Enhancing a User Matchmaking Algorithm using Personalized PageRank

Santipong Thaiprayoon
santipong.thaiprayoon@fernuni-hagen.de
Fernuniversität in Hagen
Hagen, Germany

Herwig Unger
herwig.unger@fernuni-hagen.de
Fernuniversität in Hagen
Hagen, Germany

ABSTRACT

With the increasing number of users in online communities and social networking platforms, it is becoming more difficult for users to meet and connect with individuals who share similar opinions or interests. The paper proposes a user matchmaking algorithm based on personalized PageRank to provide potential friends to individual users. A set of user profiles is transformed into a graph model for efficiently discovering meaningful connections and influential users. The semantic relationship between two user profiles is then estimated using word and sentence embeddings. By incorporating both embedding models and personalized graph analytics, the proposed algorithm can capture complex semantic information and high-order user relationships, making the matchmaking process more accurate. Experiments conducted on a simulated user profile dataset show that the proposed algorithm consistently outperforms existing state-of-the-art methods in terms of the F1 score and mean average precision metrics.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; **Retrieval tasks and goals**; **Recommender systems**;

KEYWORDS

Social Networks, Matchmaking Algorithm, Friend Recommendation, Personalized PageRank, Graph Analytics, Semantic Textual Similarity

ACM Reference Format:
Santipong Thaiprayoon and Herwig Unger. 2023. Enhancing a User Matchmaking Algorithm using Personalized PageRank. In *2023 7th International Conference on Natural Language Processing and Information Retrieval (NLPIR 2023), December 15–17, 2023, Seoul, Republic of Korea.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3639233.3639346

1 INTRODUCTION

The ubiquity of online communities and social networking platforms has fundamentally transformed the way individuals connect, interact, communicate, and build meaningful relationships [14, 17]. These platforms are a collection of people with diverse backgrounds and a myriad of interests, leading to challenges in effectively identifying and recommending potential friends or collaborators. One of the most valuable tools is user matchmaking, which plays a significant role in facilitating users by suggesting potential friends or matches based on various factors associated with a particular user, such as personal information, interests, demographics, preferences, social connections, and behavioral data [20, 21].

Matchmaking algorithms are crucial for numerous online platforms, including dating applications, social networks, e-commerce sites, and online games [5]. These matchmaking algorithms utilize various methods and data sources to analyze user profiles, behaviors, and feedback to generate personalized recommendations for potential matches. For example, in the context of online dating sites, it can assist users in expanding their social circle and meeting romantic partners who share their values, goals, and personality traits. Similarly, social media platforms apply user matchmaking algorithms to match users based on their interests, preferences, and behaviors. Additionally, gaming platforms use matchmaking algorithms to match players based on their skill level. This leads to increased user engagement and satisfaction, which benefits both users and platforms.

Of course, the current state of research in the fields of user matchmaking and friend recommendation relies on a variety of techniques [2, 3, 10], including Collaborative Filtering (CF), Graph-Based Approaches (GBA), Natural Language Processing (NLP), and Machine Learning (ML). Most studies have attempted to propose a hybrid model that incorporates several techniques, such as content-based filtering, distance similarity, and user-based collaborative filtering, with semantic and social recommendations. The semantic dimension suggests semantically close friends based on calculating the similarity between members by leveraging interest and preference data. The social dimension is based on some social-behavior metrics, such as friendship and credibility degree [11]. The remaining study proposed a friend recommendation method based on the social structures and behaviors of users [22]. The degree of interaction between users is computed to recommend candidate friends using random walk algorithms with a restart model. However, previous studies still lack research on the semantic-based similarity of users in textual data and graph-based personalized matchmaking. This facilitates improving the matchmaking process, resulting in enhanced accuracy and personalized matching.

To bridge the gap, the paper proposes a user matchmaking algorithm using Personalized PageRank (PPR) and semantic analysis. The proposed algorithm aims to discover potential users with personalized and relevant recommendations based on user profiles, enhancing the social experience and satisfaction in online communities and social networking platforms. This algorithm leverages

NLPIR 2023, December 15–17, 2023, Seoul, Republic of Korea

ACM ISBN 979-8-4007-0922-7/23/12
https://doi.org/10.1145/3639233.3639346

textual profiles of users consisting of various attributes, including interests, hobbies, occupations, and biographies, to construct a weighted graph for efficiently finding hidden connections between users and influential users. The relationship between two user profiles is calculated using word and sentence embedding techniques to capture their semantic textual similarity rather than lexical similarity. This algorithm can also determine complex semantic information, individual priorities with a particular user, and high-order user relationships by utilizing the weighted graph. The PPR method then ranks the importance of each user in the graph and provides a list of matching users or friends to individual users according to PageRank score values.

The main contributions of this paper are summarized as follows:

- This paper proposes a new perspective on user matchmaking algorithms by utilizing the PPR algorithm. The goal is to automatically identify users within a virtual community who share common interests, preferences, and traits.
- This paper introduces a method for measuring the semantic similarity between textual user profiles based on word and sentence embeddings. This method could help to understand the meanings between texts and achieve superior performance in text matching.
- Extensive experiments on a simulated user profile dataset are conducted to evaluate the performance and effectiveness of the proposed algorithm in comparison to existing state-of-the-art methods.

The remainder of this paper is organized as follows: Section 2 provides a concise overview of the related work on user matching and friend recommendation. In Section 3, the proposed algorithm is described. The experimental details are presented in Section 4. Section 5 discusses experimental results. The limitations of evaluation and impact research are explained in Section 6. Section 7 gives opportunities and challenges. Section 8 concludes the paper and outlines some possible directions for future work.

