Product Recommendation System Using Graph Neural Network

Yuliana Lavryk, Yurii Kryvenchuk

Lviv Polytechnic National University, Stepana Bandery Street 12, Lviv, 79013, Ukraine

Abstract

The use of graph neural networks (GNNs) in product recommendation systems has gained attention in recent years due to its ability to capture complex relationships between products and customers. This approach can incorporate both explicit and implicit feedback from customers, as well as contextual information such as time and location. GNNs can become a powerful tool for personalized and accurate recommendations. While there are challenges associated with building GNN-based recommendation systems, the potential benefits make it an exciting area of research and development for e-commerce businesses. This paper explores the novelty and advantages of using GNNs in product recommendation systems and discusses the challenges and future directions of this approach.

Keywords

Graph Neural Networks (GNN), Recommendation System (RS), Product, GraphSAGE

1. Introduction

Building product recommendation systems using GNNs is a relatively new approach compared to traditional methods such as collaborative filtering and content-based filtering. ***Collaborative filtering*** (CF) works by analyzing the past behavior of users, such as their previous purchases, ratings, or reviews, to make recommendations for products that other similar users have liked or purchased. This approach relies on finding similarities between users based on their past behavior, which can be challenging for new users who have not provided any feedback yet [1]. ***Content-based filtering*** (CBF) works by analyzing the attributes of the items themselves, such as their features, descriptions, or images, and recommending items that are similar to those the user has previously shown an interest in. This approach is more effective for new users as it relies on the features of the items rather than past behavior. However, it can be limited by the availability and quality of item attributes [2]. ***Graph neural networks*** are a type of neural network that can operate on graph data structures, such as social networks or product co-purchasing networks. In the context of recommendation systems, GNNs can be used to learn representations of users and items based on their relationships in the graph, and to make recommendations by predicting the likelihood of a user interacting with a particular item in the future. GNNs can combine information from both collaborative and content-based approaches and can be more effective for handling complex interactions and relationships between users and items [3].

GNNs have the ability to model complex relationships between products and customers, such as the influence of a customer's friends and family members on their purchasing behavior. This approach is particularly useful when dealing with large datasets where the number of products and customers can be overwhelming.

One of the novel aspects of using GNNs for product recommendation systems is the ability to incorporate both explicit and implicit feedback from customers. Explicit feedback refers to customer ratings and reviews, while implicit feedback refers to the customer's purchase history and browsing

[1] *MoMLeT+DS 2023: 5th International Workshop on Modern Machine Learning Technologies and Data Science, June 3, 2023, Lviv, Ukraine*
EMAIL: ylilav55547@gmail.com (Yu. Lavryk); yurkokryvenchuk@gmail.com (Yu. Kryvenchuk)
ORCID: 0000-0003-4223-3403 (Yu. Lavryk); 0000-0002-2504-5833 (Yu. Kryvenchuk)

behavior. GNNs can incorporate both types of feedback to provide more accurate and personalized recommendations.

Another novel aspect of using GNNs for product recommendation systems is the ability to incorporate contextual information such as time and location. For example, a customer's purchasing behavior may change depending on the time of day or their location. GNNs can capture these contextual factors and use them to make more accurate recommendations.

Furthermore, GNNs can handle such a problem as lack of sufficient information about a new customer or product to make recommendations. GNNs can leverage the information from other similar customers or products in the graph to make recommendations for the new customer or product.

In summary, while collaborative filtering and content-based filtering are more traditional approaches, GNNs can provide a more powerful and flexible solution for product recommendation systems, especially in cases where there are complex interactions and relationships between users and items. The use of GNNs in product recommendation systems represents a significant advancement in the field of recommendation systems. The ability to capture complex relationships and incorporate contextual information makes GNNs a powerful tool for effective and high-quality recommendations.

2. Literature Review

The literature on recommendation systems covers various approaches, including collaborative filtering, content-based filtering, and hybrid models (Figure 1).

