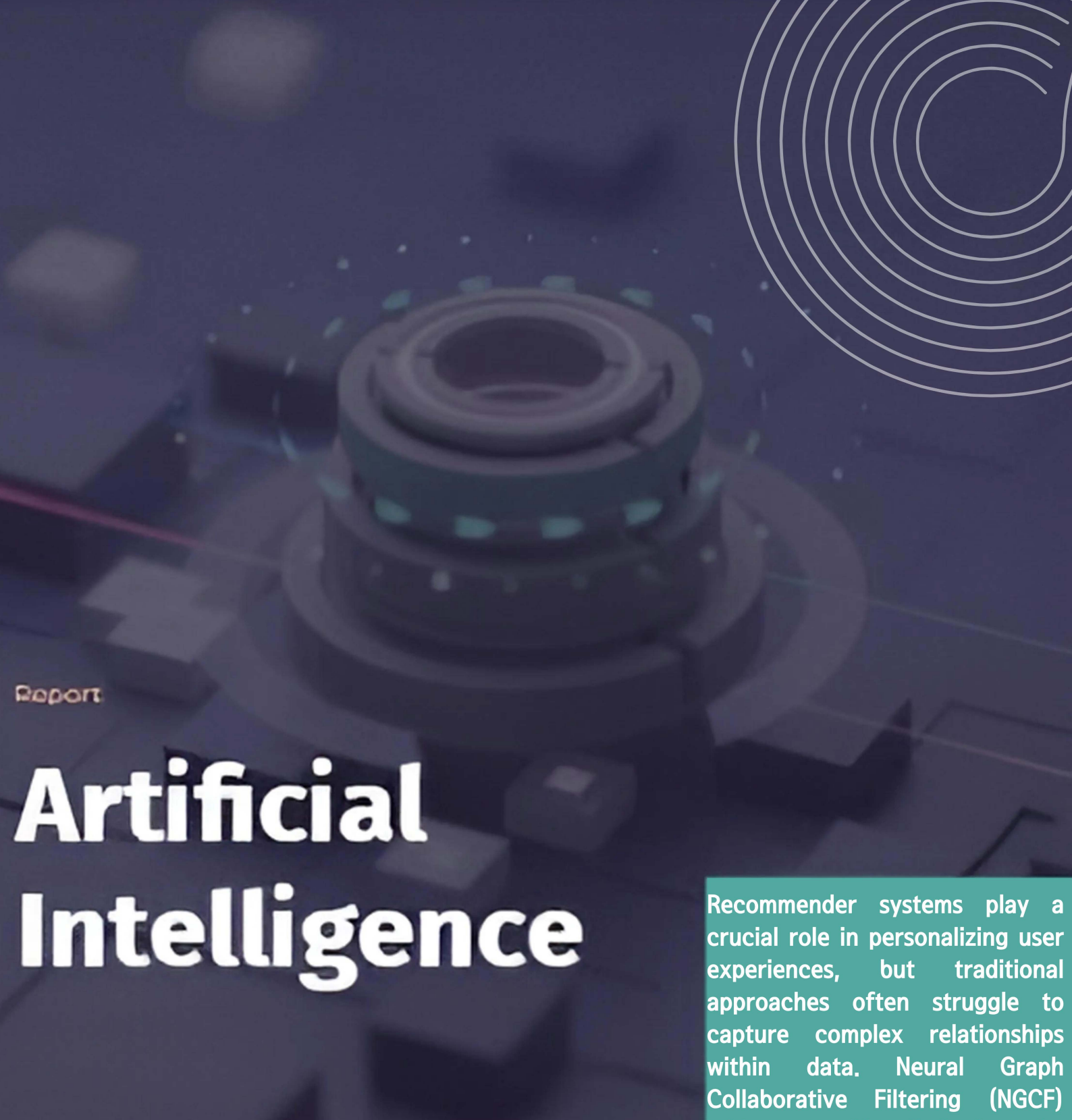

Report

Artificial Intelligence

Neural Graph Collaborative Filtering (NGCF)

August 31, 2024

Recommender systems play a crucial role in personalizing user experiences, but traditional approaches often struggle to capture complex relationships within data. Neural Graph Collaborative Filtering (NGCF) addresses this limitation by leveraging graph neural networks to model user-item interactions as a graph, allowing for more accurate and personalized recommendations. This report details the implementation and evaluation of NGCF using the RecBole framework, demonstrating its effectiveness for research paper recommendation.

