SEMARAK ILMU PUBLISHING

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Deep Learning Approach to Tweet Sentiment Analysis for Movie Recommendation Systems

Nisa Merissa[1] and Maslina Binti Zolkepli[1,*]

[1] Department of Computer Science, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, UPM Serdang, Selangor, Malaysia

ARTICLE INFO

Article history:
Received 22 November 2023
Received in revised form 22 February 2024
Accepted 15 August 2024
Available online 2 September 2024

Keywords:
Sentiment Analysis; Twitter; NLP; VADER; Deep Learning; Levenshtein Fuzzy String Matching

ABSTRACT

A deep learning approach to analyze the sentiment of user tweets is proposed to provide recommendations for movies and TV shows on streaming services. Recognizing the importance of implicit user feedback on social media, we focus on how viewers express their opinions on Twitter regarding drama series and films. To achieve this, we collect Twitter data using a Python library and REST API, then employ natural language processing techniques like TextBlob and VADER to extract show names and analyze sentiment. Our deep learning model demonstrates positive results with strong accuracy and minimal error rates. The proposed approach targets social media users who also subscribe to streaming services and opens potential applications in diverse domains like purchasing preferences, political opinions, and educational choices.

1. Introduction

In recent times, finding information or reviews on a given product is remarkably easy as people prefer to express themselves directly on social media platforms instead of specific service provider websites. This is due to the growing popularity of social media that increases the availability of user opinions that could influence others' preference in purchasing items, lifestyle choices, and opinions regarding current events such as movies, television shows, drama series, and many more. Social media platforms such as Twitter, YouTube, Instagram, and TikTok have an endless supply of user-generated opinions, mentions, and hashtags on topics that can be acquired to give a clear picture of the user's stand and preferences.

A movie recommendation system based on Twitter users' expressed interest in various shows via tweets using the deep learning method is proposed because a vast majority of existing recommendation studies do not focus on people's opinions on social media. Recommendation approach using content-based filtering methods uses item information to determine the similarities between objects, expressed as characteristics. Based on item descriptions and user preferences, content-based filtering algorithms propose appropriate items to users. Furthermore, content-based

* *Corresponding author.*
E-mail address: masz@upm.edu.my

https://doi.org/10.37934/araset.51.1.245257

filtering algorithms use only the active user's profile information or ratings to provide accurate suggestions even if other users' ratings are insufficient [1]. Another recommendation system approach is to apply collaborative filtering, where it primarily focuses on the user's rating of products [2-6]. Collaborative filtering makes product recommendations based on users' interests using two approaches, which are item-based recommendation and user-based recommendation. Item-based recommendation is a recommendation approach that searches for comparable products based on those consumers have already shown interest in or have had favorable interactions with, while user-based recommendation determines whether two users are comparable based on what they have purchased or reviewed. A hybrid recommendation strategy [7,8] introduced a new architecture for visitor recommendation systems to enhance user access to tourist resources in tourism portals. Instead of just recommending a list of tourism destinations based on the interests of the visitors, it also recommends the most relevant things and assists users in personalizing their journey.

In recommendation systems research, sentiment analysis is frequently utilized to produce recommendations where it represents a natural language processing task that attempts to extract the authors' sentiments from a piece of text [9-13]. TextBlob approach for sentiment analysis [14] provides parsed user tweets and produces the sentiment score of tweets by analyzing them using a rule-based approach. TextBlob is a Python package that provides API access to various natural language processing operations, including sentiment analysis and typo correction. TextBlob strategy is tested and proven in various settings, including movie reviews, e-commerce product reviews, and news headlines [15].

Valence aware dictionary for sentiment reasoning or VADER [16,17] is a sentiment analyzer that calculates text sentiment using a collection of linguistic features. Depending on their semantic orientation, words can be classified as positive or negative. The VADER method produced superior results than other sentiment analysis tools. Deep learning-based recommender systems differ from traditional ones in that they can deal with complicated interaction patterns and accurately reflect the user's preferences. As social media has become a vital aspect of daily life, users can now express themselves online, such as on Twitter. Twitter data, known as tweets, is beneficial since it comes straight from the user and displays their direct interest in the item.