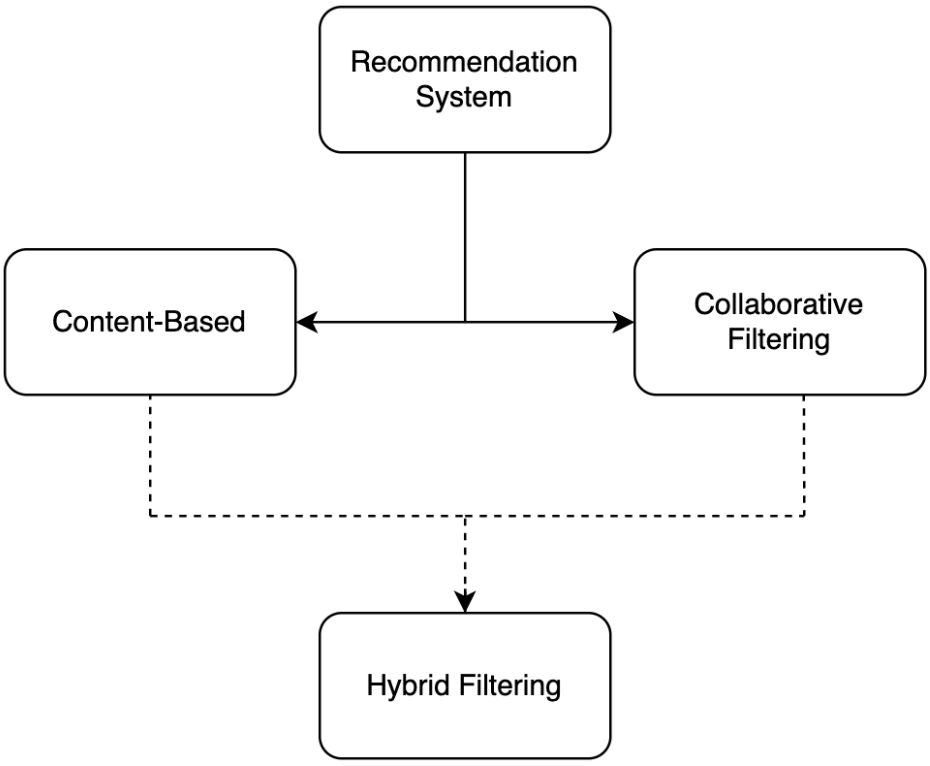

Figure 1: Common approaches to building recommendation system

More recently, the use of graph neural networks (GNNs) in recommendation systems has been gaining attention due to their ability to capture complex relationships between entities in the graph.

A study by Ying et al. [4] proposed a GNN-based recommendation system called GraphRec. The system uses a GNN to learn embeddings of users and products in a graph and uses the embeddings to make personalized recommendations. GraphRec was shown to outperform state-of-the-art collaborative filtering and content-based recommendation systems on several benchmark datasets.

Another study by Wang et al. [5] proposed a GNN-based recommendation system to learn embeddings of items and users based on the graph structure and uses the embeddings to make

recommendations. System was shown to outperform traditional collaborative filtering and content-based recommendation systems on several real-world datasets.

Furthermore, there have been studies on using GNNs for session-based recommendation systems, where the goal is to recommend items to users based on their current session rather than their entire history. A study by Wu et al. [6] proposed a session-based recommendation system called SR-GNN that uses a GNN to model the sequence of user interactions as a graph and generates recommendations based on the learned embeddings.

The authors of [7] propose a new model of a recommender system based on a dynamic graph neural network (DGN). The advantages of such a model are the ability to take into account changes in user behavior over time, as well as to process different types of data, including text, video and images. However, this type of model requires a large amount of data to train and powerful resources to run, which can often be limiting factors. The goal of my research is to build a model that will recognize deep dependencies among smaller amounts of data, which will provide high quality of recommendations and will not require resources such as DGN.

In the article [8], a new approach to the recommender system based on graph convolutional embeddings (Graph Convolutional Embeddings, GCE) is proposed. The paper demonstrates the high accuracy and efficiency of the GCE model compared to other methods used for recommender systems. But there are the same limitations as when using DGN [7], when there is a need for a very large amount of data for training and, accordingly, significant resources. Another disadvantage is possible limitations and errors when processing graphs with a complex structure. This work is another confirmation that there is a need to build a more flexible system that can work effectively without such a volume of data.

The authors of the study [9] proposed a new approach to the development of a recommender system based on context-oriented graph neural networks. The context-oriented approach is based on the analysis of more detailed factors regarding the user, which improves the quality of recommendations and provides a more personalized approach to the user. Another advantage of this method is the ability to adapt to different types of data. This article provides a detailed description of context-aware recommender systems combined with GNNs that can be used to build the own hybrid system. At the same time, this approach may require retraining the model when adding new data, as there is not enough information to make a recommendation.

