Machine Learning Project
4rd Year Computer Engineering

Recommendation System

Student Info:

Student Name	Section Number	Seat Number
Farah Mohsen Samy	3	43253045
Mariam Ehab Mostafa	2	43253079
Steven Bahaa Zarif	1	43253031
Ali Azouz Ali	1	43253040
Zeyad Ashraf Mahmoud	1	43253028

Table of Contents

Introduction

The recommendation system has become a cornerstone of modern e-commerce platforms, helping users discover products tailored to their preferences. This project focuses on building a product recommendation system leveraging graph neural networks (GNNs), particularly LightGCN, to improve prediction accuracy and scalability.

The system is designed to recommend products to users based on timestamp, user ids, product ids and ratings, providing a personalized shopping experience.

Project Overview

The core objective of this project is to develop a recommendation system using **LightGCN**.

The application is built using **FastAPI** for backend services and **PyTorch** Geometric for the implementation of the graph-based neural network.

Key Features:

- **Data Processing**: Preprocessing user and item data with label encoding and scaling for model input.
- **Model Architecture**: Using LightGCN to generate embeddings for users and items.
- **Recommendation Logic**: Computing top-k recommendations for users based on the similarity of embeddings.
- **API Integration**: FastAPI endpoints for fetching and displaying recommendations.
- **Deployment**: The application is deployable using Uvicorn.

Technology Used

Programming Languages and Frameworks:

- **Python**: Core language for data processing and model implementation.
- **FastAPI**: Framework for building RESTful APIs.
- **PyTorch Geometric**: Library for implementing graph neural networks.

Tools and Libraries:

- **Pandas**: For data manipulation and preprocessing.
- **Scikit-learn**: For label encoding and scaling.
- **Jinja2**: For rendering HTML templates.
- **Uvicorn**: For serving the **FastAPI** application.

Comparison Between NGCF and LightGCN

NGCF (Neural Graph Collaborative Filtering):

- **Model Complexity**: NGCF introduces non-linear transformations, making it more complex.
- **Feature Propagation**: Combines embeddings with multi-layer perceptrons (MLPs) for feature aggregation.
- **Performance**: Higher computational cost due to added non-linearity.

LightGCN:

- **Model Simplicity**: Removes non-linear transformations, focusing on message-passing and aggregation.
- **Efficiency**: Faster training and inference due to reduced complexity.
- **Performance**: Outperforms NGCF in many collaborative filtering tasks by directly optimizing embeddings.

Use Cases:

- **NGCF:**
  - Suitable for applications where:
    - Rich side information (features) is available.
    - The graph structure is not extremely large.
- **LightGCN:**
  - Preferred for:
    - Sparse datasets and large-scale graphs.
    - Scenarios where computational efficiency is critical.

Why LightGCN?

LightGCN was chosen over NGCF due to its simplicity, efficiency, and effectiveness.

By eliminating unnecessary non-linear components, LightGCN significantly reduces computational overhead while maintaining or improving recommendation accuracy.

The model's design aligns well with the sparse nature of user-item interaction graphs, making it ideal for large-scale datasets.

Including the most essential component of GCN — neighborhood aggregation — for collaborative filtering. Specifically, after associating each user (item) with an ID embedding, we propagate the embeddings on the user-item interaction graph to refine them. We then combine the embeddings learned at different propagation layers with a weighted sum to obtain the final embedding for prediction. The whole model is simple and elegant, which not only is easier to train, but also achieves better empirical performance than NGCF.

$$\mathbf{e}_u^{(k+1)} = \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}} \mathbf{e}_i^{(k)},$$

$$\mathbf{e}_i^{(k+1)} = \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|}\sqrt{|\mathcal{N}_u|}} \mathbf{e}_u^{(k)}.$$

