

GRNNs

Introducción a las redes neuronales de Grafos

Seminario máster IA³

11 JUNIO 2024

Sobre mi

- Soy de **Alicante** ciudad.
- Estudié **física en la Universidad de Granada**, pero me especialicé en física computacional y análisis de datos.
- Hice el máster en **Ciencia de Datos** en la **UV (ETSE)**.
- He trabajado como **Ingeniero de Inteligencia Artificial** en el proyecto **SmartCity Valencia**, y en la empresa valenciana **Tyris AI**, y como **técnico superior de investigación** en el **Departamento de Ingeniería Mecánica y Biomecánica de la UPV**.
- Actualmente estoy trabajando en el instituto universitario **VRAIN (UPV)**.



Índice seminarios

Día 11 de junio (1.5 horas)

- Teoría de grafos y métricas
- Ejemplo/práctica para trabajar con grafos en python.
- Tipos y clasificación de redes.
- Fundamentos matemáticos.
- Graph Convolutional Networks (GCNs).
- Práctica sobre GCNs

Día 12 de junio (1.5 horas)

- Lo que falte de hoy
- Graph Attention Networks (GANs)
- Técnicas avanzadas
- *GraphSAGE*
- Más aplicaciones prácticas



Índice día 11 junio

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos



Índice día 11 junio

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos

2 Representación (del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

Índice día 11 junio

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos

2 Representación (del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

3 GNNs (redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento



Índice

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos

2 Representación (del dato al grafo)

- ¿Cómo se representan los datos (grafos)?
- ¿Qué ventajas y desventajas tienen? ¿Benefician de esta representación?
- ¿Cómo se convierten los datos al grafo?

3 Redes Neuronales (redes neuronales)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

¿POR QUÉ GRAFOS?



Problemas de Grafos

¿Por qué usar grafos para **representar datos**?

1. **Ciencias de la Computación:** Modelar la estructura de programas de computadora para entender cómo interactúan los distintos componentes de un sistema.
2. **Física:** Modelar sistemas físicos y sus interacciones, como la relación entre partículas y sus propiedades.
3. **Biología:** Modelar sistemas biológicos, como las vías metabólicas, en forma de redes interconectadas.
4. **Ciencias Sociales:** Estudiar y entender redes sociales complejas, incluyendo las relaciones entre individuos en una comunidad.
5. **Finanzas:** Analizar tendencias del mercado de valores y las relaciones entre distintos instrumentos financieros.
6. **Ingeniería:** Modelar y analizar sistemas complejos, como redes de transporte y redes eléctricas.

Problemas de Grafos

¿Por qué usar algoritmos de *machine learning* sobre grafos?



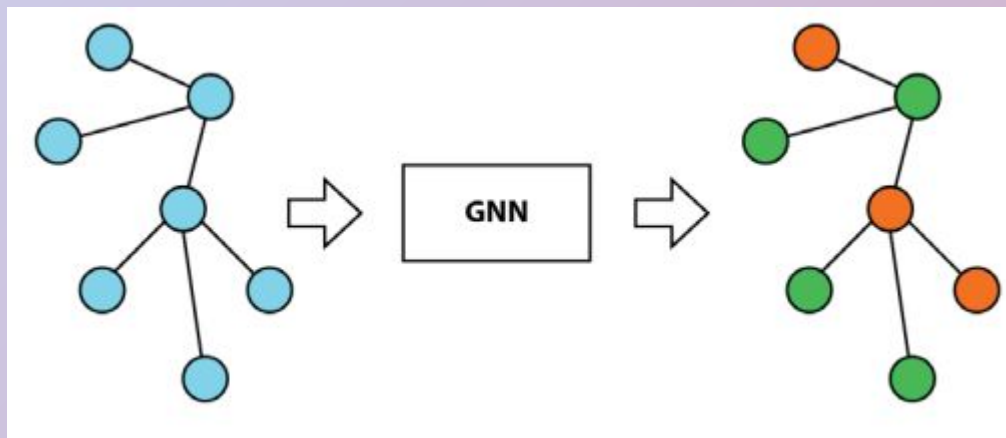
IDAL

Intelligent
Data
Analysis
Laboratory

Problemas de Grafos

¿Por qué usar algoritmos de *machine learning* sobre grafos?

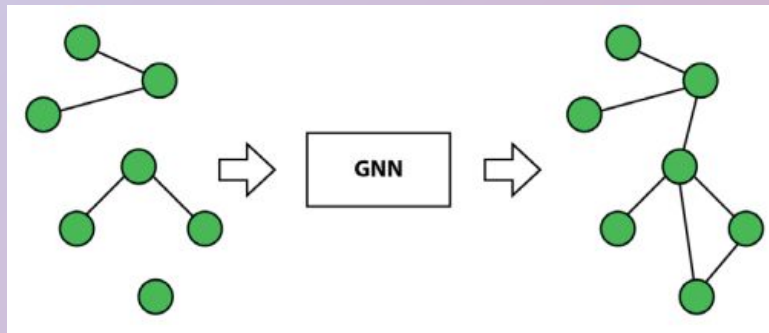
1. **Clasificación de Nodos:** Predecir la categoría de un nodo en un grafo, útil para categorizar usuarios o ítems en línea basándose en sus características.



Problemas de Grafos

¿Por qué usar algoritmos de *machine learning* sobre grafos?

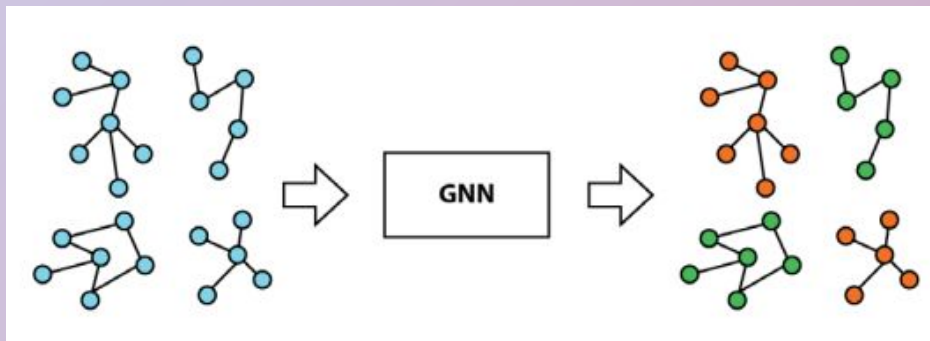
1. **Clasificación de Nodos:** Predecir la categoría de un nodo en un grafo, útil para categorizar usuarios o ítems en línea basándose en sus características.
2. **Predicción de Enlaces:** Predecir enlaces faltantes entre pares de nodos, aplicable en la completación de grafos de conocimiento para prever relaciones, como recomendaciones de amistades (RecSys).



Problemas de Grafos

¿Por qué usar algoritmos de *machine learning* sobre grafos?

1. **Clasificación de Nodos:** Predecir la categoría de un nodo en un grafo, útil para categorizar usuarios o ítems en línea basándose en sus características.
2. **Predicción de Enlaces:** Predecir enlaces faltantes entre pares de nodos, aplicable en la completación de grafos de conocimiento para prever relaciones, como recomendaciones de amistades (RecSys).
3. **Clasificación de Grafos:** Categorizar diferentes grafos en categorías predefinidas, por ejemplo, para predecir propiedades de estructuras moleculares en biología molecular.



Problemas de Grafos

¿Por qué usar algoritmos de *machine learning* sobre grafos?

1. **Clasificación de Nodos:** Predecir la categoría de un nodo en un grafo, útil para categorizar usuarios o ítems en línea basándose en sus características.
2. **Predicción de Enlaces:** Predecir enlaces faltantes entre pares de nodos, aplicable en la completación de grafos de conocimiento para prever relaciones, como recomendaciones de amistades (RecSys).
3. **Clasificación de Grafos:** Categorizar diferentes grafos en categorías predefinidas, por ejemplo, para predecir propiedades de estructuras moleculares en biología molecular.
4. **Generación de Grafos:** Generar nuevos grafos con propiedades deseadas, como en la creación de estructuras moleculares para descubrimiento de fármacos.

AlphaFold



Problemas de Grafos

¿Por qué usar algoritmos de *machine learning* sobre grafos?

1. **Clasificación de Nodos:** Predecir la categoría de un nodo en un grafo, útil para categorizar usuarios o ítems en línea basándose en sus características.
2. **Predicción de Enlaces:** Predecir enlaces faltantes entre pares de nodos, aplicable en la completación de grafos de conocimiento para prever relaciones, como recomendaciones de amistades (RecSys).
3. **Clasificación de Grafos:** Categorizar diferentes grafos en categorías predefinidas, por ejemplo, para predecir propiedades de estructuras moleculares en biología molecular.
4. **Generación de Grafos:** Generar nuevos grafos con propiedades deseadas, como en la creación de estructuras moleculares para descubrimiento de fármacos.

Más aplicaciones Prácticas: Sistemas de recomendación, para sugerir ítems a usuarios basándose en interacciones previas. Predicción del tráfico, mejorando las estimaciones de tiempos de viaje al considerar relaciones complejas entre rutas y modos de transporte ...

Teoría de Grafos

1 Teoría de Grafos

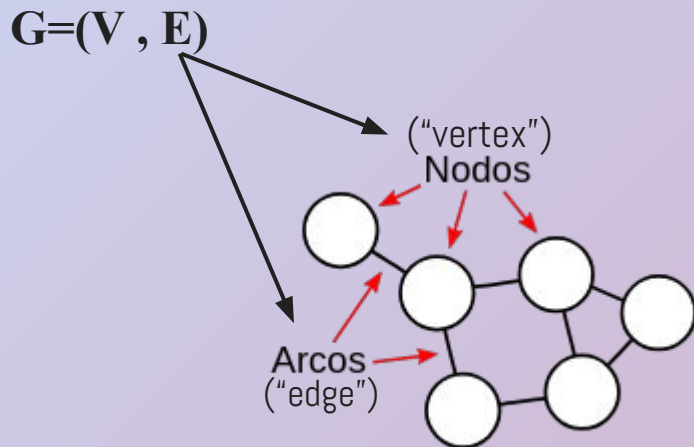
- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos



Teoría de Grafos

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos



Los nodos representan objetos como:

- Ciudades
- Personas
- Páginas web

Los arcos representan las conexiones y relaciones entre nodos.