2 RELATED WORK

The field of user matchmaking and recommendation algorithms has gained significant attention in recent years. This section provides a review of research studies on the recommendation and matchmaking of users, also known as friends, members, or individuals, which aim to recommend friends to users based on their preferences, interests, and social connections. The review of research studies discusses the different types of user matchmaking algorithms, their advantages and disadvantages, and their effectiveness in different settings.

Some of the early research studies focused on content-based filtering, which utilizes user preferences and interests to recommend friends or people. For example, a study by Jingda et al. [15] proposed an algorithm based on Latent Dirichlet Allocation (LDA) for recommending friends to learners in online education. The algorithm groups together learners with similar learning interests to find the top-k friend recommendation sequences. It does this by making learner document datasets, figuring out how similar learners are, and modeling the friend topic. Huansheng et al. [19] proposed a friend recommendation system that utilizes the Big-Five personality traits model and hybrid filtering to improve the accuracy of the system. The system considers the personality traits and harmony ratings of users instead of common physical or social features. Anju et al. [23] proposed a buddy recommendation model that employs collaborative filtering to compare and contrast similar and dissimilar user data. The model generates user recommendations based on their comparable choices, activities, and preferences. Finally, Ali et al. [4] proposed a framework for a friend recommender system that leverages hashtags to enhance the content and quality of user profiles. The framework initiates the construction of a user profile by leveraging shared hashtags. The matching approach is utilized to calculate the degree of similarity between profiles and group users with similar interests using advanced clustering methods.

Others have explored graph-based approaches, which leverage the social connections between users to recommend potential friends. For instance, Runa et al. [8] developed an integrated framework that learns user attributes, associated interactions, network structure, and timeline history from an online social network using a graph-theoretic approach to generate friend recommendations. Bu-Xiao et al. [27] developed a user similarity graph based on the interests of individual users. The LDA algorithm is used to determine topics of interest to users. The graph is constructed using the multi-view similarity method, which calculates the degree to which users share similar topics of interest. The system then makes recommendations between users by analyzing the graph. Additionally, there have been research efforts to incorporate machine learning and deep learning models to improve the accuracy of user matchmaking and recommendation systems [3].

Despite the advancements of previous approaches, a few weaknesses still need to be addressed. One weakness is that they do not fully consider the semantic relationship between users in textual data. Additionally, there is a lack of research on analyzing the connections between users in a graph model with personalization. To address these shortcomings, this paper introduces a user matchmaking algorithm from a new perspective by combining semantic analysis and the PPR method. This approach can improve the accuracy and personalization of the matchmaking algorithm.

3 METHODOLOGY

This section describes the methodology for discovering potential users using the PPR method and semantic analysis, considering textual user profiles instead of mutual friends or social graph criteria. The goal of this algorithm is to automatically match users within a virtual community who have similar interests, preferences, and traits. The processing pipeline of the user matchmaking algorithm is illustrated in Figure 1.

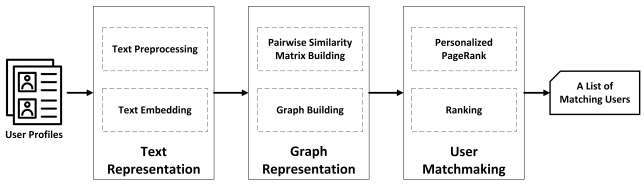

Figure 1: The Pipeline of the User Matchmaking Algorithm

The proposed algorithm starts by preprocessing each user profile. This involves cleaning the textual data, removing stop words,

and stemming words. Each text attribute of a user profile is then converted into a dense vector representation using embedding models. The cosine similarity score is calculated for each attribute pair of user profiles. These scores are then averaged to get an aggregated similarity score. The score is inserted into a similarity matrix to construct a weighted graph, where nodes represent users and edges represent their relationships. The weight of an edge can be an aggregated similarity score, which indicates the strength of the relationship between two users. The PPR method is then used to identify influential users based on a target user within the weighted graph and to generate a list of matching users who have similar profiles. The pseudocode of the user matchmaking algorithm is illustrated formally in Algorithm 1.

Algorithm 1: User Matchmaking Algorithm

Data: List of User Profiles P, Target User T
Result: Ranked List of Users According to PageRank Score to T

Function `Embed`$(text)$
  return *Vector representation of* $text$ *using embedding models*
Function `CosineSimilarity`(A, B)
  return $\frac{A \cdot B}{\|A\|_2 \times \|B\|_2}$
Function `PersonalizedPageRank`$(G, target\_user)$
  Initialize rank vector r with target user set to 1 and others set to 0
  Initialize transition matrix M from G
  while *not converged* **do**
    $r' = (1-d) \times M \times r + d \times v$
    $r = r'$
  end
  return sorted list of users based on r, excluding target user
for *each user profile* p_i *in* P **do**
  for *each user profile* p_j *in* P **do**
    $v\hat{e}c_i \leftarrow$ Concatenated `Embed`(*of all attributes of* P_i)
    $v\hat{e}c_j \leftarrow$ Concatenated `Embed`(*of all attributes of* P_j)
    $similarityMatrix[i][j] \leftarrow$ `CosineSimilarity`$(v\hat{e}c_i, v\hat{e}c_j)$
  end
end
for *each* $similarityMatrix[i][j]$ **do**
  Construct a weighted edge between user i and user j in graph G with weight $similarityMatrix[i][j]$
end
$rankedUsers \leftarrow$ `PersonalizedPageRank`(G, T)
return $rankedUsers$

