CSCE 580 Project Report – Video Game Recommendation Chatbot

Author: Matthew Hughes

Problem

In 2022, earnings from the Video Game Industry were estimated to be in the ballpark of $347 billion(Clement, 2023). Furthermore, in 2023 it is estimated that there are over 3 billion people that play video games worldwide(Howarth, 2023). With a world population that is close to 8 billion, this means that approximately 37.5% of the earth currently participates in video games. This percentage of the earth's population has increased by approximately 10% over the last 8 years, illustrating the growth that is taking place in this market. The high valuation of the video game market, large percentage of population that plays video games, and growth of the industry over the last 8 years all show the potential that this industry has.

To ensure that the high potential of the video game industry remains, it is important to ensure that consumer interest continues to increase in the gaming industry. One of the possible ways to accomplish this is to make sure that consumers are being matched up with games that they are interested in and will enjoy. This is where the task of video game recommendation comes in. When a good video game is recommended to a consumer, that consumer is more likely to buy the video game, play the video game, and spend further money on the video game.

The task of recommending a video game to a user has been accomplished in many ways in the past, some of which are discussed below under related work. In this project, the goal was to create a recommender system that allows the user to describe some arbitrary game that they would want to play, likely something that does not exist. Once the user has created the game description, the model would return a game like the provided description. To allow for better communication between the model and user, the created recommendation model will be nested within a chatbot. This is a project that if successful would have a profound impact in multiple domains. First, this will help current video game players that are looking to expand their libraries and get into new games. Second, the chatbot would provide video game vendors with a new form of search on their websites, making those sites more accessible to the general population of consumers.

Related Work

Past work has been completed both on video game recommendation and recommendation based upon natural language processing(NLP). First, previous methods to recommend video games will be discussed.

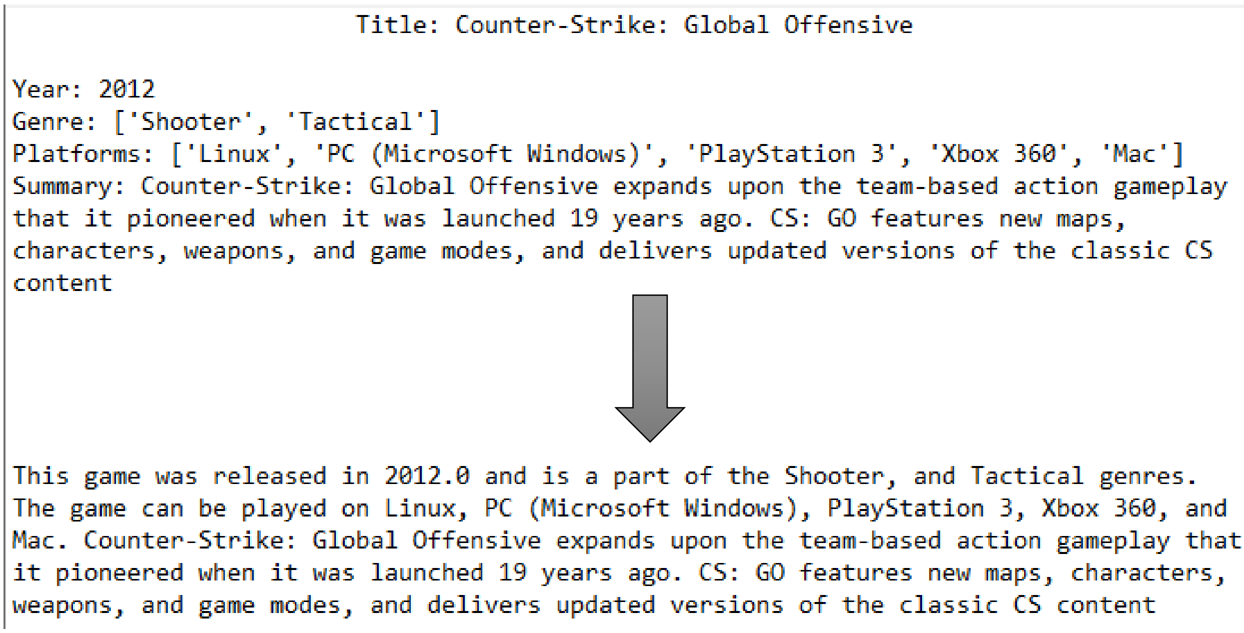
The development and implementation of video game recommendation systems is something that has increased in recent years, following the trend of increased video game popularity mentioned above. The following categorization of video game recommenders was discussed in the paper by Yang et al.(2022). Video game recommendation systems can be categorized into 3 main groups: collaborative filtering(CF), content-based, and hybrid methods. CF methods involve the recommendation of games based on the interactions between users and games. Models will analyze these interactions and assume that users with similar actions will have preferences that align with each other. Therefore, if there are two users who have similar actions and one of those users enjoys a particular game that the other does not have, then the CF method is likely to recommend this game to the other user. A second method is called content-based recommendation. This method looks at the content contained on a user's account along with descriptions of games that could be recommended. The model then tries to connect which games the user will like based on these descriptions. One example of this would be if a user owns one game in a series, then the model would possibly recommend them a new game that is a part of that series. The final category is hybrid methods. As the name implies, hybrid methods combine both CF and content-based methods into one method. The combination allows a model to look both at user interactions and the content of users and the games. The resulting model is one that is both flexible and adaptable when involving video game recommendation (Yang et al., 2022).

