Advanced User Profile Similarity Prediction: Integrating Bipartite Graphs with SimRank

Abstract

In the field of modern recommender systems, accurately predicting user similarities remains a critical challenge. This paper proposes an innovative approach that combines bipartite graphs with the SimRank structural similarity measure to enhance the prediction of similar user profiles. By representing users and their interests in a bipartite graph and applying SimRank, our method captures both direct and indirect relationships between users. Experimental results obtained with the MovieLens dataset show that this approach outperforms traditional methods such as cosine similarity, offering more accurate and consistent predictions. The integration of SimRank with a bipartite structure provides a robust framework for identifying subtle and complex similarities between users. This study highlights the effectiveness of the proposed method and paves the way for future research, including the incorporation of contextual data and advanced techniques such as Graph Neural Networks to further refine recommender systems.

1. Introduction

In the era of big data, the ability to efficiently and accurately predict user preferences has become a cornerstone of many modern applications, ranging from personalized recommendations to targeted marketing strategies. Among the various methodologies developed to address this challenge, graph-based approaches have proven to be a powerful tool for modeling and analyzing the complex relationships between entities. Graph theory, with its rich theoretical foundation and practical versatility, provides a robust framework for capturing the intricacies of interactions that define user behavior.

Recent advances in graph-based machine learning, particularly those leveraging deep learning techniques, have significantly enhanced our ability to predict similar user profiles. These methodologies transcend traditional collaborative filtering and content-based recommendation techniques by exploiting the topological structure of user-item interactions, uncovering latent patterns and connections that were previously inaccessible. The advent of Graph Neural Networks (GNNs) has further propelled this field, enabling the incorporation of both local and global graph structures into the learning process.

In this context, predicting similar user profiles based on graphs is not merely a theoretical pursuit but a practical necessity in an increasingly interconnected digital landscape. By representing users and their interactions as nodes and edges within a graph, these models can harness the power of network effects to provide more accurate and personalized predictions. This approach also aligns with the growing trend toward explainable AI, as graph-based models inherently offer a more interpretable structure compared to black-box algorithms.

This article aims to explore cutting-edge techniques in graph-based user profile prediction, focusing on recent innovations and their practical applications. We will examine the fundamental principles of graph theory applied to user profiling, review the latest developments in GNNs and their variants, and discuss the implications of these advances for real-world applications. By providing a comprehensive overview of this rapidly evolving field, we hope to highlight the potential and challenges of using graph-based methods to predict similar user profiles, thereby contributing to the advancement of personalized user experiences in the digital age.

This paper explores advanced techniques for predicting user profiles based on graphs, with a focus on recent innovations and their practical applications. We will begin by presenting various techniques for predicting similar user profiles based on graphs, detailing their advantages and disadvantages. Then, we will introduce our novel approach to detecting similar user profiles, based on representing user profiles as a bipartite graph and the SimRank structural similarity measure. Subsequently, we will conduct experiments to evaluate the effectiveness of our method. Finally, we will conclude with our findings and future research perspectives.

2. State of the Art

Graph-Based Techniques for Predicting Similar User Profiles

2.1 Graph-Based Collaborative Filtering

Graph-based collaborative filtering represents user-item interactions using nodes for users and items, and edges for interactions such as ratings. Simple to implement, this method directly leverages observed interactions. When it comes to detecting similar profiles, this approach explicitly models the relationships between users and items, thereby facilitating the identification of users with similar preferences.

However, it can suffer from scalability issues and lacks consideration of contextual information. Yang et al. (2019) show that integrating user and item attributes enhances recommendation accuracy by enriching the graphs with additional information about users and items. Moreover, [1, 2] demonstrate that the inclusion of deep learning techniques and entity representations further improves performance. For instance, [3] used a graph-based collaborative filtering approach to detect similar profiles by integrating graph embeddings to enhance recommendation accuracy.

2.2 Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) are neural network architectures specifically designed for graph data. They aggregate information from neighboring nodes to learn representations that capture both the local and global structure of the graph. When it comes to user profiles, each node can represent a user along with their characteristics and interactions with other users. By generating node embeddings, GNNs capture the features and relationships of user profiles, thus facilitating the detection of similar users. They offer advantages in modeling complex dependencies and integrating various features, but they require significant computational power and large datasets for effective training. [4] demonstrated that GNNs outperform collaborative filtering methods in terms of accuracy, although they are more computationally expensive. [1] identified scalability challenges, while [5] discussed limitations related to information propagation in large graphs.

2.3 Graph-Based Entity Representations

Graph-based entity representations, such as Node2Vec and GraphSAGE, produce vector embeddings of graph nodes, facilitating similarity comparison between users by analyzing their embeddings. These techniques offer the advantage of generating continuous node representations that can be easily integrated into various machine learning models. When it comes to detecting similar profiles, each node represents a user, and the embeddings capture the characteristics and interactions of users, effectively enabling the identification of similar profiles.