The work [10] describes the use of graph neural networks for recommender systems in electronic commerce. The study authors describe the Item Relationship Graph Neural Network (IRGNN) model, which uses graph neural networks to predict recommendations. The model uses information about the relationship between products to improve the quality of recommendations. The main advantage of IRGNN is that it can take into account complex dependencies between products, such as categories, product similarities, etc. This allows the model to make more accurate recommendations, which increases the efficiency of e-commerce. However, the authors did not investigate some key aspects of IRGNN implementation, such as speed and scalability. It is also important to be aware that this model can be more difficult to establish compared to more traditional recommendation methods. The work demonstrates a high level of benefit of using recommender systems in e-commerce.

However, there are also challenges associated with building GNN-based recommendation systems, such as scalability and the need for large amounts of data. A study by Monti et al. [11] proposed a scalable GNN-based recommendation system that can handle large-scale graphs. The system uses a variant of the GNN that allows efficient message passing and can be parallelized across multiple GPUs.

Overall, the literature suggests that GNN-based recommendation systems have shown promising results in improving the accuracy and personalization of recommendations. However, there are still challenges that need to be addressed, such as scalability and data sparsity.

3. Overview of Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) are a type of neural network designed to operate on graph-structured data. A graph is a mathematical structure consisting of a set of nodes (also called vertices) and a set of edges connecting pairs of nodes. GNNs are used to learn node and graph representations in complex, large-scale networks such as social networks, citation networks, and molecular graphs [12].

GNNs are built upon the idea of message passing, where information is propagated from node to node through the edges of a graph. Each node in the graph has a feature vector that represents its properties, and each edge has a weight that represents the relationship between two connected nodes. The goal of GNNs is to learn a representation of each node that captures its role in the graph, as well as a representation of the entire graph. GNNs consist of several layers of neural networks, each of which performs two main operations: message passing and node update. In the message passing step, the node features are updated based on the features of its neighboring nodes and the edge weights between them. In the node update step, the updated node features are aggregated to produce a new representation of the node. The aggregation function can be as simple as taking the average of the node features or more complex, such as a weighted sum or attention-based mechanism.

GNNs have been used in a wide range of applications, including recommendation systems, drug discovery, social network analysis, and image recognition. One of the strengths of GNNs is their ability to handle large-scale, complex networks with multiple types of nodes and edges. GNNs have also shown to be effective in handling noisy and incomplete data and are able to learn representations that capture the underlying structure and patterns in the data [13].

In recent years, there have been several advancements in the field of GNNs, including the development of more efficient message passing algorithms, the integration of attention mechanisms, and the use of GNNs in unsupervised and semi-supervised learning tasks. With continued research and development, GNNs are expected to become an increasingly important tool in the analysis and understanding of complex networks.

4. Challenges in building GNN-based Recommendation System

Building a GNN-based recommendation system can present several challenges, including:

Data sparsity: In recommendation systems, the user-item interaction data may be sparse, meaning that many users have only interacted with a small number of items. This can make it difficult for the GNN to learn accurate representations of users and items based on their interactions.

Cold-start problem: When a new item is introduced to the system, there may be no user interactions with it, making it difficult for the GNN to make accurate recommendations. Similarly, when a new user joins the system, there may be insufficient data to make accurate recommendations for them.

Scalability: The size of the graph can grow rapidly with the number of users and items in the system, making it difficult to train the GNN efficiently. This can lead to longer training times and increased computational resources.

Overfitting: GNNs can suffer from overfitting when the model is too complex or when there is insufficient data to generalize accurately. This can result in poor performance when making recommendations for new users or items.

Diversity: Recommendation systems may need to balance accuracy with diversity, ensuring that the recommended items are not too similar to each other. GNNs may need to incorporate additional diversity metrics to achieve this.

Interpretability: GNNs can be difficult to interpret, making it challenging to understand how the model is making recommendations. This can be problematic for applications where interpretability is critical, such as in healthcare or finance.

Overall, building a GNN-based recommendation system requires addressing these challenges to ensure that the system is accurate, scalable, and interpretable.