Before discussing the proposed algorithm, essential notations are formally defined in Table 1. Let $U = \{U_1, U_2, ..., U_N\}$ be a set of users, $P = \{P_1, P_2, ..., P_N\}$ be a set of user profiles, and $W = \left\{w_{ij}^1, w_{ij}^2, ..., w_{ij}^m\right\}$ be a set of words. Given a set of N user profiles P_i, and $P_i = \{A_{i1}, A_{i2}, A_{i3}, A_{i4}\}$. Therefore, $w_{ij}^m \epsilon A_{ij} \epsilon P_i$ is the word m^{th} of A_{ij}, where $i = \{1, 2, ..., N\}$, and $j = \{1, 2, 3, 4\}$.

Table 1: Basic Notations and Symbols Used

Notations	Description
U	A set of all users
P	A set of all user profiles
N	The element number of user profile P
A_{i1}	The interest attribute of a user profile P_i
A_{i2}	The hobby attribute of a user profile P_i
A_{i3}	The occupation attribute of a user profile P_i
A_{i4}	The biography attribute of a user profile P_i
W	A set of all words or phrases in an attribute
$v\hat{e}c\ (A_{ij})$	A single vector representation of each attribute in P_i

The user matchmaking algorithm consists of three main parts: (1) text representation; (2) graph representation; and (3) user matchmaking. The following subsection explains each part in detail.

3.1 Text Representation

This part preprocesses user profiles and converts them to high-dimensional vector representations, called embeddings. These vector representations enable efficient comparison and matching of user profiles based on their textual information, which captures the semantic meaning of individual words and texts. The fundamental concept is that words or texts with similar meanings will have similar vectors. The text representation consists of two major components, which are described in the following sections.

3.1.1 Text Preprocessing. The text preprocessing component plays a significant role in boosting the performance of the proposed algorithm. The main idea behind this component is to clean up and transform unstructured textual profiles into a suitable format for further analysis. The following modules are involved in the text preprocessing component:

- `Tokenization`: This module breaks down the textual profiles into individual words, or tokens.
- `Lowercasing`: All the words in the profiles are converted to lowercase. This helps in standardizing the text and avoiding duplication of words due to case differences.
- `Removing Stop Words`: Stop words are common words such as "the", "is", and "and". They are removed from the profiles to reduce noise.
- `Removing Special Characters`: This module removes any special characters such as punctuation marks, symbols, and emojis from the profiles. This helps eliminate any unnecessary noise and ensure that only meaningful words are included.
- `Stemming and Lemmatization`: These modules convert words to their base or root form. This helps improve the accuracy of the proposed algorithm by grouping together words that have similar meanings.

The preprocessed user profiles are sent to the text embedding component, which generates high-dimensional vector representations.

3.1.2 Text Embedding. This component aims to convert each text attribute of a user profile into a dense vector representation by

utilizing two embedding models. The purpose of these dense vector representations is to understand the semantic relationships between users.

- `Term-based Embedding`: This embedding model considers the text attributes of a user profile, including interests, hobbies, and occupations. Each text attribute is mapped into the word vectors of all words in the attribute using Word2Vec, an existing pre-trained word embedding model with 300-dimensional word vectors [18]. These word vectors are then averaged as the mean vector to get a single vector, representing the attribute. For the Out-Of-Vocabulary (OOV) issue, words that do not exist in the vocabulary of the model are ignored. In the case of multi-word attributes, such as "data analysis", the average of the vectors of individual words is computed to represent the entire attribute. In the case of word order alternation, the meaning between two terms is considered to be the same. For instance, the terms "information system" and "system information" have the same meaning. Therefore, both terms have the same meaning in context. The text-based embedding is defined by Equation 1:

$$\hat{vec}\left(A_{ij}\right) = \frac{1}{n} \sum_{m=1, w_{ij}^m \epsilon A_{ij}}^{n} vec\left(w_{ij}^m\right) \text{ for } j\epsilon\{1,2,3\} \tag{1}$$

  where $\hat{vec}\left(A_{ij}\right)$ is a single vector representation of each attribute, n is the number of words, and $vec\left(w_{ij}^m\right)$ is a vector representation of a word.
- `Context-based Embedding`: This embedding model focuses on the biography attribute in a user profile. The biography attribute is directly converted into a single vector representation using a pre-trained Bidirectional Encoder Representations from Transformers (BERT), a transformer-based model [6]. This model is capable of understanding the context and semantic meanings of texts in an effective way. Instead of aggregating word vectors, this model considers the entire text as input and generates comprehensive embeddings that capture the overall semantic content. This ensures that the vector representation captures not only individual word meanings but also the overall context and theme of the biography attribute. The context-based embedding is defined by Equation 2:

$$\hat{vec}\left(A_{i4}\right) = BERT_{w_{i4}^m \epsilon A_{i4}}(A_{i4}) \tag{2}$$

  where $\hat{vec}\left(A_{i4}\right)$ is a single vector representation of the biography attribute.

Finally, the embedded vector representations are utilized to compute the semantic similarity scores between different user profiles in the next part.

3.2 Graph Representation

This part uses a pairwise similarity matrix to construct a weighted graph model that represents the semantic relationships between different users. The edges between nodes indicate their scores for semantic similarity. The graph representation consists of two main components, which are described in the following sections.

