NLP-Based Recommender System

Project Plan

Aarnav Gupta
55990960

2022-23

Supervisor: Dr Ma Chen

Table of Contents

Motivation and Background Information

All of us interact with the web in a variety of ways. One of the many reasons that the internet is used in today's time is to consume media, purchase goods or items, and find things of our interest. Streaming services such as Netflix use recommender systems to recommend users the next movie or show to watch based on their past activity and ratings. Amazon looks at the items which users previously browsed or purchased and recommends similar items. Travel websites look at the reviews previously written by the user for different restaurants and recommends another one that they might like. Google uses the user's browser history and shows them relevant news articles, based on things they're interested in. For example, someone who looks up about the public transport service status mutiple times is more likely to find latest updates about the same on their home feed. Spotify notices what kind of songs the user enjoys, and recommends them new songs to listen to, based on their personal taste. Spotify gets such feedback by the number of times a user listens to a song, if they add it to their playlist, if they mark it as a 'liked' song, or if they skip it, or delete it from their library, among other ways.

Not every person would like the same songs, or would be looking to purchase the same items. Some would be overwhelmed to decide which movie to watch among the thousands offered on the service. A political article might be interesting for one, but not for the other. And, hence, personalised recommendations become important when the information on the web has become as vast. Recommender systems make this possible by filtering out and suggesting the most relevant item to the user, based on their personal taste. It allows the users to make better decisions regarding what they purchase or what content they choose to consume.

To collect users' preference information, the models can have the feedback in both explicit (like/dislike; rating) form or implicit (watching a film; purchasing an item) form.

Previously, matrix information was used to find a person with 'similar' interests, and recommend items based on that person's interest. So, for example, if Person 'A' loved Item 1, 2, 4, but didn't like Item 3; Person 'B' loved Item 1 and 2, and also did not like Item 3, the older algorithms would rate Person 'B''s interests to be similar to Person 'A''s, and would recommend Item 4 to 'B' as Person 'A' loved it.

But as the number of people using such services grow, and the variety of items available also increases vastly, it's difficult to find users having very similar interests to each other, and basing recommendations on that. That is why identifying items similar to the one a user liked became important to recommender systems, than finding other users who might have similar tastes. This project works on finding better ways to identify items which similar to each other.

Problem Statement, Project Objectives and Scope

Knowing what item a user likes isn't enough. It is important to identify other items which are similar to put them higher in the recommendation. These similar items can be called the item's neighbour. How we identify the neighbour of each item can significantly change the quality of the final recommendations. There are multiple ways to identify the neighbours, and among them is the use of Natural Language Processing (NLP) techniques on the description of the items.

User's feedback is stored in a sparse matrix of size $A \in R^{m \times n}$, where m is number of users and n is the number of items, with each value representing the rating from zero to five, as given by the user. The matrix is factorised into two matrices of smaller dimensions to deal with the empty values in the original matrix, with the intension that when one of these matrices is multiplied by the transpose of the other, the result is as close to the original matrix as possible. This helps the model predict the empty values in the original matrix, ie, predict whether the user would like that item or not, for which they haven't provided a feedback yet.

Natural Language Processing is a collection of computational techniques or algorithms which allow the computer to understand the natural language of humans better. The project aims to find out the best NLP technique for a recommender system by trying out different techniques on the description of the items data, combine them with the user's feedback, and generate recommendations.

The two parts of the project above will be used together to build the recommender system capable of recommending movies or books to a user based on their personal taste, basing the

model on the description of the item. That can act as a drawback as the item's description might not necessarily speak of its quality. Moreover, the model will not be very accurate for user who has not provided enough feedback for the model to understand their taste.

Major Technical Components

The project involves datasets for user-item matrix and description of the data, both of which will be processed on Python using various libraries such as PyTorch.

Movie ratings have been taken from a database on MovieLens, first published in 2016, containing 20000263 ratings across 27278 movies by 138493 users.

Python is an open-source programming language, and version 3.10 is being used for the project. Python codes are written and compiled on is run on PyCharm 2020.3.5 on a MacBook Pro (M1, 2020), runing macOS 12.5.1.

PyTorch is an open source machine learning framework for Python which is often used for Natural Language Processing. It is based on the Torch library developed by Meta AI.

Several methods from various libraries, apart from PyTorch, such as numpy, pandas, will be used to achieve the tasks of the project, such as mathematical calculations, matrix factorisation, and other smaller sub-tasks.

The two large technical parts of the project are: 1) Running different Natural Langauge Processing techniques, such as BERT (Bidirectional Encoder Representations from Transformers); 2) Factorising matrices into two smaller matrices with least error to predict the empty values in the original matrix.