5. Data for Recommendation System based on GNN

There are several sources of graph structure data that can be used to build a recommendation system. The most popular and suitable for building recommendation systems based on GNN are Amazon data. Some of the common sources include:

Amazon product metadata: This data includes information such as product categories, descriptions, and attributes. It can be used to construct a graph that connects products based on their metadata.

Amazon customer reviews: This data includes product reviews and ratings by users. It can be used to construct a bipartite graph that connects users and products based on their review history.

Amazon purchase history: This data includes information about which products were purchased by which users. It can be used to construct a bipartite graph that connects users and products based on their purchase history.

Amazon product co-occurrence: This data includes information about which products are frequently purchased together. It can be used to construct a graph that connects products based on their co-occurrence.

Amazon product browsing history: This data includes information about which products were browsed by which users. It can be used to construct a bipartite graph that connects users and products based on their browsing history [14].

It's worth noting that each of these data sources may require preprocessing and cleaning before they can be used to construct a suitable graph for training a GNN-based recommendation system. Additionally, the choice of graph structure data will depend on the specific requirements of the recommendation system, as well as the availability and quality of the data.

6. Data Preprocessing for GNN-based Recommendation Systems

Data preprocessing is a critical step in building any machine learning model, including GNN-based recommendation systems. In this section, the various data preprocessing steps involved in building a GNN-based recommendation system will be discussed.

1. Data Cleaning and Transformation: The first step in data preprocessing is to clean and transform the raw data into a format suitable for GNN-based recommendation systems. This involves removing missing values, transforming categorical variables into numerical variables, and normalizing numerical variables [15].
2. Graph Construction: GNN-based recommendation systems operate on graph-structured data, where users and products are represented as nodes in the graph and their interactions as edges. The second step in data preprocessing is to construct the graph from the cleaned data. The graph can be constructed using different approaches, such as user-item bipartite graphs or user-user and item-item co-occurrence graphs [16].
3. Node and Edge Feature Engineering: The next step is to engineer features for nodes and edges in the graph. This involves defining features for nodes and edges based on the available data, such as user demographics, product attributes, and interactions between users and products. The features can be used to learn embeddings of nodes and edges using GNNs [17].
4. Train-Validation-Test Split: The next step is to split the graph into training, validation, and test sets. The training set is used to learn the GNN model, the validation set is used to tune hyperparameters, and the test set is used to evaluate the performance of the model [18].
5. Negative Sampling: In recommendation systems, the number of negative interactions (i.e., interactions between users and products that did not occur) is much larger than the number of positive interactions. Negative sampling is a technique used to address this class imbalance by randomly sampling negative interactions for training the model [19].
6. Data Augmentation: Data augmentation is a technique used to increase the size of the training data by generating synthetic examples. In recommendation systems, data augmentation can be used to simulate different user interactions, such as adding noise to existing interactions or creating new interactions between similar products.[20].

Data preprocessing is a crucial step in building GNN-based recommendation systems. It involves cleaning and transforming the raw data, constructing the graph, engineering features for nodes and edges, splitting the graph into training, validation, and test sets, performing negative sampling, and using data augmentation to increase the size of the training data.

7. GNN Structure for Recommendation System

The structure of a graph neural network (GNN) for a recommendation system can vary depending on the specific requirements of the application. However, a typical GNN structure for recommendation systems can be divided into the following components:

Input Layer: The input layer of the GNN takes as input the node features for both users and products. These features can include categorical features such as product category, brand, and user location, as well as numerical features such as product price and user age.

Convolutional Layers: The convolutional layers of the GNN perform message passing between nodes in the graph. The message passed between nodes can be a function of the features of both nodes, as well as the relationship between them in the graph. In the case of a recommendation system, the convolutional layers can capture the similarity between users and products based on their past interactions and features.

Pooling Layers: The pooling layers of the GNN aggregate information from the convolutional layers across the graph. This can be done by computing the mean or max of the node features in a neighborhood of nodes. The pooling layers can also be used to perform graph-level feature extraction.

Output Layer: The output layer of the GNN predicts the likelihood of a user interacting with a particular product. This can be done by computing a similarity score between the user and the product based on their learned representations. The output layer can also be used to generate a ranked list of recommended products for a given user.

General structure of GNN is shown in Figure 2.

