Revolutionizing Netflix Recommendations: The Power of Transfer Learning

[a,b]*Surabhi Anuradha*[1], *P Naga Jyothi*[2], *S Sivakumar*[3]

[1a]Research Scholar, School of Computer Science and Artificial Intelligence, SR University, Warangal, India
[1b]Department of Computer Sciecne and Engineering (AIML), Keshav Memorial Institute of Technology, Hyderabad, India
anuradha@kmit.in

[2]Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to Be University), Visakhapatnam, India
npothaba@gitam.edu

[3]Department of Chemistry, Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India
sivakumar.chemistry@anits.edu.in

ABSTRACT

Major e-commerce companies grapple with managing vast data sets, posing a challenge to sense and fulfill user preferences. Despite existing Recommendation Systems (RS) for movies, songs, and products, misconceptions in recommendations persist. Navigating Netflix's extensive library poses selection challenges for users. This paper introduces an advanced recommender system utilizing text-based similarity and Word2Vec embedding. The model suggests movies based on diverse attributes like titles, genres, and descriptions. Employing techniques such as Transfer Learning, Similarity Matrix creation, and Content-Based Filtering with Word2Vec embedding, the system offers a top-10 list resembling the user's selected title. Validated through A/B testing, the user interface allows inputting preferred Netflix titles, delivering personalized recommendations based on content attributes, enhancing the streaming experience.

Keywords : content-based filtering, Word2Vec, transfer learning, Similarity Matrix, Precision, A/B testing, Click-Through Rate

I. INTRODUCTION

Nowadays, entertainment plays a significant role in the day-to-day life of humans. Movies and songs are the source of entertainment. The issue is information retrieval from enormous data as the user intends to get results with high response time, accuracy, and relevance. To execute with high user choices, the model has to perform different techniques to select the relevant features, perform filtering, and deliver the top list based on user interests. The recommender system solves the problem of analysing the content and outcomes of the tailored content to the user [1]. There is a tremendous increase of data on various platforms like YouTube, Spotify, Netflix, and all other e-commerce sites. There are mixed proven results by the recommender system of Amazon, which increased its revenue by 30 percent and Best Buy by 23.7 percent, YouTube by 70 percent on videos and 60 percent in views, Netflix by 6.7% every year with 220.6 million subscribers worldwide by the implementation of recommendation algorithms [2]. More than 71% of e-

commerce sites recommend products on their homepage, which has helped them increase engagement, conversions, and revenue. While recommendations contributed to only 7% of visits, they accounted for 26% of income [3]. Netflix offers a vast library of shows and movies, making choosing the next one to watch challenging. To simplify this process, create a recommendation system that suggests Netflix shows based on your preferences. The RSs are most frequently used in domains like movies, product reviews, and documents to ease access. For instance, Movie data like Netflix, Prime, and MovieLens are predominant contributors to e-content. According to IMDb, worldwide movie production results in the creation of 4500 movies, generating approximately 9000 hours of content. There is demand for the RS to ease access and handle this massive amount of data to facilitate viewers' interests accurately. The recommender system is developed based on essential fundamental aspects, which are content-based and collaborative filtering. Users determine content-based recommendations through their preferences for products, movies, videos, and audio content. If user A likes a product X, the content-based approach will find similar products to X and will be recommended to user A. The collaborative approach suggests that if user A's profile is similar to user B's, when user B likes a movie X, then user A is recommended X, and vice-versa follows [4].

