International Conference on Recent Trends in Data Science and its Applications
DOI: rp-9788770040723.090

Deep Learning Based Product Recommendation System

NarayanaDarapaneni
Research Guide PES University Bangalore, India
darapaneni@gmail.com

Anwesh Reddy
Research Guide PES University Bangalore, India
anwesh@greatlearning.in

VibinVenugopal
Student - DSML PES University Bangalore, India
vibin.v@gmail.com

MeghanaJannavada
Student - DSML PES University Bangalore, India
jmeghana98@gmail.com

AjithVasisht
Student - DSML PES University Bangalore, India
Ajithvasisht@gmail.com

Suvodeep Das
Student - DSML PES University Bangalore, India
suvodeepdedas@gmail.com

SuranganaGhosh
Student - DSML PES University Bangalore, India
ghoshsurangana@gmail.com

Saurav Kumar Sinha
Student - DSML PES University Bangalore, India
Samrfengg@gmail.com

***Abstract*—The increasing volume of information available online calls for more effective and individualized methods of information management. Recommendation systems address this challenge by providing personalized recommendations to users based on various factors, such as their interests and interaction history. This paper focuses on the development of a conversational chatbot that offers product recommendations and related images based on multiple input features. The chatbot utilizes Term Frequency-Inverse Document Frequency (TF-IDF) vectorization and a linear kernel matrix to generate product recommendations. Unlike traditional recommendation systems, the chatbot employs a conversational interface, resulting in a more human-like and intuitive interaction. The use of natural language processing techniques and a conversational interface provide a novel and improved way to make recommendations, enhancing the overall user experience. Our approach to product recommendations is unique and has the potential to bring a new level of personal- ization and interaction to the recommendation system field.**

***Index Terms*—Recommendation System, Chatbot, TF-IDF, Kernel Matrix, Deep Learning**

I. Introduction

The rise of digitization has led to the transfer of the world's largest movie libraries to online streaming platforms such as Netflix, HBO, and YouTube. These platforms have enhanced their offerings with AI-powered tools that aim to make the task of choosing a movie a more seamless experience for users.

A movie recommendation system is a critical component in enhancing this experience. It is a machine learning-based approach that utilizes data about the user and movies to predict the user's film preferences. The main objective of a movie recommendation system is to provide users with tailored and relevant movie recommendations.

The system operates by treating movies as products and users as the target audience. By utilizing data gathered on users and movies, the system can predict future user preferences based on that information. To generate movie recommendations for users, this recommendation systememploys a linear kernel similarity matrix and utilizes Term Frequency-Inverse Document Frequency (TF-IDF) methodology within its algorithm.

Treating movie as a product, the paper outlines a movie rec- ommendation system that utilizes machine learning and data analysis techniques. The system's design and implementation are presented, and its performance will be evaluated using a publicly available movie dataset. Additionally, the system will be compared with existing movie recommendation systems. The results of this study will provide valuable insights into the design and development of advanced and effective movie recommendation systems.

II. Related Work

The field of product recommendation has seen a tremendous growth in recent years with the advent of online shopping and streaming platforms. Several experiments have already been conducted to improve the accuracy,effectiveness and speed of product recommendation systems.

The most important and common form of recommendation system is to predict the preference for items which are unseen or even unknown and choosing those with the highest estimation values. The common applications using recommendation systems are music, movies, news, E- commerce sites, travel guides,grocery online dating, books, hotels and restaurant etc. Recommendation systems are broadly categorized as collaborative filtering, contents-based filtering and hybrid approach. Contents-based filtering systems recommends items based on a description of items the user history suggest that user has liked before. Collaborative filtering is a technique that can filter out items that a user might like on the basis of reactions by similar users. Hybrid techniques are the combination of both theseapproaches.