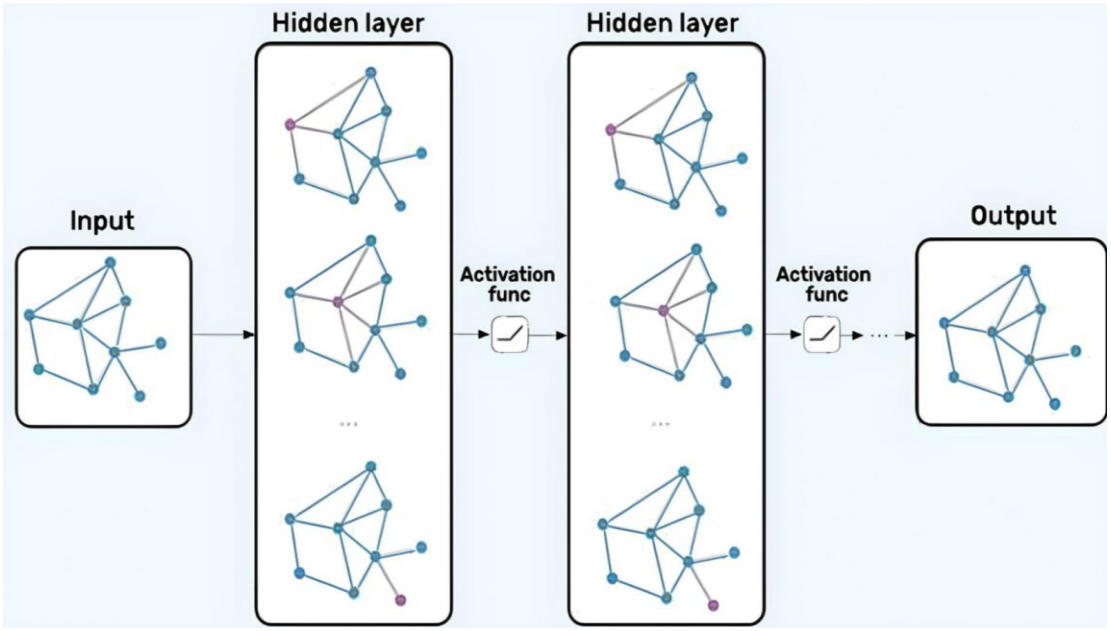

Figure 2: General structure of GNN. [21], via AI Summer. (https://theaisummer.com/Graph_Neural_Networks/).

Overall, the structure of a GNN for recommendation systems involves learning representations of users and products based on their interactions and features in a bipartite graph (Figure 3). This allows for the generation of personalized recommendations for users based on their past behavior and preferences.

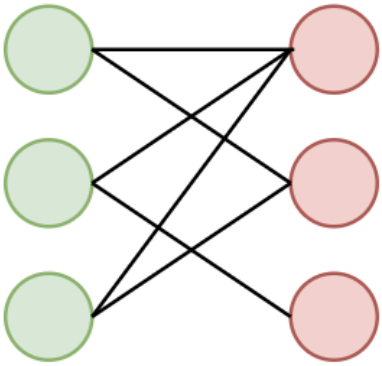

Figure 3: Example of Bipartite Graph

8. GNN Model Architecture for Product Recommendation System

The GNN model architecture for product recommendation involves several components, including graph construction, node and edge feature engineering, GNN layers, and output layer. In this section, we will discuss each of these components in detail.

- Graph Construction: The first step in building a GNN-based recommendation system is to construct the graph from the preprocessed data. The graph can be constructed using different approaches, such as user-item bipartite graphs, user-user and item-item co-occurrence graphs, or heterogeneous graphs that include multiple types of nodes and edges.
- Node and Edge Feature Engineering: The next step is to engineer features for nodes and edges in the graph. This involves defining features for nodes and edges based on the available data, such as user demographics, product attributes, and interactions between users and products. The features can be used to learn embeddings of nodes and edges using GNNs.
- GNN Layers: The core of the GNN-based recommendation system is the GNN layers that learns node embeddings by aggregating information from their neighboring nodes and edges in the graph. The GNN layers typically consist of two steps: message passing and node update. In the message passing step, each node sends a message to its neighboring nodes based on their edge features. In the node update step, each node aggregates the received messages and updates its own embedding. This process is repeated for multiple layers to capture increasingly complex relationships in the graph.
- Output Layer: The final step in the GNN model architecture is the output layer, which generates the recommendations for each user based on their learned node embeddings. The output layer can be implemented using various approaches, such as dot product, cosine similarity, or neural networks that combine user and item embeddings.
- Several variations of the GNN model architecture have been proposed for product recommendation systems. For example:
  - GraphSAGE (Hamilton et al.) [22] learns a node's representation by aggregating information from its neighbors, which can be useful for product recommendations in cases where there is limited information about a particular item.
  - PinSage (Wang et al.) [23] uses a graph neural network to learn embeddings of items and users based on the graph structure. Additionally, there have been studies on using attention mechanisms to improve the performance of GNN-based recommendation systems.
  - Graph Convolutional Network (GCN) - one more popular graph neural network architecture that has been applied to product recommendation systems. The basics are convolutional neural networks which operate directly on graphs [24].
  - Graph Attention Network (GAT) - another popular graph neural network architecture that can be used for product recommendation systems. GATs use attention mechanisms to weight the importance of different nodes in a graph, which can be useful for learning personalized recommendations [25].
- Neural Graph Collaborative Filtering (NGCF) - This is a graph neural network architecture that has been specifically designed for collaborative filtering tasks such as product recommendations. NGCF