II. LITERATURE REVIEW

Roy et al., according to reviewed reviews on movies, books, and products on RS applications, despite considerable developments in RSs, they suffer from limitations like cold start, sparsity, scalability, serendipity, etc. Since many selection techniques exist for developing application-based RS, the author discussed different types of recommender systems, Content-based, collaborative, and hybrid RS models with advantages and drawbacks. As per the discussion, the limitation of content-based is that in-depth knowledge of features is required for an accurate recommendation. However, detailed information needs to be included for all items. So, it leads to a limited capacity for users to express their choices and interests. The advantage of content-based is it adapts quickly as per the user preferences. Security and privacy can be achieved as the user profile is not exposed to any other user. It did not suffer from problems with cold-start if the new item had a sufficient description. Most commonly applicable for personalized RS for news publication webpages, etc. Collaborative RS works efficiently based on how accurately the algorithm finds the neighborhood of a target user. It suffers from cold start problems and privacy issues as users share the data. The advantage is it does not require any item knowledge for generation recommendation. The utility matrix predicts the preferences based on the neighborhood interests. Broadly, the content-based RS is divided into memory-based and model-based. Hybrid filtering combines two or more to overcome the limitations of one another to improve the accuracy of prediction. The hybridization approaches like mixed, cascade, switched, and weighted are discussed. The paper briefly discussed the challenges of RS, like cold start problems, shilling, synonymy, latency, sparsity, grey sheep, and scalability. The review screened out 60 papers from 360 papers and categorized them based on application and techniques from 2011 to 2021 [5].

Jiang Zhang et al. developed personalized movie RS by using a scalable collaborative filtering technique called Weighted KM slope-VU. K-Means clustering algorithm is applied on user profile attributes. The virtual score is calculated for each cluster, and a virtual opinion leader evaluates the other items in the cluster. The Weighted KM slope VU is applied to user rating data and reduces the dimensions of the user item matrix, and prediction is made on the VU item matrix. Finally, he developed a web application and collected feedback from registered users. The most commonly used metric, RMSE, is used for predicting user rating. The performance, when compared to SVD++, was slightly lower because of outdated movies listed, and prediction is made on virtual users, not on real-time users [6].

Dev Kumar discussed Comparative study of movie recommendation systems based on the rating given by the users. The collaborative filtering mechanism is used, and the similarity index is calculated on the

feature's importance score on user-user similarity. The model was developed on 13 handcrafted parts and prediction is made using the XGBoost regressor .The results are compared between SVD+, KNN with SVD and XGBoost Regressor ML algorithms. The error function is suggested by the author to calculate the error between the rating given by the user and the global average of all rating biases to another user. L2 regularization is meant to observe different ratings like global average, movie bias, and user bias terms on a large dataset. The pattern for some users may be higher to lower than the global average, or it may be vice versa to minimize the error for accurate results. The results obtained on a partial dataset with fewer dimensional recommenders are attempting to catch the preferences and inclinations [7].

Abdullah Havolli et al. focus on developing a content-based RS based on text-based features. The Term Frequency-Inverse Document Frequency (TF-IDF) Vectorizer technique is used for text vectorization to find the movie's similarities. Adamic Adar graph measure is used to find the closeness of shared neighbors. The model works on contexts like actors, directors, countries, and categories to evaluate similar movies and recommends the relevant film to a user. The critical aspects of his work are pre-processing exploratory data analysis and system modeling. The TF-IDF is used to calculate the frequency and find the relevant word in the series of text to define the text vectorizer. The limitation of the model is that the Adamic Adar graph measures the similarity of an undirected graph. The results show the connection of the edge and present all the relations between nodes. His future work will incorporate Artificial Intelligence and Machine Learning algorithms to find similarities between movie algorithms [8].

Suraj Suresh K et al. developed an RS based on the KNN ML algorithm; the design created involves candidate generation, content-based filtration, and ranking stages. The collected videos from the YouTube corpus undergo content-based filtration based on the previously watched video. The model recommends the following video. Data usage is framed in a Utility matrix w.r.t to significant items and user choices. The degree of preference relationship is calculated based on the user items, the likes and dislikes. The ranking works on module matching. It selects a few things from a list of consumer appeals. The ranking system retrieves ranking objects based on item properties, and user-development needs to be revised to scale for extensive data [9], [10].

This paper presents a recommender system tailored to offer movie suggestions to users based on a comparison of various content attributes, including movie titles, genres, and descriptions. The primary focus of this research is to recommend the top 10 movies that are most similar to a user's selected movie title, leveraging techniques such as Transfer Learning with pre-trained models, creating a Similarity Matrix, and Content-Based Filtering employing Word2Vec embeddings.

