Comparing Collaborative Filtering-Based Recommender and Hybrid Recommender System

Ajay Dahiya

C0783939@mylambton.ca

Akshita Khatri

C0785493@mylambton.ca

Angelica Serrano C0785370@mylambton.ca

Jonatas Aguiar C0790419@mylambton.ca

AML 3204

Instructor: Debashish Roy

Social Media Analytics

Lambton College – Toronto

Abstract— Artificial Intelligence (AI) and Machine Learning (ML) applications have significantly grown over the years. Most business sectors have acquired, and developed systems driven by ML algorithms. The advancement of these systems allowed huge convenience, not only to the businesses but also to the users. The success of existing ML applications opened several possibilities of future enhancements. More research and experiments are being conducted to elevate the competence of the existing ML methods. In this paper, two different approaches in building a Recommender System were evaluated to find out which method will exhibit better performance.

*Keywords* – Artificial Intelligence, Machine Learning, Recommender System

# **1. INTRODUCTION**

Recommender System (RS) is a machine-learning system that suggest relevant items or products to users. RS is becoming popular on several business sectors nowadays. Aside from retail and e-commerce, RS is also growing in travel and healthcare industry. RS can predict future patterns based on past data through an array of techniques, including matrix factorization. Seeing good recommendations while browsing a movie, for instance, provides an immersive movie-watching experience and satisfaction from a customer perspective. It speeds up finding all the preferrable films, gives a rundown of similar interests of movies and genres. Sufficient data is essential in building an RS. A movie recommender requires user preferences, last watched movies, movie ratings and other features pertaining to user action towards any movie on the database. RS use three filtering algorithms: (1) Content-based filtering, which utilizes characteristics information; (2) Collaborative filtering, which uses user-item interaction; and (3) Hybrid system, which uses both information. There are three types of Collaborative Filtering: (1) Memory-based, which uses user historical data rating to compute the similarity measure between users or items. (2) Model-based, which uses machine learning algorithms to predict user's rating of unrated items; (3) Neural-based collaborative Filtering where embeddings represent each user and each movie in the data. This paper aims to analyze and compare the performance of Collaborative Filtering and Hybrid-based methods in generating movie recommendations. Collaborative Filtering approach has two main parts: Movie similarity and Movie rating prediction. While Hybrid-based approach has a Content-based add-on to the Collaborative Filtering-based approach. It uses both the movie and rating data and has four main sections: 1) Text preprocessing, 2) Term weighting, 3) Movie clustering and 4) Collaborative Filtering based approach. The performance comparison showed that Hybrid-based approach performed better than the traditional Matrix factorization approach.

# **2. LITERATURE REVIEW**

Recommender Systems (RSs) are software application tools that offer similar items to users [1]. RSs are created to help the users decide on choosing relevant articles that may make their movie watching experience more interesting. RSs can give personalized recommendations over general recommendations [2]. The companies like Netflix, Amazon and Facebook are using this system to increase their sales revenue growth. There are various recommender systems, as seen in Figure 4. The three essential types are (1) content-based and (2) collaborative-based and Hybrid recommender systems that have two branches: item-based and user-based. Within these two major categories lies multiple techniques or algorithms to provide recommendations. However, item-based collaborative systems performs better than user-based ones [3]. The most significant difficulty is the cold start problem, where either a new user or item is added to the system, which gets thrown deep down in the list [4]. The solution to this problem is to capture the knowledge of human to support the decision making. This approach is called Knowledge-based approach which includes an interface as well through which users query the system and interact with it. In today's internet era, the size of data is getting larger day by day and this abundant volume of data creates different problems for the recommendation systems [5]. They have several limitations; for example, Content-Based filtering techniques have issues like limited content analysis and sparsity of data [6], collaborative approaches have issues like cold-start, sparsity, and scalability. For these scenarios, the hybrid system can be used, as shown in Figure 5. The hybrid system works by combining two or more filtering techniques in different ways to increase the performance and accuracy of the recommender system has thus been proposed to alleviate these issues [7]. Hybrid systems try to leverage the strength of each method without sacrificing any capability due to inherent weakness [8]. A survey detected that the ratings from the knowledge base improve the efficiency rate of the Collaborative Filtering [9]. RS has emerged to work with web applications. It provides customization for customers of e-commerce, promoting one-to-one marketing [10].

# **3. METHODOLOGY**

**3.1 MODEL DESIGN**

Matrix Factorization was utilized to build a Collaborative Filtering-based Recommender System. Pytorch, on the other hand, was used to build a Hybrid RS. The figures shown at Figures 1 and 2 describes the comprehensive process or algorithms applied on the two methods used in this project.