uses a weighted sum of the embeddings of a user's neighbors to generate personalized recommendations [23].

The GNN model architecture for product recommendation involves constructing the graph, engineering node and edge features, using GNN layers to learn node embeddings, and using an output layer to generate recommendations for each user. Several variations of the GNN model architecture have been proposed, each with its strengths and limitations. In this work several different GNN models will be trained and compared so that the general pipeline will look like shown in Figure 4.

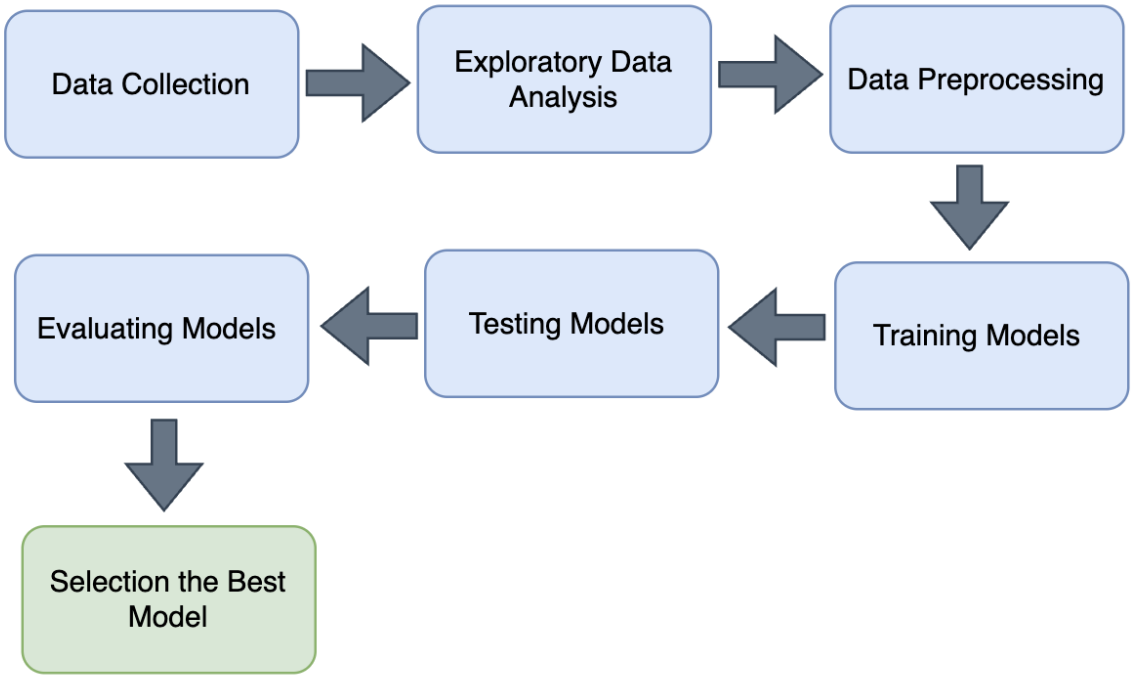

Figure 4: Pipeline of building product recommendation system using GNN

Next, one of the final steps - model evaluation - will be described.