III. METHODOLOGY

A. Content-Based Filtering

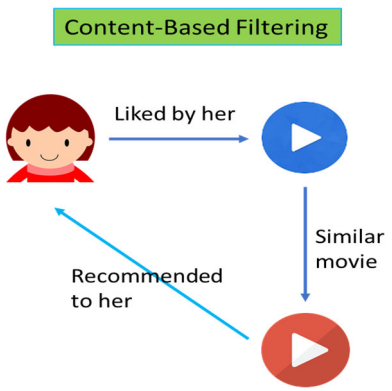

Fig. 1. Content-Based Filtering [9]

The central objective of content-based recommender systems (as shown in Figure 1) is to assess item similarities. For instance, let key_i represent the i^{th} keyword of item x_j, wt_{ij} denotes the weight assigned to key_i for $item_j$, and thus, the content of $item_j$ can be expressed as shown in (1).

$$Content(x_j) = \{wt_{1j}, wt_{2j}, \ldots..\} \quad (1)$$

As previously discussed, content-based recommender systems recommend items akin to the user's previous preferences. Hence, a user's taste can be characterized based on their historical preferences. Let's denote $ContentProfile(u)$as the preference vector of the user u is defined in (2).

$$ContentProfile(u) = \frac{1}{|L(u)|} \sum_{x \in L(u)} Content(x) \quad (2)$$

Here, $L(u)$ represents the set of items previously liked by user u [11]. Following the calculation of content vector $Content(.)$ and user preference vector $ContentProfile(.)$ for all users, given a user u, and item x, the user's preference for the item is determined by the similarity between $ContentProfile(u)$ and $Content(x)$ as shown in (3).

$$Preference(s, x) = similarity(ContentProfile(u), Content(x)) \quad (3)$$

As stated by Kim Falk in his Practical Recommender Systems [9], Content-based filtering is a method employed to generate recommendations for similar items, often referred to as "More Like This" suggestions. These recommendations are tailored based on the content's metadata and characteristics rather than subjective opinions about the quality of movies. Imagine a scenario where a user relies solely on the content's metadata to make recommendations. Similarly, Raza et al. [12] and Abdulkarem et al. [13] conducted surveys to provide an overview of current trends and methodologies in contextual aware-based recommender systems.

B. Dataset Utilized

In our research, we utilized the "Netflix_titles.csv" dataset, which encompasses a variety of attributes such as "show-id," "title," "type" (indicating TV show or movie), "director," cast details, production information including country and release year, genre classifications, and content duration, accompanied by brief descriptions. This substantial dataset, consisting of nearly 9000 unique samples, formed the foundation for our recommender model.

C. Feature Set Selection

In our approach to create a content-based recommender system, we adopted a meticulous approach, selecting attributes with precision to offer tailored, context-aware recommendations to users. This process involved leveraging the rich information embedded within the dataset. The primary emphasis was on handpicking attributes that seamlessly aligned with the intended analysis. The overarching objective was to furnish recommendations for titles closely resembling a specified one. This undertaking included a thorough evaluation of the dataset's characteristics, aiming to optimize the performance of our recommendation model. The chosen attributes for this study comprised 'title,' 'type,' 'listed-in,' and 'description.

D. Proposed Model

Given a user's title preference, the Recommender algorithm searches for genres and description similarities to discover matching shows or movies. Top recommendations will be shown to the user via the user interface based on the relevancy scores determined as depicted in Figure 2. This process involves searching for content that aligns with visitors' perceptions of similarity, creating a personalized recommendation approach based on content attributes.