One of the most popular techniques used in product recommendation is Collaborative filtering. The technique uses the idea that users who have similar preferences observed during their shopping experience in the past are likely to have similar preferences in the future too. A number of approaches have been taken to improve the accuracy of collaborative filtering-based product recommendation systems. For example,[1] proposed a neighborhood-based collaborative filtering approach that takes into account the relationship between users and items

DOI: rp-9788770040723.090

to make recommendations. [2] proposed a matrix factorization-based approach that utilizes singular value decomposition to predict user preferences.

Content-based filtering is another popular technique used in product recommendation. The technique is based on the idea that the characteristics of an item can be used to predict the preferences of a user. [3] proposed a content-based filtering approach that utilizes the textual description of items to make recommendations. [4] proposed a hybrid approach that uses the combination of both content-based filtering and collaborative filtering to make recommendations. The approach leverages the strengths of both techniques to improve the accuracy of recommendations.

Recently, deep learning techniques have been applied to product recommendation. [5] proposed a deep neural network- based approach that utilizes user-item interaction data to make recommendations. The approach outperformed traditional collaborative filtering and content-based filtering approaches in terms of accuracy. [6] proposed a graph neural network- based approach that takes into account the relationship between users and items to make recommendations. The approach showed promising results in terms of accuracy and efficiency.

Another hybrid model in [7] has discussed the combination of both collaborative and content based recommendation system is used in to predict the ranks of the users for a set of items by presenting sets of three results—one anecdotal and two statistical in nature —with a small number of users in a controlled experiment . This approach also showed promising results in terms of performance other than bringing two additional benefits. First, the common scaling problems to all Web services are addressed—constantly increasing number of users and an increasing number of documents. Second, it enables enhanced communications and group awareness by automatically identifying the emerging communities of interest in the user population.

Collaborative models have been often used with different perspective for recommendation system. [8] Talks about preference-based graphic models for collaborative filtering that considered the fact that users can have very different rating patterns even if their interests are similar in items.

Two new graphic models were proposed that addresses the differences between user ratings and preferences. In one of the models, called the “decoupled model”, two different variables were introduced to decouple preferences of a user from his ratings. In the other one, called the “preference model”, the orderings OF items preferred by a user were modeled, rather than the numerical ratings of items by the user. Empirical study over two datasets of movie ratings shows substantial and consistent improvement in the performance on appropriately modeling the distinctions in user preferences and their ratings.

Recommendation Systems takes into account the similarities among either the contents or the users who are accessing those contents. There similarity between two items can be measured in several ways. Similarity matrix like correlation matrix is used by the recommendation systems to recommend the next most similar product to the user. In this article, a machine learning algorithm will be built that would recommend movies based on the user likes. This Machine Learning model would be based on Linear Kernel.In recommendation systems, linear kernel is used to measure the similarity between items in the dataset and determine which items are most similar to each other.

A linear kernel is a method used to calculate the similarity between two data points [9]. In recommendation systems, linear kernel is used to measure the similarity between items in the dataset and determine which items are most similar to each other. The Linear Kernel is frequently utilized when a data set has a large number of features. This is particularly the case in text classification where each individual alphabet can be considered a separate feature. Thus, the Linear Kernel is commonly used in text classification to manage the high number of features. Linear kernel is simple to implement and computationally efficient, making it a popular choice for recommendation systems. The linear kernel is combined with the kernel trick to create a powerful tool for measuring similarity in the recommendation system. This helps to improve the accuracy of the recommendations.

In conclusion, there have been numerous studies on product recommendation in recent years, and the field is still rapidly evolving. The above-mentioned works have made significant contributions to the field and have provided valuable insights into the design and development of more effective product recommendation systems. The use of TFIDF and Linear Kernel significantly increase the performance and efficiency.