Teoría de Grafos

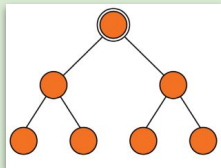
1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos

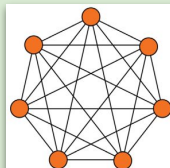
- Dirigido o no dirigido (dirección en conexiones).
- Conexiones ponderadas ("weighted")
- Conexo o inconexo.

Los más famosos:

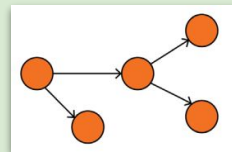
Árbol



Completo



Directed Acyclic Graph (DAG)



Teoría de Grafos

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos

Términos:

- **Grado de un nodo:** Número de conexiones que un nodo tiene.
- **Vecinos:** Nodos conectados por un solo arco (grado 1).
- **Nodos adyacentes:** Si tienen un vecino común.
- **Camino ó "path":** Conjunto de arcos que conectan dos nodos.
 - **Sencillo:** No se visitan los nodos más de una vez.
 - **Cíclico:** El primero y último nodo son el mismo. (Acíclico es cuando NO hay ciclos)

Teoría de Grafos

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos

Métricas:

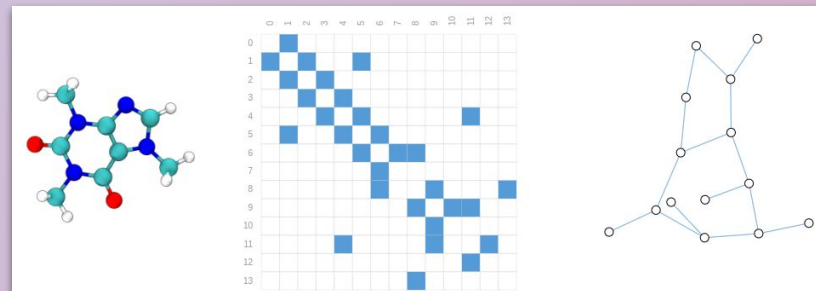
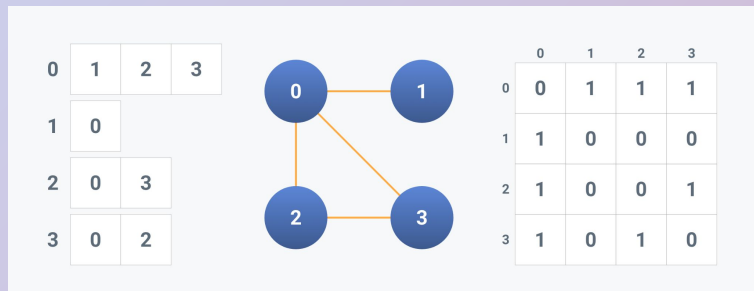
- **"Graph's minimum cut"**: Mínimo número de conexiones a eliminar para hacerlo inconexo. Se usa mucho en problemas de optimización de flujos, agrupación, detección de comunidades.
- **Grado de centralidad (*Degree Centrality*)**: Grado del nodo, es decir, el número de conexiones directas que el nodo tiene con otros nodos.
- **Centralidad de Cercanía (*Closeness Centrality*)**: Qué tan cerca está un nodo de todos los demás nodos en el grafo, correspondiendo a la longitud promedio del camino más corto entre el nodo objetivo y todos los otros nodos en el grafo.
- **Centralidad de Intermediación (*Betweenness Centrality*)**: Mide el número de veces que un nodo aparece en el camino más corto entre pares de otros nodos en el grafo.
- **Densidad**: Indica cuán conectado está un grafo. Se trata de una relación entre el número real de aristas y el número máximo posible de aristas en el grafo.

Representación de Grafos

2 Representación
(del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

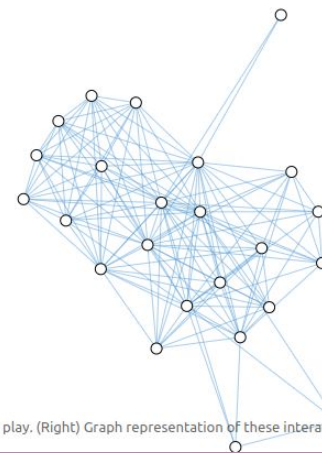
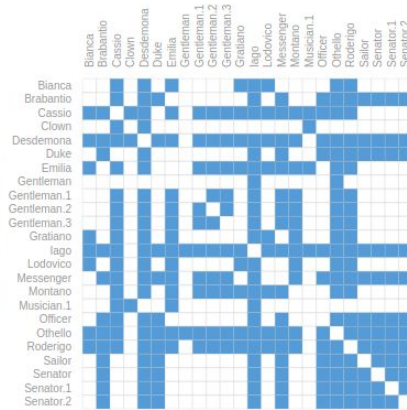
Según el caso concreto convendrá más una representación u otra. La **representación más habitual**, es mediante **la matriz de adyacencia**:



Representación de Grafos

2 Representación
(del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?



(Left) Image of a scene from the play "Othello". (Center) Adjacency matrix of the interaction between characters in the play. (Right) Graph representation of these interactions.

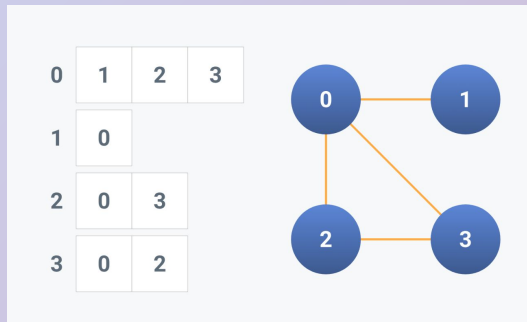


Representación de Grafos

2 Representación (del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

Si tenemos un **grafo muy disperso** (*sparse*), donde hay muchos más nodos que conexiones, entonces lo más recomendable es trabajar con **listas de adyacencia**.



Por ejemplo:

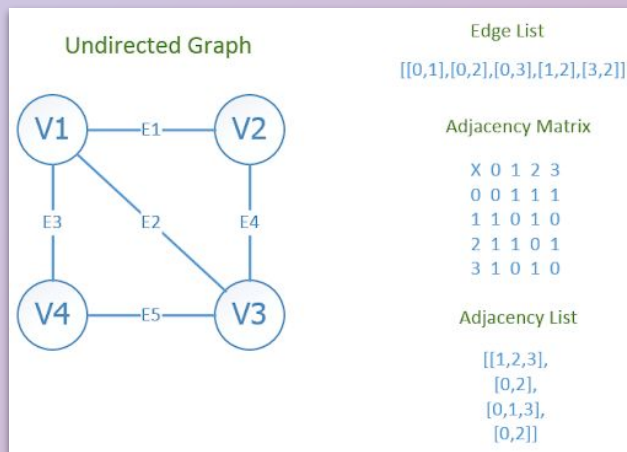
- Recomendadores:
 - Gustos musicales en spotify.
 - Follows en Twitter.
 - ...
- Ranking de páginas web (PageRank de Google)

Representación de Grafos

2 Representación
(del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

Otra forma de representar estos **grafos dispersos** es mediante las **listas de arcos** (“*edge list*”):



Representación de Grafos

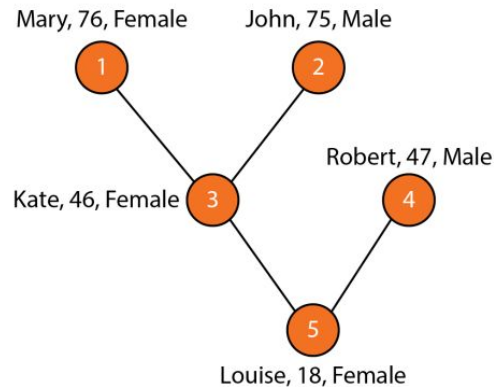
2 Representación (del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

Tabular dataset

ID	Name	Age	Gender
1	Mary	76	Female
2	John	75	Male
3	Kate	46	Female
4	Robert	47	Male
5	Loise	18	Female

Graph dataset



Representación de Grafos

2 Representación (del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

Unos datos se beneficiarían de la representación en grafo, ¿cómo lo definimos?

- **Definir los Nodos:** Identifica las entidades que serán representadas como nodos en el grafo (ejemplo: usuarios, ciudades, productos).
- **Identificar las Conexiones:** Determina qué tipo de relaciones representarán las aristas entre los nodos (ejemplo: amistades, rutas, transacciones).
- **Crear las Listas de Adyacencia:** Inicia con una lista vacía para cada nodo y añade nodos adyacentes basado en las conexiones reales entre ellos, y de aquí podemos pasar a la matriz de adyacencia.
- **Considerar la Direccionalidad:** Decide si el grafo es dirigido (las relaciones tienen una dirección) o no dirigido (las relaciones son bidireccionales).
- **Ponderar las Aristas (opcional):** Asigna pesos a las aristas si algunas conexiones tienen mayor importancia que otras.

Prácticas

Veamos su aplicación en código

Práctica sobre la teoría de grafos

Colab Notebook:

<https://colab.research.google.com/drive/1T1XRBD8anFx0C12S2gVPC7Mm7qvtTwqG?usp=sharing>



Algoritmos con grafos

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos

2 Representación (del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

3 GNNs (redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento



Algoritmos con grafos

1 Teoría de Grafos

- ¿Qué partes tiene un grafo?
- ¿Qué tipos hay?
- Fundamentos básicos

2 Representación (del dato al grafo)

- ¿Cómo los interpretamos (grafos)?
- ¿Qué datos se benefician de esta representación?
- ¿Cómo pasamos de los datos al grafo?