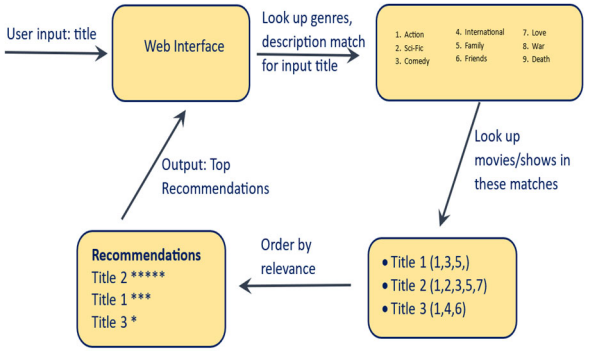

Fig. 2. Proposed Recommender System Pipeline

IV. IMPLEMENTATION

A. Using pre-trained Word2Vec Model

The development of an efficient Netflix movie recommendation system involved an innovative approach, utilizing a pre-trained word2vec model for effective mapping of high-dimensional vectors and user/item similarity calculation [14]. Drawing inspiration from the "prefs2vec" model, a word2vec-based technique, the strategy aimed to reduce computing complexity, enhancing the recommender system's speed [15]. Built on transfer learning principles, the model explored intricate connections within movie titles, genres, and textual descriptions. Employing the "word2vec-google-news-300" pre-trained model, initially trained on extensive Google News articles, enriched the system's understanding of movie-related language and semantics. The Gensim library facilitated the import of the Word2Vec model, providing versatility for natural language processing tasks. The extracted vocabulary from the model was then compared with tokenized attributes in the dataset. References to models like "prefs2vec" and the "word2vec-google-news-300" were integral to the methodology, showcasing the depth of knowledge harnessed for personalized and context-aware movie recommendations [16], [17].

B. Calculating Word Vectors for Netflix data

Continuing the utilization of the pre-trained Word2Vec model for Netflix recommendations, a crucial step involved identifying similarities among shows based on titles, genres, and descriptions. The establishment of 'matrix_netflix_vocab' acted as a repository for refined show data, entailing a meticulous examination of textual attributes for each show in the Netflix dataset. The model compared words in the title, genres, and description to the vocabulary of the loaded Word2Vec model, aligning show data with the captured semantic understanding. Subsequently, the processed show data was organized into another table, maintaining the original dataset structure but refining textual attributes to include only words with corresponding word vectors. This transformation proved essential for accurately calculating similarity scores and generating relevant show recommendations based on aligned word embeddings, significantly contributing to data preparation for the recommendation system and enhancing precision and relevance in show suggestions.

C. Recommendation Function

The recommendation function is defined in such a way to store processed show data based on the user's input title. It then traverses the Netflix dataset, aligning the title, genres, and description of each show with

word embeddings from the loaded Word2Vec model. A 'matrix_similarity' list captures similarity scores for genres, descriptions, and titles between the input show and others in the dataset [18]. Three similarity scores are calculated, and if genre similarity exceeds a threshold for shows with different types, scores are stored in 'matrix_similarity'. This data is used to create a table with recommended shows, their titles, and similarity scores. A final score is computed for each recommendation, and the top 10 shows with the highest scores are organized for user-friendly suggestions. The 'recommendation' function, detailed in Tables 8 provides context-aware and personalized Netflix show suggestions based on user input.

D. Evaluation Metrics

Precision

Precision@K is a metric used to evaluate the quality of recommendations in a content-based recommendation system [11], [19], [20]. It measures the proportion of relevant items in the top K recommendations as defined in (4).

$$Precision@K = \frac{Number\ of\ Relevant\ Items\ in\ Top\ K\ recommendations}{K} \tag{4}$$

Mean Average Precision (MAP)

MAP is a metric used to assess the quality of a ranking. It involves calculating precision at various cutoff points, starting from 1 and going up to the total number of recommended items, often denoted as 'k.' To provide a more comprehensive evaluation, the process continues by averaging the precisions over the first item, then the next two, and so on until it covers all recommended items (up to 'k'), which is defined in (5).

$$AP = \frac{1}{r} \sum_{k=1}^{K} (Precision@k * Relevant@k) \tag{5}$$

$$rel(k) = \{1, \quad if\ item\ at\ k^{th}\ rank\ is\ relevant$$
$$0, \quad otherwise$$
$$where, r - total\ relevant\ items$$