3.2.1 Pairwise Similarity Matrix Building. This component calculates the cosine similarity between every pair of user profiles based on their vector representations and inserts them into a pairwise similarity matrix. This matrix construction consists of the following three steps:

- `Cosine Similarity Calculation`: The cosine similarity calculation is used to compute the semantic similarity score between two single vectors for every attribute pair. This calculation is performed for each attribute pair in the user profiles, resulting in a similarity score for each pair. The calculation of the semantic similarity between any two single vectors vec_1 and vec_2, representing attributes from user profiles, is defined by Equation 3:

$$cosine\_similarity\_score(\hat{vec}_1, \hat{vec}_2) = \frac{\hat{vec}_1 \cdot \hat{vec}_2}{||\hat{vec}_1||_2 \times ||\hat{vec}_2||_2} \tag{3}$$

  where $\hat{vec}_1$ and $\hat{vec}_2$ are the vector representations of two attributes, $\hat{vec}_1 \cdot \hat{vec}_2$ is the dot product of the vectors, $||\hat{vec}_1||_2$ denotes the L2 norm of the vector representation of $\hat{vec}_1$, and $||\hat{vec}_2||_2$ denotes the L2 norm of the vector representation of $\hat{vec}_2$. The cosine similarity value ranges from -1 to 1, where a value of 1 indicates perfect similarity and a value of -1 indicates complete dissimilarity.
- `Similarity Score Aggregation`: To combine different attributes of a user profile, this step calculates an aggregated similarity score for two user profiles by taking a simple average of all the cosine similarity scores obtained in the previous step across their multiple attributes. The simple average is calculated by summing up all the similarity scores and then dividing by the number of scores. This gives equal importance to every score. The score aggregation is calculated using Equation 4:

$$aggregated\_similarity\_score(U_1, U_2) = \frac{\sum_{j=1}^{q} cosine\_similarity\_score_j}{q} \tag{4}$$

  where $cosine\_similarity\_score_j$ is the cosine similarity of the j^{th} attribute, and q is the total number of attributes.
- `Matrix Insertion`: The aggregated similarity score of two users is inserted into a pairwise similarity matrix. This is a square matrix that stores the similarity scores between pairs of users. The matrix is of size $N \times N$. The N is the total number of users. Each row and column represent a user, and each entry represents the similarity score between two users.

The pairwise similarity matrix serves as a foundation for building a weighted graph model, enabling further analysis and visualization of the relationships between different users.

3.2.2 Graph Building. A weighted graph, $G = (V, E, S)$, where V is the set of nodes, E is the set of edges, and W is the set of weights, is built from the pairwise similarity matrix. Each node V_h in the graph represents a user. If the aggregated similarity score between two nodes exceeds a certain threshold set to 0.70, they are connected with an edge. The weight S_h of the edge E_h is then set by an aggregated similarity score, which indicates the strength

of the relationship between them. A high score refers to a strong relationship, and a weak relationship has a low score. The edge weighting is defined by Equation 5:

$$S_h = \begin{cases} aggregated\_similarity\_score\,(U_1, U_2), & \text{if } aggregated\_similarity\_score > \tau \\ \text{No } aggregated\_similarity\_score, & \text{Otherwise} \end{cases} \tag{5}$$

where τ is a predefined threshold.

3.3 User Matchmaking

This part enables the PPR algorithm to take into account the preferences and interests of users to generate a list of matching users. Each user is assigned a ranking based on their connections and similarity scores. The user matchmaking algorithm consists of two main components, which are described in the following sections.

3.3.1 Personalized PageRank. The PPR algorithm is a graph-based algorithm that assigns a score to each node in a graph based on their connections and importance. This score reflects the influence of each node in the graph. In the context of user matchmaking, the PPR algorithm can be used to identify the most influential users based on their similarity to a target user. The PPR algorithm takes into account the relevance of each connection and similarity score, ensuring that users with more meaningful connections are ranked higher. The PPR algorithm modifies the original PageRank algorithm slightly to produce personalized matchmaking for users. Instead of jumping to any random node, the algorithm biases the random walk towards a specific starting node or a set of nodes based on a personalization vector. This provides a set of nodes that are reachable from the starting node. The steps of the PPR algorithm for user matchmaking are explained below.

- `Step 1`: Start with a vector r where the entry corresponding to the target user of interest is set to 1, and all others are set to 0.
- `Step 2`: The equation 6 for updating the PageRank vector r' in each iteration can be represented as:

$$r' = (1 - d) \times M \times r + d \times v \tag{6}$$

  where d is the damping factor, typically set to 0.85, M is the transition matrix derived from the weighted graph, and v is a personalization vector which represents the probability of jumping to the node corresponding to the target user.
- `Step 3`: This process must be repeated until it converges.
- `Step 4`: After convergence, the resulting r gives a ranking of higher scores to users that are reachable from the target user.

The PPR algorithm is an efficient method for matching users based on their similarity to a target user. The algorithm starts by setting the entry corresponding to the target user to 1 in the starting vector r. It then updates the vector iteratively until convergence. The algorithm assigns higher scores to users who are reachable from the target user. This ranking helps identify potential matches based on their connectivity and relevance to the target user.

3.3.2 Ranking. After convergence, the component sorts the user indices based on their PageRank scores for the target user in descending order. The top-k users are the users who are most similar to the target user, where k is a threshold or specific number.

