International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-11 Issue-3, September 2022

Personalized Recommendations of Products to Users

Spoorthi Chinivar

Abstract: ***Many organizations utilize recommendation systems to increase their profitability and win over their customers, including Facebook, which suggests friends, LinkedIn, which promotes employment, Spotify, which recommends music, Netflix, which recommends movies, and Amazon, which recommends purchases. When it comes to movie recommendation system, suggestions are made based on user similarities (collaborative filtering) or by considering a specific user's behavior (content-based filtering) that he or she wishes to interact with. Using TF-IDF, cosine similarity method for content-based filtering, and deep learning for a collaborative approach, this study compares two movie recommendation system. The proposed systems are evaluated by calculating the precision and recall values. On a small dataset, a content-based filtering methodology had a precision of 5.6% whereas a collaborative approach had a precision of 57%. Collaborative filtering clearly worked better than content-based filtering. Future improvements involve creating a single hybrid recommendation system that combines a collaborative and content-based approach to improve the outcomes.***

Keywords: Movie Recommendation System, Content-Based Filtering, Collaborative-Based Approach, Deep Learning, Tf-Idf, Cosine Similarity.

I. INTRODUCTION

Personalized product recommendations are when a website provides a series of unique product recommendations based on each visitor's behavior and profile. Recommender systems are utilized in a wide range of contexts, including e-commerce and online advertising. To filter through millions of items of information and provide its customers with personalised suggestions, online businesses like YouTube, Netflix, Amazon, Pinterest, and a long list of others rely on recommender systems. Recommender systems have been researched and shown to provide both Internet firms and their customers significant advantages [12]. With this in mind, the goal is to formulate machine learning and deep learning strategies for movie recommender systems. There are two primary categories of recommender systems: collaborative filtering, which may be further split into memory-based and model-based techniques, and content-based filtering. Content-based filtering methods are based on item descriptions and user-preferred profiles. For simplicity, it suggests similar movies based on a specified movie (movie name would be an input) or based on all movies watched by the user (user is an input).

Manuscript received on 26 August 2022 | Revised Manuscript received on 08 September 2022 | Manuscript Accepted on 15 September 2022 | Manuscript published on 30 September 2022.
***** Correspondence Author
Spoorthi Chinivar*, Masters, Data Science, FAU Erlangen, Germany. E-mail: srchinivar@gmail.com

It extracts features of items and can also look at the user's history to make recommendations. Collaborative filtering is a technique for predicting a user's interests by integrating preferences from several other users. It suggests unwatched movies to a user from a collection of movies that have already been seen by users who share similar interests.

There are many ways to implement a recommendation engine. Some of the limitations of today's systems include the lack of data, cold start problems, and changes in user preferences. Some examples of highly efficient recommendation systems include engines used by multinational technology giants such as YouTube, Netflix, Amazon and Pandora. These systems process huge amounts of data at various stages, including the training stages [2]. The main aim of this work is to build an efficient movie recommendation system based on 'Movie Lens' dataset using Content-based and Collaborative filtering approach and compare the results of both.