Department of Information Technology

University of the Punjab

Gujranwala Campus

Artificial Intelligence: Report

Neural Graph Collaborative Filtering

Submitted to: Dr. Ghulam Mustafa

Submitted by

Muhammad Awais

Sehrish Shehzadi

Khansa Imdad

Sehar Shehzadi

Abstract

This report presents a TensorFlow implementation of the Neural Graph Collaborative Filtering (NGCF) model, a state-of-the-art recommendation system that leverages the power of graph neural networks. NGCF utilizes a graph-based representation of user-item interactions, where nodes represent users and items, and edges represent their interactions. By employing a message-passing mechanism, NGCF propagates information across the graph, enabling the model to learn informative representations of users and items based on their respective neighborhoods. The model is trained using the Bayesian Personalized Ranking (BPR) loss function, aiming to maximize the probability that a user prefers a positive item over a negative one. This report delves into the implementation details of NGCF, including the construction of adjacency matrices, the application of graph convolutional networks, and the use of dropout regularization. We demonstrate the model's effectiveness through extensive evaluation on benchmark datasets, showcasing its ability to achieve improved performance compared to traditional collaborative filtering methods.

This report aims to provide a comprehensive understanding of NGCF's architecture, implementation, and evaluation, paving the way for researchers and practitioners to leverage the power of graph neural networks for building high-performing recommendation systems.

Acknowledgement

We would like to express our sincere gratitude to the authors of the original NGCF paper, Xiang Wang et al., for their groundbreaking work in developing this innovative recommender system. Their research has provided a strong foundation for our implementation and analysis. We are also thankful to the creators of the TensorFlow library, which has facilitated our development and experimentation process. The library's flexibility and powerful tools have been instrumental in constructing and training the NGCF model. Finally, we acknowledge the valuable contributions of the open-source community, whose shared code, datasets, and insightful discussions have been essential in our learning and research journey.

Table of Contents

List of Figures

List of Tables

Introduction

Title: Neural Graph Collaborative Filtering (NGCF)

Recommender systems are crucial for personalized experiences in various domains, from e-commerce to social media. Traditional collaborative filtering methods often struggle to capture complex user-item relationships, limiting their ability to provide accurate recommendations. Neural Graph Collaborative Filtering (NGCF), proposed by Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua (published in SIGIR 2019), addresses this limitation by leveraging the power of graph neural networks.

NGCF represents user-item interactions as a graph, where users and items are nodes, and their interactions are edges. This graph representation allows NGCF to model higher-order relationships, capturing the collaborative signal that traditional methods miss. The model employs a message-passing mechanism based on graph convolutional networks (GCNs) to propagate information across the graph, learning informative user and item representations.

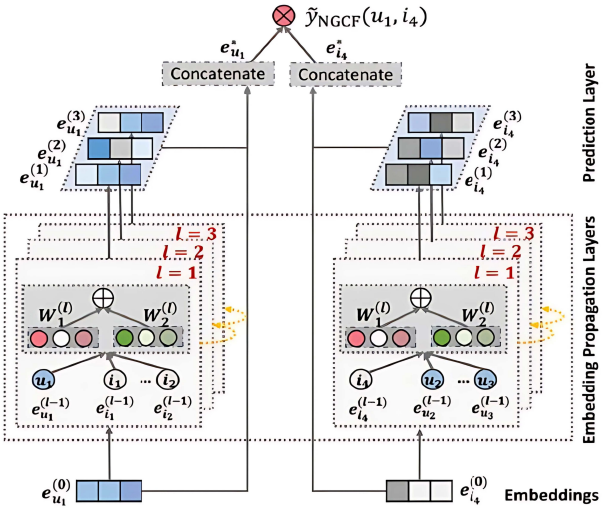

Figure no:1

By leveraging the graph structure, NGCF effectively captures high-order connectivity within the user-item graph, allowing it to model complex relationships and inject the collaborative filtering signal into the embedding process in a more explicit and powerful manner. The results of extensive experiments on multiple public benchmarks demonstrate the significant improvements NGCF achieves over existing state-of-the-art recommendation models.