This procedure is applied to each recommendation individually. To evaluate a recommender system, you can then compute the mean of these individual average precisions across all recommendations as defined in (6).

$$MAP = \frac{\sum AP@k}{Total\ Queries} \tag{6}$$

It evaluates the quality of a ranking by calculating the precision at different levels, starting from the first item and progressively considering more items up to the total number of recommended items (often denoted as 'k'). To extend this evaluation further, MAP computes the average of these precisions over the first item, then the next two items, and so on until it reaches the last item in the ranking. This provides a comprehensive measure of the ranking's overall quality.

Discounted Cumulative Gain (DCG)

The MAP merely considers relevance, but the DCG considers levels of relevance. To calculate DCG, assign a relevancy score to each item [21]. In the case of a recommender system, it might be the item's expected rating or profit. Relevance is often referred to as benefit in this context. It could also be referred

to as the discounted cumulative relevance. The relevance is discounted by the position in the list, and the relevance scores are combined together to calculate the discounted cumulative gain [11] as shown in (7).

$$DCG = \sum_{rank=1}^{k} \frac{Gains}{log_2 (rank+1)} \quad (7)$$

where, Gains: The True_scores at each rank, which represent the actual relevance or quality of the recommendations. log2(rank + 1): The logarithm base 2 of the rank (adjusted by 1 to avoid division by zero). k is the rank at which we are evaluating the results. Once the result of DCG is obtained, NDCG, which is the Normalized DCG can be computed using the formula presented in (8).

$$NDCG = \frac{DCG}{IDCG} \quad (8)$$

Where IDCG is the Ideal DCG, which is the DCG that would be achieved if all the True_scores were perfectly ranked in descending order [11], [21], [22].

E. Results & Discussions

Upon dataset loading and removal of redundant columns, Table I displays selected columns such as titles, genres, and descriptions for a sample of five shows. Following the subsequent pre-processing workflow, processed tokens undergo transformation into vectors using a pre-trained Word2Vec model, as described in the implementation section. Utilizing these vectors and employing vector similarity methods facilitate the identification of the most similar words for all processed tokens within the Netflix dataset. Table II illustrates a representation of the most similar words identified by the Word2Vec model for a randomly chosen word, 'thriller.'

TABLE I. FEATURE SELECTION OF THE NETFLIX DATASET

Show_id	Title	Listed_in	Description
s1	Dick Johnson Is Dead	Documentaries	As her father nears the end of his life, filmmaker Kirsten Johnson stages his death in inventive and comical ways to help them both face the inevitable.
s2	Blood & Water	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town teen sets out to prove whether a private-school swimming star is her sister who was abducted at birth.
s3	Ganglands	Crime TV Shows, International TV Shows, TV Action & Adventure	To protect his family from a powerful drug lord, skilled thief Mehdi and his expert team of robbers are pulled into a violent and deadly turf war.
s4	Jailbirds New Orleans	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down among the incarcerated women at the Orleans Justice Center in New Orleans on this gritty reality series.
s5	Kota Factory	International TV Shows, Romantic TV Shows, TV Comedies	In a city of coaching centers known to train India's finest collegiate minds, an earnest but unexceptional student and his friends navigate campus life.

TABLE II. MOST SIMILAR WORDS GENERATED BY THE WORD2VEC (W2V) MODEL FOR THE WORD 'THRILLER.'

Sl. No.	Similar Word	Score
1	thriller	1.00000
2	thrillers	0.78241
3	suspense	0.74405
4	psychological	0.72550
5	spy	0.71650
6	supernatural	0.65463
7	drama	0.64413
8	potboiler	0.64344

After computing these scores for all the textual information within the features title, genres, and description, the ultimate score is derived by aggregating these individual scores. The Table III presents a list of top recommendations for the title 'Black Panther,'

F. Result Validation

Precision and Mean Average Precision (MAP) serve as evaluation metrics for our recommendation system, focusing on queries like "Black Panther" and "Article 15" (Table IV). MAP, averaging precision (AP) for both queries, yields 0.925, indicating the system's effectiveness in returning relevant results. A MAP score of 1 signifies perfection; thus, 0.925 is commendable. High precision across cutoffs is observed, though the "Black Panther" query's top 10 results show room for improvement with a precision drop to 0.8. Overall, the system excels in delivering relevant recommendations, with potential enhancements for specific queries.