For example, [6] demonstrated that using Node2Vec for learning user representations in social networks significantly improves the accuracy of similar profile detection. Similarly, [7] showed that GraphSAGE can be used to predict links between users, thereby identifying similar profiles based on their interactions. However, these techniques may lose essential structural information and require preprocessing of the graphs.

According to [8], Node2Vec outperforms other embedding methods in terms of node representation quality, which enhances recommendation performance. Additional studies, such as those by [9], also confirm the effectiveness of graph-based approaches for learning embeddings.

2.3 Graph Clustering Algorithms

Graph clustering algorithms, such as Louvain and K-Means, detect groups or communities of similar users by analyzing their connections within the graph. These techniques enable the identification of user communities and are scalable for large graphs. When it comes to detecting similar profiles, these algorithms can group users with similar behaviors or preferences, thereby facilitating targeted analysis and recommendations.

However, they may not capture subtle similarities between users and are sensitive to initialization parameters as well as clustering method choices. According to [10], the Louvain algorithm is particularly effective for large-scale community detection, producing high-quality partitions in real-world graphs. [11] complement this analysis by exploring the challenges of community detection in dynamic graphs.

Further studies, such as those by [12], show that integrating graph embeddings into clustering algorithms improves the accuracy of similar profile detection. For example, [13] used DeepWalk to generate node embeddings, thus enhancing clustering performance for user community identification.

2.4 Label Propagation

Label Propagation is a method where known node labels are propagated through the graph to predict the labels of unknown nodes, leveraging the graph structure. When it comes to detecting similar profiles, this method can assign similar labels to users with similar connections or behaviors, thus facilitating the identification of groups of similar profiles. This technique is appreciated for its simplicity and speed of execution on medium-sized graphs. However, it can be sensitive to noise and errors in the data and may struggle with very dense or sparse graphs.

[14] showed that Label Propagation is effective for classifying social graphs, though improvements are needed for large-scale graphs. More recently, [3] integrated Label

Propagation with deep learning techniques to enhance the accuracy of similar profile detection. [12] proposed improvements to the Label Propagation algorithm for handling large-scale graphs with promising results. [3] also demonstrated the effectiveness of Label Propagation in recommendation applications, showing better performance in detecting similar user communities.

2.5 Community Detection

Community detection in graphs involves identifying dense subgraphs where nodes are more connected to each other than to the rest of the graph. This method allows for a natural segmentation of users, facilitating targeted analyses such as product recommendations or the detection of similar behaviors. By considering user profiles as nodes in the graph, community detection helps group users with similar interests or behaviors into distinct communities. However, applying this method can be challenging in sparsely connected graphs, and community detection algorithms can be computationally expensive.

According to [15], community detection algorithms, such as those based on modularity and spectral partitioning techniques, offer significant improvements in terms of accuracy and efficiency, even for large-scale graphs. For example, [16] developed community detection methods that integrate deep learning techniques to enhance accuracy and scalability. Additionally, [17] proposed graph embedding-based approaches for more effective community detection, even in dynamic and evolving graphs, thereby enabling better identification of similar user profiles.

3. Our Approach

As mentioned earlier, in the domain of predicting similar user profiles, various paradigms have been explored, including graph-based collaborative filtering, Graph Neural Networks (GNNs), graph-based entity representations, clustering algorithms, label propagation, and community detection. However, each of these approaches has specific limitations that can impact the accuracy and efficiency of recommendations. To overcome these challenges, we propose an innovative method for detecting similar user profiles by combining the power of a bipartite graph with the structural similarity measure SimRank. This approach leverages the detailed structure of user-interest relationships to provide a more accurate assessment of similarities, thereby enhancing the performance of recommendation systems.

Our method is based on constructing a bipartite graph, where users and their interests are represented as two distinct sets of nodes. The edges between these sets illustrate interactions between users and interests, providing a structured view of their preferences. By applying SimRank, a structural similarity measure, we assess the similarity between users based not only on the similarity of their interests but also on their connections within the graph. This approach captures complex and subtle relationships by considering both direct interactions and indirect relationships through shared interests.

By integrating this approach into user profile analysis, we aim to improve the accuracy of similarity predictions by combining the rich structural information of the bipartite graph with the robust capabilities of SimRank. This framework offers a promising alternative to existing methods, blending the simplicity and effectiveness of SimRank with the flexibility of a bipartite model to identify similar user profiles in complex recommendation systems.

