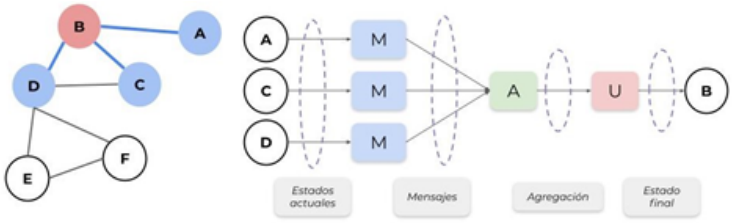

Figura 2.8: Flujo de Message passing para un nodo del grafo.

2.3.7.2. Graph Convolution Network (GCN)

Es un tipo específico de MPNN, donde se utilizan convoluciones de grafos para agregar información de los nodos adyacente de un nodo en un grafo.

La operación de convolución en el grafo produce la suma normalizada de las características de los nodos vecinos[9]. Esta normalización garantiza que la información agregada sea ponderada correctamente, es decir, evitar que un nodo con gran cantidad de vecinos tenga una representación desproporcionada y que luego tenga una influencia mayor en la representación otros nodos en las siguientes capas.

La notación de los embeddings de los nodos $h_i^{(l+1)}$ en la capa $l+1$ de la red está dado por:

$$h_i^{(l+1)} = \sigma\left(b_i^{(l)} + \sum_{j \in N(i)} \frac{1}{c_{ji}} h_j^{(l)} W^{(l)}\right) \tag{2.1}$$

Donde $N(i)$ es el conjunto de nodos vecinos del nodo i, $c_{ji} = \sqrt{|N(j)|}\ \sqrt{|N(i)|}$ y σ la función de activación, $b_i^{(l)}$ corresponde al sesgo en la capa l de la red.

2.3.7.3. Graph Attention Network (GAT)

Otra variante de MPNN son las Graph Attention Networks (GAT). A diferencia de una Red Neuronal de Convolución, GAT incorpora un mecanismo de atención que permite que cada nodo pondere de forma diferenciada, indicando la importancia de las representaciones de cada vecino para la actualización de las características de un nodo[10].

Los coeficientes se calculan por un mecanismo el cual calcula un puntaje para cada par de nodos. Luego estos puntajes se normalizan por medio de la función SoftMax (Figura 2.9).

Así tenemos:

$$z_i^{(l)} = W^{(l)} \boldsymbol{h_i^{(l)}} \tag{2.2}$$

$$e_{ij}^{(l)} = \text{LeakyReLU}\Big(\boldsymbol{a}^{(l)T}\Big(\boldsymbol{z}_i^{(l)} \parallel \boldsymbol{z}_j^{(l)}\Big)\Big) \tag{2.3}$$

$$\alpha_{ij}^{(l)} = \frac{e_{ij}^{(l)}}{\sum_{\{k \in N(i)\}} \exp\Big(e_{ik}^{(l)}\Big)} \tag{2.4}$$

$$h_i^{(l+1)} = \sigma\left(\sum_{jinN(i)} \alpha_{ij}^{(l)} z_j^{(l)}\right) \tag{2.5}$$

FIXME: Arreglar label pasos ecuaciones Donde (2 gat) es la transformación lineal del embedding de la capa anterior $h_i^{(l)}$ con $W_i^{(l)}$ una matriz de pesos entrenable.

La ecuación (FIXME:3) calcula un puntaje de atención entre dos vecinos. Primero concatena los embeddings z de dos nodos. Luego realiza el producto punto entre este y una matriz entrenable $a^{(l)}$ y aplica una función LeakyReLU al final.

En (FIXME:4) se aplica una función softmax, con el objetivo de normalizar los puntajes de atención en las aristas entrantes de cada nodo.

Finalmente, en la ecuación (FIXME:5), al igual que en GCN, se lleva a cabo la agregación de los nodos vecinos, pero en este caso, se escala por el puntaje de atención. Se utiliza σ como la función de activación que se aplicará a la capa.

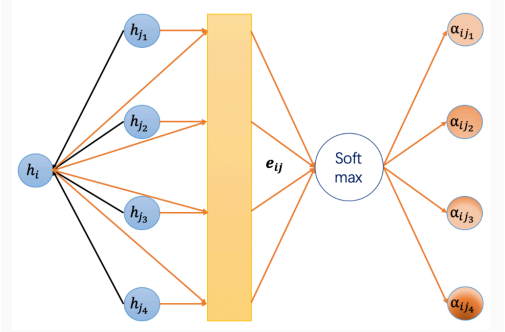

Figura 2.9: Pipeline del mecanismo de cálculo de las ponderaciones para una GAT.

2.3.7.4. GraphSAGE

A diferencia de métodos anteriores que requerían la representación completa de un grafo en la memoria, GraphSAGE puede aprender a generalizar a grafos de diferentes tamaños y estructuras, lo que lo hace particularmente útil para aplicaciones en redes sociales, biología, recomendación, etc.

GraphSAGE [11]

2.3.8 Procesamiento de Datos

2.3.9 Entrenamiento

Existen 2 approach oara llevar a cabo el entrenamiento de una GNN, estos son:

- Inductive Learning: Se entrena el modelo en un subconjunto de nodos y luego se evalúa en un conjunto de nodos no vistos.
- Transductive Learning: Se entrena el modelo en todo el grafo y luego se evalúa en un subconjunto de nodos.