TABLE III. SIMILARITY SCORES FOR TOP 10 RECOMMENDATIONS RELATED TO 'BLACK PANTHER'

Sl. No.	Recommendation	Score_Title	Score_ Category	Score_ Description	Final_ Score
1	Chappie	0.11271	1.00000	0.62195	1.73465
2	Clash of the Titans	0.04708	1.00000	0.67404	1.72112
3	Green Lantern	0.11550	1.00000	0.60476	1.72026
4	Halo: The Fall of Reach	0.05258	1.00000	0.65536	1.70794
5	Stargate	0.10318	1.00000	0.59298	1.69616
6	Illang: The Wolf Brigade	0.21976	0.90278	0.56643	1.68897
7	Men in Black	0.29745	0.88576	0.50308	1.68628
8	DC's Legends of Tomorrow	0.08603	0.90935	0.68981	1.68519
9	Superman Returns	0.09047	1.00000	0.59395	1.68442
10	Season of the Witch	0.12699	1.00000	0.55690	1.68389

TABLE IV. RELEVANCE METRICS FOR SAMPLE QUERIES

	Precision@k				Average Precision@k			
Query	P @1	P @3	P @5	P@ 10	AP @1	AP @3	AP @5	AP@ 10
Black Panther	1	1	1	0.8	1	1	1	0.95
Article 15	1	1	1	0.6	1	1	1	0.9

In the context of our content-based recommender system, the model is executed on an A/B test as well to gauge its performance and its impact on user satisfaction. Our primary aim was to determine whether the content-based recommender system led to significant enhancements in user engagement and overall satisfaction in comparison to the previous recommendation method. Users in the Control Group were retained under the existing recommendation system, while those in the Experimental Group were exposed to our newly developed recommender system. Our analysis revealed a notable increase in Click-Through Rate (CTR) among users in the Experimental Group over the course of one week, providing strong evidence for the effectiveness of our recommender system.

V. CONCLUSION

To summarize, our research paper has detailed creation and deployment of an advanced movie recommendation system designed for Netflix users. By leveraging cutting-edge techniques, including Content-Based Filtering and Word2Vec embeddings, to provide users with a powerful tool for discovering movies that closely match their preferences, thereby enriching their streaming experience on the Netflix platform. This research paves the way for ongoing exploration and enhancements in the realm of recommendation systems, catering to the diverse and ever-evolving tastes of Netflix's global user base. As a project culmination, introduced interactivity by developing a user-friendly interface, ensuring a delightful and engaging interaction with the recommendation system. The future scope of research work is potential for exploring hybrid recommendation models that integrate Content-Based filtering with Collaborative approaches, taking into account user interactions and preferences to provide recommendations that are both accurate and diverse. Additionally, the work can be considered implementing user feedback mechanisms that enable continuous improvement of recommendations through user ratings and reviews. The mechanism can be used for developing the job recommendation system from the LinkedIn profiles with description and profile comparison. These steps will contribute to further enhancing the effectiveness and relevance of our recommendation system.

REFERENCES

[1] Mishra, Nitin, Saumya Chaturvedi, Aanchal Vij, and Sunita Tripathi. "Research problems in recommender systems." In *Journal of Physics: Conference Series*, vol. 1717, no. 1, p. 012002. IOP Publishing, 2021.

[2] Xie, Xing, Jianxun Lian, Zheng Liu, Xiting Wang, Fangzhao Wu, Hongwei Wang, and Zhongxia Chen ,“ Personalized Recommendation Systems: Five Hot Research Topics You Must Know,” Microsoft Research Lab - Asia, Personalized Recommendation Systems: Five Hot Research Topics You Must Know, 2018.