3.1 Bipartite Graphs

Bipartite graphs are structures where nodes are divided into two distinct sets, and edges only connect nodes from different sets. They are particularly useful for modeling relationships between two types of entities, such as users and their interests. In user profile detection, a bipartite graph allows for a clear and structured representation of interactions between users and interests. This approach facilitates the capture of complex relationships and subtle similarities between profiles, using structural similarity measures such as SimRank. By leveraging the data structure, the bipartite graph enhances the accuracy of recommendations by effectively identifying similar profiles in complex recommendation systems.

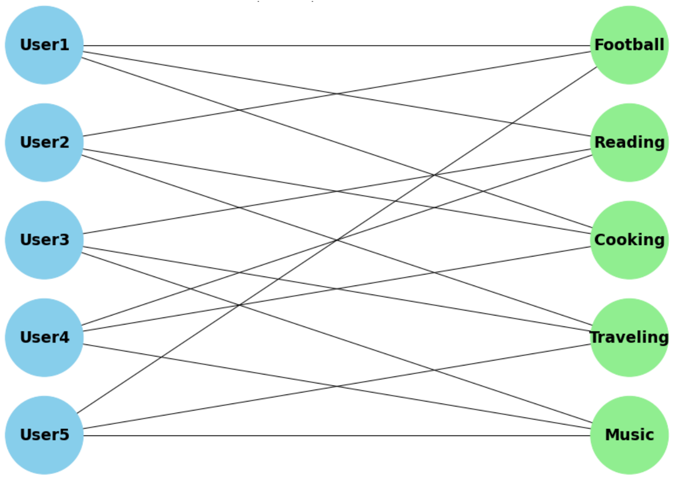


Figure 1: Bipartite Graph: Users and Interests

3.2 The SimRank Structural Similarity Measure

SimRank is a structural similarity method that evaluates the similarity between two nodes based on the similarity of their neighbors [18]. The SimRank formula is defined as follows:

$$\mathrm{SimRank}(u,v) = \frac{C \cdot \sum_{i \in \mathrm{Neighbors}(u)} \sum_{j \in \mathrm{Neighbors}(v)} \mathrm{SimRank}(i,j)}{|\mathrm{Neighbors}(u)| \cdot |\mathrm{Neighbors}(v)|}$$

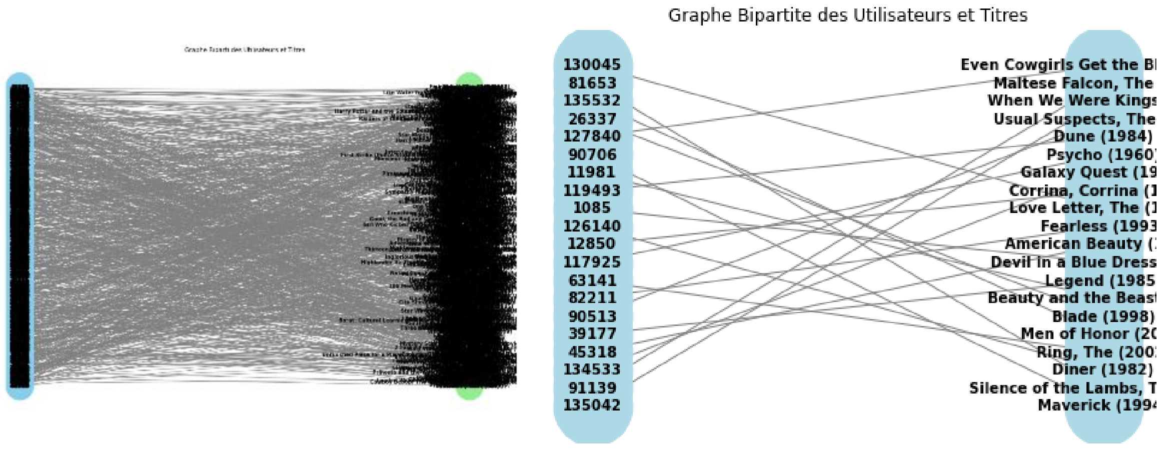
Where Neighbors (u) and Neighbors(v) are the sets of neighbors of nodes (u) and (v), respectively, and (C) is a decay factor that controls the importance of the similarity of neighbors.

When applied to a bipartite graph, SimRank is particularly effective for detecting similar user profiles by leveraging the structure of interactions between users and interests. In a bipartite graph, users and their interests are represented as two distinct sets of nodes, with edges connecting only nodes from different sets. By using SimRank, we can measure the similarity between users by analyzing not only the common interests they share but also by considering the similarity of their indirect interactions through these interests. This approach captures subtle and complex relationships between users, thereby improving the accuracy of recommendations. By integrating SimRank with a bipartite graph, we harness SimRank's ability to assess similarities in depth while benefiting from the clear and structured representation provided by the bipartite model.