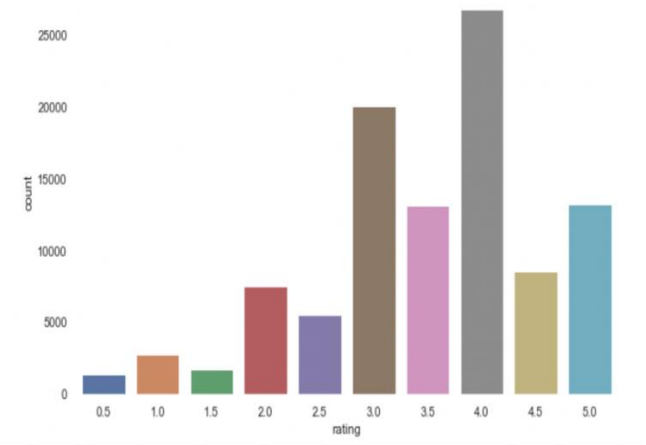
II. RELATED WORK

The most common way to perform content-based filtering is with a similarity metric. A similarity metric is a mathematical measure used to determine how similar a vector is to a given vector. They are computed from the item's feature vector and the user's preference feature vector from past records. Then some of the most similar items are recommended. Similarity metrics are mainly cosine similarity (cosine angle between vectors), dot product (product of cosine angle and magnitude of vectors), Euclidean distance (element-wise squared distance between two vectors), and Pearson similarity (ratio between the covariance and the standard deviation of both vectors). B.S. Oladapo [1] applied content-based technique in paper recommendation system. The author used Jaccard similarity coefficient also known as Jaccard index to compute similarity between users' query (users' attributes) and the attributes of the papers. The recommendations suggested by the system were sent via emails to the intended users. N. Muthurasu et. al. [2] created a movie Recommendation System using Term Frequency-Inverse Document Frequency (TF-IDF) and Cosine Similarity Method. Donghui Wang [3] proposed a recommender system on computer science publications referred to as the Publication Recommender System (PRS). This system is based on a new content-based filtering (CBF) recommendation model using chi-square and SoftMax regression, which are combined to construct a real-time online system. The most general method of achieving collaborative filtering is to use a k-nearest-neighbor approach between users as proposed by W.

Croft et. al. [4]. Clustering also helps in finding similar groups in user space. K-Means clustering is the most common clustering algorithm. An effective collaborative movie recommendation system was proposed which uses a hybrid K-Means clustering with a Cuckoo search algorithm for optimization by Katarya R et. al. [5]. There is another popularly known technique called, matrix factorization for performing collaborative filtering as described by C.Volinsky et. al. [6]. Xiangnan He et. al. [7] presented a neural network architecture for modeling latent features of users and objects and developing a general framework for neural network-based collaborative filtering. Snehal R Chavare et. al. [8] described a recommendation system by translating the inner product of traditional matrix factorization into deep learning. A collaborative filtering deep learning technique based on an autoencoder accurately predicted movie ratings as presented by Jeffrey Lund et. al. [9].

III. DATASET DESCRIPTION

The proposed model uses the Movie Lens dataset. The Movie Lens dataset was collected by Group Lens research project at the University of Minnesota [13]. This dataset contains 3683 movie tags of 9742 movies with 100836 ratings ranging from 0.5 to 5 (see Graph 1). The dataset is preprocessed and contains only users who have rated 20 or more different movies. Data is contained in links.csv, movies.csv, ratings.csv, and tags.csv files. No demographic information is included. Each user is represented by his/her ID and no other information is provided.



Graph 1: Count of ratings of all movies in dataset.

IV. PROPOSED SYSTEM

Proposed systems include implementation of a content-based filtering approach using TF-IDF, a cosine similarity method as explained in paper [2] and a collaborative approach using deep learning by authors Ram Murti Rawat et. al. [10]. The main advantage of this system is that the algorithms are designed to work efficiently even on small datasets. Recommendations are based on movie plot information, history, user profile information, and user likes and dislikes.

A. TF-IDF and Cosine Similarity

This method is commonly used as part of content-based recommender systems. It consists of two terms, Term Frequency (TF) and Inverse Document Frequency (IDF). Term frequency deals with the frequency of interests and favorites in user profiles. Inverse document frequency, on the other hand, deals with the inverse of term frequency in the total data provided by user profiles. These two concepts are combined to provide recommendations to users based on data provided by user profiles as explained in [14].

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

$$tf_{i,j} = number\ of\ occurences\ of\ i\ in\ j$$

$$df_{i,j} = number\ of\ documents$$

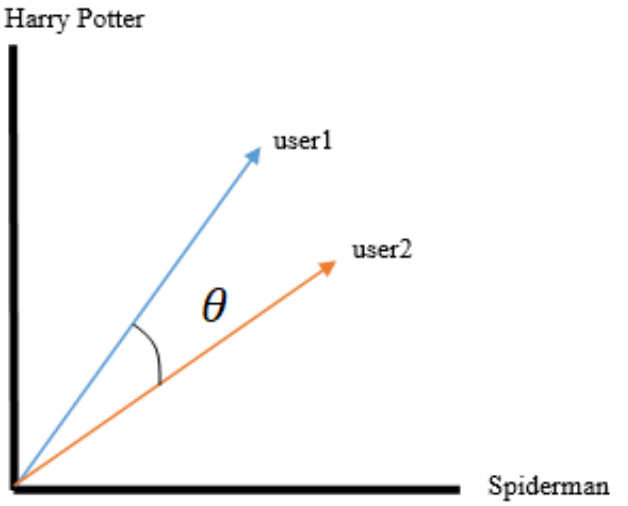
$$N = total\ number\ of\ documents$$

Here, we have the term frequency, or the number of times a certain term (such as a genre) appears in a document (such as a movie's genres), multiplied by the right-side term, which essentially scales the term frequency according to the number of times a particular word appears in all documents (movies).

