# Ampliacion GNN

Ana López Palomo y Andrea Manuel

November 2023

## 1 Comprensión de GraphSAGE

GraphSAGE, que significa "Graph Sample and Aggregated", es un algoritmo de aprendizaje de representación para grafos. SU objetivo principal es aprender representaciones vectoriales de nodos en un grafo de manera efectica, capturando información contextual de la vecinidad en cada nodo. Está diseñado para trabajar con grafos de gran escala.

El algoritmo sigue los siguientes pasos:

### 1.1 Muestreo de vecinos

GraphSAGE se basa en la idea de muestrear los vecinos de nodos para capturar información local. Para cada nodo en el grafo, se muestrean vecinos y se utilizan para construir representaciones. Este muestreo permite que el algoritmo capture información contextual diversa, y se basa en en una técnica llamada descenso de gradiente de mini lotes.

El descenso de gradiente de miniotes funciona al dividir un conjunto de datos en lotes más pequeños, y durante el entrenamiento calculamos el gradiente de cada uno de los mini lotes. Los mini lotes tienen varios beneficios:

1. Precisión mejorada: ayudan a reducir el sobreajuste, así como la variación en las tasas de error.

2. Mayor velocidad: los mini lotes se procesan en paralelo, tardando menos en entrenarse que los lotes más grandes.

3. Escapabilidad mejorada: un conjunto de datos completo puede exceder la memoria de la GPU, pero los lotes más pequeños pueden sortear esta limitación.

Por otra parte, el muestreo de vecinos tiene el siguiente proceso:

1. El muestreados selecciona aleatoriamente un número definido de vecinos. Si coge el vecino del nodo es un salto, si coge el vecino del vecino son dos saltos, etc.

2. El muestreados genera un subgráfico que contiene el objetivo y los nodos muestrados.

Este proceso se repite para cada nodo en una lista o en la totalidad del grafo. Como crear un subgrafo para cada nodo no es eficiente, los agrupamos por lotes.

Además, el muestreo de vecinos tiene otro beneficio. A veces, hay nodos muy populares que actúan como centros. Calacular las incrustaciones para estos nodos tira mucho del coste computacional, ya que tienen muchos vecinos. El GraphSAGE soluciona este problema, al considerar un número fijo de vecinos.

## 1.2 Agregación de vecinos

Después de muestrear vecinos, se realiza una operación de agregación para combinar la información de los vecinos y actualizar la representación del nodo original. El documento original presenta tres formas de agregar características: aregador medio, agregador LSTM y agregador de agrupaciones.

1) El agregador medio es el más simple. Las características ocultas del nodo de destino y sus vecinos seleccionados seleccionados se promedian, y se aplica una transformación lineal con una matriz de pedos. Es decir:

$$\mathbf{h}_i' = \mathbf{W} \cdot mean_{j \in \tilde{N}_i}(\mathbf{h}_j)$$

Donde:

$$N_i \tag{1}$$

son los nodos, y la W es la matriz de pesos.

2) El agregador LSTM tiene una arquitectura secuencial: asigna un orden a los nodos desordenados. Por esta razón, los autores los barajan al azar, obligando al LSTM a considerar sólo las características ocultas. Sin embargo, es la técnica con mejor rendimiento en sus puntos de referencia.

3) EL agregador de agrupación alimenta el vector oculto de cada vecino a una red neuronal de avance. Luego, se aplica una operación máxima por elementos al resultado para mantener el valor más alto para cada característica.

## 1.3 Adaptación de GraphSAGE

Como sabemos GraphSAGE es un modelo de aprendizaje automático diseñado para trabajar con datos de grafos y es entrenado sobre subgrafos. A continuación vamos a ver cómo ha sido adaptado para usarlo en Pinterest y UberEats.

Primero empecemos con Pinterest, se utiliza una variación de GraphSAGE más adecuada, PinSAGE. Esta es utilizada para mejorar el sistema de recomendaciones de pines, donde aprende representaciones de nodos basadándose en iteracciones pasadas de usuarios con pines similares y proporciona recomendaciones más personalizadas. También se utiliza para ayudar en la segmentación de usuarios y personificación de la experiencia para diferentes segmentos.

