RecommenderPlus: New Content-based User-centered Game Recommendation System

Tianrui Liu
The Grainger College of Engineering
University of Illinois at Urbana-Champaign
Champaign, USA
tl49@illinois.edu

***Abstract*—Many techniques have been incorporated into game recommendation systems, and they showed excellent accuracy in the recommendation. However, most of them are not user-centered, and users could only provide the game names to get their recommended list without any interaction. Therefore, we firstly conducted user research to evaluate what kind of factors will affect the games recommendation system. By analyzing the result of user research data, we concluded the influential factors our users genuinely care about. Then we designed a prototype/UI and deployed an application called RecommenderPlus, a user-centered steam games recommendation system based on the requirements of our users. Also, we introduced its design process and the challenges we met. After that, we analyzed and evaluated our model's performance and discussed its advantages and limits.**

Keywords*—Human-computer interaction, Recommender system, Information retrieval, Text mining

I. Introduction

Nowadays, fantastic new games are popping up, and people have no idea which one to pick since getting familiar with new games is quite time-consuming. Thus, many game recommendation systems or applications try to help them filter their dream games. "Video Game Recommendation Engine" is an online recommendation website. [1] It could accurately predict users' potential favorite games, but the user could type a tree of games' names [1]. So they cannot find their ideal games according to other factors, such as game description, positive review rate, playtime, and studio.

Another example is called "Game Finder". [2] It allows users to enter the game they like and outputs the top ten games from high scores to low scores. Nevertheless, users cannot input multiple games or other factors to get recommended games [2]. There is no interaction with this system, either.

Thus, the first problem we need to solve is figuring out what kinds of factors users mostly care about in the games recommendation system [3]. We launch our first research question, which is "What kind of factors will influence games recommendation system based on user's experiences and interactions?". In order to evaluate users' ideas, we design a questionnaire that contains twelve potential factors that may influence the recommendation system. After collecting the evaluation form, we analyze the data and select the essential factors that users care about [4].

Then, we implemented a new application called RecommenderPlus. This user-centric steam games recommendation system includes these factors and designs a new UI for better user interaction. We will discuss this application from three fields: data collecting, front-end design, and back-end design.

At last, we analyze the performance of our model and analyze its current advantages and limits. In addition, we mention the challenge we met and future work on the games recommendation system.

This paper makes three primary contributions:

(1) User study: We designed the user questionnaire and collected data [5]. Then we analyzed the collected forms and filtered the critical factors that affect the games recommendation system from the user's aspect.

(2) Deployment: We designed a new recommendation system based on steam games and implemented it. We collected game information from the steam database and designed the interactive user interface for a better experience. Then, we implemented the natural language processing [6] algorithm "TF-IDF" in the back-end part.

(3) Analyzing model performance: After testing our new application by design test dataset, we analyzed its performance and discussed the advantages and drawbacks of recommended games and user interactions.

The following research questions were asked before the experiment and responded to our questions and hypotheses in the results.

- **RQ1:** What factors will influence the games recommendation system based on users' experiences and interactions?
- **RQ2:** What is the best method for content-based games recommendation? Moreover, why do we choose this "TF-IDF" game-similarity algorithm?
- **RQ3:** What are the advantages and disadvantages of our algorithm?
- **RQ4:** How did our "User-centered games recommendation system" model perform based on its experiment results and user interactions?

[1] https://apps.quanticfoundry.com/recommendations/gamerprofile/video game

[2] https://gameslikefinder.com/

II. Related Work

In the application of online game platforms, the recommendation of online games is an essential function. A robust recommendation algorithm can enhance the user experience and allow users to find their favorite games and game categories quickly [7].

Collaborative filtering [8] is one of the more reliable recommendation algorithms. It can filter content that is difficult for machines to analyze automatically and can share the experience of others. Incomplete or imprecise content analysis is avoided. It can also filter based on some complex and challenging formulated concepts. However, it is difficult to find out the similarity relationship between products. So how quantifying the similarity between products is always a widespread issue.

Therefore, we chose the same more reliable algorithm model TF-IDF [9] algorithm. TF-IDF often plays the role of indicator quantification in recommendation algorithms. It has been widely used in recent years in recommendation problems. Furthermore, the famous item penalty coefficient is introduced by optimizing and improving the calculation of Pearson's correlation coefficient. It reduces the influence on the weight of coefficients. The recommendation effect of the coefficients is improved.

Furthermore, the application in the game domain has some related work at the current time. A weighting method based on semantic TF-IDF is proposed in the paper of Mir Saman et al [10]. This vector is used to redefine the semantic weights and thus the similarity of tweets. Moreover, experiments are conducted on Twitter. A similar method for calculating similarity using game descriptions for analysis is also included in our data. Here we mention in particular a game tagging recommendation system called AURYGA [11]. This is an enhanced game tagging system available. It uses new analytic techniques in order to help users define tags. It uses a well-founded stochastic model. It also uses text descriptions of PC games stored on the Steam platform to predict GAme tags. The system also uses the TF-IDF encoder to provide a base model to retrieve the list of tags.