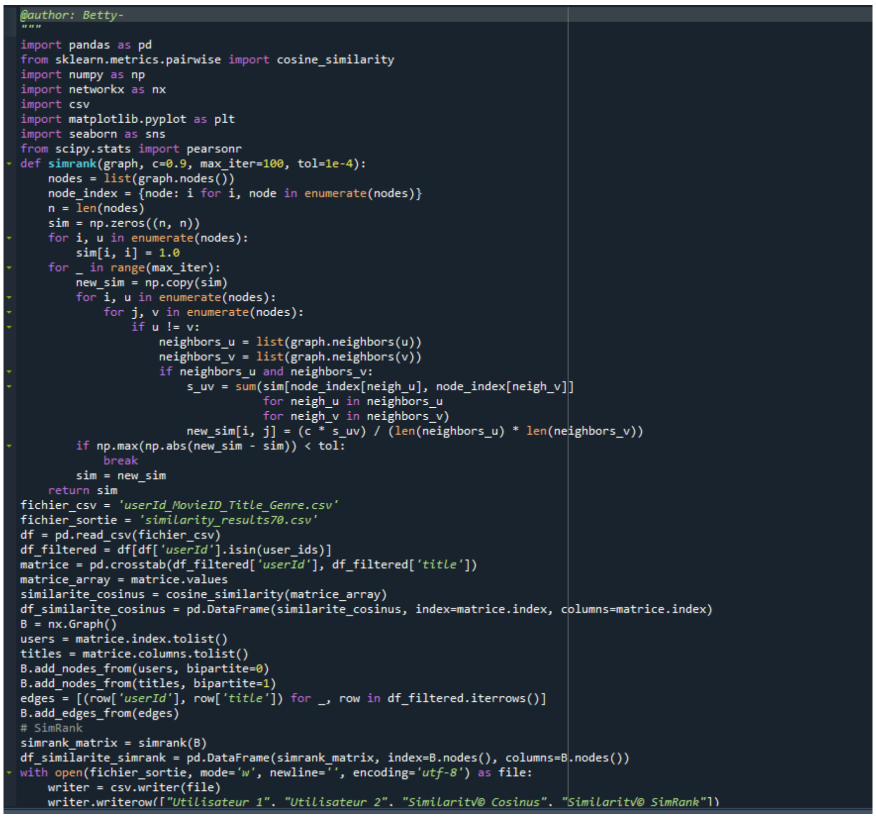
3.3 Experimentation

For our experimentation, we utilized the MovieLens Dataset, which is well-known and widely used in the research of movie recommendation systems. This dataset contains detailed information about users' movie preferences, including the ratings they have given to various films. The dataset is available in various sizes, ranging from 100,000 ratings to 27 million ratings, making it adaptable to the specific needs of different experiments.

For our study, we focused on a sample of 1,013 diverse movie titles viewed by 70 users. Initially, we represented these users and the movies they watched in the form of a weighted bipartite user-movie graph, as illustrated in the figure below.

