Comparison of different recommender systems

Žiga Drab[1], Žan Kogovšek[2], Grega Rovščček[3], and David Trafela[4]

[a]University of Ljubljana, Faculty of Computer and Information Science, Večna pot 113, SI-1000 Ljubljana, Slovenia

The manuscript was compiled on May 11, 2023

In recent years, recommender systems have gained significant popularity and there has been an increase in the utilization of graphs to represent data. This project compares traditional recommender systems with graph-based approaches in order to determine if the latter can provide recommendations with similar accuracy to the former. Several methods for recommender systems are reviewed, including collaborative filtering, matrix factorization, neural networks, personalized PageRank and graph embeddings. Results are obtained using book rating dataset and show that graph-based approaches perform similarly to traditional methods in terms of RMSE and MAE scores. Furthermore, graph-based approaches outperform traditional methods in case of precision@k and recall@k. This demonstrates that graph-based approaches are a viable alternative for recommender systems, particularly for situations where graphs are the natural way to represent data and where traditional methods become memory inefficient due to increase in dataset sizes.

Recommendation systems are a class of artificial intelligence applications that have become increasingly prevalent in our daily lives. From personalized movie and music recommendations to product suggestions on online shopping platforms, these systems have revolutionized the way we consume information and make purchasing decisions. As the amount of available data continues to grow exponentially, it becomes increasingly challenging for individuals to find the information they need without assistance. In response to this challenge, researchers have developed a variety of recommendation techniques, which analyze user data such as browsing history, purchase history, and social network interactions to generate personalized recommendations.

In recent years, graph-based recommendation systems have emerged as a popular approach, particularly for systems, where the data is represented as a graph or a graph can easily be constructed, with users and items as nodes and edges representing their interactions. Graph-based recommendation systems leverage the power of graph theory to generate recommendations by analyzing the structural properties of the graph.

The goal of this project was to compare different recommendation systems, mainly traditional approaches and graph-based approaches and through this comparison evaluate the following hypothesis:

Graph-based recommendation systems can provide recommendations with similar accuracy compared to traditional systems.

Related work

Traditional methods for recommendation systems include collaborative filtering, content-based filtering and matrix factorization. These approaches are described in works (1), (2) along with their main advantages and disadvantages. As described in (3) a popular measurement of the distance between users, items and groups they belong to is calculated by using an implementation of k-nearest neighbors named K-Means.

Approaches that use neural networks for recommendation systems have gained popularity in last decade, especially after producing accurate recommendations by leveraging the power of deep learning to learn complex user-item interactions. Several studies have evaluated the performance of these neural network-based approaches on various recommendation tasks, including movie and music recommendation. For instance, a study by He et al. (4) proposed a new recommendation framework based on deep learning, which only uses ID information of users and movies. Moreover, neural networks have been shown to outperform traditional collaborative filtering approaches, which seem to struggle with sparse data sets (5), by first extracting the features of users and items from the rating matrix and then concatenating them together as the input data for the DNN model.

According to (6) one of the most popular approaches to graph-based recommendation systems involve random walks and a rendition of that named Personalized PageRank seems to perform especially well for recommendations (7) and even outperforms collaborative filtering (8) in certain scenarios. Graph embeddings can also be used for recommendation systems as we can measure user-user and item-item proximity and based on that deduct the similarity (9). Advanced techniques mentioned in (6) include graph neural networks and graph factorization machines. Of these, graph neural networks have emerged as the state-of-the-art method in recent years, and are widely adopted in practical applications (10).

Results

Table 1 presents the MAE and RMSE scores for the models that predict the rating of unseen items. As all these methods run relatively fast, the complete dataset was used. In order to account for the time complexity of Node2Vec algorithm, shorter random walks and less repetitions were used, giving it a slight handicap.

Table 1. MAE and RMSE for different recommendation models.

Method	MAE	RMSE
Matrix factorization (SVD)	0.65	**0.83**
Collaborative filtering (User-User)	0.65	0.84
Collaborative filtering (Item-Item)	0.65	0.85
Deep Neural Network	0.65	0.84
Node2Vec Embeddings -> DNN	0.76	0.94
Item-Item projection with Collaborative filtering	**0.63**	0.85

Due to the fact that certain methods only generate a

All authors contributed equally to this work.