The resultant weight is higher when there are fewer films that belong to a certain genre (df_i). The main purpose of the logarithm is to minimize large discrepancies caused by the right-hand component by smoothing the division result.

The movie genres and titles are transformed as vectors in a geometric space. Therefore, the angle between two vectors represents the closeness of those two vectors. Cosine similarity is a complex concept that has been widely discussed in information retrieval. This algorithm converts text documents into concept vectors. This model can find similarities between two data sets by determining the cosine value between the two vectors. The smaller the angle between two vectors, the larger the cosine and the higher the similarity. The cosine similarity of two vectors A and B is represented using a dot product and magnitude as,

$$\cos(\theta) = \frac{A \cdot B}{||A||\,||B||}$$



Graph 2: Cosine similarity between movie 'Harry Potter' and 'Spiderman'

Retrieval Number: 100.1/ijrte.C72740911322
DOI: 10.35940/ijrte.C7274.0911322
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Graph 2 shows the cosine angle between two movies 'Harry Potter' watched by user1 and 'Spiderman' enjoyed by user2. This explains how relevant these two movies are for users user2 and user1, respectively.

B. Embedding Layer

During an embedding, discrete or categorical variables are converted into a vector of continuous numerical values. In the context of deep learning, embeddings are multidimensional vectors that use previously learned continuous values to represent discrete or categorical variables. By easily expressing categories in a modified space that can be fed into neural networks and other machine learning approaches, embeddings are used to improve the representation of categorical data.

C. Embedded-based Deep Learning

Given that many recommender systems employ this strategy, collaborative filtering recommendation system becomes significant technique. The two primary categories of collaborative filtering algorithms are memory-based and model-based. Memory-based collaborative filtering finds users who are similar to a certain user in user space and dynamically suggests movies to that user. This method has two drawbacks: a scarcity of data and significant computing complexity. In a model-based collaborative filtering technique, item recommendations are made using the model that has already been built based on user analyses of previous items. Different machine learning and neural network algorithms are used to create the models.

The suggested network makes ratings predictions for movies that people haven't seen. Because machine learning methods require numerical inputs, the system embeds the user ID and movie ID into fixed-size continuous vectors.

D. Network Architecture

The deep learning network receives user and movie IDs directly as the model looks for connections between IDs that don't exist. A user ID embedding vector and a movie ID embedding vector are produced from the embedding spaces of 610 users and 9742 films, respectively. The dropout layer receives the vectors after they have been concatenated. To prevent overfitting the model, a dropout layer is employed. Dropout randomly sets the layer's outgoing edge to 0 with a predetermined probability at each training epoch. The input that has been received is passed on to the hidden layer, the subsequent layer. ReLU activation features and multiple fully connected hidden layers with dropout probabilities are introduced. Output characteristics and dropout probabilities might vary for each hidden layer. Based on the output characteristics of the preceding layer, the input features are chosen for next layer. Finally, for the input user id and model id, the network must send the predicted score as output. This is accomplished by using an output layer with a sigmoid activation function and rescaling it later in the 1–5 range. At each training epoch, a backpropagation algorithm is used to adjust the neuron weights in each layer. The structure of the proposed network is as shown in Fig. 1 similar to network proposed by Ram Murti Rawat et. al. [10].