For this reason, we have asked some questions for investigation, particularly in Fig. 1. There are questions about users' opinions on how to search or discover the content of the recommended games they are interested in. We found that out of 683 valid data, more than 58% of users consider "Games played by the user (names and playtime)" to be the most important, but the current game recommendation system cannot cover their needs.

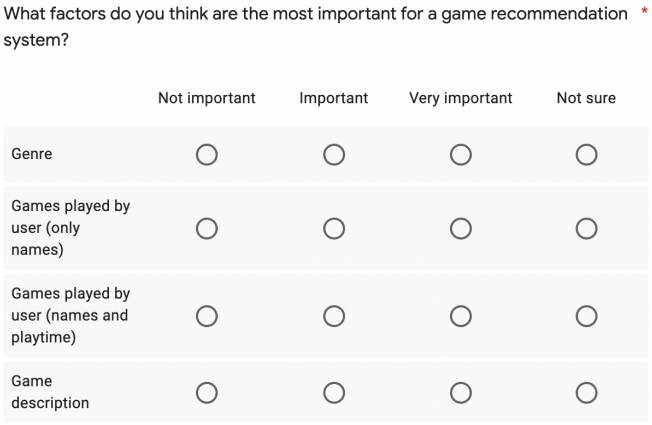


Fig. 1. The four questions we are most concerned about in the investigation and evaluation.

III. Data Collection

After much player data is collected, Player data needs to be cleaned and processed first. Thus, the validity and accuracy of the prediction process can be improved. The following steps of pre-processing work [12] are needed for the collected player game history data.

- Data cleaning. The main work of this step is to remove irrelevant and duplicate data from the original data. In this step, the data irrelevant to the prediction theme need to be filtered out through in-depth analysis. Among the many game history data, our model will focus on the data of the following five categories of attributes. They are denoted as the number of total game reviews, the number of positive game reviews, the number of lousy game reviews, the game description, and the game name. For the data of these five attributes, it is necessary to judge whether the outliers are deleted or not according to the specific situation.
- Data transformation. This process requires the normalization of the data. Some attributes of each game data are transformed into the following appropriate form. For example, all the data is calculated based on the value of the "number of positive reviews" attribute of the game and the "total number of ratings". The game's specific rating value is obtained. The result is calculated to two decimal places. In the same way, several other categories of attributes can be processed to serve the subsequent implementation of the classification algorithm.
- Our original data comes from the open-source database "Steam games complete dataset" of Kaggle. The total data contains 60,000 data with different game names. And after our formatting and cleaning. 51933 pieces of valid data are kept for algorithm development and operation [3].
- The current data is flawed. The last update of the dataset was three years ago. So our data is flawed in that some games cannot be updated. Because Steam can only show the popularity of games, for example, very positive, optimistic, most damaging, negative, etc. Furthermore, only two decimal places are kept because of the single calculation method. These factors can affect the accuracy and precision of the calculation. We will present these remaining flaws for effective improvement in the future.

[3]https://www.kaggle.com/datasets/trolukovich/steam-games-complete-dataset

IV. Methodology

With the data ready, the next step is to build the functional backend system. First, we extract the dataset as a .csv file and convert it into a data frame. Next, we use BeautifulSoup to scrape each game's description from its URLs. We process the description only to contain raw text without additional symbols. Now we add 'Game description' as an additional column in the data frame (Fig. 2).

	Positive	Negative	Game_name	Positive_Rate	Link	Game_description
0	39146	3404	DOOM	0.92	https://store.steampowered.com/app/379720/DOOM/	Developed by id software the studio that pio...
1	409937	426671	PLAYERUNKNOWNS BATTLEGROUNDS	0.49	https://store.steampowered.com/app/578080/PLAY...	LAND LOOT SURVIVE Play PUBG BATTLEGROUNDS ...
2	4991	2039	BATTLETECH	0.71	https://store.steampowered.com/app/637090/BATT...	From original BATTLETECHMechWarrior creat...
3	101940	65175	DayZ	0.61	https://store.steampowered.com/app/221100/DayZ/	DayZ is a hardcore openworld survival game w...
4	8495	2986	EVE Online	0.74	https://store.steampowered.com/app/8500/EVE_On...	THE 1 SPACE MMO EVE Online is the largest...

Fig. 2. Dataframe with Game Description

The main idea of the next task is to measure the similarity of two pieces of text. We do this by constructing topic models using TF-IDF weighting. For each document (in this case, game description), we give a weight to each word in this document. If a word occurs more frequently, it is given a relatively higher weight. Also, common words with little meaning, such as 'a', 'the' or 'to', are assigned with low weights (Fig. 3).