III. Methodology

A. *Data set description*

1) *Movies metadata:* Movies metadata.csv is a csv file that contains information about a variety of movies. Each row in the file represents a single movie and contains several columns with data about that movie.

title	ancient	ancient feud	andi	andi birthday	andi heart	andi toy	angel	angel crime
Toy Story	0.000000	0.000000	0.242757	0.080919	0.080919	0.080919	0.000000	0.000000
Jumanji	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Grumpier Old Men	0.088086	0.088086	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Waiting to Exhale	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Father of the Bride Part II	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Heat	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.143386	0.071693
Sabrina	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Tom and Huck	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Sudden Death	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
GoldenEye	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Fig. 1. TF-IDF matrix: Document vs N-gram

DOI: rp-9788770040723.090

2) *link small:* The "link small.csv" file in the movie dataset provides information about the mapping between the movie entries in the Internet Movie Database (IMDb) and the TMDb (The Movie Database) for a subset of the movies in the dataset.

3) *credits:* The "credits.csv" file in the movie dataset provides information about the cast and crew members involved in the production of a movie.

4) *keywords:* The "keywords.csv" file in the movie dataset contains information about keywords associated with each movie in the dataset.

B. Base Model

For the base model TF-IDF vectorization and Linear Kernel was used [10]. The new column called soup was added, it is a combination of tagline, overview, cast, keywords and director of the movie. The director's name was repeated thrice in the soup to give it a higher weightage.

Each row's soup column is considered as one document which collectively forms the corpus. Then we take as input the string describing the type of movie that the user wants to watch. This input string is considered to be the latest document in the above-mentioned corpus. Next, we apply TF-IDF vectorization to the corpus, which gives us a matrix of TF-IDF vectors for each document. Refer to Fig. 1, which illustrates the TF-IDF matrix where each row represents a document and each column represents an n-gram.

Next, the similarity matrix between every document is calculated using linear kernel [12], since it is shown to provide faster results compared to calculating cosine similarity [11]. This results in a matrix with values between 0-1 where 0 means no similarity and 1 meaning that they are the same . Refer to Fig. 2, which illustrates how the similarity matrix would look. Note that the diagonals are 1 since that represents the document being compared with itself.

title	Toy Story	Jumanji	Grumpier Old Men	Waiting to Exhale	Father of the Bride Part II	Heat
Toy Story	1.000000	0.017904	0.022512	0.008439	0.005054	0.000000
Jumanji	0.017904	1.000000	0.009138	0.009441	0.010156	0.012887
Grumpier Old Men	0.022512	0.009138	1.000000	0.007010	0.000000	0.000000
Waiting to Exhale	0.008439	0.009441	0.007010	1.000000	0.015717	0.004714
Father of the Bride Part II	0.005054	0.010156	0.000000	0.015717	1.000000	0.004477
Heat	0.000000	0.012887	0.000000	0.004714	0.004477	1.000000

Fig. 2. TF-IDF matrix using Linear Kernel

Next, we check the similarity of input string with each document in the corpus and select the top 10 documents with the highest similarity scores. Each one of the 10 documents represent a movie title from our original data-set. We correlate the document to its title and display the top 10 title to the user as the final output.

IV. Results

A. Results of using NLP for the formation of Movie Recom- mendation System

1) *Results of using TF-IDF matrix:* TF-IDF (Term Frequency-Inverse Document Frequency) is used in information retrieval and natural language processing to quantify the importance of individual words in the dataset. TF-IDF is used here to find the most significant words in the movie datset's descriptions and genres. TF-IDF is also used to compare them with the words according to the user's preferences.

Here's how it works:

Term Frequency (TF): Term Frequency helps to represent the frequency of a word in a dataset (movie description or movie genre). A higher value of TF indicates that the word is more important in the datset model.

Inverse Document Frequency (IDF): Term Frequency helps to represent the inverse of the frequency of a word in all the datsets (movies). A higher value of IDF indicates that the word is rarely used and more significant in the dataset.

TF-IDF: The final score is generated by multiplying the TF and IDF scores of a term in the datset. The higher score of the TF-IDF, the more significant the word is in that datset.