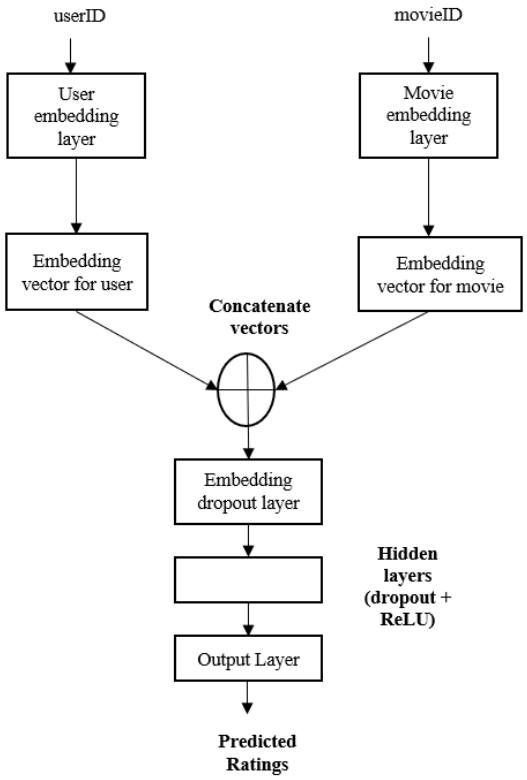


Fig. 1: Neural Network Architecture

E. Evaluation

First, some pre-processing of the data is done on. The dataset is divided into three parts in which 70% is for training data, 10% for validation, and the remaining 15% for testing. To check the performance of recommender system model, the RMSE is calculated during training, validation and testing. Root mean squared error (RMSE) is a common metric used to calculate system accuracy. RMSE is computed by taking the square root of the mean squared error. The average is calculated by dividing the sum of the squared differences between the actual output and the predicted output by the number of samples. The equation below represents the RMSE.

$$RMSE = \sqrt{\sum_{k=0}^{N} \frac{(predicted_k - actual_k)^2}{N}}$$

N is the total number of instances. The smaller RMSE value means better performance for the system. Thus, a system which produces smaller RMSE value will provide better recommendations. To compare the performance of the content-based and collaborative recommender system model, the precision and recall is calculated. Precision and recall are binary metrics used to evaluate models with binary output. Thus, there is a need to translate our numerical problem (ratings usually from 1 to 5) into a binary problem (relevant and not relevant items). To do the translation, assume that any true rating above 2.5 corresponds to a relevant item and any true rating below 2.5 is irrelevant. A relevant item for a specific user-item pair means that this item is a good recommendation for the user in question.

In the context of recommendation systems, recommending the user the top-N items is probably what we are most interested in doing. Therefore, it is more logical to compute precision and recall metrics for the first N items rather than the entire set of objects. As a result, there is the concept of precision and recall at k, where k is a user-definable integer that is specified by the user to correspond to the purpose of the top-N suggestions as explained in [15].

Precision at k is the proportion of recommended items in the top-k set that are relevant.

$$Precision@k = \frac{\#\ of\ recommended\ items\ @k\ that\ are\ relevant}{\#\ of\ recommended\ items\ @k}$$

Recall at k is the proportion of relevant items found in the top-k recommendations.

$$Recall\ @k = \frac{\#\ of\ recommended\ items\ @k\ that\ are\ relevant}{total\ \#\ of\ relevant\ items}$$

V. RESULTS AND DISCUSSION

The precision and recall score obtained for content-based recommendation system (TF-IDF and Cosine Similarity) on test data is 5.6% and 4.3% respectively. Whereas the same evaluation metric provided 57% precision and 68% recall in collaborative-based approach (deep learning).

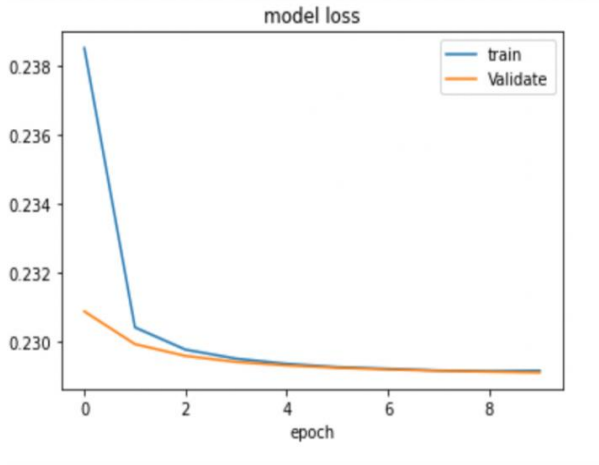


Fig. 2 : Training and Validation Loss over Number of Epoch for Collaborative filtering approach

Fig. 2 shows how the training and validation errors are changing with increasing training epochs of the collaborative filtering model. The training error is decreasing continuously along with validation error till epoch 6 but almost stays the same after that. For deeper neural network at epoch 10 RMSE is 0.22.

Therefore, the better prediction accuracy results can be explained by the fact that the use of users and items embeddings with deep neural network is very efficient.

The drawback of the used recommendation system is limited content analysis which leads to less accuracy of the recommendation system. Collaborative approach faces a cold-start problem where the profile of new user or item will be empty since he or she has not rated any item; hence, his or her taste is not known to the system.