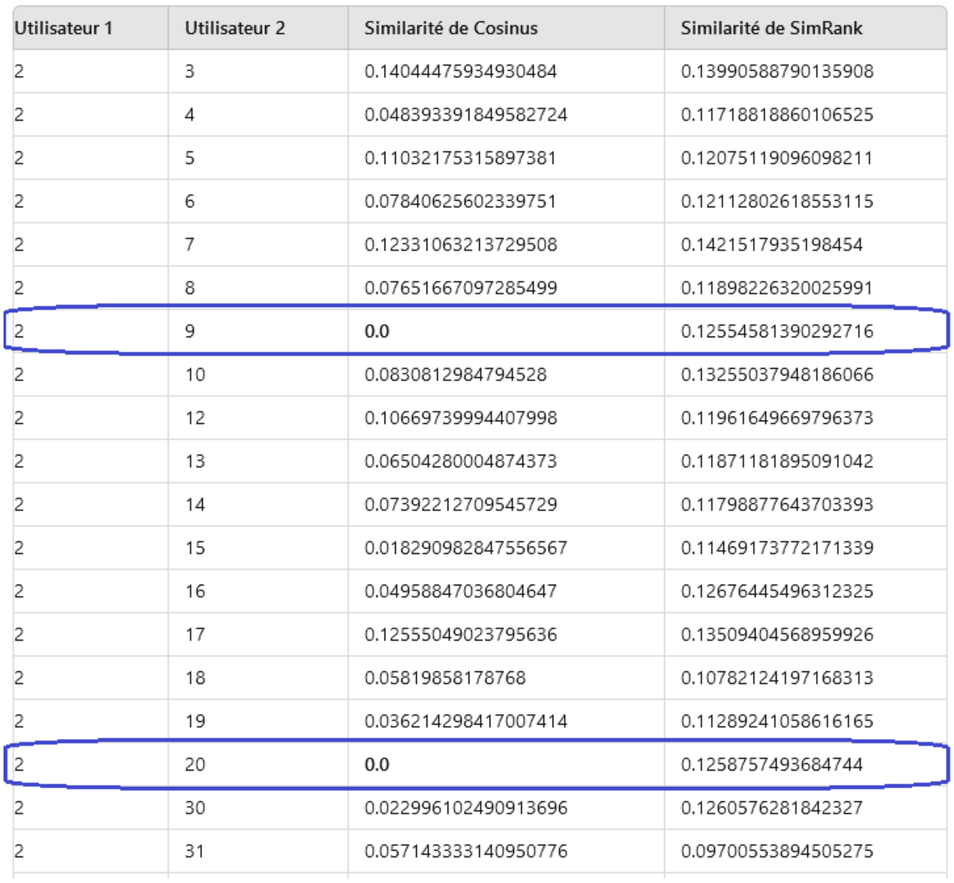


Next, we calculated the SimRank similarity within this graph to identify users with similar profiles. To evaluate the effectiveness of our approach, we also measured the similarity using the cosine method, allowing for a direct comparison between the results obtained with SimRank and those obtained with cosine similarity. Below, we present an excerpt of the Python code used as well as a sample of the results obtained.



```python
@author: Betty-
"""
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
import numpy as np
import networkx as nx
import csv
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import pearsonr
def simrank(graph, c=0.9, max_iter=100, tol=1e-4):
    nodes = list(graph.nodes())
    node_index = {node: i for i, node in enumerate(nodes)}
    n = len(nodes)
    sim = np.zeros((n, n))
    for i, u in enumerate(nodes):
        sim[i, i] = 1.0
    for _ in range(max_iter):
        new_sim = np.copy(sim)
        for i, u in enumerate(nodes):
            for j, v in enumerate(nodes):
                if u != v:
                    neighbors_u = list(graph.neighbors(u))
                    neighbors_v = list(graph.neighbors(v))
                    if neighbors_u and neighbors_v:
                        s_uv = sum(sim[node_index[neigh_u], node_index[neigh_v]]
                                   for neigh_u in neighbors_u
                                   for neigh_v in neighbors_v)
                        new_sim[i, j] = (c * s_uv) / (len(neighbors_u) * len(neighbors_v))
        if np.max(np.abs(new_sim - sim)) < tol:
            break
        sim = new_sim
    return sim
fichier_csv = 'userId_MovieID_Title_Genre.csv'
fichier_sortie = 'similarity_results70.csv'
df = pd.read_csv(fichier_csv)
df_filtered = df[df['userId'].isin(user_ids)]
matrice = pd.crosstab(df_filtered['userId'], df_filtered['title'])
matrice_array = matrice.values
similarite_cosinus = cosine_similarity(matrice_array)
df_similarite_cosinus = pd.DataFrame(similarite_cosinus, index=matrice.index, columns=matrice.index)
B = nx.Graph()
users = matrice.index.tolist()
titles = matrice.columns.tolist()
B.add_nodes_from(users, bipartite=0)
B.add_nodes_from(titles, bipartite=1)
edges = [(row['userId'], row['title']) for _, row in df_filtered.iterrows()]
B.add_edges_from(edges)
# SimRank
simrank_matrix = simrank(B)
df_similarite_simrank = pd.DataFrame(simrank_matrix, index=B.nodes(), columns=B.nodes())
with open(fichier_sortie, mode='w', newline='', encoding='utf-8') as file:
    writer = csv.writer(file)
    writer.writerow(["Utilisateur 1", "Utilisateur 2", "SimilaritÃ© Cosinus", "SimilaritÃ© SimRank"])
```