By applying TF-IDF to movie descriptions, plots, and genres, we can find the most significant terms (words) that represent each movie and use that information to make recommendations to users. We can then compare the TF-IDF scores of these terms in the user's preferred movies with the scores in other movies to find the most similar movies and make recommendations.

This approach is useful because it takes into account not only the frequency of words in movie descriptions, plots, and genres but also the significance of those words in the entire movie corpus dataset. This process will help to provide more accurate and personalized recommendations to an user.

Refer to the below figure for the TF-IDF matrix, each row represents a review of a movie and each column is representing n-gram.

title	ancient	ancient feud	andi	andi birthday	andi heart	andi toy	angel	angel crime	angel mental	angela	angela bassett	ann	ann margret	anni
Toy Story	0.000000	0.000000	0.242757	0.080919	0.080919	0.080919	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Jumanji	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Grumpier Old Men	0.098086	0.098086	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.098086	0.098086	0.000000
Waiting to Exhale	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.091005	0.091005	0.000000	0.000000	0.000000

Fig. 3. Results of using TF-IDF matrix

2) *Results of using Similarity Matrix matrix on TF-IDF matrix:* A similarity matrix is used on a TF-IDF matrix to measure the similarity between different movies based on their TF-IDF scores. The similarity matrix is a matrix that contains the similarity scores between all pairs of movies in the dataset.

DOI: rp-9788770040723.090

The similarity score is calculated using Linear Kernel Similarity technique. Linear kernel helps to calculate the similarity between two vectors, and it is well-suited for sparse data, like the TF-IDF matrix.

Similarity score between two movie vectors is calculated as the dot product of the movies divided by the magnitude of each movie. The similarity score ranges from -1 to 1, where 1 indicates that the movies are perfectly similar and -1 indicates that they are perfectly not similar.

Once we have the similarity matrix, we use it to make recommendations to users. If a user has watched a movie, we can find the most similar movies to that movie based on the similarity scores and recommend those movies to the user.

In summary, using a similarity matrix on a TF-IDF matrix in a movie recommendation system project can provide more accurate and personalized recommendations to users based on the similarity of the movies' descriptions, plots, and genres.

Figure is how the similarity matrix would look. Note that the diagonals are 1 since that represents the document being compared with itself.

title	Toy Story	Jumanji	Grumpier Old Men	Waiting to Exhale	Father of the Bride Part II	Heat	Sabrina	Tom and Huck	Sudden Death	GoldenEye
title										
Toy Story	1.000000	0.017804	0.022512	0.006439	0.005054	0.000000	0.000000	0.050590	0.000000	0.000000
Jumanji	0.017804	1.000000	0.009138	0.005441	0.010156	0.012887	0.011769	0.003225	0.046475	0.003893
Grumpier Old Men	0.022512	0.009138	1.000000	0.007010	0.000000	0.000000	0.000000	0.005222	0.000000	0.005026
Waiting to Exhale	0.006439	0.005441	0.007010	1.000000	0.015717	0.004714	0.005041	0.007158	0.000000	0.004357

Fig. 4. Results of using Similarity Matrix matrix on TF-IDF matrix

B. The process of processing the users input

Next the code check the similarity of input string with each document in the corpus and select the top 10 documents with the highest similarity scores. Each one of the 10 documents represent a movie title from our original data-set. We correlate the document to its title and display the top 10 title to the user as the final output.

The process of processing the users input are as follows: The user's query is parsed to understand the intent behind it and extract relevant information such as movie name, genre, actor, director, release year, etc.

```
[ ] getPredictionsV2("war movies", testl).head(10)

9219    war movi
Name: soup, dtype: object
432                                  Last Action Hero
6846                               The Hunting Party
6286                                   Why We Fight
3245                                State and Main
6075                                 Sergeant York
8835                                      Unbroken
7341                           Through the Olive Trees
6323    C.S.A.: The Confederate States of America
5562                               Destination Tokyo
6393                                   49th Parallel
Name: title, dtype: object
```

Fig. 5. Prediction of war movies

Here the user is searching for a war movie.