By incorporating the semantic similarity of users into textual data, it is possible to obtain an in-depth understanding of user preferences, interests, and traits. This approach can enhance the accuracy and effectiveness of personalized user matchmaking by considering not only social structures and behaviors but also the content shared by users. Additionally, integrating personalized graph-based approaches into matchmaking algorithms can further improve the quality of personalized user matchmaking by leveraging the semantic relationships between users within a graph.

4 EXPERIMENTS

This section describes a series of experiments conducted on a simulated user profile dataset to compare the performance of the proposed algorithm to existing state-of-the-art methods. The primary objective of the experimental design is to address the following Research Questions (RQs):

- `RQ1`: Can the proposed algorithm outperform state-of-the-art graph methods?
- `RQ2`: Which graph-based method has better performance, and why is that?

The following subsections detail the experimental procedures, including dataset descriptions, evaluation metrics, baseline methods, and experimental settings.

4.1 Dataset Description

User profiles typically contain private and sensitive information about individuals, making it difficult to collect and access this information without their consent. To address this issue, AI text generation systems, such as ChatGPT [12] and Google Bard [16], are used to generate a user profile dataset that simulates human language autonomously. The simulated user profile dataset includes 150 user profiles with various attributes, including names, interests, hobbies, occupations, and biographies. The attributes of interests, hobbies, occupations, and skills are typically described in the form of controlled vocabulary. On the other hand, a biography attribute is a short text written in natural language format that provides a background summary of a user. Lastly, this dataset serves as a benchmark test for user matchmaking algorithms. By using this dataset, researchers can evaluate the performance of their algorithms without having to collect and access personal information.

The simulated user profile dataset is randomly divided into two sets: a training set and a test set. The training set is used to construct a graph structure and learn features of the proposed algorithm, while the test set is used to evaluate the performance of the proposed algorithm. The training set contains 100% of the users in the whole dataset, while the test set contains 20% of the users in the whole dataset selected using a random sampling technique. Meanwhile, the test set is prepared as a ground-truth dataset, where human judgment is used to assign matches manually.

4.2 Evaluation Metrics

The accuracy of the proposed algorithm and state-of-the-art methods is measured using common evaluation metrics for recommender systems [25]. This includes precision at k denoted as $P@k$, recall at k denoted as $R@k$, F1 score at k denoted as $F1@k$, and mean average precision at k denoted as $MAP@k$. The following evaluation metrics are described below:

- `Precision@k`: This metric measures the accuracy of the top-k users by computing the fraction of relevant users that are found in the top-k users. A high precision indicates that most of the top-k users are matched, while a low precision indicates that many irrelevant users are matched among the top-k users. The precision at k is defined by Equation 7 as follows:

$$P@k = \frac{\text{Number of Relevant Users in Top-}k\text{ Matches}}{k} \tag{7}$$

- `Recall@k`: This metric measures the fraction of relevant users that are found in the top-k users. A high recall indicates that most of the relevant users are found among the top-k users. The recall at k is defined by Equation 8 as follows:

$$R@k = \frac{\text{Number of Relevant Users in Top-}k\text{ Matches}}{\text{Total Number of Relevant Users}} \tag{8}$$

- `F1 Score@k`: This is a weighted average of precision and recall at k that balances both metrics into a single number. It ranges from 0 to 1, where higher values indicate better performance. Equation 9 is defined as follows:

$$F1@k = \frac{2 \times P@k \times R@k}{P@k + R@k} \tag{9}$$

- `Mean Average Precision@k`: This metric provides an overview of the quality of the ranking algorithm, which is the mean of the Average Precisions (APs) across all users. The mean of the AP score for each user and the MAP are defined by Equations 10, 11, and 12, respectively, as follows:

$$MAP@k = \frac{\sum_{u=1}^{|U|} AP@k_u}{|U|} \tag{10}$$

$$AP@k = \frac{\sum_{k=1}^{K} P@k \times rel_k}{\text{Total Number of Relevant Users}} \tag{11}$$

$$rel_k = \begin{cases} 1, & \text{If a user at } k \text{ rank is relevant} \\ 0, & \text{Otherwise} \end{cases} \tag{12}$$

where $AP@K$ is the average precision at k, rel_k is a binary indicator that is 1 if the user at rank k is relevant and 0 otherwise, $P@k$ is the precision at rank k, $|K|$ is the number of relevant users, and $|U|$ is the number of users.

4.3 Baseline Methods

To demonstrate the effectiveness of the proposed algorithm, existing state-of-the-art methods, considering graph-based approaches, were compared. These methods are briefly described below:

- `Random Walk` [7]: This method simulates random paths in a graph by following some rules. The basic idea is to start at a node in the graph and then choose one of its neighbors to move to. This process is repeated until a certain number of steps are taken or a certain condition is met.
- `Random Walk with Weighted Edges` [1]: This method generates random paths in a graph, where the probability of moving from one node to another depends on the edge weights. The basic idea is that it starts at a node in the graph and then randomly chooses one of its neighbors to move to, according to the weights of the edges.
- `SimRank` [28]: This is a graph-based method for calculating the similarity between two nodes. The similarity of two nodes is determined by the similarity of the nodes that they are connected to. In other words, two nodes are considered similar if they are connected to similar nodes. The nodes with the highest SimRank scores are considered to be the most similar to the starting node.
- `Girvan-Newman Algorithm` [24]: This method detects communities or clusters, which are groups of nodes that are more densely connected to each other than to the rest of the graph. The method works by iteratively removing the edges with the highest betweenness centrality, which measures how frequently an edge lies on the shortest path between any two nodes.
- `Louvain` [9]: This method detects communities in a graph. It maximizes a modularity score for each community, where modularity measures the quality of assigning nodes to communities.
- `Node2Vec` [26]: This method embeds nodes in a graph into low-dimensional vector representations that preserve neighborhood relationships, such that nodes that are similar in the graph are also similar in the embedding space. It uses skip-gram models to learn node embeddings based on the context of the random walks and a random walk strategy to find structural similarities between nodes in the graph.