Text, letter

Description automatically generated

***Figure 1 -*** *Matrix Factorization Algorithm*

Text

Description automatically generated

***Figure 2 -*** *Hybrid-Based algorithm*

**3.2 HYBRID RECOMMENDER SYSTEM**

The Hybrid Recommender shown in Figure 3 merges information used on both Collaborative Filtering and content-based methods. The content-based filtering leverages movie characteristics or features, such as sentiment scores (positive, neutral, or negative) and genres. Collaborative Filtering, on the other hand, is based on movie ratings from users. It is possible to merge movie features, user ratings, and the embedding layers, described in section 3.5. The merged data were fed into a neural network that outputs a value based on an activation function. If the output does not match an expected value, the neural network updates its weights so that, after a new run, the result gets closer to the expected value.

Each run is called an epoch. The expected output value is the rating users gave to the movie they watched. If a trained neural network can accurately predict the user rating, it is possible to create a recommendation list of films a user would love.

Diagrama

Descrição gerada automaticamente***Figure 3 -*** *Hybrid Recommender System Design*

**3.3 DATASET DESCRIPTION**

This project leveraged two datasets from the 'ml-100k' datasets collection distributed by the Group Lens Research Project in University of Minnesota. The first dataset, named u.data, is a collection of 100000 ratings given by 943 users to 1682 movies. Each instance in this dataset has a user id, an item id, a rating, and a timestamp. The second dataset, named u.item, describes the movie features, such as genres, movie id, movie title, release date, and video release date. The essential attributes to compose a Hybrid Recommender are the 19 genres that describe each movie. Genres on this dataset are: Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, and Western.

**3.4 MATRIX FACTORIZATION**

Matrix Factorization is one of the most widely used technique for dimensionality reduction. It works by compressing the initial sparse [user][item] and introduce them as separate matrix that will show user and items as unspecified feature vectors. This equation shows the breakdown process, where f is number of latent factors:

**ratings Matrix[user][item]** = sum (userFeature[f][user] \* item Feature[f][item])

The result of this equation will produce two matrices, userFeature of size (user \* f) and itemFeature of size (f \* item). Refer to Figure 4 for detailed procedure.

Diagram

Description automatically generated

***Figure 4*** *- Diagram of Matrix Factorization*

**3.5 NEURAL EMBEDDING LAYER**

The Neural Embedding Layer is a tensor, a vector of vectors, composed of random numbers like a conventional matrix. The number of vectors is the number of rows, and the size of each vector is the number of columns. When instantiating an object with Pytorch, we need to specify the number of vectors (embeddings) and the size of each vector. For example, on the left side of figure 5, we have an object instantiated with five vectors with size 3. This object has five vectors, and each vector has three random values. The right side of the same picture is an example of applying this to the Hybrid Recommender system. We can create an embedding layer with five vectors corresponding to each user, with three latent features. If we pass a list of users to the embedding layer, this object will return the corresponding vector to each user.

Diagrama

Descrição gerada automaticamente

***Figure 5*** *– Latent features generation with embedding layers*

**3.6 SENTIMENT SCORE CALCULATION**

Vader, a popular sentiment analysis model, was used to calculate the sentiment scores of each tweet extracted from Twitter. Vader's Sentiment Intensity Analyzer library produced a dictionary output that displays sentiment (key) and polarity scores (value). To determine the possible sentiment of each tweet, the weight of the compound can be used as an indicator. Scores between -0.05 to 0.05 can be considered neutral, and anything above 0.05 is positive, otherwise negative. The average (compound) score for each movie was calculated using Pandas groupby aggregate mean function.

**3.7 ACTIVATION FUNCTION OF HYBRID RECOMMENDER SYSTEM**

The Rectified Linear Unit (ReLU) activation function was used to build Hybrid Recommender system. Many neural networks architectures use ReLU activation function. Some of the reasons behind this trend are: 1) ReLU calculation is simpler compared to tanh and sigmoid activation functions: 2) ReLU can output actual zero values, leading to a desirable property to accelerate learning called sparse representation: 3) ReLU optimizes the model better as it is closer to linear behaviour, unlike the tanh and sigmoid functions. The rectified linear activation function graph can be seen in Figure 6, and its formula is denoted by (1):

(1)

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

***Figure 6 -*** *ReLU graph*

# **4. EXPERIMENTS**

The dataset taken from Movie Lens contains 100k movie ratings. In this section, we will throw some light on the experiment design, dataset preparation from Twitter tweets (taking account of tag words), evaluation metrics, results, and analyses of hyperparameter selection and recommendation models.