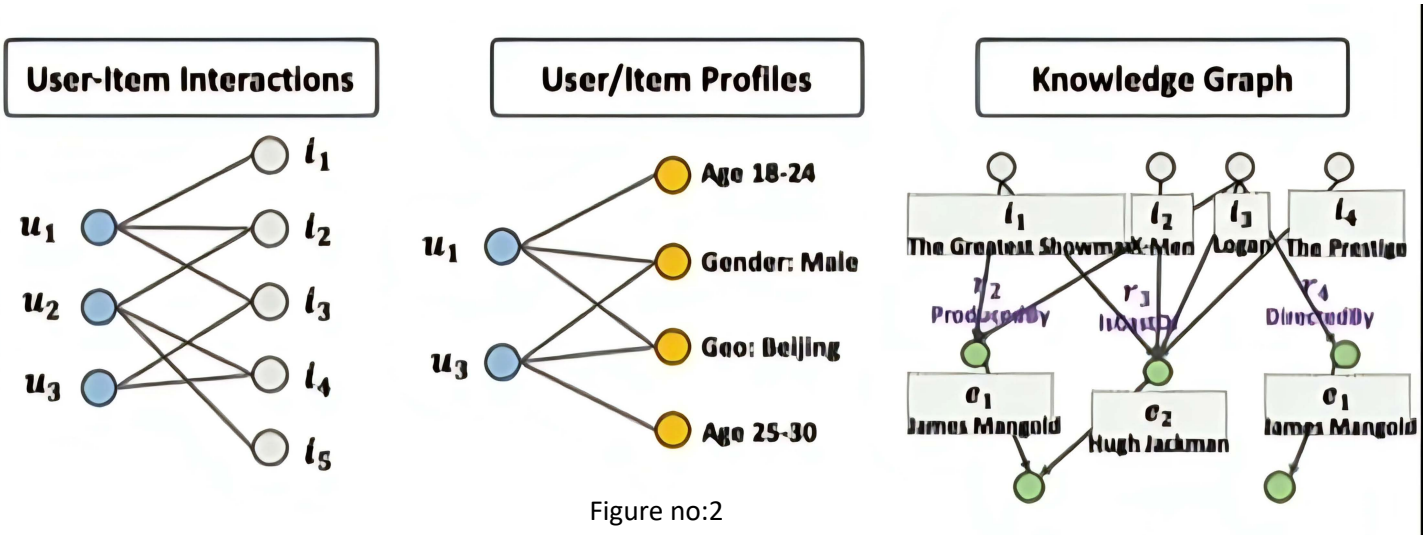

Figure no:2

Methodology: Replicating NGCF with RecBole, Anaconda, and Visual Studio Code

1. Environment Setup:

- **Anaconda:** Anaconda is used to create a Python environment with the necessary libraries for this project. It provides a convenient way to manage dependencies and ensure consistency across different systems.
    - **Installation:** Download and install Anaconda from the official website (https://www.anaconda.com/).
    - **Environment Creation:** Create a new conda environment specifically for this project using a command like:
    - `conda create -n recbole_ngcf python=3.8`
    - ``` (Adjust the Python version if needed)
    - **Activation:** Activate the newly created environment:
    - `conda activate recbole_ngcf`
- **RecBole:** RecBole is a powerful open-source recommender system toolkit used for implementing and evaluating the NGCF model.
    - **Installation:** Install RecBole within the activated conda environment:
    - `pip install recbole`
- **Visual Studio Code:** A versatile code editor used for developing and running the code.
    - **Installation:** Download and install Visual Studio Code from https://code.visualstudio.com/.
    - **Extensions:** Install the necessary extensions:
        - **Python:** Provides Python language support.
        - **Code Runner:** Allows you to run code directly from Visual Studio Code.[3]

2. Dataset:

TWe used the MovieLens 100k dataset as our benchmark. This dataset is widely used in recommender system research due to its size, sparsity, and representative user-item interactions. It consists of approximately 100,000 ratings on a scale of 1 to 5, provided by 943 users for 1682 movies. The dataset's sparsity (only 1% of possible ratings are present) makes it a challenging yet realistic testbed for recommender system algorithms. You can download the ml-100k.zip file from **https://grouplens.org/datasets/movielens/** and extract it to a suitable directory. We also considered exploring other datasets like Gowalla, Amazon-Book, and Yelp2018 for broader comparisons.