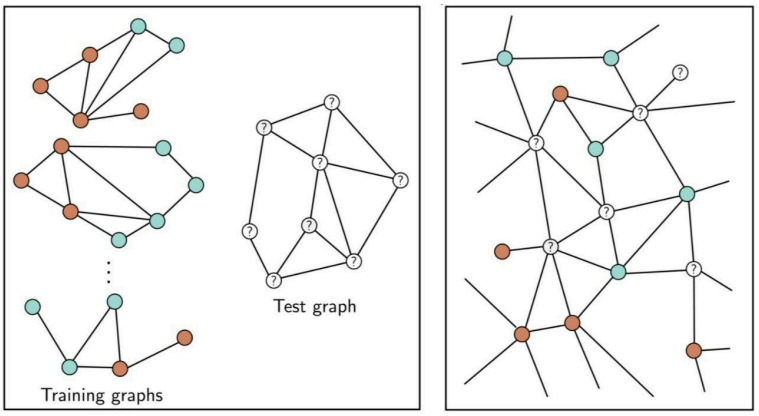

Figura 2.10: Inductive y transductive settings para entrenar y testear un modelo GNN.

En el caso de la tarea de classificación de nodos, en el enfoque inductivo, se entrena el modelo en un subconjunto de nodos y luego se evalúa en un conjunto de nodos no vistos. En cambio para el enfoque transductivo, Se tiene un solo gran grafo de donse un subconjunto de este es seleccionado para entrenar el modelo y el resto para testearlo.

2.3.9.1. Optimización

El entrenamiento de uan Red tiene como fin encontrar los pesos de los parametros que minimicen la función de perdida. Este error calcula la diferencia entre los valores que tira la Red Neuronal y los correctos. Con este error se realiza lo uqe se conoce como *backpropagation* que consiste en calcular el gradiente de la función de perdida con respecto a los pesos de la Red. Este gradiente se utiliza para actualizar los pesos de la Red de forma que se minimice la función de perdida.(Buscar referencia)

Para backpropagation se utiliza el método del gradiente con el fin de optimizar los parámetros de la red y encontrar el menor error posible. Para esto debemos calcular la derivada del error respecto a cada parámetro. Esta derivada nos permitirá movernos en la dirección de la pendiente en bajada y así disminuir el error.

Algunos optimizadores Comunes:

- **Descenso del Gradiente con Momentum**:

Introduce un término de «momentum» en el cálculo del gradiente para evitar oscilaciones y hacer que el proceso de optimización sea más suave y eficiente. Ayuda a superar barreras locales en la función de pérdida.

- **Descenso de Gradiente Estocástico con Nesterov Momentum**:

Variante del descenso de gradiente con momentum que realiza una «anticipación» antes de calcular el gradiente. Mejora la velocidad de convergencia en algunas situaciones.

- **Adam**: Adaptive Moment Estimation,Ajusta dinamicamente la tasa de aprendizaje para cada parametro , utilizando el momentum de primer y segundo orden de los gradientes.
- **RMSprop**: Root Mean Square Propagation, Ajusta la tasa de aprendizaje de forma adaptativa para cada parametro, utilizando el promedio de los cuadrados de los gradientes.

Existen diferentes tipos de optimización que se pueden utilizar para entrenar una Red Neuronal, estos son:

- **Stochastic Gradient Descent**: Para cada elemneto se calcula el gradiente y se realiza un update de los pesos. Cada update esta basado en el gradiente calculado de u nodo seleccionados de forma aleatoria del dataset.

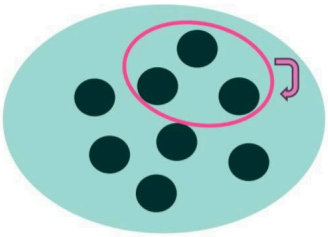

Figura 2.11: Stochastic Gradient Descent.

- **Batch Gradient Descent** : Para todo el dataset se calcula u promedio del gradiente y lueo se realiza el update de los pesos. El datset entero se usa en cada iteración del entrenamiento. El pesos de los parametros se update una vez por cada epoch. Hay un risego de overfittig ya que el modelo es expuesto de forma reptida eln el mismo orden.

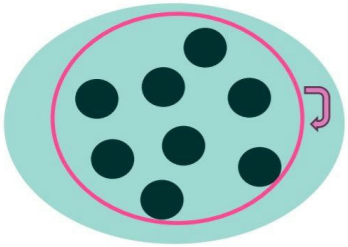

Figura 2.12: Batch Gradient Descent.

- **Mini Batch Gradient Descent**: Se divide el dataset en pequeños subconjuntos y se calcula el gradiente para cada uno de ellos y se realiza el update de los pesos.

Consiste en subdividir el dataset en sets más pequeños llamados mini-batches. El peso de los parametros se actualiza una vez por cada mini-batch. Se introduce un hiprpaametro para estre caso orrespondiente al tamaño del mini-batch.

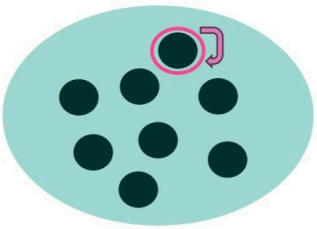

Figura 2.13: Batch Gradient Descent.