Figure 2:Extrait du code



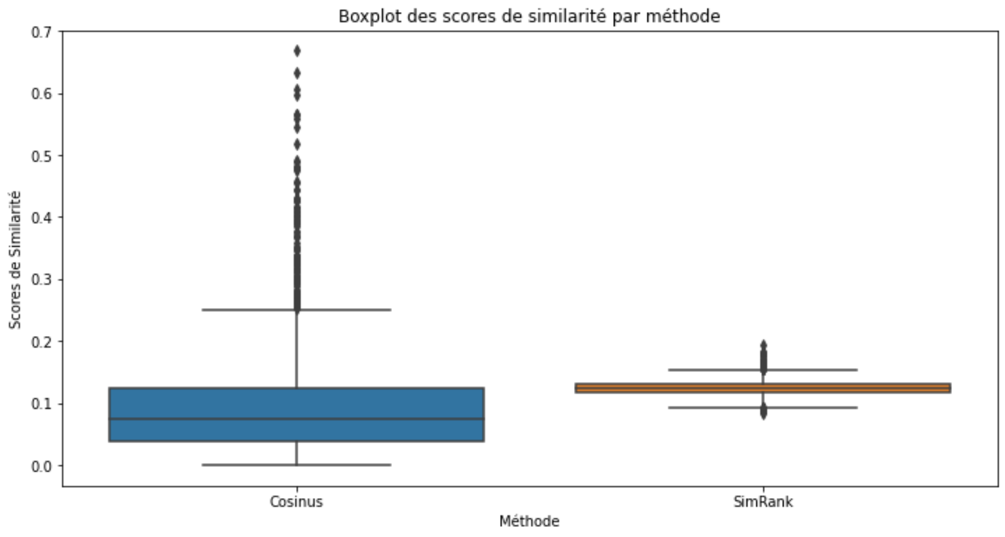
Utilisateur 1	Utilisateur 2	Similarité de Cosinus	Similarité de SimRank
2	3	0.14044475934930484	0.13990588790135908
2	4	0.048393391849582724	0.11718818860106525
2	5	0.11032175315897381	0.12075119096098211
2	6	0.07840625602339751	0.12112802618553115
2	7	0.12331063213729508	0.1421517935198454
2	8	0.07651667097285499	0.11898226320025991
2	9	0.0	0.12554581390292716
2	10	0.0830812984794528	0.13255037948186066
2	12	0.10669739994407998	0.11961649669796373
2	13	0.06504280004874373	0.11871181895091042
2	14	0.07392212709545729	0.11798877643703393
2	15	0.018290982847556567	0.11469173772171339
2	16	0.04958847036804647	0.12676445496312325
2	17	0.1255504902379563 6	0.13509404568959926
2	18	0.05819858178768	0.10782124197168313
2	19	0.036214298417007414	0.11289241058616165
2	20	0.0	0.1258757493684744
2	30	0.022996102490913696	0.1260576281842327
2	31	0.057143333140950776	0.09700553894505275

Figure 3:Extrait des résultats

In this excerpt, it is interesting to note that the cosine similarity measure between User 2 and User 9, as well as between User 2 and User 20, yields a score of 0, indicating that no movies were watched in common. In contrast, the SimRank structural similarity measure assigns non-zero scores for these same pairs due to the phenomenon of similarity propagation. To illustrate this concept, consider three users: Ibtissam, Leila, and Malika. Ibtissam likes the movies *Jumanji* and *City of Lost Children*, Leila likes *Jumanji* and *Twelve Monkeys*, and Malika likes *Twelve Monkeys* and *Seven*. If we calculate the cosine similarity measure between these users, we obtain a value of 0 between Ibtissam and Malika because there is no common movie between them. However, with the SimRank similarity measure, we obtain a non-zero value because Ibtissam is similar to Leila (common movie: *Jumanji*), and Leila is similar to Malika (common movie: *Twelve Monkeys*). Thus, it is likely that Ibtissam is similar to Malika due to this chain of similarities.

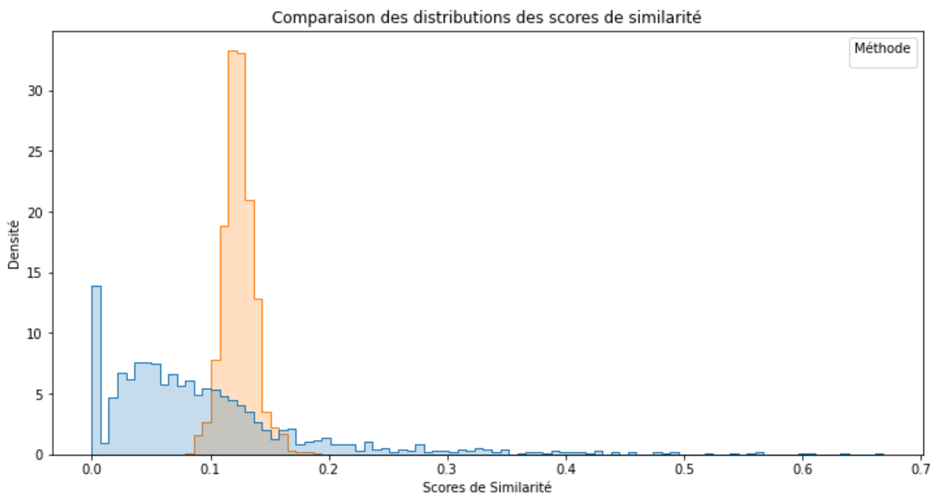
3.3.1 Distribution of Similarity Scores: A Comparative Analysis

a. Le boxplot



The results show that the median similarity scores for the Cosine method are around 0.1, while those for SimRank are higher, nearing 0.25, indicating overall higher scores for SimRank. In terms of dispersion, the Cosine method exhibits wide variability in scores, with extreme values (outliers) reaching up to 0.6-0.7, suggesting a significant diversity in the calculated similarities. In contrast, SimRank demonstrates much lower dispersion, with scores largely concentrated around the median, indicating greater homogeneity. Outliers, although present in both methods, are much more frequent with Cosine, reflecting greater variability, while SimRank shows better-controlled variability with fewer outliers.