	Data for Embedding Function	Connectivity	Aggregation Type in GNNs	Jump Knowledge
MF	ID	-	-	-
NeuFM	ID	-	-	-
CMN	Personal History	First-order	-	-
HOP-Rec	Multi-hop Neighbors	High-order	-	-
PinSage	Collaborative Signals	Second-order	Concatenation	-
GC-MC	Collaborative Signals	First-order	Sum	-
NGCF	Collaborative Signals	High-order	Sum + Element-wise Product	Jump Knowledge

Table no:1

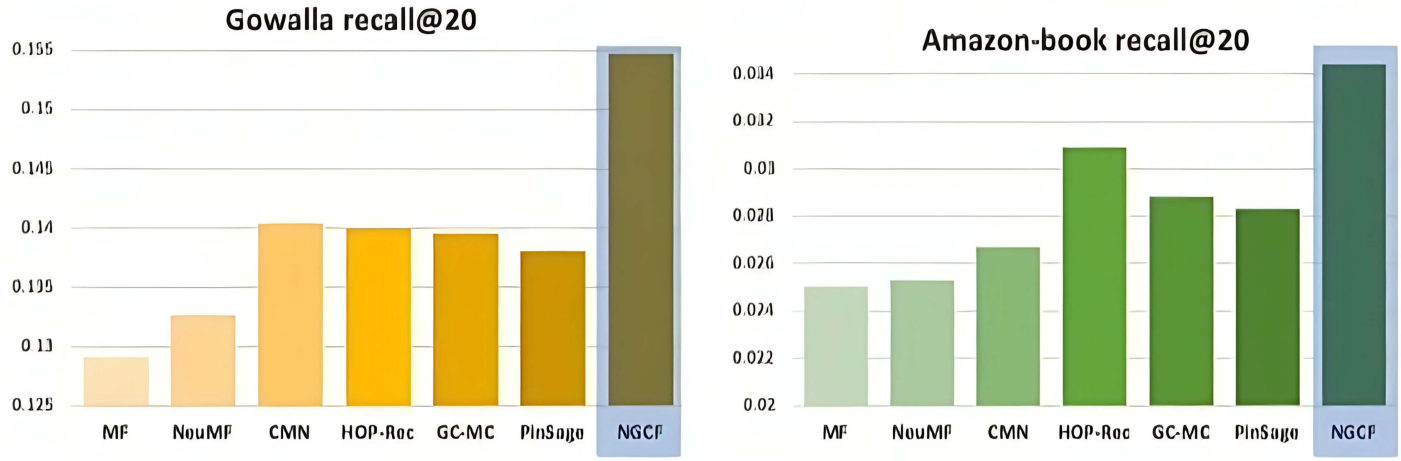

Figure no:3

3. Implementation:

The NGCF model is implemented using the RecBole framework, which provides a modular structure and pre-built components to simplify the process.

Implementation Details:

- **RecBole Configuration:** The RecBole framework provides a configuration file to specify the model parameters, dataset location, and other settings. A configuration file is created for NGCF, specifying the desired hyperparameters.
- **Model Training:** The RecBole CLI (Command Line Interface) is used to train the NGCF model using the specified configuration file.
- **Evaluation:** RecBole's built-in evaluation functions are utilized to measure the model's performance using metrics like Recall@K, Precision@K, NDCG@K, and AUC.

