Game Recommendation Model with Steam API

Junhwi Jeong
Department of Informatics
Indiana University
Bloomington, IN
jeonjunh@iu.edu

Elizabeth Jiang
Department of Computer Science
Indiana University
Bloomington, IN
elijiang@iu.edu

Jack Liang
Department of Computer Science
Indiana University
Bloomington, IN
jacklian@iu.edu

***Abstract*—The rapid expansion of the Steam platform has made a vast array of video games accessible to users worldwide. However, this abundance of choice poses a significant challenge: helping users discover games that match their interests and preferences. Many users purchase games only to realize that the gameplay or themes do not align with their expectations, resulting in underutilized libraries and wasted expenses. To address this issue, we developed a personalized game recommendation model that leverages the existing game libraries and engagement histories of users. Using game metadata such as genre and tags, we applied dimensionality reduction techniques like t-SNE and clustering algorithms like k-means to group games and users based on shared characteristics. By analyzing these clusters, our model identifies patterns in user preferences and generates tailored game recommendations. This approach not only enhances user satisfaction by increasing the likelihood of discovering enjoyable games but also minimizes the frequency of regrettable purchases, ultimately enriching the gaming experience for users.**

***Definitions*—t-SNE, k-Means, Steam, Game, Classification**

I. Introduction

The rapid growth of game distribution platforms like Steam, a popular service that allows users to buy and play digital games, has changed how people find and play video games. With a constantly growing selection of games in different genres, themes, and styles, users can enjoy many gaming experiences right from home. However, this accessibility comes with a challenge: How can users effectively identify games that align with their individual preferences and avoid making purchases that they may later regret? For many players, navigating such an extensive catalog can be overwhelming, leading to purchases made without enough information. This problem is significant because it not only results in financial loss for the users but also reduces overall satisfaction with their gaming experience. Current solutions range from simple user ratings and review-based systems to more advanced recommendation algorithms used by digital platforms. Although review-based recommendations offer some guidance, they often fail to capture the more subtle preferences of users. Collaborative filtering, used by platforms such as Netflix and Amazon, has been applied to gaming, but it faces challenges in recommending games to new users because of limited data. In recent years, more advanced models have been developed to improve the accuracy of recommendations. For example, content-based filtering looks at the characteristics of the games themselves, such as genre, gameplay style, and ratings. Hybrid approaches combine both content-based and collaborative filtering methods to take advantage of the strengths of each. Despite these improvements, challenges still exist in making recommendations that better match users' changing interests and behaviors. To address these limitations, we propose a personalized game recommendation model that utilizes data mining techniques to build comprehensive user profiles. By analyzing users' game libraries and engagement histories, our model incorporates game metadata, including genre, user ratings, and popularity, to create tailored recommendations. By using clustering techniques like k-means, the model groups users based on similar gaming interests and behaviors, allowing for recommendations that align with the preferences of each group. This approach helps match users with games more accurately, improving user satisfaction and lowering the chance of bad purchases. Focusing on data to personalize the experience, the model aims to better meet users' expectations and improve their overall gaming experience.

II. Previous Work

Similarity between users or items is typically calculated using metrics like cosine similarity or Pearson correlation coefficient. Matrix factorization, a popular collaborative filtering technique, reduces dimensionality by identifying hidden factors that explain user preferences. Although effective in providing new recommendations, collaborative filtering faces challenges like the cold start problem (difficulty recommending for new users or items) and data sparsity, which can lower prediction accuracy [1].

Evaluation metrics are essential in assessing the effectiveness and accuracy of a recommendation system, as they provide concrete measures of how well the system aligns with user preferences. Metrics such as recall and precision are crucial, as they help determine not only how well the system retrieves relevant games (recall) but also the accuracy of these recommendations (precision). Top-N accuracy (e.g., Top-5 recommendations) gives insights into how well the model ranks games that users are likely to enjoy, allowing for a more user-centered assessment by focusing on the most relevant recommendations.By using these metrics, a recommendation system can be fine-tuned to deliver more relevant, personalized suggestions, ultimately enhancing user satisfaction and reducing the likelihood of recommending mismatched games. [2]

In terms of pre-processing data, data collection may also include user-generated review sentiment, as sentiment scores reveal user satisfaction, especially when combined with engagement metrics. It was highly recommended to clean the dataset by filtering out users with minimal activity and inactive profiles, as including only active users enhances the model's focus on genuinely engaged player profiles. [3]

III. Methodologies

A. Data Collection and Pre-processing

For this research, a synthetic dataset of game tags and user preferences was used. The dataset was sourced from the Steam API that grabbed users with 5 games, filter the users game list to their top five games based on playtime (Algorithm 1). From this, a comprehensive list of unique tags was created, allowing each game to be represented as a binary vector (Algorithm 2). A matrix was constructed where each row corresponded to a game and each column to a tag, with binary values indicating the presence or absence of a tag for a specific game. This "tag frequency matrix" became the foundation for clustering and recommendations.