9. Evaluation of GNN-based Recommendation System

Evaluation of GNN-based recommendation systems is critical to determine their performance and effectiveness in recommending products to users. In this section, we will discuss the metrics and techniques used to evaluate the performance of GNN-based recommendation systems.

1. Evaluation Metrics: There are several metrics used to evaluate the performance of recommendation systems, including precision, recall, F1-score, and mean average precision (MAP). Precision measures the fraction of recommended products that are relevant to the user, while recall measures the fraction of relevant products that are recommended. The F1-score is the harmonic mean of precision and recall. MAP is a measure of the average precision at different levels of recall. Also, one more wide-spread metric to evaluate the performance of a recommendation system is top-K accuracy. It measures the percentage of test instances where the true positive recommendation is in the top-K predicted recommendations [26]. These metrics can be used to evaluate the accuracy, relevance, and diversity of recommendations generated by GNN-based recommendation systems.
2. Splitting Strategy: To evaluate the performance of GNN-based recommendation systems, the data is typically split into training, validation, and test sets. The training set is used to train the GNN model, the validation set is used to tune hyperparameters, and the test set is used to evaluate the

performance of the model on unseen data. Different splitting strategies can be used, such as random splitting, temporal splitting, or user-based splitting, depending on the characteristics of the dataset.

3. Ablation Study: Ablation study is a technique used to evaluate the contribution of each component of the GNN model to the overall performance. This involves training and testing the GNN model with and without each component to determine its impact on the performance. Ablation study can help identify which components are essential for the performance of the GNN-based recommendation system [27].

Several studies have evaluated the performance of GNN-based recommendation systems using the above techniques. For example, the study by Ying et al. [22] evaluated the performance of GraphSAGE on the MovieLens dataset and compared it with baseline methods, while the study by Wang et al. [23] evaluated the performance of PinSage on the Pinterest dataset and used ablation study to identify the contribution of each component to the performance.

The evaluation of GNN-based recommendation systems involves using evaluation metrics, splitting strategy, baseline methods, and ablation study to determine their performance and effectiveness. These techniques can help identify the strengths and limitations of GNN-based recommendation systems and guide further improvements in the model architecture.

10. Future Directions and Conclusion

In this research work, the use of graph neural networks (GNNs) in product recommendation systems was explored . Discussed the background of recommendation systems, reviewed the literature on GNN-based recommendation systems, and presented the data preprocessing, model architecture, and evaluation techniques for GNN-based recommendation systems.

In summary, GNN-based recommendation systems offer several advantages over traditional recommendation systems, including the ability to capture complex relationships among products and users, and the ability to incorporate auxiliary information such as product attributes and user profiles.

However, there are still several areas for future research and improvement. One area is the development of more sophisticated GNN architectures that can capture higher-order relationships among products and users. Another area is the incorporation of temporal dynamics into GNN-based recommendation systems, which can capture changes in user preferences over time.

Additionally, the scalability and efficiency of GNN-based recommendation systems need to be improved to handle large-scale datasets with millions of products and users. Furthermore, the interpretability of GNN-based recommendation systems needs to be improved to enable users to understand how the recommendations are generated and provide feedback.

In conclusion, GNN-based recommendation systems have shown promise in improving the accuracy and relevance of product recommendations. However, there is still much work to be done in developing more sophisticated GNN architectures, incorporating temporal dynamics, improving scalability and efficiency, and enhancing interpretability. This research work provides a useful starting point for future research in this exciting and rapidly evolving field.

11. References

[1] P. Seeda, A Complete Guide To Recommender Systems — Tutorial with Sklearn, Surprise, Keras, Recommenders, 2021. URL: https://towardsdatascience.com/a-complete-guide-to-recommender-system-tutorial-with-sklearn-surprise-keras-recommender-5e52e8ceace1.

[2] P. Kordík, Machine Learning for Recommender systems — Part 1, 2019. URL: https://medium.com/recombee-blog/machine-learning-for-recommender-systems-part-1-algorithms-evaluation-and-cold-start-6f696683d0ed.

[3] S. Virinchi, Using graph neural networks to recommend related products, 2022. URL: https://www.amazon.science/blog/using-graph-neural-networks-to-recommend-related-products.

[4] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, Dawei Yin, Graph Neural Networks for Social Recommendation, World Wide Web Conference (WWW '19), May 13–17, New York, USA, 2019. doi: https://doi.org/10.1145/3308558.3313488.