[3] https://webengage.com/blog/10-product-recommendations-strategies-to-triple-your-e-commerce-sales/

[4] Kumaar, Hareesh, S. Srikumaran, and S. Veni. "Content-based Movie Recommender System Using Keywords and Plot Overview." In *2022 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, pp. 49-53. IEEE, 2022.

[5] Roy, Deepjyoti, and Mala Dutta. "A systematic review and research perspective on recommender systems." *Journal of Big Data* 9, no. 1 (2022): 59.

[6] Zhang, Jiang, Yufeng Wang, Zhiyuan Yuan, and Qun Jin. "Personalized real-time movie recommendation system: Practical prototype and evaluation." *Tsinghua Science and Technology* 25, no. 2 (2019): 180-191.

[7] Kumar, Dev. "Comparative study of movie recommendation system using feature engineering and improved error function." In *2022 International Conference on Futuristic Technologies (INCOFT)*, pp. 1-6. IEEE, 2022.

[8] Havolli, Abdullah, Arianit Maraj, and Lorik Fetahu. "Building a content-based recommendation engine model using Adamic Adar Measure; A Netflix case study." In *2022 11th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1-8. IEEE, 2022.

[9] Srivastava, Mehul. "YouTube and Movie Recommendation System Using Machine Learning." In *2023 Second International Conference on Electronics ad Renewable Systems (ICEARS)*, pp. 1352-1356. IEEE, 2023.

[10] Tai, Yifan, Zhenyu Sun, and Zixuan Yao. "Content-based recommendation using machine learning." In *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1-4. IEEE, 2021.

[11] Falk, Kim. *Practical recommender systems*. Simon and Schuster, 2019.

[12] S. Raza and C. Ding, "Progress in context-aware recommender systems—an overview," Computer Science Review, vol. 31, pp. 84– 97, 2019.

[13] H. F. Abdul karem, G. Y. Abozaid, and M. I. Soliman, "Context Aware recommender system frameworks, techniques, and applications: A survey," in 2019 International Conference on Innovative Trends in Computer Engineering (ITCE). IEEE, 2019, pp. 180–185.

[14] Bansal, Ashish. *Advanced Natural Language Processing with TensorFlow 2: Build effective real-world NLP applications using NER, RNNs, seq2seq models, Transformers, and more*. Packt Publishing Ltd, 2021.

[15] D. Valcarce, A. Landin, J. Parapar, and A. Barreiro, "Collaborative ´ filtering embeddings for memory-based recommender systems," Engineering Applications of Artificial Intelligence, vol. 85, pp. 347–356, 2019.

[16] Hagiwara, Masato. *Real-world natural language processing: practical applications with deep learning*. Simon and Schuster, 2021.

[17] Kaneko, Masahiro, and Danushka Bollegala. "Dictionary-based debiasing of pre-trained word embeddings." *arXiv preprint arXiv:2101.09525* (2021).

[18] Fkih, Fethi. "Similarity measures for Collaborative Filtering-based Recommender Systems: Review and experimental comparison." *Journal of King Saud University-Computer and Information Sciences* 34, no. 9 (2022): 7645-7669.

[19] Charu, C. Aggarwal. *Recommender systems: The textbook*. 2016.

[20] Banik, Rounak. *Hands-on recommendation systems with Python: start building powerful and personalized, recommendation engines with Python*. Packt Publishing Ltd, 2018.

[21] Li, Yang, Kangbo Liu, Ranjan Satapathy, Suhang Wang, and Erik Cambria. "Recent Developments in Recommender Systems: A Survey." *arXiv preprint arXiv:2306.12680* (2023).

[22] Y. Wang, L. Wang, Y. Li, D. He, W. Chen, and T.-Y. Liu, "A theoretical analysis of ndcg ranking measures," in Proceedings of the 26th annual conference on learning theory (COLT 2013), vol. 8, 2013, p. 6.