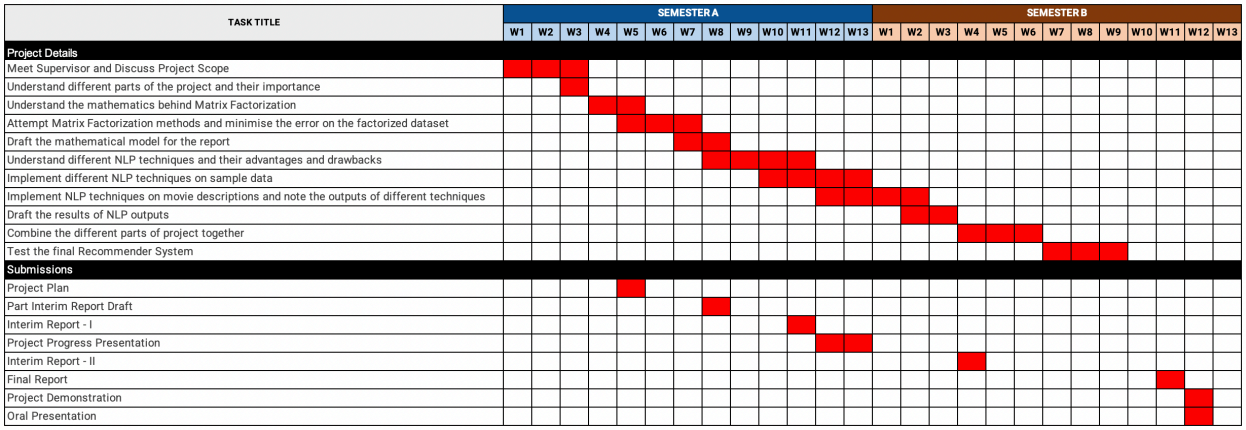
Expected Results & Deliverables

At the end, the project expects to have a recommender system machine learning model trained to provide personalised recommendations to users based on the previous feedback provided by them. The best NLP technique to compare items will be first identified as different techniques may prioritise different keywords of the item description, changing the results significantly. It will then be used with the factorized matrices to build a recommender system. The project will be completed as a Python project, using multiple pre-installed libraries, and will be hosted on GitHub.

This project should be able to signify the importance of choosing the correct NLP technique to identify the neighbours, and elaborate on what the advantages of some NLP techniques are over the others. The final model will be tested by using a test set and a model which keeps the importance of positions in recommendations intact while giving a score.

Project Schedule

The project started in September 2022 and is expected to be wrapped up by April 2023. First interim report will be completed by November 2022, with the final report and presentation in April 2023.



The Gannt chart above elaborates on the project schedule.

References

1. Chowdhary, K. R. (2020). Natural Language Processing. *Fundamentals of Artificial Intelligence*, 603–649. https://doi.org/10.1007/978-81-322-3972-7_19

2. Harper, F. M., & Konstan, J. A. (2016, January 7). The MovieLens Datasets. *ACM Transactions on Interactive Intelligent Systems, 5*(4), 1–19. https://doi.org/10.1145/2827872

3. Ma, C., Kang, P., Wu, B., Wang, Q., & Liu, X. (2019, January). Gated attentive-autoencoder for content-aware recommendation. In *Proceedings of the twelfth ACM international conference on web search and data mining* (pp. 519-527).

4. *Matrix Factorization*. (n.d.). Google Developers. Retrieved September 14, 2022, from https://developers.google.com/machine-learning/recommendation/collaborative/matrix

5. Ricci, F., Rokach, L., & Shapira, B. (2021, November 22). Recommender Systems: Techniques, Applications, and Challenges. *Recommender Systems Handbook*, 1–35. https://doi.org/10.1007/978-1-0716-2197-4_1

6. Rocca, B. (2021, December 10). *Introduction to recommender systems - Towards Data Science.* Medium. Retrieved September 16, 2022, from https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

7. Saunders, D. (2022, January 6). *How Can You Tell if Your Recommender System Is Any Good?* Medium. Retrieved September 16, 2022, from https://towardsdatascience.com/how-can-you-tell-if-your-recommender-system-is-any-good-e4a6be02d9c2

8. Smith, B., & Linden, G. (2017). Two decades of recommender systems at Amazon. com. *Ieee internet computing*, 21(3), 12-18.

9. Thandapani, S. P. (2022, June 17). *Recommendation Systems: Collaborative Filtering using Matrix Factorization — Simplified*. Medium. Retrieved September 15, 2022, from https://medium.com/sfu-cspmp/recommendation-systems-collaborative-filtering-using-matrix-factorization-simplified-2118f4ef2cd3