Machine Learning PRE-redes neuronales

3 GNNs (redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento



Algoritmos con grafos

Graph learning techniques

Graph signal processing

Métodos tradicionales de procesamiento de señales a grafos, como la transformada de Fourier de grafos y el análisis espectral. Revelan propiedades intrínsecas del grafo, como su conectividad y estructura.



Algoritmos con grafos

Graph learning techniques

Graph signal processing

Métodos tradicionales de procesamiento de señales a grafos, como la transformada de Fourier de grafos y el análisis espectral. Revelan propiedades intrínsecas del grafo, como su conectividad y estructura.

Matrix Factorization

Busca encontrar representaciones de baja dimensión de matrices grandes. Identificar factores latentes o patrones que expliquen las relaciones observadas. Proporcionar una representación compacta e interpretable de los datos.



Algoritmos con grafos

Graph learning techniques

Graph signal processing

Métodos tradicionales de procesamiento de señales a grafos, como la transformada de Fourier de grafos y el análisis espectral. Revelan propiedades intrínsecas del grafo, como su conectividad y estructura.

Matrix Factorization

Busca encontrar representaciones de baja dimensión de matrices grandes. Identificar factores latentes o patrones que expliquen las relaciones observadas. Proporcionar una representación compacta e interpretable de los datos.

Random Walk

Concepto matemático utilizado para modelar el movimiento de entidades en un grafo. Al simular Random Walks sobre un grafo, se puede recopilar información sobre las relaciones entre nodos, lo cual puede dar lugar a "embeddings", representaciones muy útiles para el entrenamiento de otros modelos de aprendizaje automático.



Algoritmos con grafos

Graph learning techniques

Graph signal processing

Métodos tradicionales de procesamiento de señales a grafos, como la transformada de Fourier de grafos y el análisis espectral. Revelan propiedades intrínsecas del grafo, como su conectividad y estructura.

Matrix Factorization

Busca encontrar representaciones de baja dimensión de matrices grandes. Identificar factores latentes o patrones que expliquen las relaciones observadas. Proporcionar una representación compacta e interpretable de los datos.

Random Walk

Concepto matemático utilizado para modelar el movimiento de entidades en un grafo. Al simular Random Walks sobre un grafo, se puede recopilar información sobre las relaciones entre nodos, lo cual puede dar lugar a "embeddings", representaciones muy útiles para el entrenamiento de otros modelos de aprendizaje automático.

Deep Learning

Subcampo del aprendizaje automático centrado en redes neuronales. Muy útiles para codificar, representar y operar eficazmente con datos de grafos como vectores ("embeddings").



Algoritmos con grafos

Graph learning techniques

Graph signal processing

Métodos tradicionales de procesamiento de señales a grafos, como la transformada de Fourier de grafos y el análisis espectral. Revelan propiedades intrínsecas del grafo, como su conectividad y estructura.

Matrix Factorization

Busca encontrar representaciones de baja dimensión de matrices grandes. Identificar factores latentes o patrones que expliquen las relaciones observadas. Proporcionar una representación compacta e interpretable de los datos.

Random Walk

Concepto matemático utilizado para modelar el movimiento de entidades en un grafo. Al simular Random Walks sobre un grafo, se puede recopilar información sobre las relaciones entre nodos, lo cual puede dar lugar a "embeddings", representaciones muy útiles para el entrenamiento de otros modelos de aprendizaje automático.

Deep Learning

Subcampo del aprendizaje automático centrado en redes neuronales. Muy útiles para codificar, representar y operar eficazmente con datos de grafos como vectores ("embeddings").



Redes neuronales de Grafos

3 GNNs
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

Simplificación del cálculo (hay muchas variantes, por ejemplo, cómo agregar los nodos):

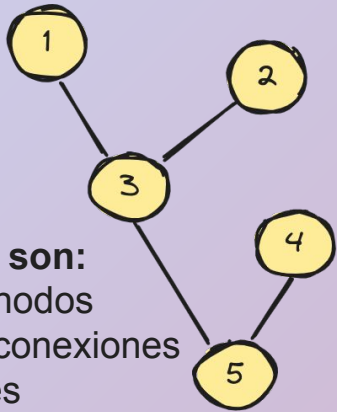
1. Cada nodo tiene unas variables (por ejemplo edad, salario, estado civil, ...)
2. Las relaciones entre nodos también puede llevar consigo variables.
3. Al final, lo que buscamos casi siempre es:
 - a. Clasificación/Regresión de una clase/valor sobre cada nodo.
 - b. Clasificación/Regresión de una clase/valor sobre cada conexión.
 - c. Clasificación/Regresión de una clase/valor sobre el grafo completo.



Ejemplo cálculos con GNNs

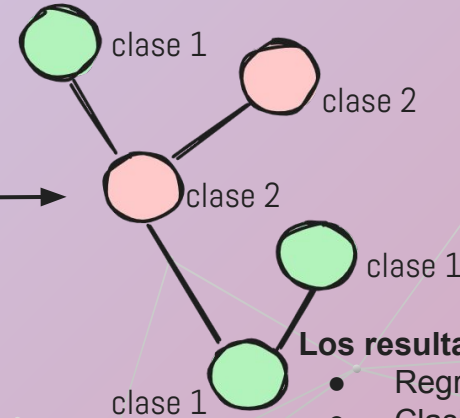
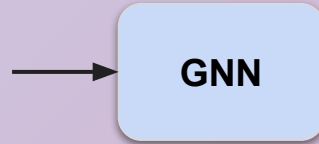
La GNN más sencilla (convolución \rightarrow GCNs):

- Arquitectura **graph in - graph out**.
- La conectividad no cambia.
- Message-passing framework



Los datos de entrada son:

- Variables de los nodos
- Variables de las conexiones
- Variables globales



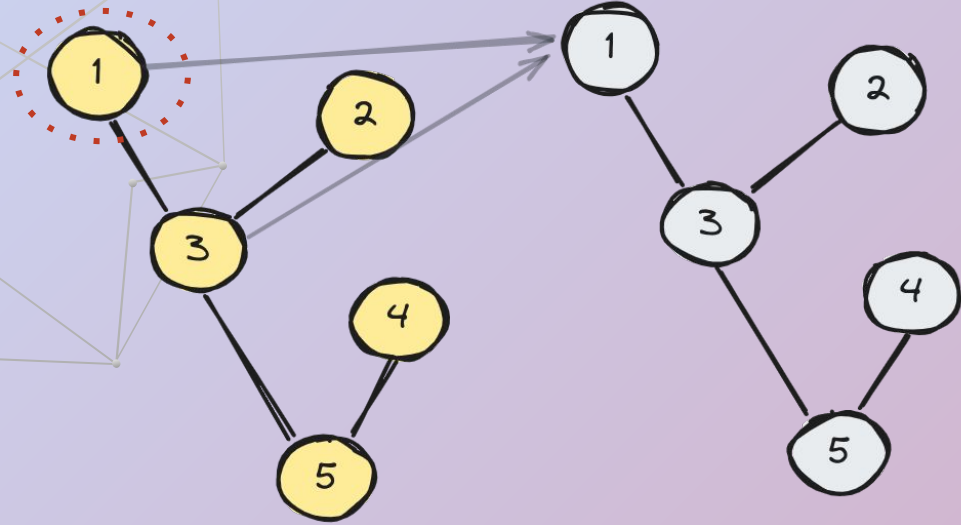
Los resultados de salida son:

- Regresión sobre nodos
- Clasificación de nodos
- ...



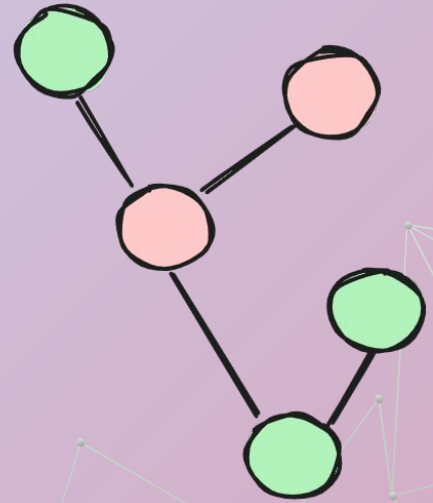
Ejemplo cálculos con GNNs

Input



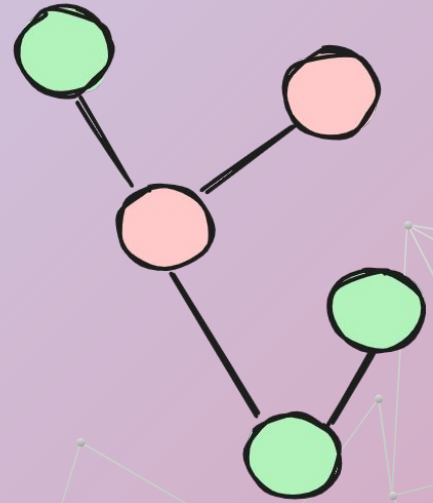
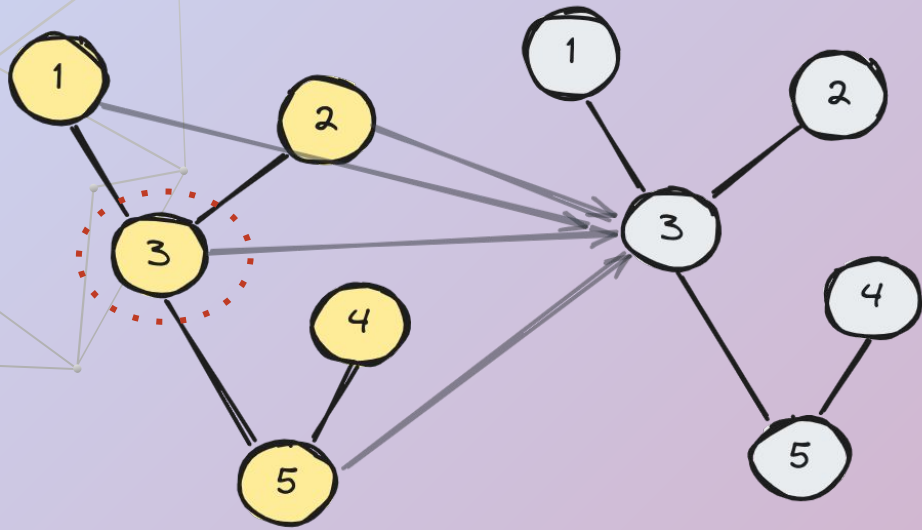
Output

Con 1 capa oculta.