Despite the prevalence of existing recommendation systems, effectively capturing and utilizing user sentiment on social media for personalized movie and TV show recommendations remains a challenge. This study addresses this gap by proposing a deep learning-based approach that analyzes Twitter data to unlock personalized content suggestions based on user preferences and emotions, thereby potentially revolutionizing the way users discover and enjoy entertainment on streaming platforms.

2. Methodology

2.1 Twitter, IMDB and Netflix Dataset

Three types of databases are required for the proposed recommender system. The first is user tweets from Twitter, followed by the IMDB database, which contains ratings for related movies and television shows. Finally, the Netflix shows database, from which the system will recommend movies and television shows. To collect users' tweets from Twitter, the dataset is created from social media data Twitter uses REST API. Different keywords are used for each genre and collected directly into the database. The data contains four collections for each genre as Comedy, Romance, Horror, and Action. In addition, because Twitter has unstructured data, the ratings and program lists for all current Netflix shows are gathered using the exact names of the shows. The snapshot of the extracted words from Twitter is shown in Figure 1.

user_id	user_name	text
72627052	TrivWorks	SNL
14970333	Monsters and Critics	Netflix
113123861	Niru	BlackMoneyLove
1315197590892404700	Anar	Terter
2242846435	Phytoversity	SOTWPF
713396467898183700	TV Cancel Beast	ElfOnTheShelf
1033917030058156000	Tales from the Vi...	action
72631996	IRS #COVIDreliefIRS	IRS
367589500	Shah Shahid	AvatarTheLastAirb...

Fig. 1. Extracted words from Twitter using Twitter REST API

For IMDB data collection, rating records from IMDB are collected using web scraping with python of each show extracted from tweets. For Netflix show data, ratings and television show lists of all current Netflix shows are collected using the exact names of the shows from Kaggle [18].

2.2 TextBlob, Vader and Deep Learning Based Sentiment Analysis

The research intends to propose a deep learning-based recommender system that uses social media data to recommend movies or shows on Netflix, based on users' sentiment where two algorithms will be used, one is a neural network approach by using a single layer perceptron which uses a matrix factorization with a singular neural network and the other one is a deep learning approach that uses a multi-layer perceptron with matrix factorization and multiple neural networks. Figure 2 shows the main steps of the sentiment analysis process flow.

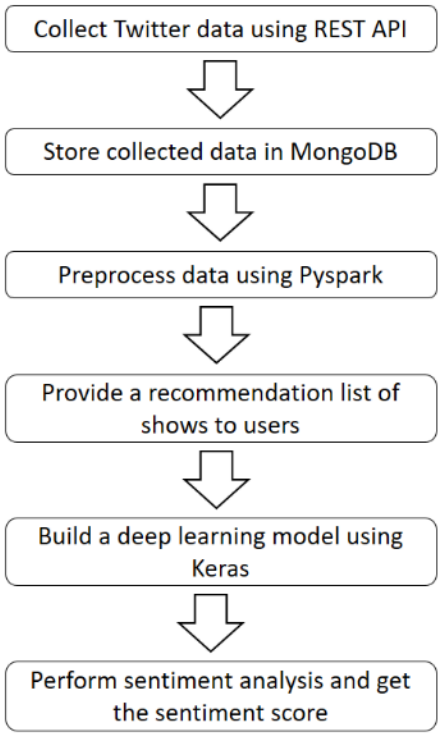

Fig. 2. Sentiment Analysis on Users' Tweet for Recommendation Systems' Flow

After storing the retrieved tweets in MongoDB and preprocessing them, the sentiments of these tweets are analyzed using TextBlob and Valence aware dictionary for sentiment reasoning (VADER). For a given input sentence, the TextBlob sentiment analyzer returns two properties. The first is polarity, a float that ranges from [-1,1], with -1 denoting negative sentiment and +1 denoting positive sentiment. The second one is subjectivity, a float that ranges from [0,1]. Personal opinion, emotion, or judgment are all examples of subjective sentences. The text blob sentiment method has two components which are Polarity and Subjectivity. Figure 3 shows the framework for the recommendation approach based on sentiment analysis using TextBlob and VADER.