Girvan-Newman and Louvain are used to partition groups of similar users. Consequently, users from the same cluster are matched, with the underlying assumption that users in the same group as the target user are likely to be similar.

4.4 Experimental Settings

Extensive experiments were conducted on the simulated user profile dataset, which imitates real-world user preferences and characteristics, ensuring the validity of the experimental results. The proposed algorithm was evaluated against existing state-of-the-art methods using both the F1 and MAP metrics for the top 5, 10, 15, and 20 candidate users.

All experiments were written in the Python programming language and executed on a Personal Computer (PC) with an Intel (R) Core (TM) i5-4570 CPU at 3.20 GHz and 8 GB of DDR3 RAM, which was running on an Ubuntu 22.04 LTS Linux server. Each method was implemented and tested under the same experimental conditions for a fair comparison. The hyperparameters of each method were safely set to default values according to the NetworkX library [13], aiming to maintain consistency and eliminate any potential bias in the comparison of the different methods.

5 EXPERIMENTAL RESULTS AND DISCUSSION

The experimental results on the simulated user profile dataset are reported in Table 2 to address RQ1 and RQ2, which are concerned

with the effectiveness of the proposed algorithm in predicting user matches. The best results are in boldface, and the second-best results are underlined.

Table 1 shows that the proposed algorithm significantly outperforms all state-of-the-art methods for user matchmaking in terms of F1 and MAP metrics across the top 5, 10, 15, and 20 candidate users. The proposed algorithm achieves an average improvement of more than 8% over the best baseline method. The second-best method is Node2Vec, which uses random walks to embed nodes in a low-dimensional vector space and a skip-gram model to learn their representations. This method is able to capture the structural relationships between nodes in a graph. The third-best method is SimRank, which measures the similarity between two nodes based on their structural context in the graph. This method does not consider the weights of the edges between nodes. The worst methods are Random Walk, Girvan-Newman, and Louvain. These methods do not consider edge weights in their computations. Random Walk is a simple method that selects one adjacent node at random and assigns equal weights to each edge. Girvan-Newman and Louvain focus primarily on detecting communities or clusters within graphs. Their goal is to partition a graph into clusters, which is not directly optimized for finding similar users.

There are several reasons for the higher performance of the proposed algorithm. Firstly, the proposed algorithm based on the PPR method assigns weights to each edge, representing semantic relationships between users in the graph. This allows it to capture the relationships between users based on their profiles more accurately. For example, if two users have similar interests, the PPR method assigns a higher weight to the edge between them. Secondly, PPR introduces a personalized aspect to the ranking mechanism. This means it is designed to prioritize nodes based on a particular source node, ensuring the results are more tailored and relevant. For example, if a user is looking for friends who are interested in hiking, the PPR method will prioritize nodes connected to the user node with a high weight for hiking interests. Lastly, this personalization factor ensures that a traversal of the graph does not travel too far from the starting node. This characteristic is especially valuable in contexts where the graph is large and the goal is to find nodes similar to a particular starting node. For example, if a user is looking for friends in a large city, the PPR method will prioritize nodes that are close to the user.

In summary, the experimental results indicate that the proposed algorithm is significantly effective in predicting users who are similar to the target user in terms of preferences, interests, and traits. This is because the algorithm uses both edge weights calculated by text embedding techniques and personalized nodes in the random walk process. Text embedding techniques allow the algorithm to capture the semantic similarities between nodes, while personalized nodes allow the algorithm to capture the preferences and interests of the target user. Therefore, the proposed algorithm is a suitable and effective solution for user matchmaking on a variety of real-world online platforms.

6 LIMITATIONS

One major limitation of this evaluation is the lack of public or real-world datasets suitable for evaluating user matchmaking. This means that the findings of this study may need to be more generalizable to real-world scenarios. Additionally, the study considered a relatively small dataset, which limits the scalability of the methods. Future work could involve scalability testing on larger datasets for comprehensive evaluations. Additionally, deeper dives into the hyperparameters of personalized PageRank, such as the number of iterations, could provide further improvements. It is essential to tune these learning parameters carefully to achieve the best results.

7 OPPORTUNITIES AND CHALLENGES

User matchmaking systems have become key components of many social media platforms and online services. Their primary goal is to connect users based on shared interests, habits, or other contextual information, offering matching friends or users that meet individual needs. However, these systems also face some opportunities and challenges.

User matchmaking systems have the potential to significantly enhance effectiveness and accuracy by integrating contextual information about users, such as geographical location, social connections, and behavioral data. By leveraging this information, these systems significantly impact the effectiveness of user matchmaking, leading to increased user satisfaction and retention. Intelligent matchmaking systems should consider various factors and utilize machine learning and data analysis techniques to provide more accurate and personalized matches by learning and adapting user data over time. However, integrating these multiple data points into a user matchmaking algorithm requires advanced techniques and a deep understanding of human behavior.