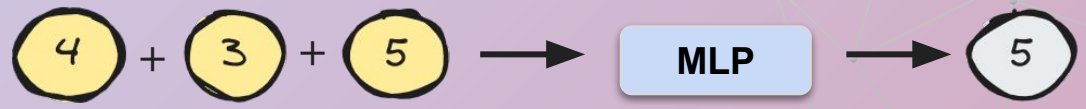
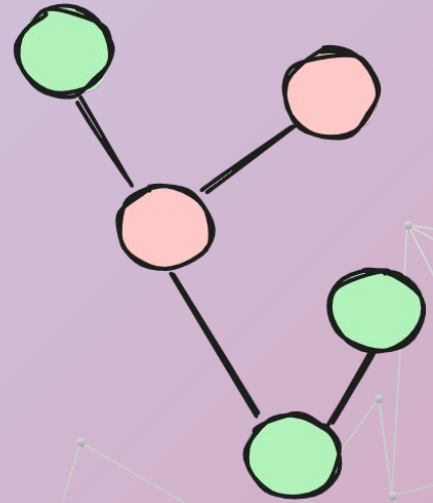
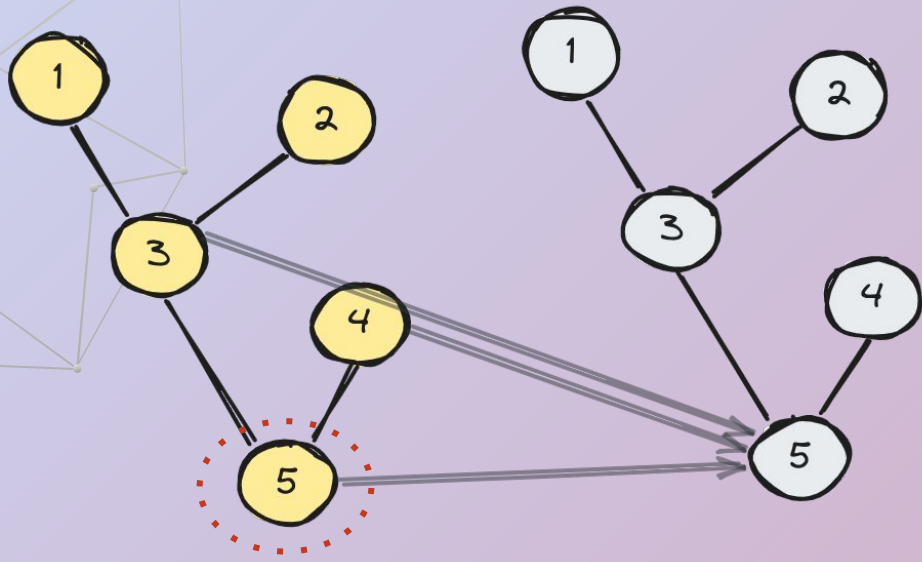
Ejemplo cálculos con GNNs

Con 1 capa oculta.



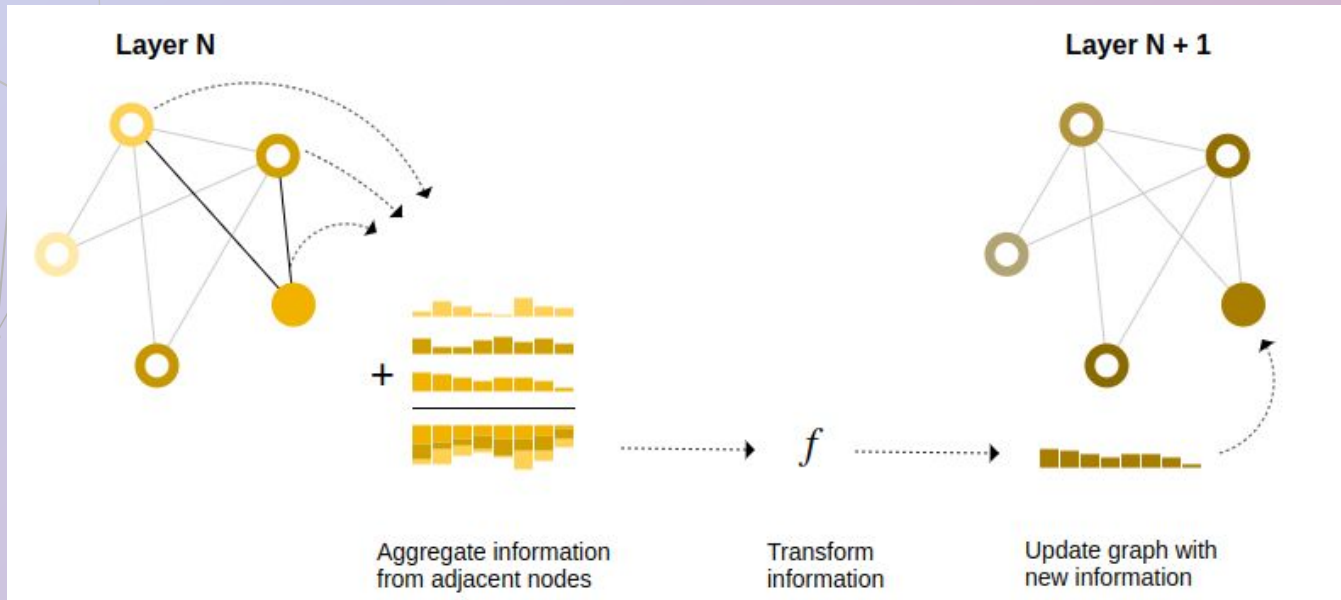
Ejemplo cálculos con GNNs

Con 1 capa oculta.



Ejemplo cálculos con GNNs

Otra visualización:



Redes neuronales de Grafos

3 GNNs
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

El concepto (framework) de “**message passing**” (2017 by Gilmer et al.). Generalizar las GNNs.

Message Passing

1. Generación del mensaje

Cada nodo genera mensajes para sus vecinos usando sus características propias y, opcionalmente, las de los vecinos y las aristas.

2. Agregación del mensaje

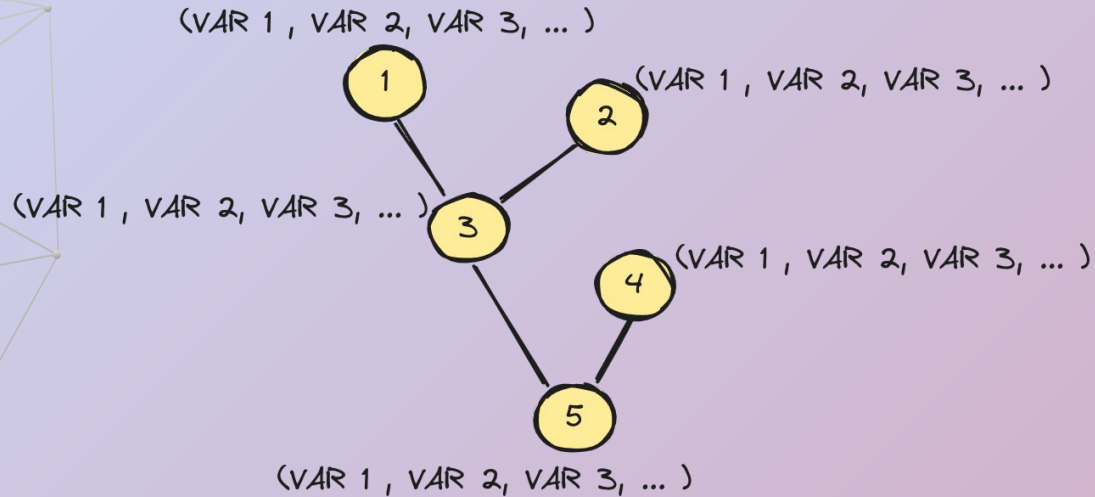
Cada nodo combina mensajes de sus vecinos mediante una función que no cambia con las permutaciones, típicamente sumando estos mensajes

3. Actualización del estado.

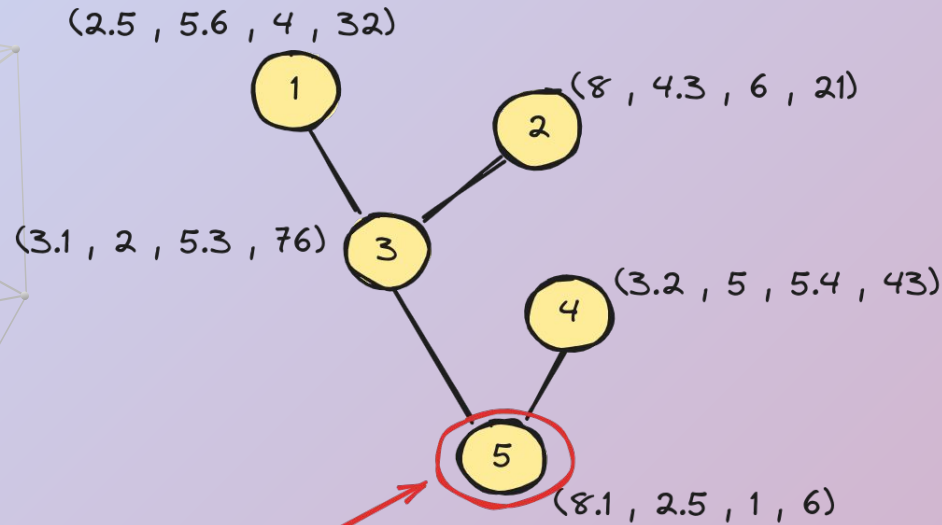
Actualiza sus propias características integrando estos mensajes agregados. Incluyéndose, a veces, el estado propio con los estados de los vecinos.