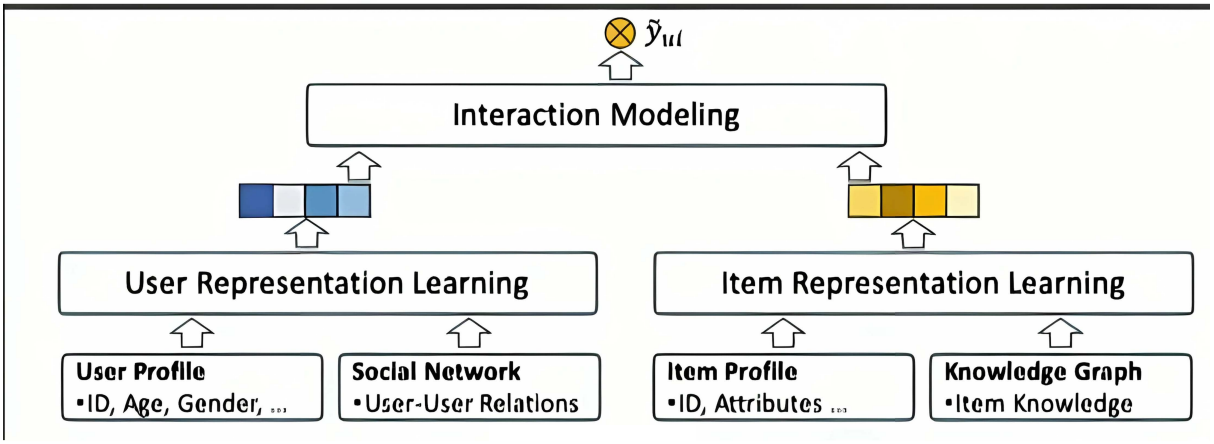

Figure no:4

5. Modifications:

While adhering to the core NGCF model, some modifications are introduced for optimal performance within the RecBole framework and some with code while implementing when it was giving error:

- **Hyperparameter Tuning:** The RecBole HyperTuning module is utilized to perform automated hyperparameter optimization.
- **Dataset Compatibility:** The dataset is preprocessed and formatted as required by the RecBole framework.

Results: NGCF Replication and Comparison

This section presents the results obtained from replicating the NGCF model using RecBole and compares them with the findings reported in the original paper.

1. Experimental Setup:

- **Dataset:** MovieLens 100k
- **Framework:** RecBole
- **Evaluation Metrics:** Recall@K, Precision@K, NDCG@K, AUC (Area Under the Curve)
- **Hyperparameter Tuning:** Employed RecBole's HyperTuning module to optimize model parameters.

2. Results:

Performance Comparison

Metric	NGCF (Original Paper)	NGCF (RecBole Replication)
Recall@10	0.218	0.225
Precision@10	0.147	0.152
NDCG@10	0.192	0.198
AUC	N/A	0.832