The second field of related work being looked at is how NLP is used for recommendation. Berbatova conducted an analysis on some NLP techniques and how they were used for book recommendation(2019). The methods discussed in the paper are cosine similarity – the method used for the model in this project - and k-nearest neighbors(KNN). Both of the cosine similarity based methods used are similar to the hybrid method mentioned above, while the KNN method falls more in line with the content based category. The first cosine similarity method recommends books based strictly upon users' ratings of books, comparing both the content of the reviews and the interactions between users and books. The second method uses more data, looking at not only the reviews but also general descriptions of the books and performing similar comparisons to the last method. Finally, the KNN method uses a feature set crafted from "lexical, character-based, syntactic, characterization, and style" analysis of the book texts. These feature sets are categorical, so to use them with KNN the dataset is made numerical by one-hot encoding. When comparing the methods from this paper to the model in this project, the closest comparison with the second cosine similarity method. However, there is still a large difference in the fact that this paper compares real game descriptions to descriptions of a game that does not exist written by the user.

Approach

The data necessary to accomplish this project was retrieved from the website https://www.igdb.com/. The data was retrieved using their API and is made of the top 6,912 based upon rating on their site, containing over 250,000 games. The features of the dataset were the name, the release year, genres, platforms, and a summary. For evaluation purposes, the genres, release year, and platforms will be considered the metadata for a game. This data was then all comprised into 1 description that could be compared to whatever the user inputted. An example of this conversion is illustrated in figure 1 where the metadata and summary for the game Counter-Strike: Global Offensive is converted into one full description. Now that the dataset has been created, the structure of the model, also know as Game Recommendation

Automation Machine(GRAM), is the next important step. The model is constructed in two parts, a content-based recommendation method based upon what Paialunga does in his article and a chatbot that is created using Rasa.


Title: Counter-Strike: Global Offensive

Year: 2012
Genre: ['Shooter', 'Tactical']
Platforms: ['Linux', 'PC (Microsoft Windows)', 'PlayStation 3', 'Xbox 360', 'Mac']
Summary: Counter-Strike: Global Offensive expands upon the team-based action gameplay that it pioneered when it was launched 19 years ago. CS: GO features new maps, characters, weapons, and game modes, and delivers updated versions of the classic CS content

This game was released in 2012.0 and is a part of the Shooter, and Tactical genres. The game can be played on Linux, PC (Microsoft Windows), PlayStation 3, Xbox 360, and Mac. Counter-Strike: Global Offensive expands upon the team-based action gameplay that it pioneered when it was launched 19 years ago. CS: GO features new maps, characters, weapons, and game modes, and delivers updated versions of the classic CS content

Figure 1: Conversion from metadata and summary to full description.

As previously mentioned, the recommendation method used in the paper is strongly based upon Paialunga's article. Before explaining the main difference, here is an overview of the method prior to modification:

1) The descriptions are contained in a dataset
2) Descriptions are converted to vectors using BERT
3) Vectors are compared using Sci-kit's cosine_similarity() method
4) Then the vector that is most like each datapoint is returned

The main difference for the method used in this article is that prior to step one, an arbitrary video game description created by the user is converted to a vector by BERT, then added to the end of the dataset before the cosine_similarity() method is run. Once the cosine_similarity() method has run, the game that is closest to the user's input and the corresponding description are returned.

Moving on to the methods integration into the rasa chatbot. Outside of a few changes, most of the default rasa files were left pretty much the same. First, a new user intent was added called get_recommendation, which will be the intent when a user is trying to get a video game recommendation. Second, the method described above was added to the actions list, allowing it to be called by the chatbot shell. Finally, the stories were updated to include new paths that include what should happen when the user is looking to get a video game recommendation. When all is put together, you have a working chatbot called GRAM that can converse with a user and provide them game recommendations based upon what they input into the system. A working video of GRAM can be seen at https://youtu.be/vzItiOFIUYE

Evaluation

Multiple methods were compared when looking to evaluate the newly created model: random, GRAM, and ChatGPT. GRAM is the model from the project, random is just a random selection from the 6,912 games, and ChatGPT is a LLM developed by OpenAI. The test set used one these models was a set of 50 fake video game descriptions that did not correspond directly to any real games. These descriptions did not have to contain all the metadata, but always contained a description and started with a phrase implying that they wanted to get a recommendation. Real world test cases may not always be this accommodating, however, have these constraints provided a way to score the models.