But, content-based filtering approach have the ability to recommend new items even if there are no ratings provided by users. So even if the database does not contain user preferences, recommendation accuracy is not affected. The deep learning technique with embedding layers has an additional advantage of scalability.

VI. CONCLUSION AND FUTURE WORK

In this work, a content-based recommender system is implemented using TF-IDF, cosine similarity, and a collaborative recommender system using an embedding-based deep learning approach. In a collaborative approach, user ratings of movies in the test data are predicted using deep learning methods, and an evaluation metric is calculated based on these predictions. Based on metrics, the collaborative model outperforms content-based on precision and recall scores.

Perhaps the biggest issue facing recommender systems is that they need a lot of data to effectively make recommendations. Vast amounts of data required to produce accurate recommendations open scope to other applications such as incorporation of big data tools and efficient data processing methodologies. The key to finding the perfect recommendation system for the user's needs lies in combination of content-based and collaborative approach into hybrid recommendation system. By implementing both in a single system, it can overcome drawbacks from both types of filtering techniques [11].

REFERENCES

1. B.F. Oladapo . "A Research Proposal on Paper Recommender System", unpublished, 2013.
2. N. Muthurasu, Nandhini Rengaraj, Kavitha Conjeevaram Mohan . "Movie Recommendation System using Term Frequency-Inverse Document Frequency and Cosine Similarity Method", IJRTE, 2019.
3. Donghui Wang, Yanchun Liang, Dong Xu, Xiaoyue Feng, Renchu Guan. "A content-based recommender system for computer science publications", Knowledge based Systems, 2018.
4. W.Croft, T.Strohman, and D.Metzler "SearchEngines: Information Retrieval in Practice", Pearson Education, Inc., 2105.
5. Katarya R, Verma OP. "An effective collaborative movie recommender system with cuckoo Search". Egyptian Informatics Journal, 2017. [CrossRef]
6. C.Volinsky, R.Bell, Y.Koren. "MatrixFactorization Techniques for Recommender Systems", IEEE, 2009.
7. Xiangnan He, Lizi Liao, Hanwang Zhang. "Neural Collaborative Filtering", International World Wide Web Conference Committee, 2017.
8. Snehal R . Chavare, Chetan J. Awati , Dr. Suresh K. Shirgave. "Smart Recommender System Using Deep Learning", Sixth International Conference on Inventive Computation Technologies, 2021. [CrossRef]
9. Jeffrey Lund, Yiu-Kai Ng. "Movie Recommendation using deep learning approach". IEEE Computer Society, 2018.
10. Ram Murti Rawat, Vikranth Tomar, Vinay Kumar. "An Embedding-based Deep Learning Approach for Movie Recommendation", IEEE Xplore, 2020.
11. Robin Burke. "Hybrid Recommender Systems : Suvery and Experiments", User Modeling and User-Adapted Interaction, 2002.
12. https://towardsdatascience.com/prototyping-a-recommender-system-step-by-step-part-1-knn-item-based-collaborative-filtering-637969614ea
13. F.Maxwell Harper and Joseph A. Konstan. 2015. "The MovieLens Datasets: History and Context.", ACM Transactions on Interactive Intelligent Systems (TiiS), 2015.
14. https://towardsdatascience.com/content-based-recommender-systems-28a1dbd858f5
15. https://medium.com/@m_n_malaeb/recall-and-precision-at-k-for-recommender-systems-618483226c54

Spoorthi Chinivar, Passionate and challenge-driven Master's degree student with a major in Data Science at Friedrich Alexander University - Erlangen with 2 years of work experience. I believe in sincerity, perfectionism and above all positive thinking. I value the job, which will give me an opportunity to apply my knowledge and provide me with an environment that stimulates learning. Key Technical and Inter-Personal Skills:

- Experience with AWS Cloud Services including AWS Lambda, Athena, S3, RDS, Code Commit, Step Function.
- Strong expertise in python programming using different libraries like Pandas, NumPy, SciPy and Matpotlib.
- Skilled in Data visualization tools such as Spot Fire, Power BI.
- Excellent interpersonal, communications and presentation skills.
- Strong analytical and problem-solving skills.
- Firm knowledge in AI, ML, Deep Learning, Software Engineering

Retrieval Number: 100.1/ijrte.C72740911322
DOI: 10.35940/ijrte.C7274.0911322
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)