When to Choose LightGCN

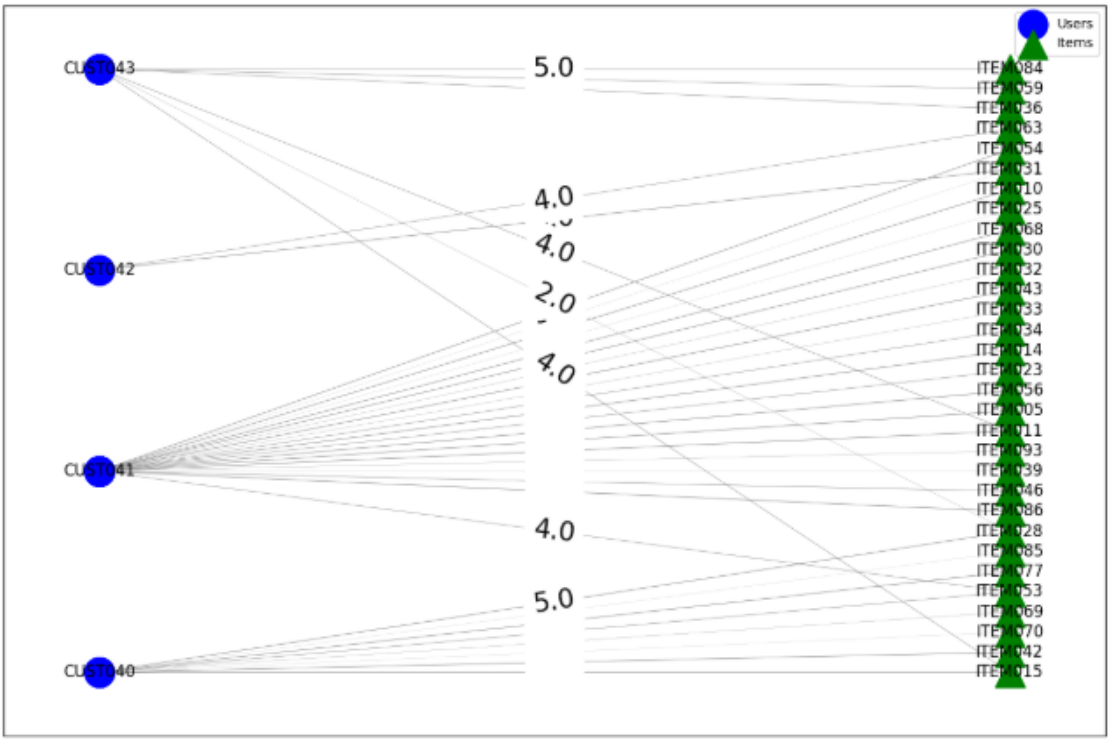
- **Large-scale graphs:** When dealing with millions of users and items.
- **Sparse datasets:** Where data is limited or scattered.
- **Limited resources:** When computational power or memory is constrained.
- **Faster development and training cycles:** Ideal for quick prototyping and large-scale deployments.

Visualization

Graph-based visualization of user-item interactions for the recommendation system. Key elements include:

- **Clusters**: Users (blue nodes) are grouped into clusters based on similarities in their interactions with items (green nodes).
- **Connections**: Edges represent relationships between users and items, with weights (e.g., 4.0, 5.0) reflecting interaction strength or ratings.
- **Insights**: This structure illustrates how LightGCN leverages graph-based embeddings to capture relationships and cluster similar users or items, improving recommendation accuracy.

This visualization emphasizes the role of graph neural networks in identifying patterns and relationships for personalized recommendations.



Conclusion

This project demonstrates the potential of graph-based approaches in building efficient and accurate recommendation systems.
By leveraging **LightGCN**, we achieved a balance between computational efficiency and recommendation quality.
The use of **FastAPI** ensures scalability and ease of integration into real-world applications, making this system a robust solution for personalized product recommendations.
Moreover, the project highlights the importance of careful model selection in addressing specific challenges such as scalability and data sparsity.
LightGCN's architecture proved to be a perfect fit for handling the large and sparse user-item interaction data, making it possible to deliver relevant recommendations effectively.

Table 5: Performance of the 3-layer LightGCN with different choices of normalization schemes in graph convolution.

Dataset	Gowalla		Yelp2018		Amazon-Book	
Method	recall	ndcg	recall	ndcg	recall	ndcg
LightGCN-L_1-L	0.1724	0.1414	0.0630	0.0511	**0.0419**	**0.0320**
LightGCN-L_1-R	0.1578	0.1348	0.0587	0.0477	0.0334	0.0259
LightGCN-L_1	0.159	0.1319	0.0573	0.0465	0.0361	0.0275
LightGCN-L	0.1589	0.1317	0.0619	0.0509	0.0383	0.0299
LightGCN-R	0.1420	0.1156	0.0521	0.0401	0.0252	0.0196
LightGCN	**0.1830**	**0.1554**	**0.0649**	**0.0530**	0.0411	0.0315

Method notation: -L means only the left-side norm is used, -R means only the right-side norm is used, and -L_1 means the L_1 norm is used.

Future work could explore integrating additional user and item features, such as demographic data and textual product descriptions, to further enhance the system's capabilities.
Additionally, deploying this system in a real-world scenario could provide valuable insights into its performance and areas for improvement.

Overall, this project serves as a foundational step in utilizing graph-based neural networks for recommendation tasks.