Ejemplo message passing



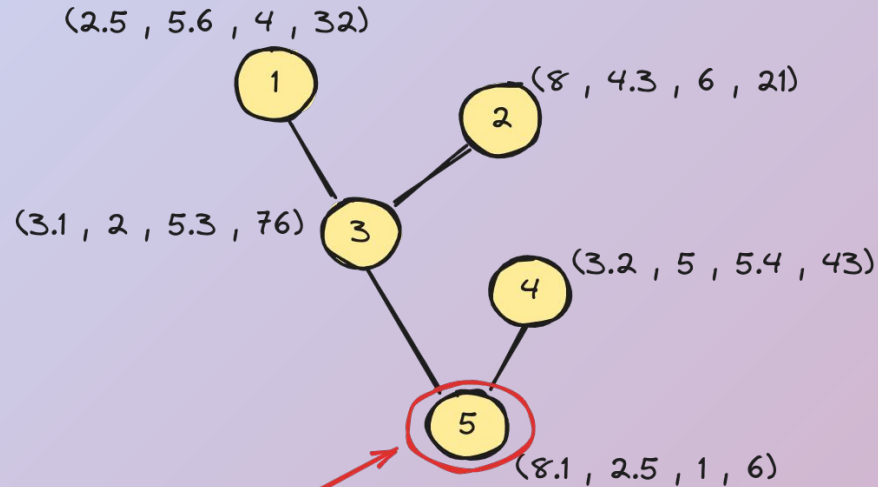
Ejemplo message passing



Calculamos nueva
representación de este nodo.



Ejemplo message passing



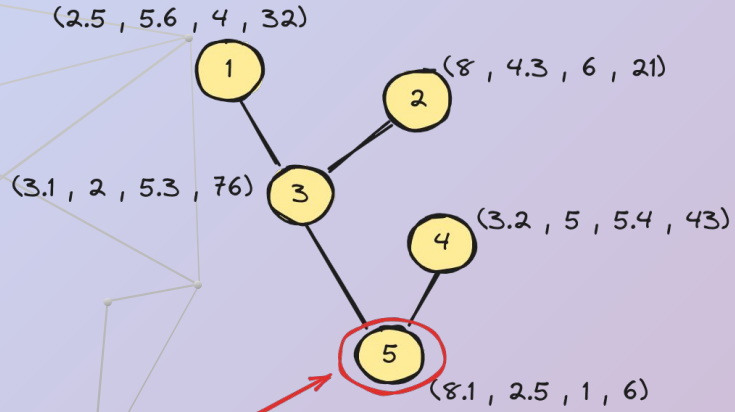
Calculamos nueva
representación de este nodo.

Para calcular **5** en **L=2** necesitamos **3** y **4** en **L=1** ...

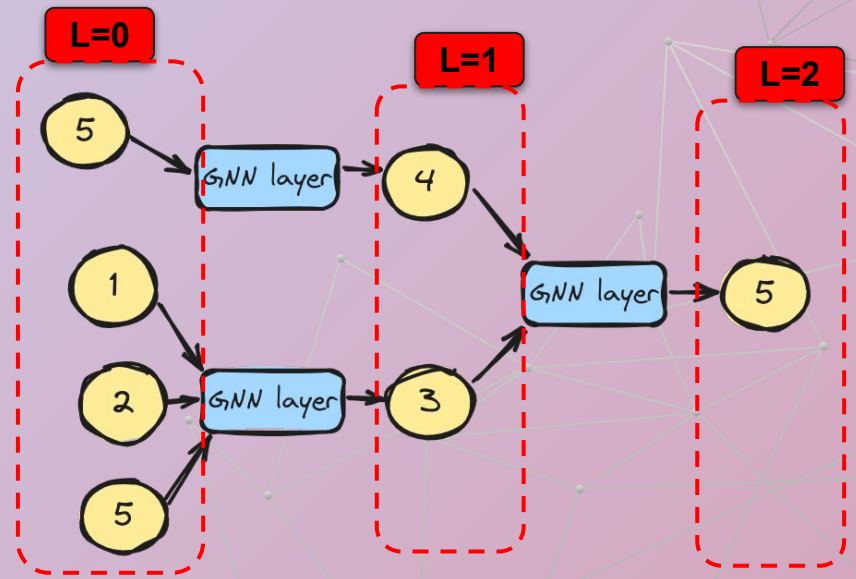
Los cálculos son iterativos, y pueden tener diferente profundidad.

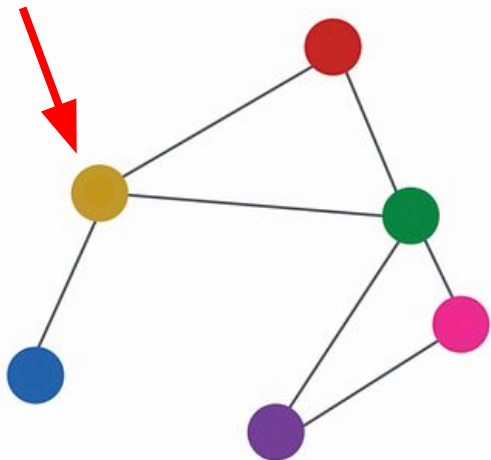


Ejemplo message passing



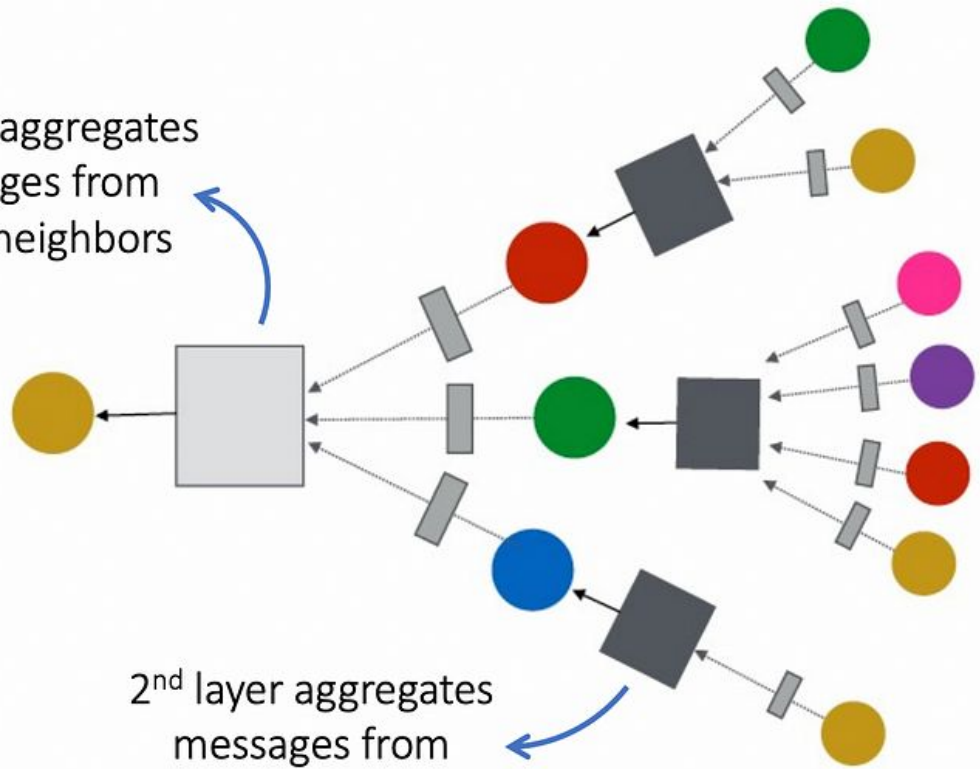
Calculemos nueva representación de este nodo.





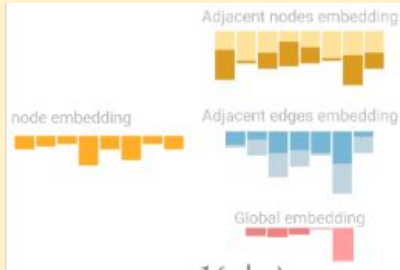
Input Graph

1st layer aggregates messages from 1-hop neighbors



2nd layer aggregates messages from 2-hop neighbors

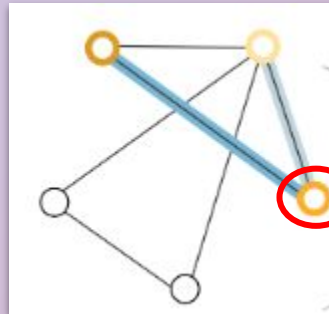
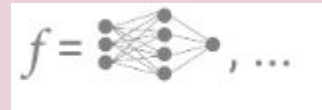
Mensaje



Agregación

Función invariante a permutaciones.
E.g. Sum, max, mean,
...

