Machine Learning Engineer Nanodegree

Capstone Proposal

Christian Beckmann
December 27th, 2017

Boardgame recommandations using Natural Language Processing and Transfer Learning

Domain Background

According to Statista.com[1] about 70% of the Germans play at least occasionally board, card or strategy games and 44% want to buy at least one game in the next 12 months. There are about 1,500 new board games produced each year for the German market alone[2] which compete for new customers.

The assumption that digital games (computer games, mobile games) are pushing back the use of board games can be refuted with the sales figures of the board games industry of the last 20 years (e.g. sales growth of Ravensburger AG from EUR 282 million in the year 2006 to a total of EUR 474 million in the year 2016[2]). This shows that even with the ongoing digitalization of games this market stays stable and will be relevant in the future.

[1] "Ein Brettspiel kommt selten allein" - Statista.com https://de.statista.com/infografik/11589/anzahl-gesellschaftsspiele-im-haushalt/
[2] "Brettspiele" - Institut für Ludologie https://www.ludologie.de/spiele/brettspiele/

Problem Statement

At the time of writing there are 29,495 games listed on http://www.spiele-check.de in 49,145 different versions and variations, as stated above there are about 1,500 new games released each year. This high amount of games available makes it difficult for customers to find new games to buy and play that match their preferences.

To help the users to find games i will use the information available for all of the games (see section below) and the current user as the input to create a list of recommended games for this user based on similarity of the contents.

Datasets and Inputs

For this project I will use the data of user generated contents of http://www.spiele-check.de which consists of:

- Metainformation of each game:
  - Title
  - Pictures of packaging and contents
  - Description
  - Author
  - Release date

    - Publisher
    - Number of players
    - Time needed for playing
    - Time needed for preparation
    - Awards
    - Rating (all users)
    - Percentage of characteristics
        - Strategy
        - Luck
        - Player interactions / negotiations
        - Knowledge
        - Dexterity

- User information:
    - Games owned
    - Games played (number of times played)
    - Ratings

All this data is available as an export of a MySQL database (thanks to the owner of spiele-check.de) and will be preprocessed and prepared to be used in this project.

Solution Statement

There are two approaches that can be used to solve this problem.

Collaborative filtering based recommendations

In this approach is based on finding clusters of users with similar interests or, in this case, games and use this to make predictions about what games the current user may like. This is done by comparing the current user to the other users which are in the same cluster and compare the games owned or rated by these. The elements with the most positive accordance will than be used to build the list of recommendations for the user.

The advantage is that with this approach it is possible to find new elements to recommend which may not be in the scope by looking at the user or his games alone.

A drawback is that for this approach the data of all users has to be evaluated and according to german privacy laws a user has the right to withdraw the allowance for this. This would lead to additional overhead to implement functionality to exclude such users from the clustering and evaluation of the data.

Content based recommendations

The second approach - and the one i will use in this project - is to use the information available for each of the games and compare it to the games that the current user ownes or has rated to generate recommendations based on similarity.

In this project the similarity will be calculated by using the text based information of the games and by using feature based similarity using the images of the packaging of the games.

The drawback of this approach is that with this solution we will only find games and content which strongly relates to the already known preferences of the user which means that recommending new contents (yet unknown to the user) is not possible.

As this solution is based only on the current user it is much easier to comply to the privacy rules if the user withdraws his allowance to use his data for personalization.

Benchmark Model

The recommender system will be compared to the actual implementation for recommendations which is based on the overall statistics of all games without any personalization. An example can be found at the bottom of each game page, e.g. http://www.spiele-check.de/14476-Robinson_Crusoe_-_Abenteuer_auf_der_verfluchten_Insel.html.

Evaluation Metrics

The evaluation of the solution will be done by doing an A/B test on the website http://www.spiele-check.de and comparing the click through ratio of the new solution with the current statistical algorithm based on the overall statistics of all games.

Project Design

For this project the following software and libraries will be used:

- Python with Keras / Tensorflow backend (https://keras.io)
- scikit learn (http://scikit-learn.org)
- MySQL (https://www.mysql.com/)
- Flask (http://flask.pocoo.org/)
- Piwik (https://piwik.org/)

The solution will consist of the folling parts:

1. Comparing the text based meta information of the games

In this part of the solution the text based metainformation will be analyzed and used to get the most similiar games.

In the first step the data will be preprocessed with the following steps:

- remove stop words
- tokenize the text
- apply stemming algorithm

After this preprocessing we need to vectorize the text information to calculate the cosine similarity for each game to each other game.

The vectorization can be done using Term Frequency / Inverse Document Frequency or with a doc2vec / word2vec approach. As this project is based on german text information I will have to do an evaluation of different approaches / libraries to be sure that it can handle german language properly. The libraries evaluated for this project will be NLTK, gensim and spaCy which all have different pros and cons.

The cosine similarities are later used as one input for the calculation of the final score for each element.

2. Comparing the images of the game packaging

In this part of the project I will use the pre-trained ImageNet models available for the Keras library (https://keras.io/applications/) to extract the features of the images as vectors, which are then used to calculate the similarity of the images using cosine similarity.

It is based on the concept outlined at https://nycdatascience.com/blog/student-works/deep-learning-meets-recommendation-systems/.

To do this we need the following steps:

- extract the image urls for the packaging from the data
- download the pictures
- load and initialize the InceptionResNetV2 model from Keras with parameter include_top=True and weights='imagenet"
- create feature vectors for the images using the model predict function
- calculate cosine similarity for the images.

With the similarity scores from both parts of the algorithm we can now calculate the final similarity scores as a weighted average of these both.

3. Web application

To make the solution available for users the last part of the project is a web application which acts as a HTML frontend for the users.

If a user accesses the HTML page the following steps have to be performed:

- load information of owned games for this user
- load games ratings for this user
- use the scores calculated for the games to find the most similar ones
- show the list of most similar games with basic information for each to the user ordered by score descending

If the user clicks on a game entry we will also need a detail page which will show the information of each game as outlined in the section "Datasets and Inputs" above.

To perform the A/B test for evaluation we also need a function to show a non personalized version of the result page which is based on the overall statistical data of all games (like "best rated") on random basis. To track the results of the test tracking is implemented using Piwik. In Piwik we will then have the data available to compare the CTR of the statistical solution to the solution created in this project.