**4.1 EXPERIMENT DESIGN**

To create an experimental design for the model's proper functioning, the following steps were carried out:

* Downloaded and prepared all the data needed. Refer to section 3.3 for detailed description of Movie Lens dataset.
* Twitter developer credentials (Access token and Consumer API keys) were initially defined to obtain Twitter function access. Subsequently, Twitter authentication handler and API objects were created to be able to call the Twitter cursor function.
* Using Tweepy, Twitter data was extracted for each movie using the generated tag words as the search value.
* Data pre-processing and cleansing steps were performed on the twitter data. NLTK library was used to remove stopwords and apply lemmatization. Python regex module was also utilized for removing punctuations and non-word elements.
* Sentiment scores for each tweet were calculated using VADER or Valence Aware Dictionary for Sentiment Reasoning. Then, tag words were mapped against the movie titles from the original u.data.
* The singular Value Decomposition (SVD) technique was used to decompose a matrix into several component metrics.

**4.2 DATASET PREPARATION**

The datasets used for sentiment analysis are: 1) *u.item*, to get the movie list and generate tag words, and 2) Twitter data to get sentiment scores. To prepare the tag words, a series of steps were performed, such as:

1. Removed the trailing (year) for each movie titles.
2. Reformat the movie titles with incorrect construction (e.g “Birgcage, The” and “Time to kill, A”).
3. Removed space for each formatted movie titles.

To clean the twitter data, the following pre-processing steps were performed:

1. Dropped null and duplicate values.
2. Removed non-alphanumeric characters such as punctuation marks and emojis.
3. Removed twitter links.
4. Removed stopwords by importing nltk.corpus stopwords library.
5. Lemmatized each words using another nltk library called nltk.stem.WordNetLemmatizer

A limit of 5 tweets per movie was enforced during extraction. Based on the collected tweets, from Figure 7,"The Big Squeeze" topped the movie with the highest positive review. Followed by "Til There was you," with a 93% average sentiment score.

**Chart

Description automatically generated**

***Figure 7 -*** *Top 10 movie*s

From Figure 8, The movie with lowest sentiment score turned out to be "Killer: A Journal of Murder (1995)” with -0.84 followed by "Faster Pussycat! Kill! Kill!” with -0.83

Chart, funnel chart

Description automatically generated

***Figure 8 -*** *Bottom 10 movie Graph*

**4.3 EVALUATION METRICS**

Two metrics have been used to compare the two models: Root Mean Square Error (RMSE) and Mean Absolute Error (MAE).

RMSE (2) measures the error of a model in predicting quantitative data.

(2)

Here, 𝑦̂1, 𝑦̂2, ..., 𝑦̂𝑛 are the predicted values, 𝑦1, 𝑦2, ..., 𝑦n are the observed values, and 𝑛 is the number of observations.

MAE (5) is one of the many metrics for summarizing and assessing the quality of a machine learning model. Here, error refers to the subtraction of Predicted value from Actual Value as below.

(3)

The prediction error takes each record and convert them all to positive. To take the Absolute Error (4) this equation is applied.

(4)

Finally, we calculate the mean for all recorded absolute errors (Average sum of all fundamental errors).

(5)

Here, 𝑦̂1 is the predicted value, 𝑦1 is the observed value, and 𝑛 is the number of observations.

Precision (6) is calculated as the number of true positives divided by the total number of true positives and false positives.

(6)

Recall (7) is calculated as the number of true positives divided by the total number of true positives and false negatives.

(7)

F-score (8) combines precision and recall into a single measure that captures both properties.

(8)

**4.4RESULTS AND ANALYSES**

**4.4.1. HYPERPARAMETER SELECTION**

In this section, two different experiments were executed. For both models (Matrix Factorization and Hybrid Recommender), the calculated RMSE were generated when running the model with latent vectors of sizes 20, 30, 40, and 50. Line diagram was plotted with RMSE x Latent Feature sizes for each model.

From the graph seen at Figures 9 , Matrix Factorization model demonstrated a linear relation between the number of latent features and RMSE. Additionally, RMSE decreases as the number of latent features increases. The lowest RMSE value was around 1.85.

The Neural Network graph on Figure 10 demonstrated that RMSE decreases as the embedding layers size increase until it reaches the size of 40, when RMSE starts to remain constant at value 0.91 with lower fluctuation. Comparing both results, we can say that Hybrid Recommender model gave lower RMSE value than the first one, Matrix Factorization.

Gráfico, Gráfico de linhas

Descrição gerada automaticamente***Figure 9 -*** *Matrix Factorization - RMSE x Number of latent features*

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

***Figure 10 -*** *Hybrid Recommender - RMSE x Number of latent features*

**4.4.2 RESULTS OF RECOMMENDATION MODELS**

The model that was identified to have lowest RMSE from the previous section was selected to be used further in this section.