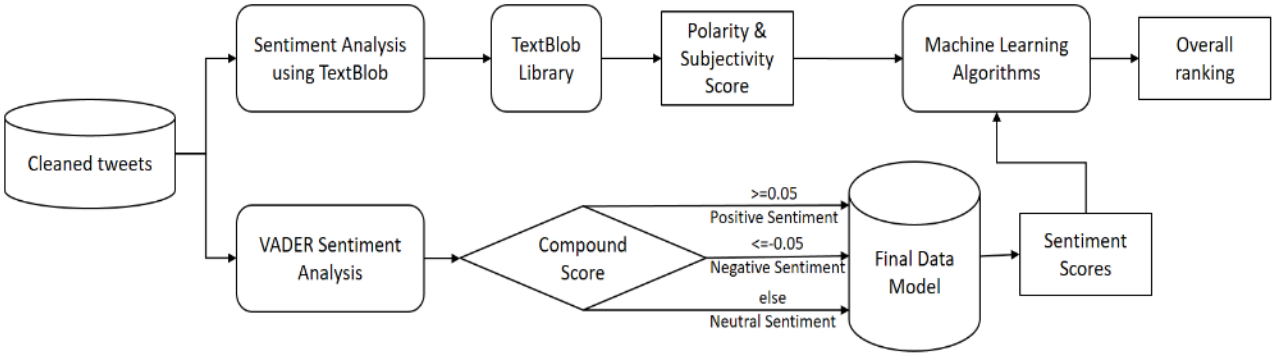

Fig. 3. Framework for recommendation approach based on sentiment analysis using TextBlob and VADER

TextBlob analyzes the text data and assigns scores to rate it positive or negative sentiment. The polarity value lies between -1 and +1, where -1 means negative sentiment and one means positive sentiment. Subjectivity checks if the text data is a public or factual statement. Subjectivity score lies between [0,1], where a higher value means public opinion and a lower value means factual statement. Sentiment matching is accomplished by calculating the polarity of a text and labelling it as negative if the result is less than 0, positive if the result is more than 0, and neutral if the result is 0.

VADER sentiment returns the likelihood that a given input sentence will be Positive, Negative, or Neutral. It is a dictionary-based approach that maps each word with its emotion by building a lexicon. In this process, the subjectivity of the text data is handled by the sentiment process itself. VADER sentiment analysis has four components, i.e., positive, negative, neutral, and compound scores. The compound score value is between [-1,1], -1 means least preferred and +1 means most preferred. Text data sentiment for VADER is positive if the compound score is greater than or equal to 0.05, and negative if the compound score is less than or equal to -0.05. It is a neutral sentiment if the sentiment score is neither of the above. A compound score of 0.05 is a typical threshold value [19].

Both sentiment analysis approaches will produce sentiment scores that will become the input for the machine learning algorithms to produce an overall ranking of the films and shows, as shown in Figure 3.

After TextBlob and VADER sentiment analysis are performed on users' tweets, two machine learning techniques are applied to the sentiment analysis result to compare the performance of each technique. The first one is a single layer perceptron, and the other one is multi-layer perceptron [20], which is a deep learning technique. Learning curves to show the progress of each machine learning model are produced at the end of the sentiment analysis.