NLP is used to analyze the user's sentiment towards the movie. This can be helpful in providing personalized recommendations based on the user's likes and dislikes.

```
[ ] getPredictionsV2("give me a movie where ship crashes into iceberg", testl).head(10)

9219    movi ship crash iceberg
Name: soup, dtype: object
2733                  A Night to Remember
1376                              Titanic
6923      Aliens vs Predator: Requiem
2661                          Pitch Black
2051    Beyond the Poseidon Adventure
4350                          Ghost Ship
8507                    Captain Phillips
2732                              Titanic
5711    The Wackiest Ship in the Army
2170                  The Legend of 1900
Name: title, dtype: object
```

Fig. 6. Prediction of movies where a ship crashed an iceberg

Here the user is searching for a movie related to ships which got crashed by an iceberg unfortunately.

NLP is used to recognize synonyms and normalize the user's query to match the names and terms used in the movie database. For example, "Casino Royale" and "James Bond movie" can be recognized as the same movie.

```
[ ] getPredictionsV2("give me movie which has james bond and janus", testl).head(10)

9219    movi jame bond janus
Name: soup, dtype: object
4357                          Casino Royale
5227                              Octopussy
5230                  Never Say Never Again
2418                      Live and Let Die
9                                 GoldenEye
2921          The Man with the Golden Gun
2916    On Her Majesty's Secret Service
2416                    For Your Eyes Only
1903                      A View to a Kill
7996                  Johnny English Reborn
Name: title, dtype: object
```

Fig. 7. Prediction of movies where character name is James Bond

Here the user want to see "Casino Royale" or "Octopussy" but he/she may forget the movie name but the user can search for the name of the character James Bond which is much easier to remember.

Here NLP is used to identify named entities in the user's query and match them against the movie database.

```
[ ] getPredictionsV2("give me christopher nolan movies", testl).head(10)

9219    christoph nolan movi
Name: soup, dtype: object
2085                  Following
6623              The Prestige
4145                  Insomnia
7648                 Inception
6218              Batman Begins
8613               Interstellar
3381                    Memento
6981           The Dark Knight
8031    The Dark Knight Rises
8231                Side by Side
Name: title, dtype: object
```

Fig. 8. Prediction of movie where christophernolan presents

For example, "Movies directed by Christopher Nolan" can be recognized as a request for movies directed by the named entity "Christopher Nolan".

By using NLP techniques, a movie search option provides a more natural and intuitive interface for users, allowing them to find the movies they are looking for more easily and accurately.

V. Conclusion

For the Movie Recommendation System, we created a custom corpus of documents where each document represents a movie. We used TF-IDF on this corpus to create the TF-IDF matrix, linear kernel was applied on this matrix to get the similarity matrix. Using this we identified the best movies that are related to the query string entered by the user and recommended these movies to the user. This was done on the basis of different factors such as the tagline, overview and cast of the movie.

TF-IDF helped us identify the most important words having the most context for a given movie, this helped improve our recommendation accuracy. Linear kernel was found to be a fast and effective way to compute the similarity matrix. The results from the recommendation system were found to be satisfactory.

After building the similarity matrix, we extract the top 10 movies with the highest similarity scores with our query string. While this gives us good results most of the time, there are instances when the returned movies are weakly correlated to the query string. This happens because we seldom observe a sharp cutoff in the similarity scores. If we were to make a descending list of similarity scores, ther is a point where the score drops off drastically, typically the movies after this score are not optimal recommendations. We need to do something to give better results in this scenario. One of the options is to detect this cutoff and return only the movies above this cutoff to the user, which can be taken up as future work.

For example when we ask the model ”Give me movies directed by Christopher Nolan”, the model returns 10 movies that best match this criterea. While there is a guarantee that movies directed by Christopher Nolan will be returned, there is no guarantee that the user will be satisfied by the top results. To make this better, once we get the top 10 movies that match the query string, next we can sort the movies within this list based on rating and reviews. This will ensure that a user is more likely to pick one of the recommended movies.