For each model, the following experiments were executed:

* Created Top 20 recommendation list for a user.
* Checked if the movies in recommended list were watched by the user.
* Based on these suggestions - RMSE, MAE, Precision, Recall, and F1-Score were calculated.
* Repeated the process for all users.
* Calculated the mean of all metrics.

Table 1 shows the metrics of the top-10 recommendation list for each model. The Hybrid Recommender had lower RMSE and MAE values than the ones presented by the Matrix Factorization model. Recall of Matrix Factorization was 14% higher than the Recall of the Hybrid Recommender model. However, precision and F1-score of the Neural Network Recommender were 53% and 37%, respectively, higher.

|  |  |  |
| --- | --- | --- |
| **Metrics of Top-10 recommendation** | | |
|  | **Matrix Factorization** | **Hybrid Recommender** |
| **RMSE** | 1.289 | 0.879 |
| **MAE** | 1.099 | 0.826 |
| **Precision** | 0.143 | 0.220 |
| **Recall** | 0.862 | 0.755 |
| **F1-score** | 0.229 | 0.315 |

***Table 1 -*** *Metrics of Top-10 recommendation*

Table 2 shows the metrics of the Top 20 recommendation list for each model. All the metrics of the Hybrid Recommender model were better than the Matrix Factorization ones. Precision, Recall, and F1-score of Neural Collaborative Filtering were 24%, 57%, and 24%, respectively, higher than the ones provided by the traditional method.

|  |  |  |
| --- | --- | --- |
| **Metrics of Top-20 recommendation** | | |
|  | **Matrix Factorization** | **Hybrid Recommender** |
| **RMSE** | 1.435 | 0.925 |
| **MAE** | 1.238 | 0.834 |
| **Precision** | 0.209 | 0.259 |
| **Recall** | 0.739 | 0.903 |
| **F1-score** | 0.295 | 0.367 |

***Table 2 -*** *Metrics of Top-20 recommendation*

# **5. CONCLUSION**

Two different recommendation systems were implemented to build a Recommender System (RS). Traditional Matrix Factorization and Hybrid Recommender (driven by the newly Neural Collaborative Filtering) were the primary methods used and analyzed during development phase.

Three experiments were conducted to evaluate the performance of each model.

1. From the generated graphs, it was demonstrated that RMSE decreases as the number of latent features increases from 20 to 50.
2. Top 10 recommendation list was generated from each model. Based on the comparison result, all metrics, except recall, were better in the list generated by Hybrid Recommender.
3. Top 20 recommendation list was generated from each model. Based on the comparison result, metrics of the Hybrid Recommender were better than the ones of Matrix Factorization.

In conclusion, Hybrid Recommender exhibited better performance than the other method. Thus, it can be a potential method to improve the traditional recommender systems that drive industry today.

# **6. REFERENCES**

1. Ricci F., Rokach, L., Shapira, B. (2015) Introduction and Challenges. In: Ricci F., Rokach L., Shapira (Eds). *Recommender Systems Handbook*. (1-34). <https://doi.org/10.1007/978-1-4899-7637-6_1>

2. Ricci, F., Rokach, L., Shapira, B. (2011) Introduction to Recommender Systems Handbook. In: Ricci F., Rokach L., Shapira B., Kantor P. (Eds). *Recommender Systems Handbook*. (1-35). <https://doi.org/10.1007/978-0-387-85820-3_1>

3. Joel Azzopardi, Department of Artificial Intelligence Faculty of ICT University of Malta, [Item-based vs User-based Collaborative Recommendation Predictions](http://www.keystone-cost.eu/ikc2017/slides/presentation_17.pdf)

4. Melville, P., Sindhwani, V. (2010). Recommender System. Encyclopedia of Machine Learning. <https://doi.org/10.1007/978-0-387-30164-8_705>

5. Resnick, P. J., Varian, H. R., (1997). Recommender Systems. *Communications of ACM, 519(3),* 56-58. <https://doi.org/10.1145/245108.245121>

6. G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," in IEEE Transactions on Knowledge & Data Engineering, Apr. 25, 2005, pp. 734-749.

7. B. Mobasher, “Recommender Systems,” Kunstliche Intelligenz, Special Issue on Web Mining, BottcherIT Verlag, Bremen, Germany, 2007, pp. 41-43.

8. M. Y. H. Al-Shamri and K. K. Bharadwaj, "Fuzzy-genetic approach to recommender systems based on a novel hybrid user model," Expert systems with applications, vol. 35, no. 3, pp.1386-1399., Oct. 2008.

9. Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction 12*, 331–370.

<https://doi.org/10.1023/A:1021240730564>

10.Jiawei Han, Jian Pei, in [Data Mining (Third Edition)](https://www.sciencedirect.com/book/9780123814791/data-mining-concepts-and-techniques), 2012