[1]To whom correspondence should be addressed. E-mail: zk0821@student.uni-lj.si / zd7631@student.uni-lj.si

list of recommended items, the MAE and RMSE scores cannot be computed. Therefore, the evaluation metrics used were precision@k and recall@k, which are shown in Figure 1. As most of the graph-based methods rely on random walks they typically have a high time complexity. To account for this a sample size of 10% was used for all methods, including traditional ones. Code is available at: `https://github.com/ziga7631/INA-Final-Project`

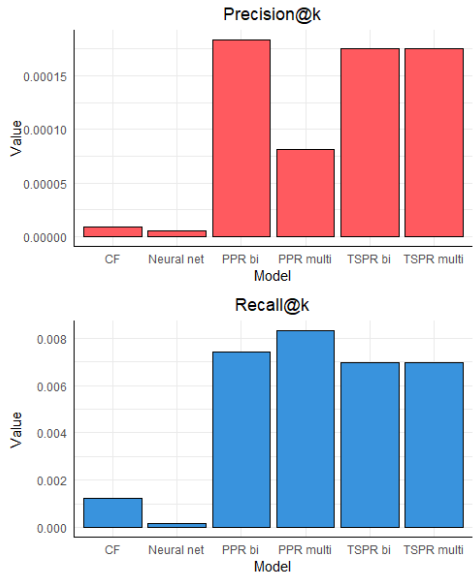

Fig. 1. Precision@k and Recall@k results

Discussion

Overall, the results show that graph-based recommender systems exhibit similar performance to traditional approaches in terms of MAE and RMSE scores. Notably, Node2Vec embeddings to deep neural networks show inferior performance compared to other methods in terms of both metrics. This may be attributed to the handicap given to Node2Vec.

In terms of RMSE, the best performing method is Matrix Factorization (SVD), while the bi-partite projection collaborative filter achieves the best MAE score, demonstrating the potential of our proposed method as a recommender system.

Analyzing precision@k and recall@k showed that Personalized Pagerank, including both traditional and topic specific approach, drastically outperformed traditional recommender systems by accurately identifying relevant recommendations as well as finding all relevant recommendations within the sampled data set.

According to the observations above, in general, graph-based recommender systems perform similarly or even better than traditional approaches, except for graph embeddings. This finding approves our original hypothesis.

Even though precision@k and recall@k are both commonly used evaluation metrics for recommender systems, they do not provide a complete assessment of the system's performance. In practice click-through rate (CTR) is the most important metric as it measures the ratio of clicks on recommended items to the number of times the item was presented to users. This metric is important as it provides an insight into the real-world effectiveness of the system in generating user engagement and interest (11).

With the growth of datasets, traditional matrix user-item approaches become memory inefficient due to the sheer sparsity. Advanced graph-based techniques, such as graph neural networks and graph factorization machines, have emerged as a promising alternative as they address the problem of memory efficiency with their ability to handle large and complex data sets. However, to fully explore their potential and achieve promising results in terms of recommender systems, further research in machine learning with graph-based techniques is required.

Methods

Data.

Goodreads book data set. The Goodreads dataset is a publicly available dataset of book ratings and reviews collected from the Goodreads website. For our experimentation we used the Goodreads 10k dataset (12), which is a subset of the larger Goodreads dataset, and contains information on approximately 10,000 books, 53,000 users, and 950,000 ratings. The dataset includes information on book titles, authors, publication dates, and user ratings, but does not include user reviews or demographic information. The ratings in the dataset range from 1 to 5.

Data gathering and preprocessing. For the task at hand, we choose Goodreads dataset to compare properties of recommender systems. In order to better understand the data, we conducted exploratory data analysis. We noticed that some users rated less than 3 books, which provides insufficient information to make meaningful recommendation. Because of this cold-start problem, we removed them from the dataset, and popular items can be suggested to them. We filled in some missing data and also noticed some inconsistencies regarding authors' names, where multiple spaces existed between the surname and the given name, which we removed in the preprocessing stage. Finally, we extracted some additional information from external sources, like book genres and number of pages.

Following the preprocessing phase, we created a user-item bipartite graph, along with a multipartite graph, that includes also genres and author nodes. While we initially planned to include additional node groups like languages, book lengths, and book age, we later excluded them due to increased complexity, which usually reflects in worse recommendations. We used two user-item weighting schemes: normal ratings and sigmoid ratings, that emphasize items rated above a user's average. To avoid users with more ratings having more influence on recommendation system than others, we normalized these weights by each user's total weights, ensuring that influence is based on user-item interaction quality, not rating quantity.

Traditional approaches.

Matrix factorization. Matrix factorization breaks down a large user-item interaction matrix into two lower dimensional matrices, which represent latent features of users and items respectively. The product between these two matrices allows us to quickly and efficiently compute the missing ratings. The decomposition is performed with the SVD method due to its effectiveness and ability to handle missing values (13).