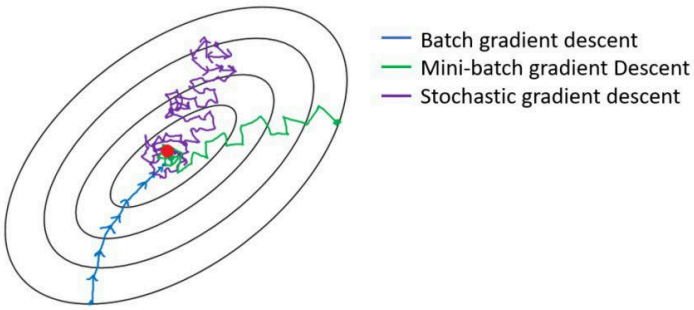

Figura 2.14: Tipos de Entrenamineto.

2.3.9.2. Sampling

Sampling (muestreo en español) en MAchine Learning coresponde a la tecnica utilizada para seleccionar subconjuntos de datos para entrenar o evaluar un modelo, en ve de utilizar el conjunto de datos completo. Esta tecnica se usa con le fin de : Cuando se trabaja con dataset muy grandes, como por ehjemplo..., es computacionalmente costoso porcesar todos los datos en cada iteración del enrenamiento, otra razon es la generalización, al muestrear diferentes subconjuntos de datos en diferentes iteraciones, el modelo tiene más probabilidades de generalizar de forma correcta y no sobreajustar los datos. En GNN, el sampling es esencial debido a la naturaleza estructurada y muchas veces masiva de los grafos. existen varias tecnicas especificas par atrabajar el sampling en este tipo de Redes Neuronales.

- **Random Node Sampling**: Se selecciona de forma aleatotia un subset de nodos del grafoo completo. Reduce el costo computacional en comparacion a entrenar todos los nodos de un grafo. Hay redundancia al calcular los embeddings si esque dos nodos comparten el mismo vecino, el embedidng de dicho nodo será calculado dos veces.

- **Neighbor Sampling**: Se selecciona un numero expecifico de vecinos para cada nodo en cada capa de la Red. Esto evita

PAPERS: GraphSAGE[11], PAPERS: PinSAGE[12], [13]

- **Layer Sampling**: Realiza un uestreo por capas de forma aleatoria e indeoendetie en tre capas. Sin embaro esto puede causr tener nodos aisaldos. Tiene como objetivo evitar el calculo redundante en el muestrei po rnodos. Permute un uso más eficiente de memoria.

Ejemplo de Papers FstGCN[14], Adaptative Sampling[15], LADIES[16]

- **Subgraph Sampling**: Extrae subgrafos de manera aleatoria o divide el grafo original en subgrafos. Estos se entrenan como muestras de datos independientes. Reduce el tamaño significativamente de la data que la GNN tiene que procesar en cada iteracion.

PAPERS: GraphSAINT [17], ClusterGCN [18]

¿Qué es Bacth Normalization? Tien el fin de estabilizar el training d ela GNN. Reescala cada salida de las layers ara que su promedio y varianza a traves del batch B

$$m_h = \frac{1}{|B|} \sum_{i \in B} h_i \tag{2.6}$$

$$s_h = \sqrt{\frac{1}{|B|} \sum_{i \in B} (h_i - m_h)^2} \tag{2.7}$$

Se calcula el mean y variance sobre |B| embeddings en un batch B.

$$h_i \longleftarrow \frac{h_i - m_h}{s_h + \epsilon} \tag{2.8}$$

$$h_i \longleftarrow \gamma h_i + \delta \tag{2.9}$$

Normalizar los node embeddingsusando el mean y variance calculado en el paso anterior.

Beneficios de Batch Normalization: Podekos ocupar learn rates más grandes. BN hace que cambie la loss de forma más smoothly lo que conlleva a ques epuedan ocupar learning rates más grandes.

Por otro lado BN injetca ruido al proceso de training porque la onormalizacion depende de las estadistcias de todo el batch. Nos permite controlar la varianza de los embeddings aprendidos en re capas consecutivas de la GNN..

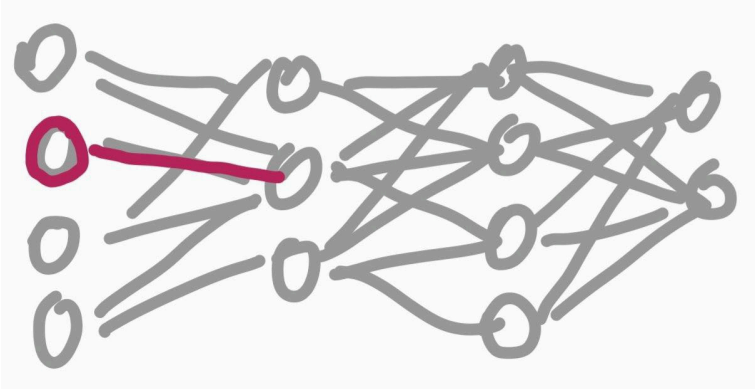

Figura 2.15: Explciacion FIXME:.