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Fig. 3. Algorithm for TF-IDF Weighting

Then we process the user input data. The input would be a list of game names with their respective hours played by the user. We need to assign a weight to each game that corresponds to the amount of time the user has spent on it. We used a naive approach: dividing the playtime into different intervals and assigning a proper weight to each interval. Specifically, if a game has been played for less than 2 hours, then its weight is 0.1; if a game has been played for within 2 to 10 hours, then its weight is 0.5; if a game has been played for within 10 to 50 hours, then its weight is 0.8; if a game has been played for within 50 to 100 hours, then its weight is 1; and lastly, if a game has been played for over 100 hours, then its weight is 1.2.

Then we can use the topic models to compute the similarity of each of the games that the users have played with all the games in our dataset. Moreover, we can find the top 10 most similar games to each of the games in the input list. Furthermore, each of these ten games will have a weight too. For example, if game A is in the input, we find the top 10 most similar games to game A: game A1 to game A10, where A1 is the least similar to A and A10. Then A1 will weigh 1, A2 will weigh 0.99, A3 will weigh 0.98, etc. In the end, A10 will weigh 0.91. Multiplying this weight with the original weight of game A in the input list, we can now have a final score for recommended games A1 to A10. Furthermore, after finding the ten most similar games for each game in the input, we rank the results by their final score and keep the top 10 results overall.

Next, we use React to build the user interface and Flask to connect the frontend with our functional backend system. Users can add their favorite games, where a table will show the games they added. Then users can click the button to generate the recommended games that our backend system generates.

V. Results

As for RQ1, our model will focus on the data of the following five categories of attributes among the many game history data. They are denoted as the number of total game reviews, the number of positive game reviews, the number of lousy game reviews, the game description, and the game name.

As for RQ2, our "TF-IDF" is a more reliable algorithm than the past games recommendation algorithms. TF-IDF often plays the role of indicator quantification in recommendation algorithms. Furthermore, the penalty coefficient is introduced by optimizing and improving the calculation of Pearson's correlation coefficient. In addition, it reduces the influence on the weight of coefficients.

As for RQ3, the advantages are that our algorithms emphasize the user's experience, and it introduces the penalty coefficient by optimizing and improving the calculation of Pearson's correlation coefficient. Also, the recommendation effect of the coefficients is improved. As for the disadvantages, we just utilized a simple formula to convert playtime into coefficient. Also, we currently use only positive and negative user reviews as rating criteria.

This is the page where the application starts in Fig. 4. Users can add their favorite games and playtime. Before adding games, the user must click the setup data button, which will load the back-end dataset for the recommendation.

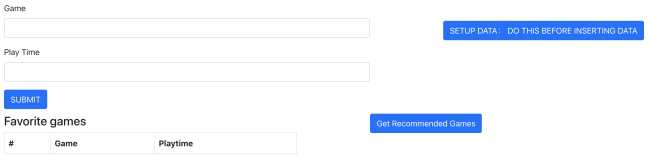
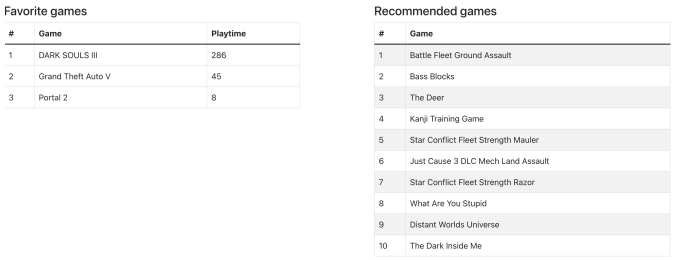


Fig. 4. Application Start Page

This is the page in Fig. 5 where our application gives users their recommended games based on their entered games. Furthermore, here is a showcase of the test results. The figure shows that we imported three different games (DARK SOULS III - 286 hours, Grand Theft Auto V - 45 hours, Portal 2 - 8

hours). The top 10 most relevant recommended game names are listed according to the results returned by our algorithm.



Favorite games

#	Game	Playtime
1	DARK SOULS III	286
2	Grand Theft Auto V	45
3	Portal 2	8

Recommended games

#	Game
1	Battle Fleet Ground Assault
2	Bass Blocks
3	The Deer
4	Kanji Training Game
5	Star Conflict Fleet Strength Mauler
6	Just Cause 3 DLC Mech Land Assault
7	Star Conflict Fleet Strength Razor
8	What Are You Stupid
9	Distant Worlds Universe
10	The Dark Inside Me

Fig. 5. Recommendation Page

As for RQ4, the time complexity of our model is O(MN), M is the number of input data, and N is the number of data from the dataset. We invited a couple of our friends to test the user experience of our model. They thought it was a pretty concise and efficient games recommendation system. Nevertheless, they may want more factors of filters. Moreover, the extension to modify the recommendation list and add some games to their favorites was mentioned. We conducted 100 small-scale tests on our models, and the correct rate was about 88%-94%. As for the model's accuracy/running performance, larger-scale tests should be implemented for our model [4].