b. The Distribution of Similarity Scores Between Users



The Cosine and SimRank methods exhibit distinct score distributions. Cosine shows a high density near zero with a long right tail, indicating a wide variability in scores with some high values. In contrast, SimRank has a distribution more concentrated around 0.1-0.15, with a sharp drop-off outside this range.

Cosine has a low and scattered density peak, suggesting that most scores are very low, while SimRank has a more pronounced density peak, indicating that most scores cluster within a narrow range. In terms of dispersion, Cosine covers a wide range of scores, whereas SimRank is more concentrated, with few high scores.

In summary, Cosine is useful for capturing a broad range of similarities, while SimRank is better suited for more uniform and consistent similarity results. These differences influence the choice of method depending on the objectives of the study on user similarity.

c. Summary

The comparative analysis of the Cosine and SimRank methods reveals notable differences in the distribution of similarity scores between users. The Cosine method results in greater variability, with scores showing a wide diversity, including a significant number of pairs with very high similarities. In contrast, SimRank generates more uniform and generally higher scores, but with fewer extreme values, providing more homogeneous and consistent results.

This distinction suggests that the choice of method depends on the study's objectives. Cosine is suitable for detecting marked but rare similarities, while SimRank is better suited for obtaining more consistent and nuanced similarities between users. SimRank is particularly effective for identifying significant similarities by considering indirect relationships, even in the absence of direct common features. Its ability to propagate and transpose similarities allows it to establish deeper and more implicit connections between users.

Regarding the correlation between the two methods, it is moderate. Cosine measures similarity based on the angle between users' feature vectors, which can result in highly variable scores. SimRank, on the other hand, evaluates similarity based on the structure of interactions between users and their indirect relationships, producing more uniform scores. This distinct approach explains the moderate correlation observed between the two methods.

3.4 Results and Discussion

In the rapidly evolving field of user profile prediction, our study highlights the superior effectiveness of integrating bipartite graphs with the SimRank structural similarity measure. This innovative approach combines the robustness of bipartite graphs with the analytical depth of SimRank, offering a powerful solution for identifying similar user profiles in complex recommendation systems.

Bipartite graphs, by distinctly representing users and their interests, provide a clear structure of interactions. This separation facilitates the analysis of complex relationships and subtle similarities between users, leveraging the fine structure of the data. However, constructing and analyzing bipartite graphs can become computationally expensive in terms of memory and processing time, especially with large-scale datasets. Addressing the scalability and efficiency of algorithms for complex bipartite graphs is crucial to manage these computational challenges.

Additionally, bipartite graphs can be very sparse if interactions between users and interests are limited, which may impact the quality of recommendations. The sparsity of data can influence the results, as it may lead to less accurate similarity measurements and recommendations. Exploring methods to mitigate the effects of data sparsity, such as incorporating additional data sources or using advanced imputation techniques, could enhance the performance of the recommendation system.

Applying SimRank to these graphs allows for an understanding beyond simple direct matches by also capturing indirect relationships between users through their shared interests. This method reveals deep similarities that might not be visible with more superficial methods.

Our experimental results, based on the MovieLens dataset, illustrate the significant advantage of this combined approach. While traditional methods such as cosine similarity yield varied results with wide dispersion, SimRank offers a more homogeneous and consistent evaluation of similarities. By considering not only direct interactions but also indirect connections, SimRank enables more accurate predictions of similar users, even in the absence of direct common features.

In conclusion, for applications requiring a nuanced understanding of user relationships, the approach based on bipartite graphs combined with SimRank proves to be an optimal solution. It provides enhanced precision in recommendations and the ability to identify similar profiles in a more nuanced and contextual manner. For any application or research aiming to improve recommendation personalization and explore the complex dynamics of user interactions, integrating bipartite graphs with SimRank represents a strategic and promising choice.

4. Conclusion

This paper has demonstrated the effectiveness of combining bipartite graphs with SimRank for predicting similar user profiles. By using a bipartite graph to model interactions between users and interests, and applying SimRank, we achieved more accurate and consistent results compared to traditional methods such as cosine similarity. Experiments on the MovieLens dataset showed that SimRank captures both direct and indirect relationships, thereby enhancing the quality of recommendations.