¿Qué pasa si este weigh termina siendo muy grande (encomparación a otros)? Hara que la salida d ela neurona sea muy grande, afectando asi a la siguente capa, causando inestabilidad en el training. Nomlamente se normaliza los inputs y entonces por que no normlaizar entre capas?

- Acelera el entrenamiento (podemos ocupar lr mas grandes)
- Disminuye la imortancia de los pesos iniciales
- Regulariza el modelo (un poquito)

Entonces la idea es qu elos pesos no sean muy grandes o muy chicos y asi no se ve aafectada al estabilidad del modelo.

function BATCH NORMALIZATION(X, gamma, beta, epsilon)
    ▷ Calculo mean mini-batchten
    $\mu_{\mathrm{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$
    ▷ Calculo varianza del mini-batch
    $\sigma_{\mathrm{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} \left(x_i - \mu_\beta\right)^2$
    ▷ Normalizar
    $\widehat{x_i} \leftarrow \frac{x_i - \mu_{\mathrm{B}}}{\sqrt{\sigma_{\mathrm{B}}^2 + \varepsilon}}$
    ▷ Scale and shift
    $y_i \leftarrow \gamma \widehat{x_i} + \beta$

2.3.9.3. Regularización

Una de las metas que s etiene al momento de enrenar un modelo es evitar el overfitting y por ende que pueda generalizar los resultados. Es decir logre classificar correctamente un dato nunca visto anteriormente.

Existen diferentes tecnicas para lograr esto como :

- **Dropout**: Es una tecnica que desactiva un número de neuronas de forma aleatoria.Para aplicar este método se asigna una probabilidad a cada neurona de ser desactivada en la fase de entrenamiento. Esto quiere decir que las conexiones que tenía esa neurona desaparecerán momentáneamente.

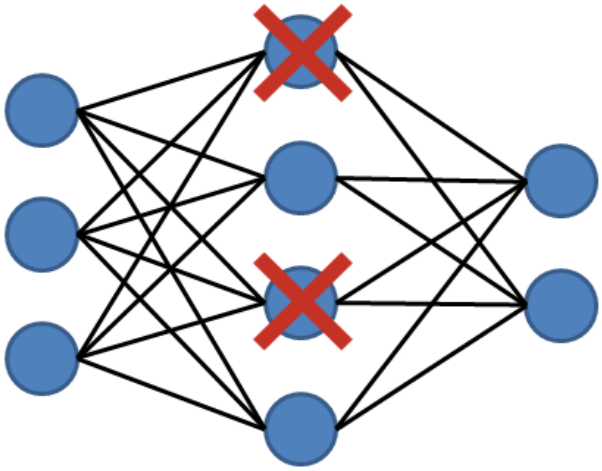

Figura 2.16: Dropout.

Para el caso de GNN En training:De forma aleatoria se seleccionan neuronas a 0 (drop out) con una probabilidad de p e una capa especifica. En Tesing: Se usan todas las neurionas. PAPER: [19]

- **Early Stoping**:
- **L1 y L2 Regularization**: blablabla
- **Data Augmentation**: blablabla
- **Batch Normalization**: blablabla

2.3.10 Evaluación

2.3.10.1. Metricas de evaluación

- True Positive (TP):
- True Negative (TN):
- False Positive (FP):
- False Negative (FN):

A partir d eestos valores podemos crear la matriz de consusión para clase que s equiere clasificar. Esta es …

Figura 2.17: Matriz de confusión.

Existen diferentes etricas para medir el desempeño de lso modelos estos son:

- **Accuracy**:
- **Precision**:
- **Recall**:
- **F1-Score**:

2.4 Internet

Antes de abordar la definición de Internet, es crucial comenzar definiendo qué es una Red. Se trata de un conjunto de computadoras conectadas entre sí, que posibilita el intercambio de datos. En este contexto, una red puede ser visualizada como un grafo, donde los nodos representan los computadores y las aristas simbolizan las conexiones entre ellos que permiten el envío de mensajes.

En el caso de Internet, se define como una extensa red creada mediante la interconexión de redes más pequeñas coonocidas como Sistemas Autónomos (AS), los que consisten en grupos de máquinas interconectadas que comparten un mismo protocolo de enrutamiento. Estos AS están gestionadas por diversas organizaciones, que pueden ser desde proveedores de acceso a Internet (ISP), grandes empresas tecnológicas, universidades o incluso agencias gubernamentales. A cada AS, se le asigna un número único conocido como ASN, el cual es utilizado parar identificar al AS. Además, a cada AS se le asigna un conjunto de direcciones IP del cual es responsable que le lleguen los paquetes.

La conexión entre computadoras y redes en Internet se establece a través de cables, ondas de radio y otras formas de infraestructura. Los datos transmitidos por estas conexiones se traducen en bits, los cuales, al ser leídos por una computadora, se descodifican y se interpretan como un mensaje.

Es relevante destacar que los datos transmitidos a través de redes informáticas se dividen en paquetes, que pueden ser subdivisiones de paquetes más grandes. Cada paquete consta de dos partes: el mensaje en sí y el encabezado. Este último contiene información importante, como las direcciones de destino y origen, así como valores que indican si es un paquete es un fragmento de un mensaje más grande, entre otros detalles. El encabezado de un paquete permite que, una vez que el paquete alcanza su destino, pueda ensamblarse adecuadamente si es necesario.