UberEats, GraphSAGE se usa para mejorar las recomendaciones de restaurantes, donde considera las iteracciones entre usuarios y restaurantes aprendiendo para hacer sugerencias más precisas. También para oprimizar entegas y análisis de comportamiendo del usuarios, como los patrones de pedido y preferencias.

## 2 Implementación

En este apartado vamos a ir explicando los resultados de la implementación del código que hay en el repositorio.

El dataset de PubMed tiene una cantidad muy baja de nodos de entrenamiento (60) para los 1000 nodos de prueba que tenemos que clasificar.

Después de imprimir información sobre el dataset y el grafo, vamos a comenzar el muestreo de vecinos a través de NeighborLoader. Mostramos cuatro gráficas, cada una un subgrafo con un tamaño y así, podemos procesarlos en paralelo.

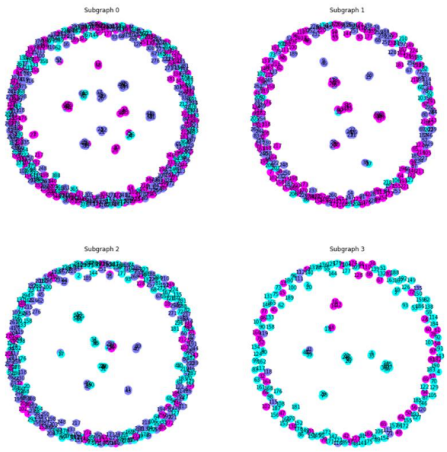


Figure 1: Enter Caption

A continuación, hemos hecho dos gráficas: la primera muestra la distribución original de los grados de los nodos y la segunda muestra la distribución que obtenemos después del muestreo de vecinos.

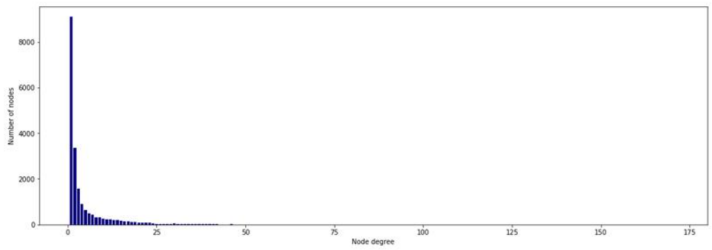


Figure 2: Original

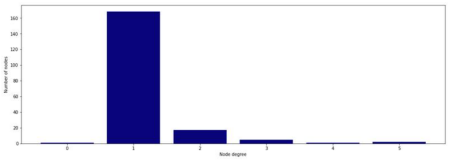

Figure 3: Tras muestreo

Vamos a utilizar SAGEConv para implementar una arquitectura GraphSAGE, que utiliza dos matrices de peso en lugar de una. Creamos una red con dos capas SAGEConv: la primera usa ReLU como la función de activación y una capa de abandono. La segunda genera directamente las incrustaciones de nodos.

Por último, hemos comparado los beneficios de GraphSAGE con un GCN y un GAT, sin ningún muestreo. Vemos que los tres modelos obtienen resultados similares en cuanto a precisión, pero la diferencia se puede apreciar en cuanto a velocidad: GraphSAGE es 88 veces más rápido de que GAT y 4 veces más rápido que GCN (en este ejemplo).

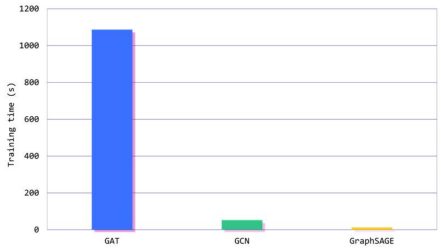


Figure 4: Comparación

Este es el beneficio de este método. Auqnue GraphSAGE pierde mucha precisión en el muestreo de vecinos, mejora en gran medida la escalabilidad.

# 3 Conclusión

GraphSAGE es una arquitectura increíblemente rápida, muy buena para procesar grafos grandes. Aunque no es tan precisa como las otras arquitecturas que hemos probado, como GCN o GAT, ofrece esta gran velocidad, gracias a una combinación entre el muestreo de vecinos y la agregación rápida.