Looking ahead, future research could explore the integration of additional data, such as temporal or contextual interactions, and improve the scalability of the approach. The use of deep learning techniques and graph neural networks could also further enrich this method, paving the way for even more sophisticated and adaptive recommendation systems.

Références

[1] Q. Zhang, L. T. Yang, Z. Chen, et P. Li, « A survey on deep learning for big data », *Inf. Fusion*, vol. 42, p. 146-157, juill. 2018, doi: 10.1016/j.inffus.2017.10.006.

[2] K. He, X. Zhang, S. Ren, et J. Sun, « Deep Residual Learning for Image Recognition », présenté à Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, p. 770-778. Consulté le: 26 juillet 2024. [En ligne]. Disponible sur: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

[3] X. Wang, X. He, Y. Cao, M. Liu, et T.-S. Chua, « KGAT: Knowledge Graph Attention Network for Recommendation », in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, in KDD '19. New York, NY, USA: Association for Computing Machinery, juill. 2019, p. 950-958. doi: 10.1145/3292500.3330989.

[4] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, et P. S. Yu, « A Comprehensive Survey on Graph Neural Networks », *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, n° 1, p. 4-24, janv. 2021, doi: 10.1109/TNNLS.2020.2978386.

[5] K. He, X. Zhang, S. Ren, et J. Sun, « Deep Residual Learning for Image Recognition », présenté à Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, p. 770-778. Consulté le: 27 juillet 2024. [En ligne]. Disponible sur: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

[6] Y. Dong, N. V. Chawla, et A. Swami, « metapath2vec: Scalable Representation Learning for Heterogeneous Networks », in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, in KDD '17. New York, NY, USA: Association for Computing Machinery, août 2017, p. 135-144. doi: 10.1145/3097983.3098036.

[7] S. Abu-El-Haija *et al.*, « MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing », in *Proceedings of the 36th International Conference on Machine Learning*, PMLR, mai 2019, p. 21-29. Consulté le: 27 juillet 2024. [En ligne]. Disponible sur: https://proceedings.mlr.press/v97/abu-el-haija19a.html

[8] A. Grover et J. Leskovec, « node2vec: Scalable Feature Learning for Networks », in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, in KDD '16. New York, NY, USA: Association for Computing Machinery, août 2016, p. 855-864. doi: 10.1145/2939672.2939754.

[9] W. Hamilton, Z. Ying, et J. Leskovec, « Inductive Representation Learning on Large Graphs », in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017. Consulté le: 26 juillet 2024. [En ligne]. Disponible sur: https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html

[10] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, et E. Lefebvre, « Fast unfolding of communities in large networks », *J. Stat. Mech. Theory Exp.*, vol. 2008, n° 10, p. P10008, oct. 2008, doi: 10.1088/1742-5468/2008/10/P10008.

[11] A. Lancichinetti et S. Fortunato, « Community detection algorithms: A comparative analysis », *Phys. Rev. E*, vol. 80, nº 5, p. 056117, nov. 2009, doi: 10.1103/PhysRevE.80.056117.

[12] X. Huang, S. Qian, Q. Fang, J. Sang, et C. Xu, « CSAN: Contextual Self-Attention Network for User Sequential Recommendation », in *Proceedings of the 26th ACM international conference on Multimedia*, in MM '18. New York, NY, USA: Association for Computing Machinery, oct. 2018, p. 447-455. doi: 10.1145/3240508.3240609.

[13] B. Perozzi, R. Al-Rfou, et S. Skiena, « DeepWalk: online learning of social representations », in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, in KDD '14. New York, NY, USA: Association for Computing Machinery, août 2014, p. 701-710. doi: 10.1145/2623330.2623732.

[14] U. N. Raghavan, R. Albert, et S. Kumara, « Near linear time algorithm to detect community structures in large-scale networks », *Phys. Rev. E*, vol. 76, nº 3, p. 036106, sept. 2007, doi: 10.1103/PhysRevE.76.036106.

[15] S. Harenberg *et al.*, « Community detection in large-scale networks: A survey and empirical evaluation », *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 6, nov. 2014, doi: 10.1002/wics.1319.

[16] C. Liu, X. Kong, X. Li, et T. Zhang, « Collaborative Filtering Recommendation Algorithm Based on User Attributes and Item Score », *Sci. Program.*, vol. 2022, p. 1-7, mars 2022, doi: 10.1155/2022/4544152.

[17] J. Yao et B. Liu, « Community-Detection Method of Complex Network Based on Node Influence Analysis », *Symmetry*, vol. 16, nº 6, Art. nº 6, juin 2024, doi: 10.3390/sym16060754.

[18] G. Jeh et J. Widom, « SimRank: a measure of structural-context similarity », in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, in KDD '02. New York, NY, USA: Association for Computing Machinery, juill. 2002, p. 538-543. doi: 10.1145/775047.775126.