B. Dimensionality Reduction with t-SNE

To visualize relationships and patterns among the games, t-Distributed Stochastic Neighbor Embedding (t-SNE) was employed. This technique reduced the high-dimensional tag frequency matrix to a two-dimensional space, retaining the local structure of the data. The t-SNE algorithm was configured with:

- `n_components: 3`, to allow visualization in a 3D scatter plot),
- `Review Sentiment:`a positive review is a strong indicator of satisfaction, while a negative review can reflect dissatisfaction.
- `Random state: 42`

The t-SNE results revealed clusters of games with similar tag distributions, making it easier to visually observe natural groupings.

C. Clustering with k-Means

Following dimensionality reduction, k-means clustering was applied to group games into similar clusters based on their tag distributions. Using ten clusters, the algorithm partitioned the tag frequency matrix by minimizing intra-cluster variance. The clustering utilized the same random seed as t-SNE for consistency and to ensure deterministic results. Each game was assigned a cluster label, which was subsequently used to guide the recommendation process.

IV. Results

A. Visualization

The results of t-SNE and k-means clustering were visualized in a scatter plot (Fig. 1). Each point on the graph represents a game, with its position determined by the t-SNE-reduced dimensions. Colors were used to distinguish between the clusters identified by k-means. To provide clarity, each point

Algorithm 1: Breadth-First Search (BFS) for User and Game Recommendations

```
Set max_users ← 500;
while queue is not empty and len(uniqueUser) < max_users do
    Pop (current_user, current_username, current_depth) from queue;
    if current_depth > 10 then
        continue;
    end
    friends ← get_friends_list(current_user);
    foreach friend in friends do
        friend_id ← friend["steamid"];
        friend_username ← friend.get("personaname", friend_id);
        if friend_id not in uniqueUser and friend["communityvisibilitystate"] == 3 then
            games ← get_friends_games_with_appinfo(friend_id
            if len(games) ≥ 5 then
                Add friend_id to uniqueUser;
                Append (friend_id, friend_username, current_depth + 1) to queue;
                Append (current_username, friend_username) to edges;
                top_games ← get_top_n_most_played(games);
                user_map[friend_id] ← [ {"app_id": game["appid"], "name": game["name"], "playtime": game["playtime_forever"]} for game in top_games ];
                if len(uniqueUser) ≥ max_users then
                    break;
                end
            end
        end
    end
end
```

Algorithm 2: Fetch Game Tags Using App ID

Input: `app_id`: Application ID of the game
Output: `genres`: List of game genres (tags) or an empty list if unsuccessful

Convert `app_id` to a string;
Construct `url` using `https://store.steampowered.com/.../{app_id}`
Send a GET request to `url` and store the response in `response`;
Parse `response` as JSON into `data`;
if *`data[app_id]["success"]` is `True`* **then**
    Extract `genres` from `data[app_id]["data"]["genres"]`;
    return List of genre descriptions from `genres`;
end
else
    Print "`No data found for app_id`";
    return an empty list;
end

was annotated with the name of the game, and a legend was included to denote the cluster affiliations.

The scatter plot offered an intuitive understanding of how games with similar tags were grouped together. For example, games with overlapping or similar tag compositions were located closer to each other within the same cluster, while distinct games formed separate clusters.

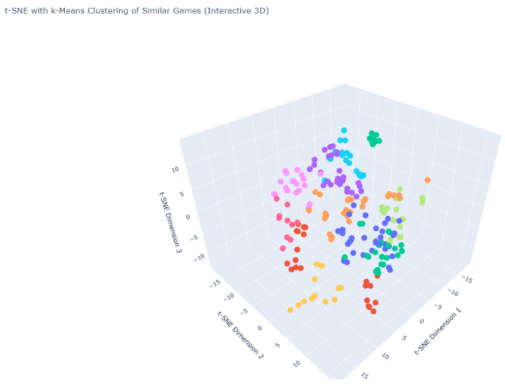

Fig. 1. t-SNE visualization of game tag distributions with k-means clustering. Each color represents a unique cluster, and annotations identify individual games.