Another challenge is that user matchmaking systems often rely on sensitive and behavioral information that users do not want to reveal. This raises concerns about user privacy and security. To protect against information leakage, these systems should implement privacy and security measures to ensure that user data is protected and anonymized. This includes processing raw data locally instead of uploading it to servers and having clear and easy-to-understand privacy policies that explain what data is collected, how it is used, and who it is shared with. Additionally, providing users with control over the information they share can help build trust and encourage participation in user matchmaking systems. For example, users should be able to choose who can see their profile or what information is visible to others. Finally, these systems must follow data protection laws and regulations, such as the General Data Protection Regulation (GDPR) in the European Union or the California Consumer Privacy Act (CCPA) in the United States. They must also take steps to prevent the spread of misinformation and moderate content to guarantee user safety. Implementing strong security measures, such as encryption and secure authentication protocols, is also crucial to safeguarding user data and ensuring their privacy. Additionally, regularly updating and auditing these systems can help identify and address any potential vulnerabilities that could compromise user information.

Last but not least, new users may also face challenges in building their profiles and establishing their preferences within the platform. Without sufficient data, matchmaking algorithms may struggle to accurately suggest compatible matches for these users. However, implementing strategies such as employing hybrid recommender

Table 2: Experimental Results with F1 and MAP Metrics Comparing with Classical Methods

Methods	F1 Score				MAP			
	F1@5	F1@10	F1@15	F1@20	MAP@5	MAP@10	MAP@15	MAP@20
Random walk	0.204	0.243	0.184	0.232	0.146	0.230	0.183	0.242
Random walk w/ weighted edges	0.252	0.296	0.271	0.265	0.188	0.270	0.282	0.273
SimRank	0.373	0.493	0.533	0.493	0.245	0.374	0.474	0.512
Girvan-newman algorithm	0.045	0.047	0.051	0.100	0.020	0.023	0.013	0.028
Louvian	0.406	0.480	0.479	0.429	0.269	0.399	0.444	0.462
Node2Vec	0.542	0.580	0.557	0.513	0.431	0.582	0.668	0.696
The proposed algorithm	**0.543**	**0.664**	**0.626**	**0.543**	**0.432**	**0.654**	**0.742**	**0.764**
Improvement (%)	0.184	14.482	12.387	5.847	0.232	12.371	11.077	9.770

systems to make initial recommendations based on similar user profiles and item characteristics or providing incentives for new users to actively engage with the platform can help overcome these challenges and improve the overall user experience.

8 CONCLUSION

This paper introduces a new perspective on user matchmaking strategies by proposing a user-based matchmaking algorithm that uses personalized PageRank. This algorithm aims to identify potential users with similar interests, preferences, and traits based on semantic analysis and graph analytics. The personalization feature of the graph allows the algorithm to specify the number of neighboring nodes surrounding a source node. The algorithm also uses advanced natural language processing techniques to gain a deeper understanding of textual user profiles. This helps improve the accuracy of the matchmaking process. Extensive experiments on a simulated user profile dataset demonstrate that the proposed algorithm consistently outperforms existing state-of-the-art methods. This suggests that the proposed algorithm is a promising approach for user matchmaking. Finally, the proposed user matchmaking algorithm can be implemented in several domains, including social networking sites, online dating platforms, and online multiplayer games. In these domains, the algorithm can match users with similar interests, preferences, and characteristics, improving the overall user experience.

To enhance the effectiveness and accuracy of the proposed algorithms, future research should incorporate additional contextual information about users. This could include various factors, such as time, location, weather, demographic information, behavioral data, historical interactions, and activity. In addition, reinforcement learning techniques are used to continuously improve and refine the process of user matchmaking by collecting both explicit and implicit user feedback derived from the interactions and characteristics of users over time.

REFERENCES

[1] Hakan Bagci and Pinar Karagoz. 2016. Context-aware location recommendation by using a random walk-based approach. *Knowledge and Information Systems* 47 (2016), 241–260.

[2] Lamia Berkani. 2020. A semantic and social-based collaborative recommendation of friends in social networks. *Software: Practice and Experience* 50, 8 (2020), 1498–1519. https://doi.org/10.1002/spe.2828 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/spe.2828

[3] Liang Chen, Yuanzhen Xie, Zibin Zheng, Huayou Zheng, and Jingdun Xie. 2020. Friend Recommendation Based on Multi-Social Graph Convolutional Network. *IEEE Access* 8 (2020), 43618–43629. https://doi.org/10.1109/ACCESS.2020.2977407

[4] Ali Choumane and Zein Al Abidin Ibrahim. 2020. Friend Recommendation based on Hashtags Analysis. *CoRR* abs/2003.03531 (2020). arXiv:2003.03531 https://arxiv.org/abs/2003.03531

[5] Indrakant Dana, Udit Agarwal, Akshat Ajay, Saurabh Rastogi, and Ahmed Alkhayyat. 2023. Recommendation Mechanism to Forge Connections Between Users with Similar Interests. In *International Conference on Innovative Computing and Communications*, Aboul Ella Hassanien, Oscar Castillo, Sameer Anand, and Ajay Jaiswal (Eds.). Springer Nature Singapore, Singapore, 455–470.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Magdalini Eirinaki, Jerry Gao, Iraklis Varlamis, and Konstantinos Tserpes. 2018. Recommender systems for large-scale social networks: A review of challenges and solutions. , 413–418 pages.

[8] Runa Ganguli, Akash Mehta, Narayan Debnath, Sultan Aljahdali, and Soumya Sen. 2020. An Integrated Framework for Friend Recommender System based on Graph Theoretic Approach. In *Proceedings of 35th International Conference on Computers and Their Applications (EPiC Series in Computing, Vol. 69)*, Gordon Lee and Ying Jin (Eds.). EasyChair, 242–255. https://doi.org/10.29007/4bwn

[9] Fabio Gasparetti, Giuseppe Sansonetti, and Alessandro Micarelli. 2021. Community detection in social recommender systems: a survey. *Applied Intelligence* 51 (2021), 3975–3995.