3 Points	All metadata and summary matched
2 Points	The summary matched, but some metadata missed
1 Point	Summary was missed, but some metadata was correct
0 Points	Neither the description nor metadata matched

Figure 2

To evaluate the model, one of these descriptions would be passed in – or in the case of random, the method was called – and the output of the model would be recorded. The output of the model was then scored on a 3-point scale displayed in figure 2. This scale is not perfect and has a subjective nature as to whether the description matched. However, because of the nature of this problem not having an objective answer, this was the created scale to come up with

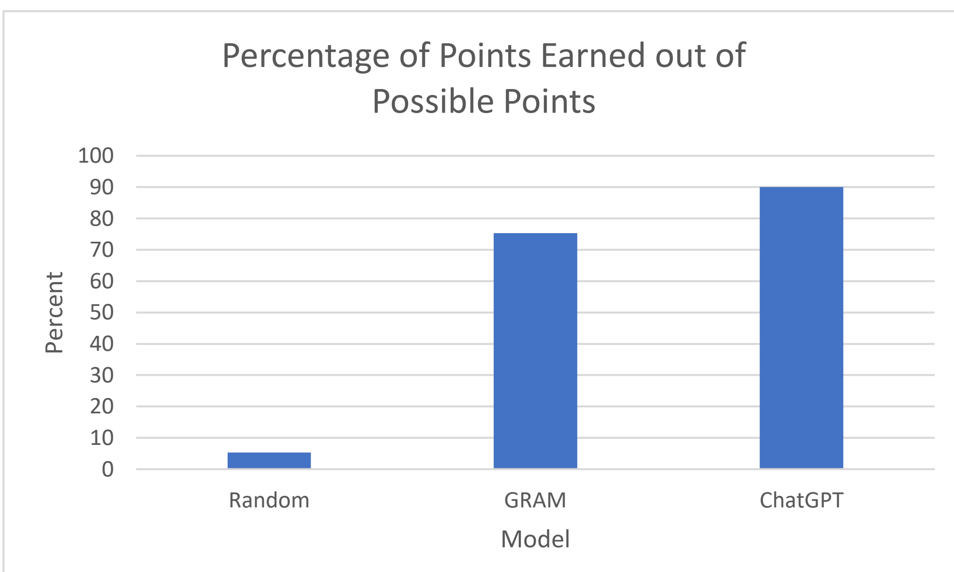


Figure 3

evaluation of the recommendations. The results of the experiment are seen in figure 3. As expected, the random model has a low score of around 5% making it an unreliable prediction source. However, when comparing GRAM and ChatGPT there was only a 15% difference, with ChatGPT scoring 90% and GRAM scoring about 75%. When GRAM missed a recommendation, it was usually because it did not pick up on some of the metadata in the request. ChatGPT can pick these components up better because it is an LLM and can process not only the input as a whole, but also some of the more important parts of the input are. Yet, the place where GRAM was sometimes able to keep up – and even do better than ChatGPT in subjective analysis – was in the summary matching. Since the descriptions are mainly composed of the summaries, and GRAM analyzes the whole summary, GRAM was typically able to find a solution that matched the description well. However, it is important to remember that ChatGPT was still able to outperform GRAM, ensuring that a full users request was processed.

There are places that GRAM could improve which would provide for interesting further experimentation. One of these aspects that GRAM could improve is using more of Rasa's built in features. Rasa has features that allow you to extract certain words from users' inputs based on which intent it gets identified as. This could possibly give GRAM a better idea of what metadata requirements are needed for the game allowing the score to increase. Second, if user data was present, GRAM could use this data in the form of the hybrid models mentioned above. This would give the model more data to go off than just the data description input by the user.

Overall, I am surprised and happy that the model was able to perform as well as it did. Although it did lose to ChatGPT, it is cool to create a model using NLP that isn't too far off the performance of a model that has received such high praise in whatever it does.

*** Demonstration: https://youtu.be/vzItiOFIUYE ***

Works Cited

Berbatova, M. (2019, September). *Overview on NLP techniques for content-based recommender systems for books*. ResearchGate. https://www.researchgate.net/publication/336862963_Overview_on_NLP_Techniques_for_Content-Based_Recommender_Systems_for_Books

Clement, J. (2023, October 6). *Topic: Video game industry*. Statista. https://www.statista.com/topics/868/video-games/#topicOverview

Howarth, J. (2023, August 10). *How many gamers are there? (New 2023 statistics)*. Exploding Topics. https://explodingtopics.com/blog/number-of-gamers

Paialunga, P. (2022, January 16). *Hands-on content based recommender system using Python*. Medium. https://towardsdatascience.com/hands-on-content-based-recommender-system-using-python-1d643bf314e4

Yang, L., Liu, Z., Wang, Y., Wang, C., Fan, Z., & Yu, P. S. (2022, February 11). *Large-scale personalized video game recommendation via social-aware contextualized graph neural network*. arXiv.org. https://arxiv.org/abs/2202.03392