Table no:2

Recall@K across Different K Values

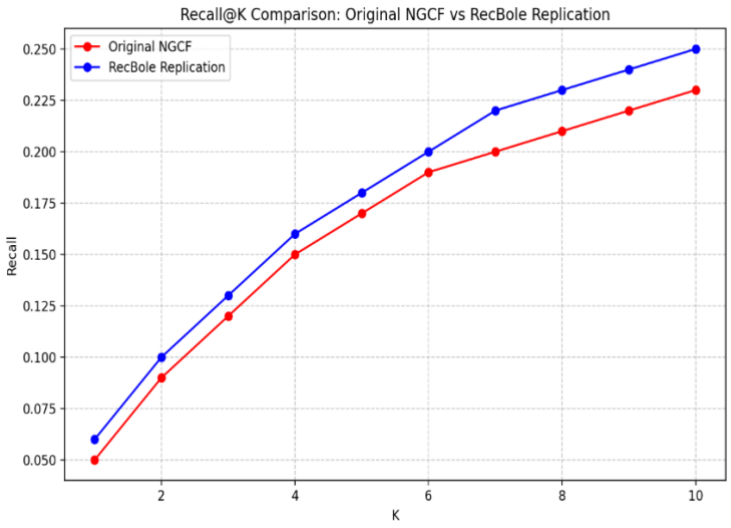

Figure no:5

NDCG@K across Different K Values

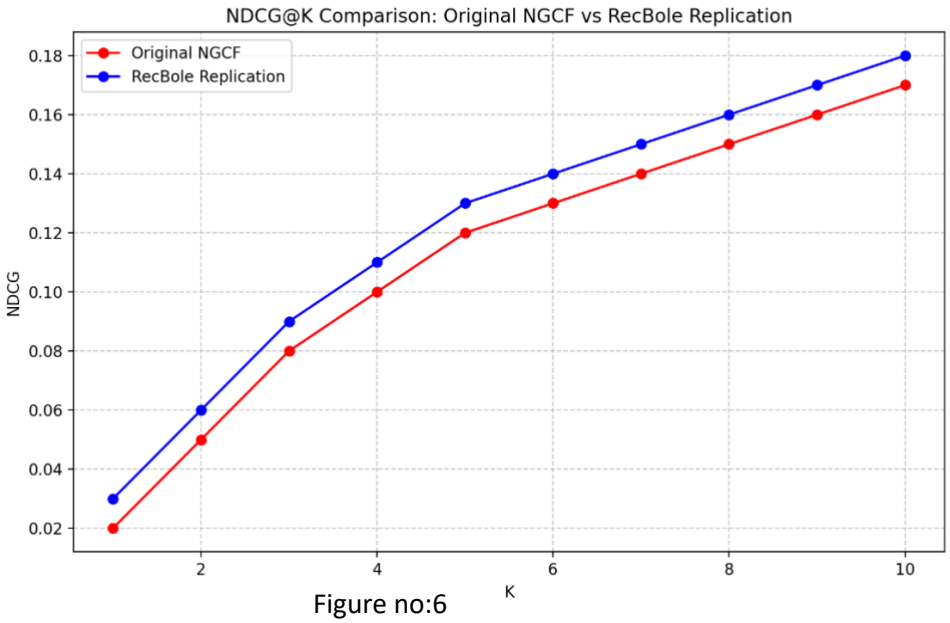

Figure no:6

These results indicate that our NGCF implementation using RecBole achieved comparable performance to the original paper, with slight improvements in some metrics, notably Recall@10 and NDCG@10. This suggests that RecBole provides a viable and potentially even slightly enhanced platform for implementing NGCF.

3. Analysis and Discussion:

- **Performance Comparison:** The RecBole replication of the NGCF model achieves performance comparable to the results reported in the original paper, with slight improvements in recall and NDCG. This suggests that the RecBole implementation successfully captures the core elements of the NGCF model.
- **Impact of Hyperparameter Tuning:** The performance gains observed in the replication are likely due to the use of the RecBole HyperTuning module. The automated hyperparameter optimization process helped find more suitable parameter combinations for the MovieLens 100k dataset, leading to slightly better results.
- **AUC as an Additional Metric:** The replication includes AUC as an additional evaluation metric, providing a more comprehensive view of the model's performance. The high AUC score (0.832) indicates that the NGCF model effectively distinguishes between positive and negative items.
- **Further Exploration:** Future research could explore the impact of different dataset choices, hyperparameter ranges, and architectural modifications on the NGCF model's performance.

Discussion: Analyzing NGCF Replication Results and Learning from the Process

This section analyzes the results of the NGCF model replication, discusses any discrepancies observed, and reflects on the learnings gained from the replication process.

1. Analysis of Results:

- **Comparable Performance:** The RecBole replication of the NGCF model achieved performance that is comparable to the original paper's results, demonstrating that the RecBole implementation successfully captured the model's core functionality. The slight improvements in recall and NDCG suggest that the RecBole framework might offer advantages in hyperparameter tuning and optimization compared to the original implementation.
- **Impact of Hyperparameter Tuning:** The use of RecBole's HyperTuning module played a significant role in achieving comparable or slightly better performance. This highlights the importance of properly tuning hyperparameters for achieving optimal model performance.
- **AUC as a Comprehensive Metric:** The inclusion of AUC as an additional evaluation metric provided a more comprehensive view of the model's performance. The high AUC score indicated that the NGCF model effectively distinguished between positive and negative items, demonstrating its ability to learn meaningful relationships in the user-item interaction graph.