K-Means Collaborative filtering. Collaborative filtering uses the past behavior of users and the ratings they have given to items. There are two main types: user-based and item-based. The prior works by identifying users with similar preferences and recommends items that those similar users have rated highly, but the user in question has not yet rated. On the other hand, item-based collaborative works similarly, except that it looks for items that are similar to the items the user has already liked and recommends them (1). The similarity between users or items, depending on the version, is determined by identifying k-nearest neighbors with K-Means algorithm.

Neural networks. The main advantage of neural network approaches is the fact that they can handle complex and non-linear relationships between user features and item features. With the use of embedding layers, neural networks are able to map high-dimensional user and item data to a dense array of real numbers in a continuous space. These latent features are then used in the final layer to predict the final rating. Figure 2 presents such a recommender system.

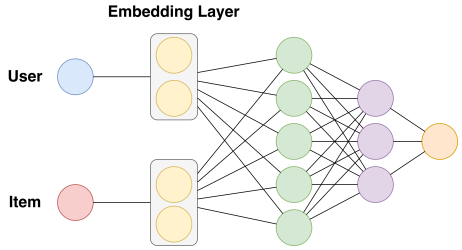

Fig. 2. Neural network recommender system

Graph-based approaches.

The PageRank. The PageRank is a graph-based ranking algorithm invented by Google's founders, Larry Page and Sergey Brin. It assigns importance of each node based on the importance of its neighbors. It uses the following equation 1, where d is a dumping factor, N is number of nodes in a graph and L is number of out-links neighbor m.

$$PR(n) = (1-d) + d \sum_{m \in M(n)} \frac{PR(m)}{L(m)} \quad [1]$$

The PageRank value of a node can also be seen as the chance that a random walker lands on it, where it follows an outbound link with probability d and jumps to a random node with probability d-1. There are several modifications to this algorithm, that use personalization and can be used for recommendation systems. Personalized Pagerank 3 uses random walk with restart, which jumps with probability 1-d to a specific node. It biases the walk and can capture user's preferences. Another approach is Topic Specific Pagerank, which similarly does not teleport to any random node, but selects one from predefined set of nodes (14).

In order to make book recommendations, we have implemented both mentioned approaches, both of which utilize a bipartite graph as well as a multipartite graph. The only distinction between both approaches is personalization vector used. In the case of Personalized PageRank, the vector only contains the query user, while for Topic Specific PageRank, it is filled with sigmoid values of book ratings, which biases the random walker towards books liked more by the user.

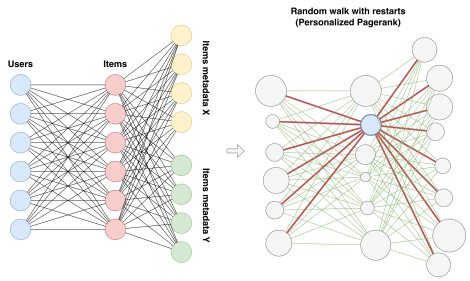

Fig. 3. Multi-partite Personalized PageRank

Node embeddings. Node embeddings are vector representations of individual nodes in a graph. They capture, in a lower-dimensional space, their properties and relationships with other nodes.

There are numerous techniques, that generate embeddings of nodes. One of them is Node2Vec. It generates node's vector representation in the way, that maximizes likelihood of preserving local neighborhoods. The fundamental idea is again a biased random walker, that generates sequences of nodes. Bias is introduced with parameters p and q, that control walker's likelihood of exploring local and not yet explored regions of a graph. Finally, sequence of nodes are used to generate nodes embeddings using a skip-gram model, which is a common approach in the field of natural language processing (15).

Node embeddings can be used in a variety of machine learning task involving graphs, such as link predictions and node classifications. Recommendation systems basically just predict links between users and items. Image 4 shows how we used this approach to predict user's rating of an item. Firstly, we used Node2Vec to generate nodes embeddings of a bipartite user-item graph. Then, we feed them to a neural network, which outputs a predicted rating.

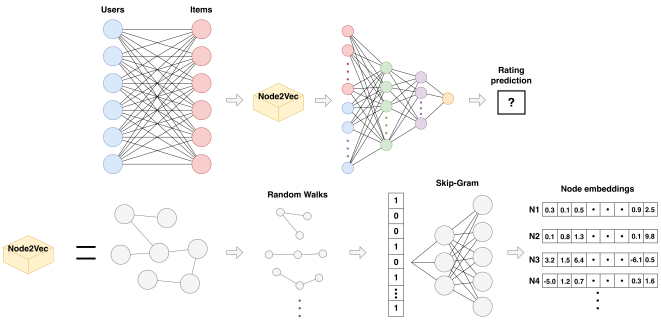

Fig. 4. Graph embeddings with DNN

Bipartite graph projection. A bipartite graph projection is a technique, that transform a bipartite graph, containing two types of nodes, into a unipartite graph, where all nodes are of the same type. In the context of user-item bipartite graph, projection could be used to obtain fully connected graph of either users or books, where weighted edge between two nodes represents some kind of interaction between them (16).