VI. Conclusion and Discussion

Evaluation of such a recommendation system is no easy task, as the accuracy of the results is essentially a subjective matter. Therefore there is no quantitative method for evaluation. We plan to conduct a large-scale user study to get as much feedback on our system's accuracy. Nevertheless, as of now, we still could not find enough participants. Thus, the user study likely suffers from potential bias due to insufficient sample size.

Improvements regarding the data section are not limited to updating data but also include the accuracy and precision of the data format. For example, we currently use only positive and negative user reviews as rating criteria; however, natural language processing techniques can be used to identify the reasonableness of user ratings for a series of filters. We are removing irrelevant language descriptions or inaccurate ratings. This will significantly improve the accuracy of the data and thus lead to the more effective development of our algorithm.

In addition, we only used a naive approach to convert playtime into coefficient by dividing the playtime into several intervals and assigning a reasonable weight to each interval. To improve the accuracy, we can use a more accurate function to map better the relationship between how much the user likes a game and how long the user plays this particular game. The initial idea is to take the logarithm of the playtime.

More functionalities can be added to the front-end, such as deleting games from the table. Also, the user interface could be more beautified.

[4]https://github.com/liutiantian233/RecommenderPlus

We also plan to implement several additional features further to enhance the user experience. For example, we planned to add a filter to let users filter out certain games/genres/studios from the recommendation list. Another idea is to use Machine Learning to build a user profile from the information provided, such as games played and how long each game has been played by the user, to give the user a more precise understanding of what type of user he/she is. These features are all centered around the central idea that we want to focus on the users and make each user's experience unique.

References

[1] A. Toskova and G. Penchev, "Intelligent game recommendation system," *AIP Conference Proceedings*, vol. 2333, no. 1, p. 050007, 2021. [Online]. Available: https://aip.scitation.org/doi/abs/10.1063/5.0042063

[2] L. De Simone, D. Gadia, D. Maggiorini, and L. A. Ripamonti, "Design of a recommender system for video games based on in-game player profiling and activities," in *CHItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter*, ser. CHItaly '21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3464385.3464742

[3] Y. Blanco-Fernández, M. López-Nores, A. Gil-Solla, M. Ramos-Cabrer, and J. J. Pazos-Arias, "Exploring synergies between content-based filtering and spreading activation techniques in knowledge-based recommender systems," *Information Sciences*, vol. 181, no. 21, pp. 4823–4846, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025511003008

[4] A. C. Rivera, M. Tapia-Leon, and S. Lujan-Mora, "Recommendation systems in education: A systematic mapping study," in *Proceedings of the International Conference on Information Technology & Systems (ICITS 2018)*, Á. Rocha and T. Guarda, Eds. Cham: Springer International Publishing, 2018, pp. 937–947.

[5] S. Dwivedi and V. S. K. Roshni, "Recommender system for big data in education," in *2017 5th National Conference on E-Learning E-Learning Technologies (ELELTECH)*, 2017, pp. 1–4.

[6] J. K. Tarus, Z. Niu, and D. Kalui, "A hybrid recommender system for e-learning based on context awareness and sequential pattern mining," *Soft Computing*, vol. 22, no. 8, pp. 2449–2461, Apr 2018. [Online]. Available: https://doi.org/10.1007/s00500-017-2720-6

[7] J. Kim, J. Wi, S. Jang, and Y. Kim, "Sequential recommendations on board-game platforms," *Symmetry*, vol. 12, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2073-8994/12/2/210

[8] J. Bobadilla, F. Ortega, A. Hernando, and A. GutiéRrez, "Recommender systems survey," *Know.-Based Syst.*, vol. 46, no. 24, p. 109–132, jul 2013. [Online]. Available: https://doi.org/10.1016/j.knosys.2013.03.012

[9] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0306457388900210

[10] M. S. Tajbakhsh and J. Bagherzadeh, "Microblogging hash tag recommendation system based on semantic tf-idf: Twitter use case," in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*. Irfan: IEEE International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), 2016, pp. 252–257.

[11] R. Rubei and C. Di Sipio, "Auryga: A recommender system for game tagging," in *The 11th Italian Information Retrieval Workshop*. Italy: The 11th Italian Information Retrieval Workshop, 10 2021.

[12] C. Fan, M. Chen, X. Wang, J. Wang, and B. Huang, "A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data," *Frontiers in Energy Research*, vol. 9, 2021. [Online]. Available: https://www.frontiersin.org/article/10.3389/fenrg.2021.652801