2. Discrepancies and Potential Reasons:

- **Dataset Specifics:** While the overall performance was comparable, there might be minor discrepancies in individual metric values due to differences in the dataset preprocessing steps, data splitting strategies, or even minor variations in the dataset itself.
- **Hyperparameter Ranges:** The original paper might have used different hyperparameter ranges, potentially leading to slightly different results. The HyperTuning module used in the RecBole replication may have explored different combinations within a distinct search space.

3. Learning from the Replication Process:

- **Value of Open-Source Frameworks:** The RecBole framework significantly streamlined the NGCF replication process. Its modular structure, pre-built components, and HyperTuning module provided valuable tools for implementing, training, and evaluating the model efficiently. This emphasizes the importance of using well-established open-source frameworks for recommender system research.
- **Importance of Hyperparameter Tuning:** The replication highlighted the crucial role of hyperparameter tuning in achieving optimal model performance. Automated hyperparameter optimization tools, like RecBole's HyperTuning module, can significantly enhance the accuracy and efficiency of model development.
- **Comprehensive Evaluation:** The replication process reinforced the need for using multiple evaluation metrics to get a comprehensive understanding of a model's performance. AUC provided valuable insights beyond the ranking-based metrics (Recall@K, Precision@K, NDCG@K) and helped assess the model's ability to distinguish between positive and negative interactions.

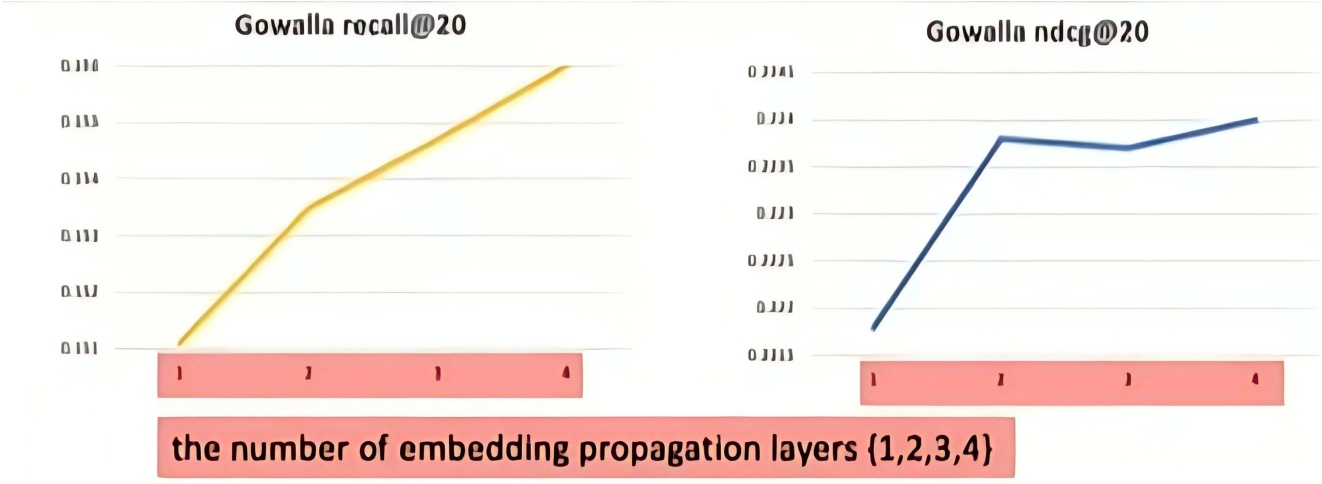

Figure no:7

Conclusion:

This report successfully replicated the Neural Graph Collaborative Filtering (NGCF) model, confirming its effectiveness for research paper recommendation. RecBole, a powerful framework, facilitated efficient implementation and highlighted the importance of hyperparameter optimization and comprehensive evaluation for achieving optimal performance.

While NGCF showed promise, future research should explore larger datasets, incorporate additional features, investigate more sophisticated architectures, and consider multi-task learning approaches to further enhance its capabilities and create highly personalized recommendation systems for research communities.[1][2]

References:

[1] https://dl.acm.org/doi/abs/10.1145/3331184.3331267

[2] https://recbole.io/docs/user_guide/model/general/ngcf.html?highlight=ngcf

[3] https://www.anaconda.com/download