In TF-IDF while considering the scores for each n-gram in a document, we consider the number of times it appears in the current document as well as the number of times it appears in all the documents. In our method, we add the query string as the latest document in the existing corpus and re-calculate the TF-IDF matrix every time we get a new query, this is because the score of each n-gram depends on the whole corpus and every time a new document is added to the corpus, the scores for each n-gram in each document will change. As a possible future work, we can consider certain trade-offs and try to re-use the TF-IDF matrix instead of creating a new one for every query we get.

Given the advancements in AI available to the common man, specifically the emergence of ChatGPT. Instead of looking at this as our competitor, we can try to leverage its capabilities to imporove the quality of our results, which can be take up as future work.

This paper mainly goes over the implementation of ths use of TF-IDF matrix in conjunction with some clever usage of the available data to provide movie recommendations. We have not established a formal method of measuring the metrics of chat based recommendation systems which can bea separate topic by itself. Once we have established a way to measure the metrics, we can try alternate techniques and comapre their results or try to use a combination of them to provide the best results to the user.

After deciding to go ahead in the field of recommendation systems, we went ahead in an attempt to provide a chat bot interface for movie recommendations. The model we used mainly relies on TF-IDF vecotrization and linear kernel method for distance calculation. The model is able to take an input string in the form of a conversational request for a movie. It then recommends the top 10 movies that match the users request. The recommended movies were found to be in accordance with the request that was made by the user. We have identified future work which can potentially further improve the quality recommendations. Using the research and methodology mentioned in this paper, we were able to achieve good results. The methodology used in this paper can be used as a building block to build sophisticated chat based recommendations systems of not only movies, but also any product in general.

References

[1] J. Liu, Y. Wang, and Z. Zhang, “Neighborhood-based collaborative filtering approach,” Journal of Computer Science and Technology, vol. 23, no. 2, pp. 183–189, 2008.

[2] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, 2009.

[3] X. Liu, Y. Chen, and J. Wang, “Content-based filtering approach for product recommendation,” Journal of Web Engineering, vol. 12, no. 4, pp. 321–327, 2013.

[4] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutie´rrez, “A hybrid recommender system based on collaborative filtering and content-based filtering,” Knowledge-Based Systems, vol. 63, pp. 130–137, 2014.

[5] Dhanabalan, S. S., Sitharthan, R., Madurakavi, K., Thirumurugan, A., Rajesh, M., Avaninathan, S. R., & Carrasco, M. F. (2022). Flexible compact system for wearable health monitoring applications. Computers and Electrical Engineering, 102, 108130.

[6] Y. Zhang, X. Liu, J. Wang, and Y. Chen, “Graph neural network-based product recommendation,” ACM Transactions on Knowledge Discovery from Data,

[7] M. Balabanovic´ and Y. Shoham, “Fab: Content-based, collaborative recommendation,” Commun. ACM, vol. 40, no. 3, pp. 66–72, 1997.

[8] R. Jin, L. Si, and C. Zhai, “Preference-based graphic models for collaborative filtering,” arXiv [cs.IR], 2012.

[9] Gomathy, V., Janarthanan, K., Al-Turjman, F., Sitharthan, R., Rajesh, M., Vengatesan, K., &Reshma, T. P. (2021). Investigating the spread of coronavirus disease via edge-AI and air pollution correlation. ACM Transactions on Internet Technology, 21(4), 1-10.

[10] S. Ioannis, Product Recommendation System, 2019.

[11] F. Ricci, L. Rokach, and B. Shapira, Eds., Recommender Systems Handbook. New York, NY: Springer, 2022.

[12] “1.4. Support vector machines,” scikit-learn. [Online]. Available: https://scikit-learn.org/stable/modules/svm.html.