2.4.1 Ruteo

El ruteo consiste en la elección de caminos que seguirá un paquete dentro de una red, con el propósito de garantizar que la información que se transmite por Internet pueda llegar a su destino mediante la ruta más eficiente. Una red está formada por múltiples maquinas a las cuales se les llama nodo y las rutas que las unen. La comunicación entre dos nodos de la red se puede establecer mediante la interconexión de diferentes caminos, permitiendo así, conectar dos nodos que no tienen una conexión directa por medio de nodos intermedios. De esta forma el enrutamiento es el proceso de seleccionar la mejor ruta entre estos nodos en base a algún parámetro o reglas.

Un enrutador o router es un dispositivo de red que se conecta a otros dispositivos y redes. Son los encargados de seleccionar las rutas que irán tomando los datos enviados.

El ruteo opera gracias a las tablas de rutas presentes en los routers y a la información proporcionada en los encabezados de los paquetes, los cuales contienen datos sobre el destino del paquete. Cuando llega un paquete a un router, se consulta la tabla de enrutamiento

para localizar la dirección destinos, y posteriormente dirigir el paquete al próximo router o punto de red.

Para ilustrar esto, supongamos un usuario accede a una página web desde su hogar. En este escenario, los paquetes viajan desde el computador hasta el router de su casa. Este router luego examina el encabezado del paquete para identificar el destino final en su tabla de rutas y lo envía al siguiente punto en red. Este nuevo punto será el encargado de realizar nuevamente el proceso de redirigir el paquete. Este procedimiento se repite en todos los routers hasta que finalmente el paquete llega al destino final.

Existen dos tipos de enrutamiento: estático y dinámico. El enrutamiento estático implica el uso de tablas estáticas, las cuales deben ser modificadas manualmente para cambiar su configuración. Es útil en situaciones donde los parámetros de red permanecen constantes. Por otro lado, en el enrutamiento dinámico, los routers se encargan de ir actualizando las tablas de enrutamiento en tiempo real, ajustándolas según las condiciones de la red. Este proceso se lleva a cabo mediante los protocolos de enrutamiento.

2.4.2 Ruteo Interno

Se encarga de gestionar las rutas a seguir de un paquete dentro de un Sistema Autónomo. En este contexto los routers ocupan protocolos de enrutamiento interno para intercambiar la información del estado de la red y las rutas disponibles. Entre los protocolos de ruteo interno se tiene:

- OSPF (Open Shortest Path First): Utiliza el algoritmo de Dijkstra para determinar las rutas más cortas entre nodos[20].
- RIP (Routing Information Protocol): Utiliza un enfoque de vector de distancia para calcular la ruta más optima, basándose en la cantidad de saltos[21].
- EIGRP (Enhanced Interior Gateway Routing Protocol):
- IS-IS (Intermediate System to Intermediate System):

2.4.3 Ruteo Externo

Se centra en la gestión de rutas entre los diferentes Sistemas Autónomos que conforman el Internet. En este caso, se usan protocolos de enrutamiento externo, que al igual que los protocolos de enrutamiento interno se encarga de intercambiar la información de las rutas disponibles, permitiendo así que paquetes viajen de manera más efectiva. Algunos protocolos de enrutamiento externos son:

- BGP (Border Gateway Protocol): Tiene un enfoque de vector de distancia. Utiliza un enfoque de vector de distancia y toma decisiones basadas en políticas de red para intercambiar información eficientemente[22].
- IS-IS (Intermediate System to Intermediate System): Protocolo de enrutamiento de estado de enlace, similar a OSPF[23].

2.4.3.1. Border Gateway Protocol (BGP)

El Border Gateway Protocol (BGP) es un protocolo de enrutamiento externo, que se utiliza para intercambiar información de rutas entre los diferentes Sistemas Autónomos que conforman el Internet (Figura 2.18). Toma decisiones de enrutamiento basadas en políticas, reglas de Red y el camino más corto (AS PATH)[22].

El protocolo BGP comienza con un handshake, el cual se hace entre dos vecinos BGP, donde los AS se ponen de acuerdo en cuanto a configuraciones y soporte que tendrán por ejemplo si soportaran IPv4 o IPv6 o ambas. Una vez establecida la conexión estas intercambian información mediante UPDATES donde se pueden agregar o quitar caminos. De este modo, los vecinos actualizarán sus tablas de rutas y propagarán estos mismos cambios a sus vecinos.

De este, modo BGP elige caminos mediante la determinación del camino más corto según saltos entre AS apara llegar a su destino, sin embargo, esta métrica para determinar el siguiente salto de un paquete no toma en cuenta factores como congestión o velocidad de conexión al momento de seleccionar una ruta.

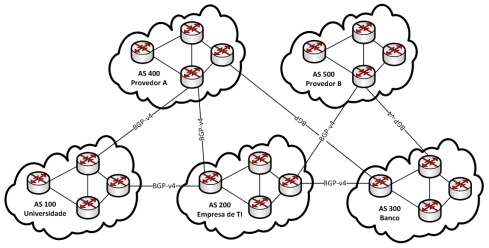

Figura 2.18: Grafo de una red con 5 Sistemas Autónomos con 4 direcciones IP cada uno.