3. Results

3.1 Experiment on User's Tweets for Sentiment Analysis

A dataset of 160,000 tweets was collected to evaluate the performance of both VADER and TextBlob in doing sentiment analysis on microblogs, with 40,000 tweets for each genre. The Twitter streaming API is used with Tweepy, a Python library, to extract the tweets. Tweepy [21] has a user-friendly interface for iterating object types and can extract up to 3,200 tweets at once. The retrieved tweets will then be loaded into MongoDB database for data preprocessing.

3.1.1 Data preprocessing

The first step of data preprocessing is data stored in MongoDB as four genres are read into spark data frames to be cleaned and combined, as shown in Figure 4.

Tweets are scraped data containing a tremendous amount of noise such as repetitive words and other irrelevant data, one of which is stop words. Aside from stop words, non-informative parts of tweets, such as special characters, punctuation, capitalization, web links, and redundant words, are also removed as they did not significantly improve the outcome of the sentiment analysis. Table 1 shows some examples of unusable parts of a tweet.

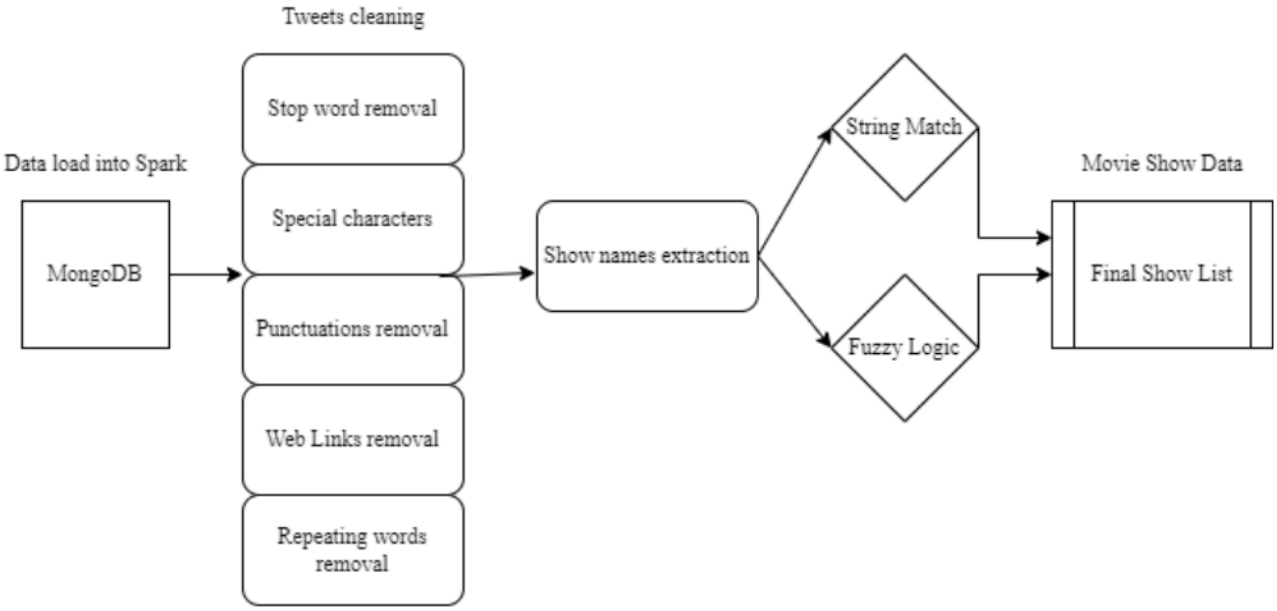

Fig. 4. Twitter data preprocessing phase

Table 1
Examples of unusable parts in a tweet

Types of noise	Examples
Stop words	be, of, the, being, am
Special characters	!, @, #, $
Punctuations	Quotation marks (” ” & ' '), Commas
Weblinks	(,), Colon (:), Semicolon (;)
Repeating words	www.imdb.com
Types of noise	helooooo, happyyyyy
Stop words	Examples
Special characters	be, of, the, being, am

The next part of data preprocessing is word extraction. Regex is used to extract show names from tweets. Words starting with hashtags, quotes, and capital letters were extracted. Actual titles from Netflix CSV are also read, and string matching is done to get exact show names. Before performing the string-matching method, the extracted words from tweets and titles from Netflix CSV are cleaned to increase the performance of the string-matching method.