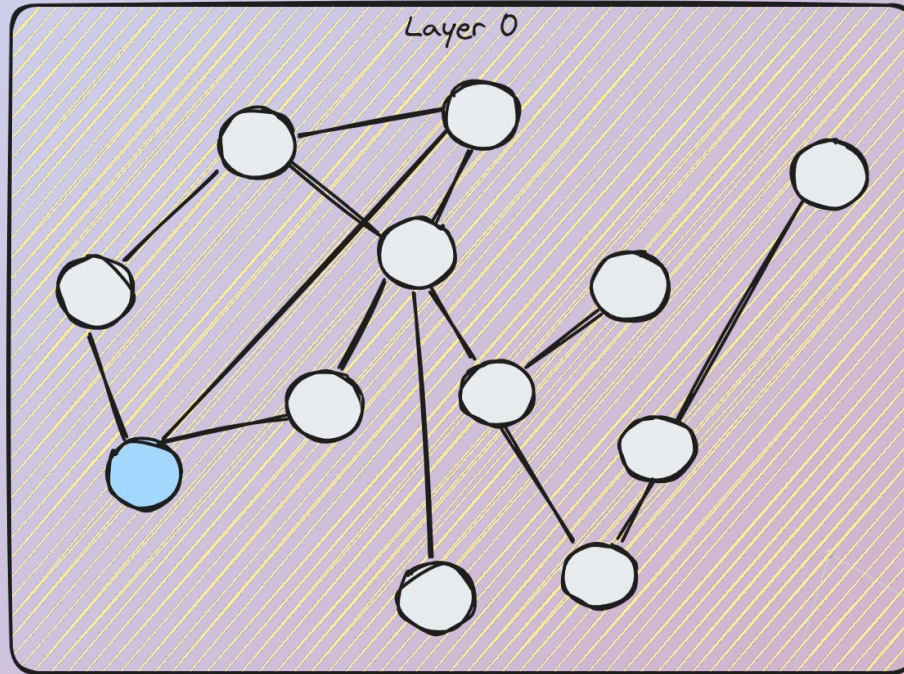
Actualización



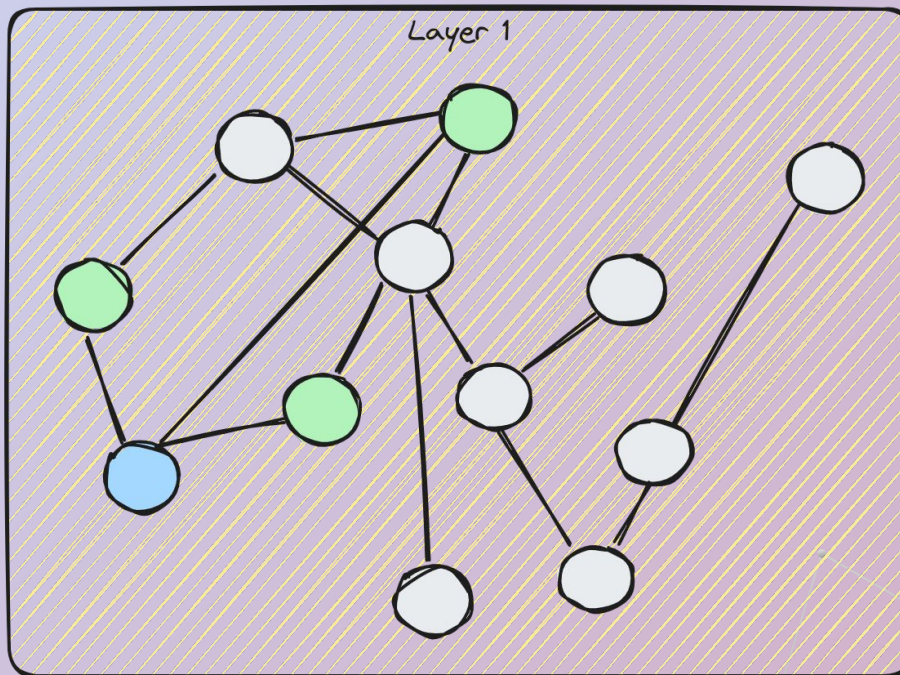
Nodo a calcular



Alcance de la red (*"receptive field"*)



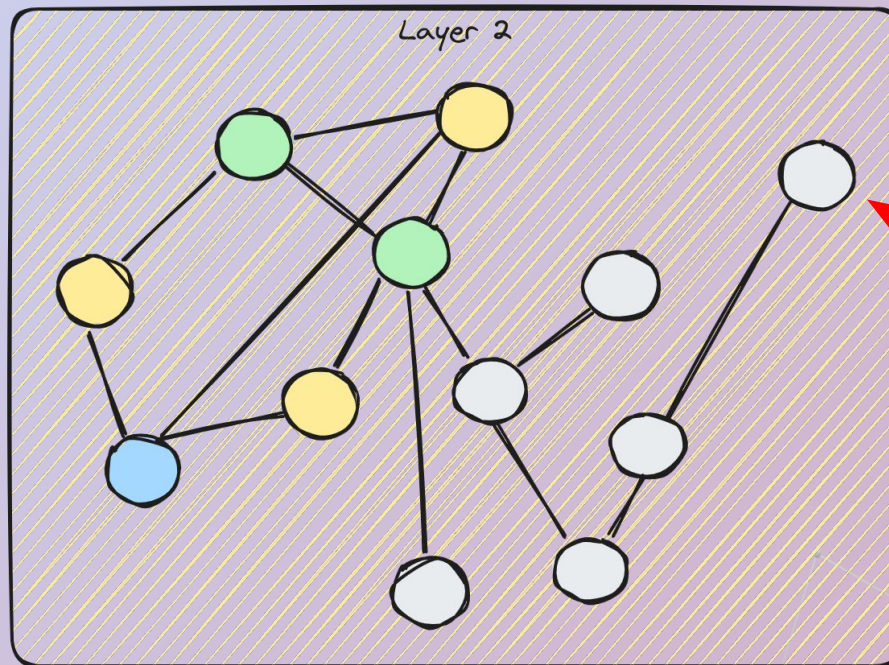
Alcance de la red (*"receptive field"*)



IDAL

Intelligent
Data
Analysis
Laboratory

Alcance de la red (*"receptive field"*)



Dificultad para abarcar toda la red.

Nota: Se proponen **soluciones** como **nodos maestros** o **vectores de contexto ampliado**.



Redes neuronales de Grafos

3 GNNs
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

Hay muchas variantes y a veces cuesta tener una visión nítida de todo, para ello nos centraremos en 4 de los conceptos/arquitecturas centrales:

- **Message Passing Neural Networks (MPNNs)**
 - **Graph Convolutional Networks (GCNs)**
 - **Graph Attention Networks (GATs)**
 - **GraphSAGE**

Nota: Todos estos modelos son formas de aplicar aprendizaje profundo a datos estructurados en forma de grafos, pero cada uno tiene características únicas y aplicaciones específicas.

Redes neuronales de Grafos

3 GNNs
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

Message Passing Neural Networks (MPNNs)

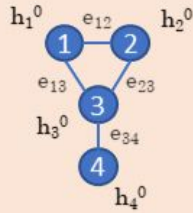
Las **MPNNs** representan un marco general para muchas de las técnicas de redes neuronales de grafos, incluidas las GCNs, GATs, GraphSAGE, etc.

El concepto central es que los **nodos intercambian mensajes con sus vecinos**, y **estos mensajes son luego agregados y usados para actualizar los estados de los nodos**.

En este marco, se pueden definir explícitamente las funciones de **generación de mensajes**, **agregación** y **actualización**, lo que ofrece una gran flexibilidad para diseñar diferentes tipos de agregaciones y transformaciones.