Capítulo 3

Datos

3.1 RIPE NCC

Regional Internet Registry for Europe, middle East an dCentral Asia

RIS (Routing Information Service) is a public service that provides information about the global Internet routing system. The service is operated by the RIPE NCC, one of the five Regional Internet Registries (RIRs). The RIS service is made up of a number of route collectors and a data repository. The route collectors collect BGP routing data from different parts of the Internet. The data repository stores this data and makes it available to the public in various formats.

Route collectors are the physical machines where RIS ingests BGP routing data. We receive this data via BGP peering sessions. Most route collectors collect data from peers at IXP peering LANs that the route collectors are physically attached to. We also have "multi-hop" route collectors, which collect BGP data from peers via BGP multi-hop sessions. The advantage of multi-hop BGP sessions is that data collection is not restricted to networks that are attached to the same peering LANs as the RIS collector. Instead, these sessions can be established with us from all over the world.

Or route collectors have names that start with "RRC" and end with a number. Starting at 00 and (as of this writing) up to 26. We typically add a small number of route collectors every year.

- LOS COLLECTOS SE ENCUENTRA EN ANEXO

La dat ase puede obtener de diferentes maneras:

- La RAW daa en MRT FIles. Se colecta dos tipo de data *dumos* y *updates*. Los *dumps* son una instantanea de la tabla de ruteo de un router, mientras que los *updates* son los mensajes BGP que se envian entre routers para informar de cambios en la tabla de ruteo.
- RIS Live: Es un servicio que permite a los usuarios conectarse a un colector de ruteo en tiempo real y recibir actualizaciones BGP en tiempo real.

(URL : https://ris-live.ripe.net/)

3.2 ROUTE VIEWS

University of Oregon RouteViews Project

RouteViews is collecting BGP Updates at the following locations

- PAPERS: https://www.routeviews.org/routeviews/index.php/papers/
- Collectors: SE ENCUENTRA EN ANEXO

3.3 CAIDA

- Center for Applied Internet Data Analysis based at the University of California's San Diego Supercomputer Center

es una organización de investigación ubicada en el San Diego Supercomputer Center (SDSC) de la Universidad de California, San Diego

CAIDA se dedica a analizar, recolectar y compartir datos sobre el comportamiento de la infraestructura de Internet con el fin de mejorar la comprensión y la estabilidad de la red global.

Que recolecta/tiene:

- Invetsigaciones
- Recoleccion de datos
- Herramientas y plataformas d emonitorea

Capítulo 4

Clasificaction tipo de relaciones entre Sistemas Autónomos

Capítulo 5

modelos

Para bordar este problema se utiulizaron .....

Capítulo 6

Experimentos

Para abordar este problema se tomarón dos enfoque diferentes.

- Clasificación Binaria: Se tomaron p2c y c2p como una misma clase y las otras siendo p2p.
- Clasificación Multiclase: Las relaciones p2c y c2p se tomaron como clases diferentes.