[5] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua, Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences, World Wide Web Conference (WWW '19), May 13–17, New York, USA, 2019. doi: https://doi.org/10.1145/3308558.3313705.

[6] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, Tieniu Tan, Session-based Recommendation with Graph Neural Networks, 2019.

[7] Zhang M., Wu S., Yu X., Liu Q., Wang L., Dynamic Graph Neural Networks for Sequential Recommendation, IEEE Transactions on Knowledge and Data Engineering, 2022. doi: https://doi.org/10.48550/arXiv.2104.07368.

[8] Duran P. G., Karatzoglou A., Vitria J., Xin X., Arapakis I. Graph convolutional embeddings for recommender systems, IEEE Access, 2021. doi: https://doi.org/10.48550/arXiv.2103.03587

[9] Li D., Gao Q., Session Recommendation Model Based on Context-Aware and Gated Graph Neural Networks, Computational Intelligence and Neuroscience, 2021. doi: https://doi.org/10.1155/2021/7266960

[10] Weiwen Liu, Yin Zhang; Jianling Wang, Yun He; James Caverlee, Patrick P. K. Chan, Daniel S. Yeung, Item Relationship Graph Neural Networks for E-Commerce, 2021. doi: 10.1109/TNNLS.2021.3060872.

[11] Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., and Bronstein, M. M., Geometric deep learning on graphs and manifolds using mixture model cnns, 2016.

[12] A. Pathak, Graph Neural Networks Explained in 5 Minutes, 2023. URL: https://geekflare.com/graph-neural-networks/.

[13] P. Sharma, What are Graph Neural Networks, and how do they work, 2023. URL: https://www.analyticsvidhya.com/blog/2022/03/what-are-graph-neural-networks-and-how-do-they-work/.

[14] G. Valdata, Building a Recommender System for Amazon Products with Python, 2022. URL: https://towardsdatascience.com/building-a-recommender-system-for-amazon-products-with-python-8e0010ec772c.

[15] D. Kumar, Introduction to Data Preprocessing in Machine Learning, 2018. URL: https://towardsdatascience.com/introduction-to-data-preprocessing-inmachine-learning-a9fa83a5dc9d/.

[16] A. Pearce, A. Wiltschko, B. Sanchez-Lengeling, E. ReifA, Gentle Introduction to Graph Neural Networks, 2021. URL: https://distill.pub/2021/gnn-intro/.

[17] A. Ali Awan, A Comprehensive Introduction to Graph Neural Networks (GNNs), 2022. URL: https://www.datacamp.com/tutorial/comprehensive-introduction-graph-neural-networks-gnns-tutorial.

[18] S. Agrawal, How to split data into three sets (train, validation, and test) And why, 2021. URL: https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c.

[19] A. I. Herghelegiu and H. Lu, Improving Negative Sampling in Graph Neural Networks for Predicting Drug-Drug Interactions, IEEE International Conference on Bioinformatics and Biomedicine, 2021. doi: 10.1109/BIBM52615.2021.9669483.

[20] Tong Zhao, Yozen Liu, Leonardo Neves, Oliver Woodford, Meng Jiang, Neil Shah, Data Augmentation for Graph Neural Networks, 2020. doi: https://doi.org/10.48550/arXiv.2006.06830

[21] S. Karagiannakos, Graph Neural Networks - An overview, 2020. URL: https://theaisummer.com/Graph_Neural_Networks/.
[22] William L. Hamilton, Rex Ying, Jure Leskovec, Inductive Representation Learning on Large Graphs, 2018. doi: https://doi.org/10.48550/arXiv.1706.02216.
[23] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, Tat-Seng Chua, Neural Graph Collaborative Filtering, 2020. doi: https://doi.org/10.1145/3331184.3331267.
[24] I. Mayachita, Understanding Graph Convolutional Networks for Node Classification, 2020. URL: https://towardsdatascience.com/understanding-graph-convolutional-networks-for-node-classification-a2bfdb7aba7b.
[25] Petar Velickovi, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, Graph Attention Networks, 2018. doi: https://doi.org/10.48550/arXiv.1710.10903
[26] Top-K Accuracy Score, 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.top_k_accuracy_score.html.
[27] Shimin Xiong, Shimin Xiong, Bin Li, Shiao Zhu, DCGNN: a single-stage 3D object detection network based on density clustering and graph neural network, 2022.