Image 5 shows the whole process of making book recommendations. We created book-book projection, where weight of the edge reflects a similarity of two books, that is proportional to a number of times they were rated above average by the same user. It was used to identified each book's nearest neighbors. For user-book predictions, we selected the top-k similar books the user rated before and predicted the rating as a weighted average of their ratings. This approach was inspired by the principle of collaborative filtering, where past interactions are leveraged to predict future preferences, but it obtains nearest neighbors in a different way.

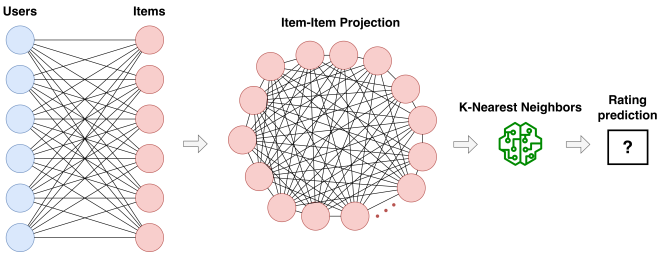

Fig. 5. Bi-partite item-item projection recommendation system using collaborative filtering

Evaluation.

Mean Absolute Error. MAE 2 is calculated as the average of the absolute differences between the predicted values y_i and actual values $\hat{y_i}$.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y_i}| \quad [2]$$

Root Mean Square Error. RMSE 3 is calculated as the square root of the average of squared differences between predicted values y_i and actual values $\hat{y_i}$.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2} \quad [3]$$

Precision@k. Precision@k measures the proportion of relevant items among the top k recommended items.

$$Precision@k = \frac{|\text{ Recommended items @k that are relevant }|}{k} \quad [4]$$

Recall@k. Recall@k, on the other hand, measures the proportion of relevant items that were actually recommended in the top k results.

$$Recall@k = \frac{|\text{ Recommended items @k that are relevant }|}{|\text{ Relevant items }|} \quad [5]$$

1. Amir Hossein Nabizadeh Rafsanjani, Naomie Salim, Atae Rezaei Aghdam, and Karamollah Bagheri Fard. Recommendation systems: a review. *International Journal of Computational Engineering Research*, 3(5):47–52, 2013.
2. Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
3. Phongsavanh Phorasim and Lasheng Yu. Movies recommendation system using collaborative filtering and k-means. *International Journal of Advanced Computer Research*, 7(29):52, 2017.
4. Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering, 2017.
5. Libo Zhang, Tiejian Luo, Fei Zhang, and Yanjun Wu. A recommendation model based on deep neural network. *IEEE Access*, 6:9454–9463, 2018.
6. Shoujin Wang, Liang Hu, Yan Wang, Xiangnan He, Quan Z Sheng, Mehmet A Orgun, Longbing Cao, Francesco Ricci, and Philip S Yu. Graph learning based recommender systems: A review. *arXiv preprint arXiv:2105.06339*, 2021.
7. Phuong Nguyen, Paolo Tomeo, Tommaso Di Noia, and Eugenio Di Sciascio. An evaluation of simrank and personalized pagerank to build a recommender system for the web of data. In *Proceedings of the 24th international conference on world wide web*, pages 1477–1482, 2015.
8. Ziqi Wang, Yuwei Tan, and Ming Zhang. Graph-based recommendation on social networks. In *2010 12th International Asia-Pacific Web Conference*, pages 116–122. IEEE, 2010.
9. Yue Deng. Recommender systems based on graph embedding techniques: A review. *IEEE Access*, 2022.
10. Jin-Young Kim and Sung-Bae Cho. A systematic analysis and guidelines of graph neural networks for practical applications. *Expert Systems with Applications*, 184:115466, 2021.
11. Renjie Zhou, Samamon Khemmarat, and Lixin Gao. The impact of youtube recommendation system on video views. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 404–410, 2010.
12. Zygmunt. Goodbooks-10k, 2023. URL https://www.kaggle.com/zygmunt/goodbooks-10k. Accessed: 2023-05-10.
13. M Sunitha Reddy and T Adilakshmi. Music recommendation system based on matrix factorization technique-svd. In *2014 international conference on computer communication and informatics*, pages 1–6. IEEE, 2014.
14. Amy N Langville and Carl D Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3): 335–380, 2004.
15. Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.
16. Tao Zhou, Jie Ren, Matúš Medo, and Yi-Cheng Zhang. Bipartite network projection and personal recommendation. *Physical review E*, 76(4):046115, 2007.