Redes neuronales de Grafos

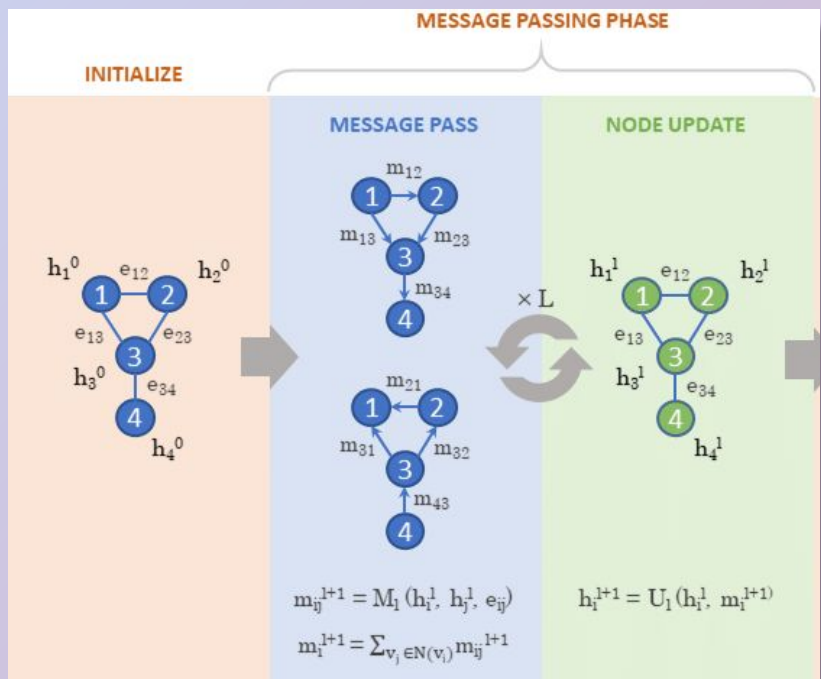
INITIALIZE



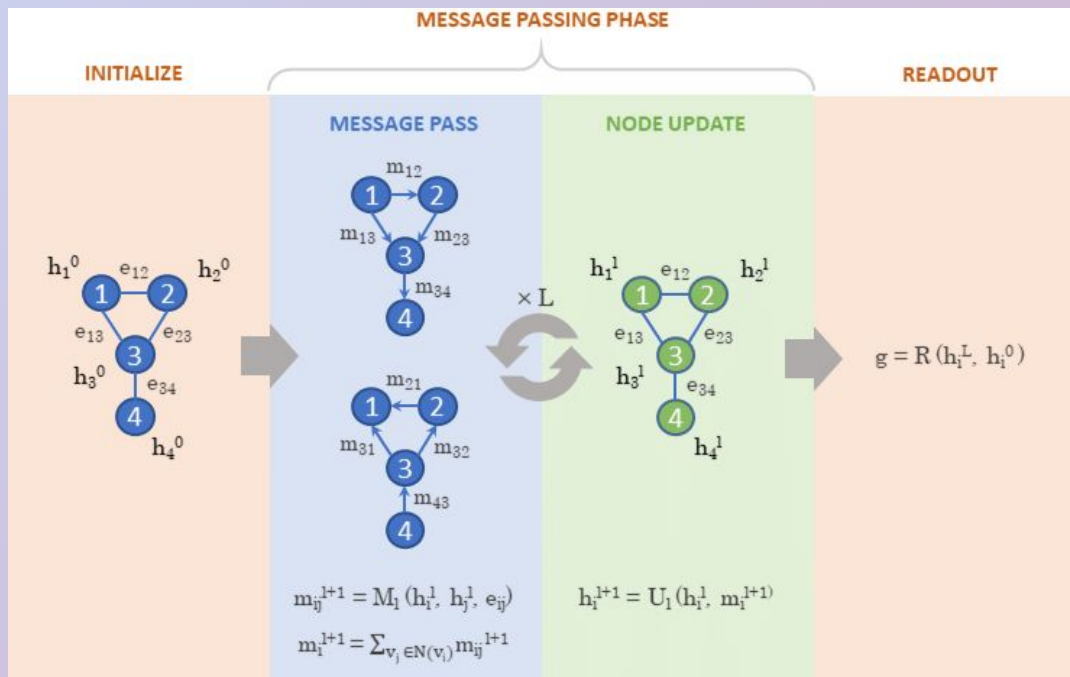
IDAL

Intelligent
Data
Analysis
Laboratory

Redes neuronales de Grafos



Redes neuronales de Grafos



Redes neuronales de Grafos

3 $GNNs$
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

Graph Convolutional Networks (GCNs):

Las **GCNs** son una técnica que generaliza las operaciones de convolución de las CNNs para trabajar sobre grafos.

La idea es utilizar la estructura del grafo para definir una operación de "convolución" que, en lugar de operar sobre una región local de una imagen (como en las **CNNs**), opera sobre los "vecindarios" de un nodo en un grafo.

Esto implica sumar y transformar las características de los nodos vecinos usando pesos compartidos, lo que permite capturar la estructura local del grafo.

Redes neuronales de Grafos

3 GNNs
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

Graph Attention Networks (GATs):

Las **GATs** introducen un mecanismo de atención en el proceso de agregación de las características de los vecinos.

En una **GAT**, la influencia de cada vecino en el nodo central se pondera mediante **coeficientes de atención**, que son aprendidos por la red. Esto permite que el modelo dé más importancia a ciertos nodos vecinos en función de cómo de relevantes son para la tarea específica, proporcionando una forma adaptativa y potente de agregación de características.

Redes neuronales de Grafos

3 GNNs
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

GraphSAGE:

Es un modelo que también se centra en cómo los nodos agregan características de sus vecinos, pero con un enfoque específico en la eficiencia y la escalabilidad.

En lugar de usar toda la información de los vecinos, **GraphSAGE** emplea un método de muestreo para seleccionar un subconjunto fijo de vecinos y luego utiliza diferentes funciones de agregación (como el promedio, la suma, o el máximo) para combinar sus características.

Esto permite que **GraphSAGE** maneje grafos grandes de manera eficiente.

Redes neuronales de Grafos

3 GNNs
(redes neuronales de grafos)

- Fundamentos
- Taxonomía (partes/clasificación)
- Entrenamiento

Relación entre los Conceptos:

Todos estos modelos comparten el principio básico de que la representación de un nodo se puede mejorar mediante la incorporación de información de sus vecinos.

Las **GCNs** y **GATs** son casos específicos dentro del marco general de las **MPNNs** donde las operaciones de agregación y actualización están más estructuradas.

GraphSAGE por otro lado, amplía estos modelos al introducir métodos de muestreo que hacen que el procesamiento de grafos grandes sea más factible.

Cada modelo ofrece diferentes enfoques y mejoras sobre cómo se puede integrar la información de los vecinos en la representación de un nodo, y el uso de uno u otro dependerá de las necesidades específicas de rendimiento, precisión y escalabilidad del problema en cuestión.



Caso concreto Redes convolucionales de Grafos (GCNs)

¿Cómo trasladar el concepto de convolución a un grafo?

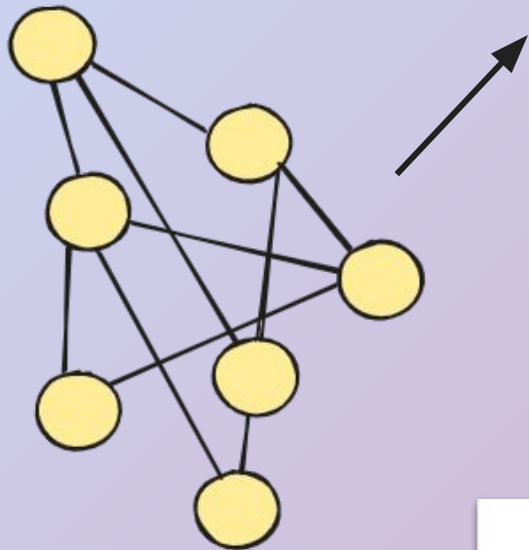


IDAL

Intelligent
Data
Analysis
Laboratory

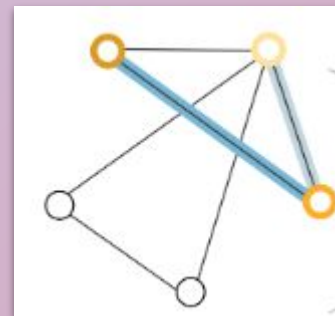
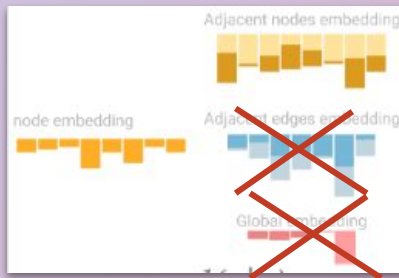
Fundamento matemático en GCNs

GCN simplificada ("vanilla")



A: Matriz de adyacencia

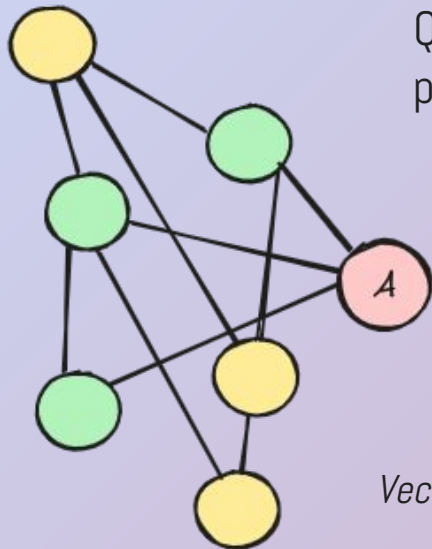
- Nos centramos solo en los nodos
- Su conectividad no es ponderada
- Cada nodo tiene un vector de características asociado.
- No tenemos variables globales



Fundamento matemático en GCNs

GCN simplificada ("vanilla")

Queremos calcular la nueva representación de **A**, usando para ello los vectores de características de sus vecinos:



$$h_A = \sum_{i \in \mathcal{N}_A} x_i W^T$$

Matriz que transforma linealmente el vector de características

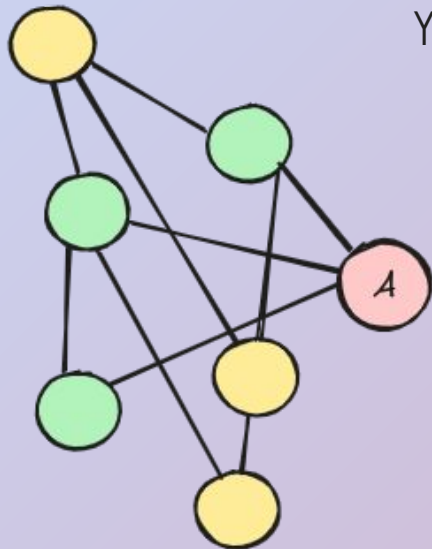
Vecinos (nodos verdes)

Cada vector de características de los vecinos:

ID	Name	Age	Gender
1	Mary	76	Female
2	John	75	Male

Fundamento matemático en GCNs

GCN simplificada ("vanilla")



Y esto lo tendremos que realizar para cada nodo del grafo:

For **A** in **nodos**:

$$h_A = \sum_{i \in \mathcal{N}_A} x_i W^T$$

Mejor y más óptimo con **álgebra matricial**.

$$\tilde{A} = A + I$$

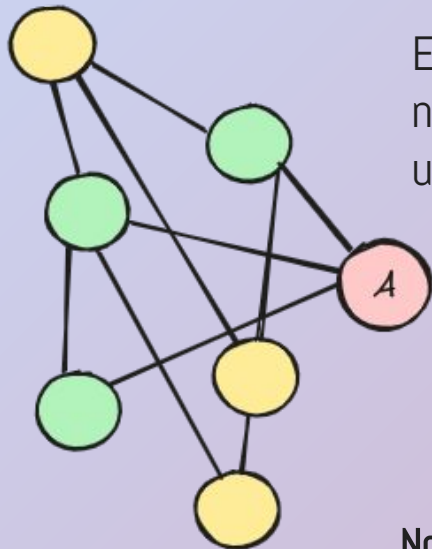
Añadimos la identidad para tener en cuenta el vector de variables del propio nodo a calcular.

$$H = \tilde{A}^T X W^T$$

Aquí estamos calculando todos los estados intermedios $H=(h_A, h_B, \dots)$ de golpe.