Once such words were extracted for show mapping, string matching and Levenshtein fuzzy string matching [22] methods are used. A full string matching is done to find an exact match between extracted words and actual show titles collected from the dataset. Levenshtein fuzzy string matching is applied for words that did not pass an exact match test.

String Matching: The technique of identifying strings that roughly fit a pattern is known as string matching. In this project, the actual show titles and the first eight characters of extracted words from tweets were substrings to find a partial match. The first eight characters were substring to find a partial match, and the reason the first eight characters are chosen is that the average length of the movie title from the year 2001-2010 is 15.83 [22]. To find a partial match for the movie title, the average length is divided into two equals 7.91 and 8 after being rounded off.

Levenshtein fuzzy string matching: This method matches the extracted words and actual titles and gives a score based on the similarity of strings. The lower the score, the higher the similarity of the string. A match score of less than 16 is selected because the average length of the movie title from the year 2001-2010 is 15.83 [22]. Levenshtein fuzzy string matching is done to the words that were unsuccessfully matched in the direct matching method. The Levenshtein distance between two strings a, b (of length $|a|$ and $|b|$ respectively) is given by $lev(a,b)$ where

$$lev(a,b) = \begin{cases} |a| & if\ |b| = 0, \\ |b| & if\ |a| = 0; \\ lev\big(tail(a), tail(b)\big) & if\ a|0| = b|0| \\ 1 + \min\begin{cases} lev(tail(a), b) \\ lev\big(a, tail(b)\big) \\ lev\big(tail(a), tail(b)\big) \end{cases} & otherwise. \end{cases} \quad (1)$$

Since two methods are applied for show extraction, any potential good data like fuzzy string matching are not excluded, which is a partial matching method. There are chances that the tweets can be unrelated to the matched names in a few cases. Such data is included to have adequate data to perform the rest of the project and, most importantly, build a deep learning model.

As far as the subjectivity of the tweets is concerned, after processing the data using natural language processing approach, filtering it using Regex, and cleaning it, bad data is removed. However, it is impossible to eliminate all insufficient data from the dataset, where the subjectivity of the data is crucial to predict the sentiment of the user-item interaction, which is mainly based on the user's reaction towards a specific topic.

There is no apparent subjectivity from the tweets mentioned as the data was collected by matching a list of show names. Nevertheless, the proposed approach considered it a tweet about the show, and sentiment analysis was performed because the main idea was to build a recommendation system based on sentiment score. Table 2 shows the result of the cleaned tweets for sentiment analysis.

Table 2
Cleaned tweets using string matching and Levenshtein fuzzy string matching

User ID	Twitter Text	Cleaned Text
8870241181 284802600	"I love that ""Ash vs the Evil Dead"" is a mixture of expertly crafted horror and well written comedy." Definitely reco...	I love that ash vs the evil dead is a mixture of expertly crafted horror and well written comedy definitely reco…
212220703	Still trying to figure out whether the new restrictions mean I have to wear a mask while moving around school tomor...	Still trying to figure out whether the new restrictions mean i have to wear a mask while moving around school tomor
2675216486	Looking sharp Henry! Can't wait to see you back in action #TheWitche r #Netflix #HenryCavill #GeraltOfRi via...	looking sharp henry cant wait to see you back in action thewitcher netflix henrycavill geraltofrivia
19084114	"@brightlightx2 @letterboxd ILOVE scary movies! Do you use the @Shudder app??? The ""Netflix""" for horror movies! So fun. to run jobs on the street so he doesn't get taken out by debt collector agents until he starts fighting against the...	i love scary movies do you use the app the netflix for horror movies so fun to run jobs on the street so he doesnt get taken out by debt collector agents until he starts fighting against the