6.1 Experimento 1:

6.2 Experimento 2:

6.3 Resultados y Analisis

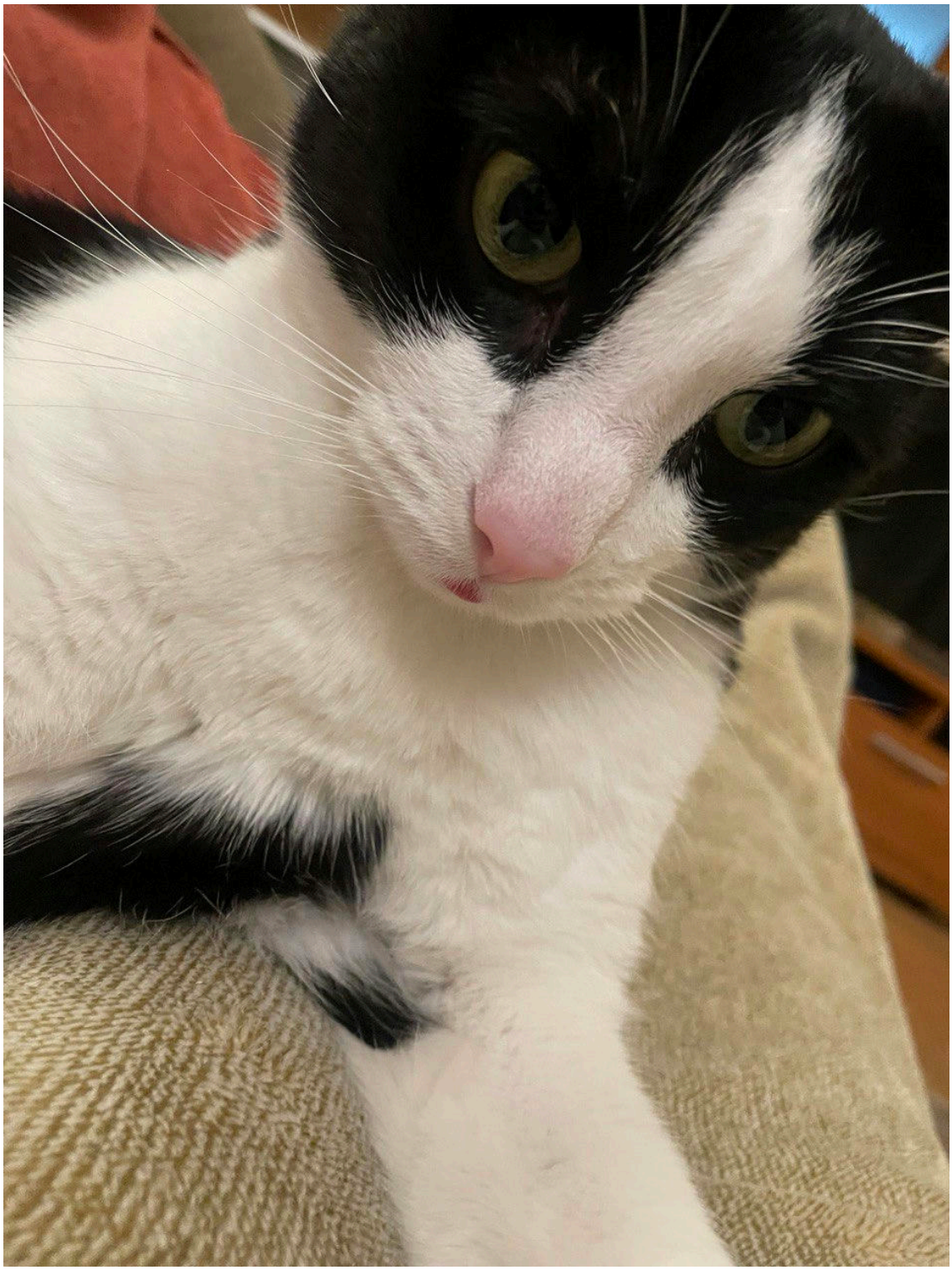

Figura 6.1: El gatito más bello del mundo.

Bibliografía

[1] Y. S. Tal Shapira, «Unveiling the Type of Relationship Between Autonomous Systems Using Deep Learning», *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020.

[2] S. H.,. Z.,. Y. Z. L. L. W. C. L. M. S. Jie Zhou Ganqu Cui, «Graph neural networks: A review of methods and applications», 2018.

[3] W. I. A. G. S. M. Nwankpa C., «Activation Functions: Comparison of trends in Practice and Research for Deep Learning. », 2018.

[4] G. E. H. R. J. W. Rumelhart D. E., «Learning representations by back-propagating errors», 1986.

[5] A. C. I. Goodfellow Y. Bengio, *Deep Learning*. MIT Press, 2016.

[6] J. A.-I. R. G.-B. T. H. A. A.-G. R. P. A. David Duvenaud Dougal Maclaurin, «Convolutional Networks on Graphs for Learning Molecular Fingerprints», 2015.

[7] Y. L. A. S. P. V. Michael M. Bronstein Joan Bruna, «Geometric deep learning: going beyond Euclidean data», *IEEE Signal Processing Magazine*, 2017.

[8] P. F. R. O. V. G. E. D. Justin Gilmer Samuel S. Schoenholz, «Neural Message Passing for Quantum Chemistry», 2017.

[9] M. W. Thomas N. Kipf, «SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS», 2017.

[10] A. C. A. R. P. L. Y. B. Petar Velickovic Guillem Cucurull, «GRAPH ATTENTION NETWORKS», *International Conference on Learning Representations*, 2018.

[11] R. Y. William L. Hamilton y J. Leskovec, «Inductive Representation Learning on Large Graphs», 2017.

[12] K. C. P. E. W. L. H. Rex Ying Ruining He y J. Leskovec, «Graph Convolutional Neural Networks for Web-Scale Recommender Systems», 2018.

[13] J. Z. Jianfei Chen y L. Song, «Stochastic Training of Graph Convolutional Networks with Variance Reduction», 2017.

[14] T. M. Jie Chen y C. Xiao, «FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling», 2018.

[15] Y. R. Wenbing Huang Tong Zhang y J. Huang, «Adaptive Sampling Towards Fast Graph Representation Learning», 2018.

[16] Y. W. S. J. Y. S. Difan Zou Ziniu Hu y Q. Gu, «Layer-Dependent Importance Sampling for Training Deep and Large Graph Convolutional Networks», 2019.

[17] A. S. R. K. Hanqing Zeng Hongkuan Zhou y V. Prasanna, «GraphSAINT: Graph Sampling Based Inductive Learning Method», 2019.

[18] S. S. Y. L. S. B. Wei-Lin Chiang Xuanqing Liu y C.-J. Hsieh, «Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks», 2019.

[19] L. F. Pál András Papp Karolis Martinkus y R. Wattenhofer, «DropGNN: Random Dropouts Increase the Expressiveness of Graph Neural Networks», 2021.

[20] J. Moy, « OSPF Version 2», *RFC 2328, https://www.rfc-editor.org/rfc/rfc2328.html*, 1998.

[21] G. Malkin, « RIP Version 2», *RFC 2328, https://datatracker.ietf.org/doc/html/rfc 2453*, 1998.

[22] E. T. L. E. S. Hares Ed. Y. Rekhter, «A Border Gateway Protocol 4 (BGP-4).», *RFC 4271*, 2006.

[23] «IS-IS for IP Internets (isis), http://www.ietf.org/html.charters/isis-charter.html».