Fundamento matemático en GCNs

GCN estandar



En el caso anterior, no estábamos teniendo en cuenta el número de vecinos que tiene cada nodo. Y al tratarse de una suma sobre vecinos, sería conveniente normalizarlo:

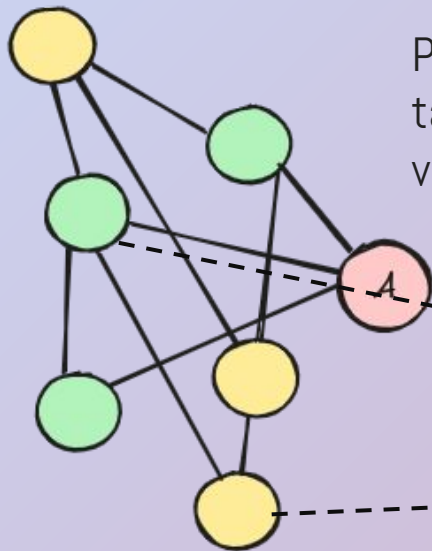
$$h_A = \sum_{i \in \mathcal{N}_A} x_i W^T$$

$$h_A = \frac{1}{\text{deg}(A)} \sum_{i \in \mathcal{N}_A} x_i W^T$$

Nota: Si un nodo A tiene 1000 vecinos y otro B solo 5, por lo general, el h_A será mucho mayor que h_B porque la suma se ha hecho 1000 veces, y en B solo 5.

Fundamento matemático en GCNs

GCN estandar



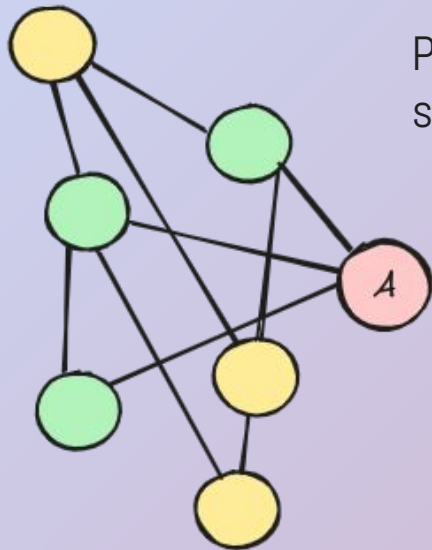
Para hacerlo todo de manera matricial, ahora usaremos también la matriz D , que recoge el grado (número de vecinos) de cada nodo.

$$D = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$



Fundamento matemático en GCNs

GCN estandar



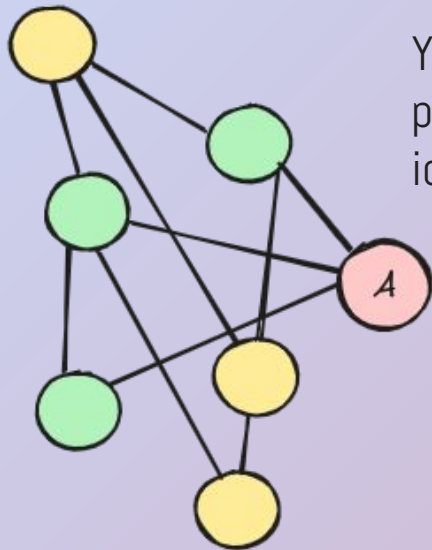
Pero para el cálculo nos interesa $1/\text{deg}(i)$ no el grado, sino su inversa:

$$D^{-1} = \begin{pmatrix} \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{3} \end{pmatrix}$$



Fundamento matemático en GCNs

GCN estandar



Y al igual que con la matriz de adyacencia, contamos al propio nodo como vecino, por eso debemos de sumarle la identidad, cada nodo tiene un vecino más (él mismo):

$$\tilde{D}^{-1} = (D + I)^{-1}$$

$$\tilde{D}^{-1} = \begin{pmatrix} \frac{1}{4} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{5} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{4} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} \end{pmatrix}$$

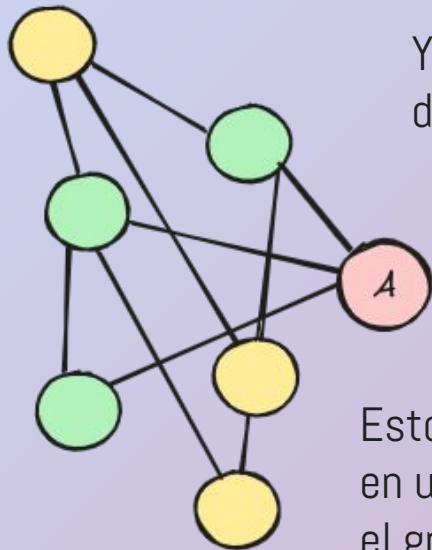


IDAL

Intelligent
Data
Analysis
Laboratory

Fundamento matemático en GCNs

GCN estandar



Y aquí tenemos dos maneras opuestas de incluir la matriz de grado en las ecuaciones:

$$\begin{aligned} \tilde{D}^{-1} \tilde{A} X W^T \\ \tilde{A} \tilde{D}^{-1} X W^T \end{aligned}$$

Esto es lo que proponíamos en un principio dividiendo por el grado de cada nodo.

Pero en el artículo original los autores descubrieron que las características de nodos con muchos vecinos, se ven demasiado "diluidas" debido a dividirse por números altos.

Propusieron una alternativa híbrida.

$$H = \tilde{D}^{-\frac{1}{2}} \tilde{A}^T \tilde{D}^{-\frac{1}{2}} X W^T$$



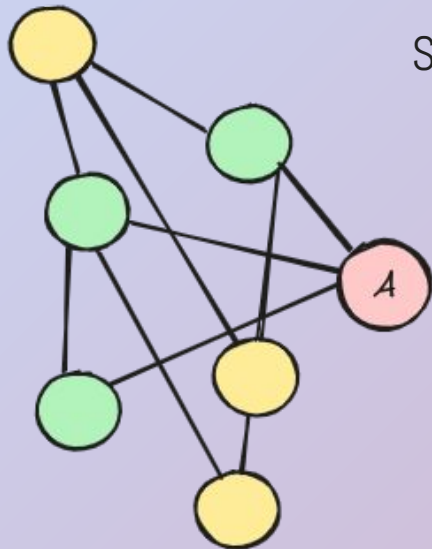
IDAL

Intelligent
Data
Analysis
Laboratory

<https://arxiv.org/pdf/1609.02907>

Fundamento matemático en GCNs

GCN estandar



Sin expresarlo con álgebra matricial:

$$H = \tilde{D}^{-\frac{1}{2}} \tilde{A}^T \tilde{D}^{-\frac{1}{2}} X W^T$$

For A in nodos:

$$h_A = \sum_{i \in \mathcal{N}_A} \frac{1}{\sqrt{\deg(i)} \sqrt{\deg(A)}} x_i W^T$$

Nota: Aquí "A" hace referencia al nodo A, no a la matriz de adyacencia.

Aquí nos quedamos el día 1



Intelligent
Data
Analysis
Laboratory

Prácticas

Veamos su aplicación en código

Práctica sobre GCNs

Colab Notebook:

<https://drive.google.com/file/d/1d38sO2ZU3OpSMYWT6wE6xHTW6TpSI2J2/view?usp=sharing>





Caso concreto Redes de Grafos con Atención (GANs)

¿Cómo implementar los mecanismos de atención en las GNNs?

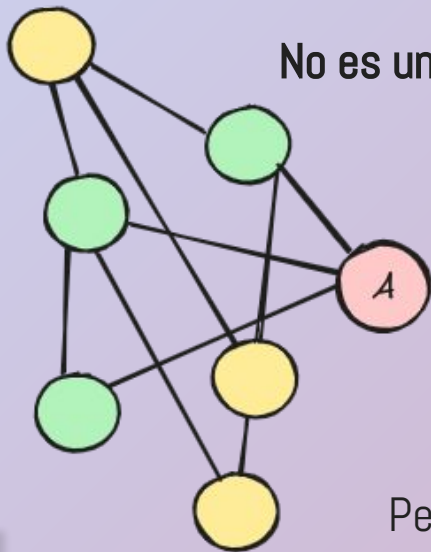


IDAL

Intelligent
Data
Analysis
Laboratory

Fundamento matemático en GANs

En el caso anterior, se le prestó la misma atención a cada uno de los vecinos del nodo a calcular:



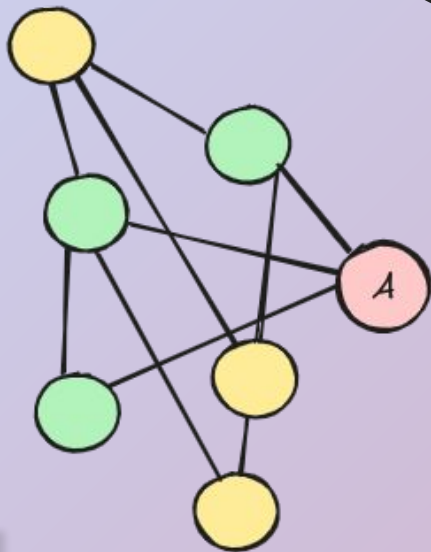
No es una suma ponderada, cada vecino tiene la misma importancia.

$$h_A = \sum_{i \in \mathcal{N}_A} x_i W^T$$

Pero esto no es cierto en general. A veces unos vecinos serán más importantes que otros.

Fundamento matemático en GANs

Añadimos un término de hace referencia a la importancia entre el nodo A y el nodo vecino "i".



$$h_A = \sum_{i \in \mathcal{N}_A} \alpha_{iA} x_i W^T$$

Prácticas

Veamos su aplicación en código

Práctica sobre GAN

Colab Notebook:

