Graph Neural Networks: An Introduction

Michael Mateju

Institute/Organization

February 5, 2025

Introduction

- Graph Neural Networks (GNNs) are gaining popularity in AI.
- Leading companies like Google, Uber, and Twitter are adopting GNNs.
- GNNs are effective in modeling relationships in complex datasets.

Advantages of Graph Machine Learning

- GNNs excel at capturing complex relationships **beyond Euclidean space**.
- They allow for **node embeddings** that represent entities in a meaningful way.
- They improve performance in **graph-based tasks** like link prediction and clustering.

Applications of GNNs

- **Drug Discovery:** Predicting molecular interactions for new medicines.
- **Social Networks:** Recommending friends and filtering misinformation.
- **Fraud Detection:** Identifying fraudulent transactions in banking.
- **Traffic Optimization:** Predicting road congestion and enhancing navigation.

Graph Neural Networks in Data Mining

- GNNs enhance traditional **data mining techniques** by understanding **relational data**.
- They can uncover hidden patterns in **large-scale graphs**.
- Applications in:
  - **Anomaly Detection:** Identifying outliers in **financial transactions**
  - **Knowledge Graphs:** Enhancing search engines with better **semantic understanding**.
  - **Clustering:** Grouping similar entities based on relationships.

Why GNNs Matter in 2024

- GNNs are transitioning from research to **industry-wide adoption**.
- They offer new ways to **leverage graph-based data** across different fields.
- The **future of AI** will involve more **graph learning** to improve predictions and decision-making.

What are Graph Neural Networks?

- Neural networks designed for **graph-structured data**.
- Unlike traditional models, they process **nodes and edges**.
- Applications: **Social networks, molecules, recommendation systems, knowledge graphs**.
- GNNs leverage message passing and graph convolution techniques to learn representations.

Graph Representation

A graph is represented as:

$$G = (V, E)$$

- V - Set of **nodes (vertices)**.
- E - Set of **edges** (connections).
- Adjacency Matrix A:

$$A_{ij} = \begin{cases} 1, & \text{if edge exists between } i \text{ and } j \\ 0, & \text{otherwise} \end{cases}$$

- Feature matrix X where X_i represents node features.

How Do GNNs Work?

Message Passing (Graph Convolution):

$$h_i^{(l+1)} = \sigma\left(W^{(l)} \sum_{j \in \mathcal{N}(i)} \frac{h_j^{(l)}}{c_{ij}} + B^{(l)} h_i^{(l)}\right)$$

- Nodes update their states by aggregating information from neighbors.
- Iterative process across layers.
- Preserves local connectivity and captures relational dependencies.

Graph Convolutional Networks (GCN)

Update Rule:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}\right)$$

- $\tilde{A} = A + I$ (Adjacency matrix with self-loops)
- $\tilde{D}$ is the degree matrix
- $W^{(l)}$ is a trainable weight matrix
- Introduces smoothing and enables feature propagation.

Graph Attention Networks (GAT)

Attention Mechanism:

$$\alpha_{ij} = \frac{\exp(LeakyReLU(a^T[Wh_i||Wh_j]))}{\sum_{k \in \mathcal{N}(i)} \exp(LeakyReLU(a^T[Wh_i||Wh_k]))}$$

- Uses learnable attention weights.
- Dynamically assigns importance to different neighbors.
- Improves performance on heterogeneous graphs.

GraphSAGE: Scalable Learning on Graphs

- GraphSAGE samples neighbors instead of using all neighbors.
- Aggregation function (e.g., mean, max-pooling, LSTM-based).
- Computationally efficient for large graphs.

Training Graph Neural Networks

- Loss function: Cross-entropy, mean squared error, contrastive loss.
- Backpropagation through graph layers.
- Mini-batch training for large-scale graphs.

Training Steps:

- For each training epoch:
  - For each node v_i in batch:
    - Aggregate features from neighbors.
    - Apply non-linear transformation.
    - Compute loss and update weights.

Applications of GNNs

Michael Mateju

Application Area	Use Cases
Social Networks	Friend recommendations, community detecti
Biology	Drug discovery, protein interactions
Finance	Fraud detection, credit risk modeling
Recommendations	Personalized content recommendations
Knowledge Graphs	Semantic search, entity linking

Challenges in GNNs

- **Scalability** - Large graphs require optimized memory usage.
- **Over-smoothing** - Deep networks make node representations indistinguishable.
- **Graph Heterogeneity** - Different node types require specialized architectures.
- **Dynamic Graphs** - Many real-world graphs evolve over time.
- **Limited Label Availability** - Semi-supervised learning often required.

Summary

- GNNs extend deep learning to graphs using message passing.
- Various architectures: **GCN, GAT, GraphSAGE**.
- Applications in social networks, chemistry, finance, and more.
- Challenges include scalability and over-smoothing.

Thank You!

Questions?