B. Recommendation System Design

The recommendation system leveraged the clustering and tag information to provide personalized game recommendations. The following process was adopted:

- User Profile Construction: Each user's profile was built by identifying their most-played games. Tags associated with these games were aggregated to form a representation of their preferences (Fig. 2)
- Cluster Identification: Using the k-means cluster labels, the system identified clusters that contained the user's top-played games. This allowed the system to focus on games that were similar to the user's preferences.
- Candidate Game Selection: Games from identified clusters were shortlisted as recommendation candidates. This step ensured the recommendations aligned with the user's preferred styles and themes.
- Similarity Scoring: The tags of the user's preferred games were used to construct a binary vector, representing their interests. Cosine similarity was computed between this vector and the tag vectors of the candidate games to measure relevance.
- Top-N Recommendations: Based on similarity scores, the top-N games with the highest scores were recommended to the user. This approach prioritized games that shared the most similar characteristics with the user's preferences.

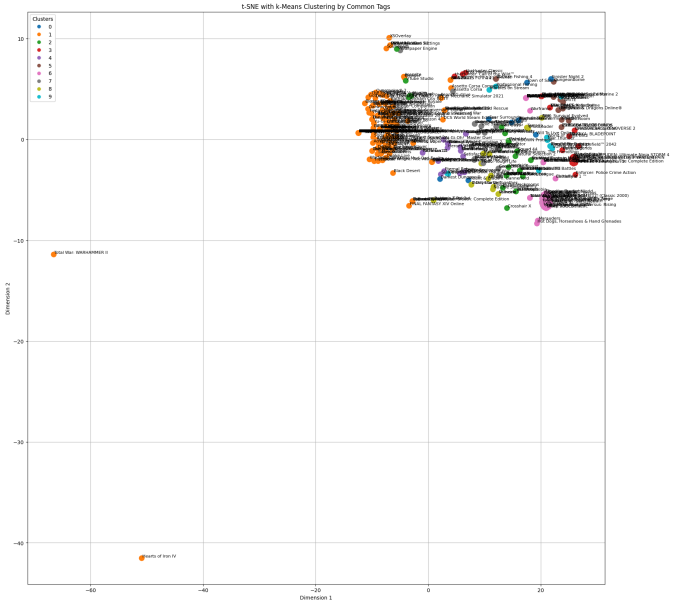

Fig. 2. Clustering of games based on their shared tags using t-SNE for dimensionality reduction and K-Means for clustering.

C. Evaluation

The recommendation system's effectiveness was tested on a randomly selected set of users. For each test user:

- Top-played games were identified.
- Recommendations were generated based on similarity and cluster affiliation.
- Candidate recommendations were manually inspected to validate relevance and alignment with user preferences.
- The system performed robustly, generating recommendations that matched user interests and uncovered new games within the same clusters.

This methodology, combining t-SNE, k-means clustering, and cosine similarity, effectively aligned user preferences with

the available dataset. Visualizing the results further supported understanding and interpretation, while the clustering approach ensured scalable and targeted recommendations.

Here are our results for sample users who needed recommended games.

TABLE I
RECOMMENDED GAMES

User ID	Most Played Games	Recommended Games
76561198195991698	Counter-Strike 2, Rainbow Six Siege, Destiny 2, GTA V, Rust	tModLoader, ARK: Survival Evolved, Terraria, Palworld
76561198377715433	Apex Legends, Brawlhalla, ELDEN RING, Terraria, Phasmophobia	tModLoader, Valheim, Palworld, Lethal Company, Terraria
76561198253696914	Assetto Corsa, Euro Truck Sim 2, BattleBit Remastered, Warframe, PAYDAY 2	Robocraft, POLYGON, Phantasy Star Online 2, Rocket League, SoulWorker

V. DISCUSSION

A. New Insights

1) User Engagement Data Provide Valuable Information: We discovered players often have a wide range of preferences that exist across multiple clusters by looking at the most-played games. For example, a user who frequently plays Counter-Strike 2 (competitive FPS) might also often play Rust (survival sandbox). This insight shows that user engagement data such as playtime would provide valuable information about cross-genre preferences that tag-based methods might overlook.

2) Social Relationships in Steam would Improve Recommendation: Analyzing social connections through the Steam friends list, beyond just individual game preferences, provided an additional insight. Players who are friends often have similar gaming habits or follow and copy their games, which suggests that integrating social relationships into the recommendation process could improve results.

3) Unexpected Relationships Between Games: We found that games with similar tags often formed clusters. However, a surprising insight was that some games from seemingly unrelated genres (like strategy and sandbox) occasionally gathered together. This highlighted how overlapping tags (such as "multiplayer" or "co-op") or common player involvement patterns can lead to surprising linkages between games. This suggests that the future model could go beyond surface-level tags and explore deeper contextual similarities.

4) Visualization Helps Interpretation: Visualizing game clusters using t-SNE and k-means was valuable in interpreting the results. The scatter plots made it easier to observe patterns, such as which genres were overrepresented in the data and where outliers appeared. It would have been challenging to get these intuitive insights using only numerical analysis, without these visual methods.

B. Limitations

While the current system demonstrates promising results, there is significant room for improvement in how games are grouped and recommended. A primary limitation lies in the reliance on tags, which, although useful, often provide only surface-level details about a game. Tags are not standardized, lacking a defined minimum or maximum limit, which can result in inconsistent or overly broad categorizations. For instance, games as vastly different as Among Us (a social deduction game) and Call of Duty (a first-person shooter) might both be tagged with "strategy," yet their gameplay mechanics, target audiences, and thematic premises are entirely distinct.

C. Future Directions

1) Semantic Analysis for Context-Aware Game Clustering: To overcome these limitations, future iterations of the system will explore semantic analysis to better understand the deeper context and themes of each game. Using advanced AI techniques like chatbots and natural language processing (NLP), the system could analyze game summaries, descriptions, and even reviews to identify nuanced similarities. This would allow the system to group games based on more meaningful criteria, such as gameplay mechanics, narrative style, or emotional tone. For example, instead of simply tagging a game with "strategy," semantic analysis might identify whether it is a cooperative, turn-based, or real-time strategy game, providing a richer and more accurate basis for clustering. By integrating semantic analysis, the system could also separate games with similar tags but fundamentally different premises. For example, Among Us could be grouped with other social deduction or party games like Goose Goose Duck, while Call of Duty could be categorized alongside other competitive first-person shooters such as Battlefield or Apex Legends. This enhanced clustering would enable more precise recommendations that truly align with the unique characteristics of each game.

2) Personalized Recommendations Using Collaborative Filtering: Additionally, the system could benefit from incorporating user-based collaborative filtering to complement game-based clustering. This approach involves analyzing the gaming preferences of individual users and identifying those with similar interests based on their top-played games. If two users share a high degree of overlap in their top five or top ten games, it is likely they have similar tastes. The system could then recommend games that one user has played but the other has not, leveraging their shared preferences to expand the recommendation pool. For example, if User A and User B both frequently play Dota 2, League of Legends, and Teamfight Tactics, and User A has also played Hearthstone, the system could recommend Hearthstone to User B. This method allows for the discovery of new games that align with a user's existing preferences while taking into account the collective experiences of others with similar tastes. The collaborative filtering approach also addresses the diversity of player interests within a cluster. Even if two players are grouped into the same cluster based on game tags or semantic analysis, their individual preferences might still differ. By analyzing their specific gaming habits, the system can tailor recommendations to their unique profiles, increasing relevance and satisfaction.

3) A Hybrid Model: Combining Semantics and User Preferences: The future system could combine semantic analysis and user-based collaborative filtering to create a hybrid recommendation model. Games could be initially grouped using semantic clustering, ensuring that games with similar premises and gameplay are categorized together. Within these clusters, user data could be used to refine recommendations

further, identifying preferences that tags or semantic analysis might overlook. For instance, while a game like Civilization VI might semantically group with other turn-based strategy games, collaborative filtering could reveal that users who play Civilization VI also frequently enjoy city-building games like Cities: Skylines.

By integrating these advancements, the system will not only address the limitations of tag-based clustering but also enhance personalization and discovery. These improvements aim to make the recommendations more accurate, contextually relevant, and adaptable to the diverse interests of individual players. Ultimately, the goal is to ensure that players receive recommendations that align closely with their gaming preferences while also introducing them to new and exciting titles they might not have discovered otherwise.

Author Contribution Statement

Special thanks to Junhwi Jeong for providing references to previous work, contributing to the introduction, and writing the discussion section. Elizabeth Jiang for contributing to the abstract and further references, working on the results section, and refining the overall manuscript. Finally, Jack Liang for his input on the methodologies used in this research and also contribution to the discussion.

References

[1] Plunkett, Barry, et al. The Steam Engine: A Recommendation System for Steam Users. brandonlin.com/steam.pdf. Accessed 11 Nov. 2024.

[2] Improving Performance for the Steam Recommender System Using Achievement Data. arno.uvt.nl/show.cgi?fid=144995. Accessed 9 Nov. 2024.

[3] Robert, et al. "Number 1 Spring 2024 Article 4 Part of the Data Science Commons Recommended Citation Recommended Citation Blue." SMU Data Science Review SMU Data Science Review, vol. 8, no. 1, scholar.smu.edu/cgi/viewcontent.cgi?article=1272&context=datasciencereview. Accessed 6 Nov. 2024.

[4] IBM. "Content-Based Filtering." Ibm.com, 18 Mar. 2024, www.ibm.com/topics/content-based-filtering. Accessed 12 Nov. 2024.