[10] Way-Siang Goh, Chian-Wen Too, Meei-Hao Hoo, and Kok-Chin Khor. 2023. Evaluation of Recommendation Models for Matchmaking. In *Intelligent Computing & Optimization*, Pandian Vasant, Gerhard-Wilhelm Weber, José Antonio Marmolejo-Saucedo, Elias Munapo, and J. Joshua Thomas (Eds.). Springer International Publishing, Cham, 843–852.

[11] Jibing Gong, Xiaoxia Gao, Yanqing Song, Hong Cheng, and Jingjing Xu. 2016. Individual Friends Recommendation Based on Random Walk with Restart in Social Networks. In *Social Media Processing*, Yuming Li, Guoxiong Xiang, Hongfei Lin, and Mingwen Wang (Eds.). Springer Singapore, Singapore, 123–133.

[12] Roberto Gozalo-Brizuela and Eduardo C. Garrido-Merchan. 2023. ChatGPT is not all you need. A State of the Art Review of large Generative AI models. arXiv:2301.04655 [cs.LG]

[13] Aric Hagberg and Drew Conway. 2020. *Networkx: Network analysis with python.* Retrieved September 10, 2023 from https://networkx.github.io

[14] Richard Harrison and Michael Thomas. 2009. Identity in online communities: Social networking sites and language learning. *International Journal of Emerging Technologies and Society* 7, 2 (2009), 109–124.

[15] Jingda Kang, Juntao Zhang, Wei Song, and Xiandi Yang. 2021. Friend Relationships Recommendation Algorithm in Online Education Platform. In *Web Information Systems and Applications*, Chunxiao Xing, Xiaoming Fu, Yong Zhang, Guigang Zhang, and Chaolemen Borjigin (Eds.). Springer International Publishing, Cham, 592–604.

[16] James Manyika. 2022. *An overview of Bard: an early experiment with generative AI.* https://ai.google/static/documents/google-about-bard.pdf

[17] Lynn A McFarland and Robert E Ployhart. 2015. Social media: A contextual framework to guide research and practice. *Journal of applied psychology* 100, 6 (2015), 1653.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs.CL]

[19] Huansheng Ning, Sahraoui Dhelim, and Nyothiri Aung. 2019. PersoNet: Friend Recommendation System Based on Big-Five Personality Traits and Hybrid Filtering. *IEEE Transactions on Computational Social Systems* 6, 3 (2019), 394–402. https://doi.org/10.1109/TCSS.2019.2903857

[20] Iván Palomares, Carlos Porcel, Luiz Pizzato, Ido Guy, and Enrique Herrera-Viedma. 2021. Reciprocal Recommender Systems: Analysis of state-of-art literature, challenges and opportunities towards social recommendation. *Information Fusion* 69 (2021), 103–127.

[21] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2015. Recommender systems: introduction and challenges. *Recommender systems handbook* (2015), 1–34.

[22] Animesh Chandra Roy and A. S. M. Mofakh Kharul Islam. 2023. Friend Recommendation System Based on Heterogeneous Data from Social Network. In *Proceedings of International Joint Conference on Advances in Computational Intelligence*, Mohammad Shorif Uddin and Jagdish Chand Bansal (Eds.). Springer Nature Singapore, Singapore, 565–580.

[23] Anju Taiwade, Nitish Gupta, Rakesh Tiwari, Shashi Kumar, and Upendra Singh. 2022. Hierarchical K-Means Clustering Method for Friend Recommendation System. In *2022 International Conference on Inventive Computation Technologies (ICICT)*. 89–95. https://doi.org/10.1109/ICICT54344.2022.9850852

[24] Sadriddinov Ilkhomjon Rovshan Ugli, Doo-Soon Park, Daeyoung Kim, Yixuan Yang, Sony Peng, and Sophort Siet. 2022. Movie Recommendation System Using Community Detection Based on the Girvan–Newman Algorithm. In *International Conference on Computer Science and its Applications and the International Conference on Ubiquitous Information Technologies and Applications*. Springer, 599–605.

[25] Daniel Valcarce, Alejandro Bellogín, Javier Parapar, and Pablo Castells. 2020. Assessing ranking metrics in top-N recommendation. *Information Retrieval Journal* 23, 4 (June 2020), 411–448. https://doi.org/10.1007/s10791-020-09377-x

[26] Janu Verma, Srishti Gupta, Debdoot Mukherjee, and Tanmoy Chakraborty. 2019. Heterogeneous edge embedding for friend recommendation. In *Advances in Information Retrieval: 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14–18, 2019, Proceedings, Part II 41*. Springer, 172–179.

[27] Bu-Xiao Wu, Jing Xiao, and Jie-Min Chen. 2015. Friend Recommendation by User Similarity Graph Based on Interest in Social Tagging Systems. In *Advanced Intelligent Computing Theories and Applications*, De-Shuang Huang and Kyungsook Han (Eds.). Springer International Publishing, Cham, 375–386.

[28] Lu Yang, Tao Hong, and Anilkmar Kothalil Gopalakrishnan. 2018. A Framework for Recommender System Based on Game Theory in Social Networks. In *2018 10th International Conference on Knowledge and Smart Technology (KST)*. IEEE, 95–100.