Anexo A

titulo anexo 1

Collectors RIPE NCC:

Nombre	Ubicación	Typo	Sponsors
RRC00	Amsterdam, NL	multihop	RIPE NCC
RRC01	London, GB	IXP	LINX, LONAP
RRC03	Amsterdam, NL	IXP	AMS-IX, NL-IX
RRC04	Geneva, CH	IXP	CIXP
RRC05	Vienna, AT	IXP	VIX
RRC06	Otemachi, JP	IXP	DIX-IE, JPIX
RRC07	Stockholm, SE	IXP	Netnod
RRC10	Milan, IT	IXP	MIX
RRC11	New York, NY, US	IXP	NYIIX
RRC12	Frankfurt, DE	IXP	DE-CIX
RRC13	Moscow, RU	IXP	MSK-IX
RRC14	Palo Alto, CA, US	IXP	PAIX
RRC15	Sao Paolo, BR	IXP	PTTMetro-SP
RRC16	Miami, FL, US	IXP	Equinix Miami
RRC18	Barcelona, ES	IXP	CATNIX
RRC19	Johannesburg, ZA	IXP	NAP Africa JB

Nombre	Ubicación	Typo	Sponsors
RRC20	Zurich, CH	IXP	SwissIX
RRC21	Paris, FR	IXP	France-IX Paris and France-IX Marseille
RRC22	Bucharest, RO	IXP	InterLAN
RRC23	Singapore, SG	IXP	Equinix Singapore
RRC24	Montevideo, UY	multihop	LACNIC region
RRC25	Amsterdam, NL	multihop	RIPE NCC
RRC26	Dubai, AE	IXP	UAE-IX, Datamena

Collectors de ROuteVIews:

- Collectors:

Host	Ubicación
amsix.ams.routeviews.org	AMS-IX Amsterdam, Netherlands
cix.atl.routeviews.org	CIX-ATL Atlanta, Georgia
decix.jhb.routeviews.org	DE-CIX KUL, Johor Bahru, Malaysia
iraq-ixp.bgw.routeviews.org	IRAQ-IXP Baghdad, Iraq
pacwave.lax.routeviews.org	Pacific Wave, Los Angeles, California
pit.scl.routeviews.org	PIT Chile Santiago, Santiago, Chile
pitmx.qro.routeviews.org	PIT Chile MX, Querétaro, Mexico
route-views.routeviews.org	Cisco IPv4 U of Oregon, Eugene Oregon
route-views.amsix.routeviews.org	AMS-IX AM6, Amsterdam, Netherlands

Host	Ubicación
route-views.bdix.routeviews.org	BDIX, Dhaka, Bangledesh
route-views.bknix.routeviews.org	BKNIX, Bangkok, Thailand
route-views.chicago.routeviews.org	Equinix CH1, Chicago, Illinois
route-views.chile.routeviews.org	NIC.cl Santiago, Chile
route-views.eqix.routeviews.org	Equinix DC, Ashburn, Virgina
route-views.flix.routeviews.org	FL-IX, Miami, Florida
route-views.fortaleza.routeviews.org	IX.br (PTT.br), Fortaleza, Brazil
route-views.gixa.routeviews.org	GIXA, Ghana, Africa
route-views.gorex.routeviews.org	IGOREX, Guam, US Territories
route-views.jinx.routeviews.org	JINX, Johannesburg, South Africa
route-views.kixp.routeviews.org	KIXP, Nairobi, Kenya
route-views.linx.routeviews.org	LINX, London, United Kingdom
route-views.mwix.routeviews.org	FD-IX, Indianapolis, Indiana
route-views.napafrica.routeviews.org	NAPAfrica, Johannesburg, South Africa
route-views.nwax.routeviews.org	NWAX, Portland, Oregon
route-views.ny.routeviews.org	DE-CIX NYC, New York, USA
route-views.paix.routeviews.org	PAIX, Palo Alto, California
route-views.perth.routeviews.org	West Australian Internet Exchange, Perth, Australia
route-views.peru.routeviews.org	Peru IX, Lima, Peru

Host	Ubicación
route-views.phoix.routeviews.org	University of the Philippines, Diliman, Quezon City, Philippines
route-views.rio.routeviews.org	IX.br (PTT.br), Rio de Janeiro, Brazil
route-views.saopaulo.routeviews.org	SAOPAULO (PTT Metro, NIC.br), Sao Paulo, Brazil
route-views2.saopaulo.routeviews.org	SAOPAULO (PTT Metro, NIC.br), Sao Paulo, Brazil
route-views.sfmix.routeviews.org	San Francisco Metro IX, San Francisco, California
route-views.siex.routeviews.org	Sothern Italy Exchange (SIEX), Rome, Italy
route-views.sg.routeviews.org	Equinix SG1, Singapore, Singapore
route-views.soxrs.routeviews.org	Serbia Open Exchange, Belgrade, Serbia
route-views.sydney.routeviews.org	Equinix SYD1, Sydney, Australia
route-views.telxatl.routeviews.org	TELXATL, Atlanta, Georgia
route-views.uaeix.routeviews.org	UAE-IX, Dubai, United Arab Emirates
route-views.wide.routeviews.org	DIXIE (NSPIXP), Tokyo, Japan