3.1.2 Sentiment analysis of user's tweet

After preprocessing the data, the next step is to perform sentiment analysis of the tweets. Sentiment analysis is analyzing the underlying emotion of the text posted by the user. VADER method is applied in the sentiment analysis of users' tweets, and the sentiment score from this method will be used in recommending users' Netflix shows. The VADER algorithm created a lexicon with a complete set of sentiment characteristics. It takes into account the emoticons used in the Western world, such as :-) represents a smiley face and is used to present joyful emotions, abbreviations for emotions (e.g., LOL and OMG are both illustrative of sentimental initialisms), and the lingo of sentimental value commonly utilized (e.g., nah, meh and giggly). VADER is a lexical dictionary created with social media in mind, and it contains the intensity values of colloquial and online slang. Table 3 shows the polarity and subjectivity score from TextBlob sentiment analysis and the sentiment score from VADER sentiment analysis.

Table 3
Polarity and subjectivity score from TextBlob & Sentiment score from VADER

User ID	Title	Cleaned Text	Genre	Polarity & Subjectivity Score	Sentiment Score
8870241181 284802600	Love	I love that ash vs the evil dead is a mixture of expertly crafted horror and well written comedy definitely reco…	Romance	(-0.175, 0.625)	0.327
212220703	Still	Still trying to figure out whether the new restrictions mean i have to wear a mask while moving around school tomor	Horror	(-0.088, 0.571)	1.000
2675216486	The Witcher	looking sharp henry cant wait to see you back in action thewitcher netflix henrycavill geraltofrivia	Action	(-0.0187, 0.283)	1.000
19084114	Love	i love scary movies do you use the app the netflix for horror movies so	Romance	(0.006,0.6)	0.292

		fun to run jobs on the street so he doesnt get taken out by debt collector agents until he starts fighting against the					

3.2 Sentiment Analysis Results and Discussions

Figure 5 shows the distribution graph of sentiment scores in the training data. It is clear from the plot that a sentiment score of 1 is highly positive scores are more in number compared to other scores. Fewer positive scores such as 0 and 0.2 are also present in a significant amount.

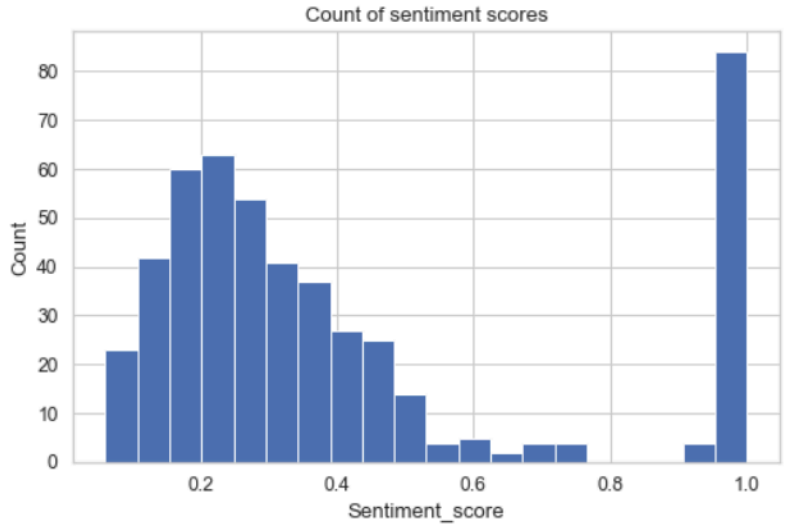

Fig. 5. Count of Sentiment Scores Graph

Figure 6 shows the distribution graph of IMDB ratings in the dataset. Ratings are plotted after normalization, and hence the range is from 0 to 1, where 0 indicates poor rating and 1 indicates a very high rating. We could find that most shows had an average rating, and a smaller number of movies had a very high rating.

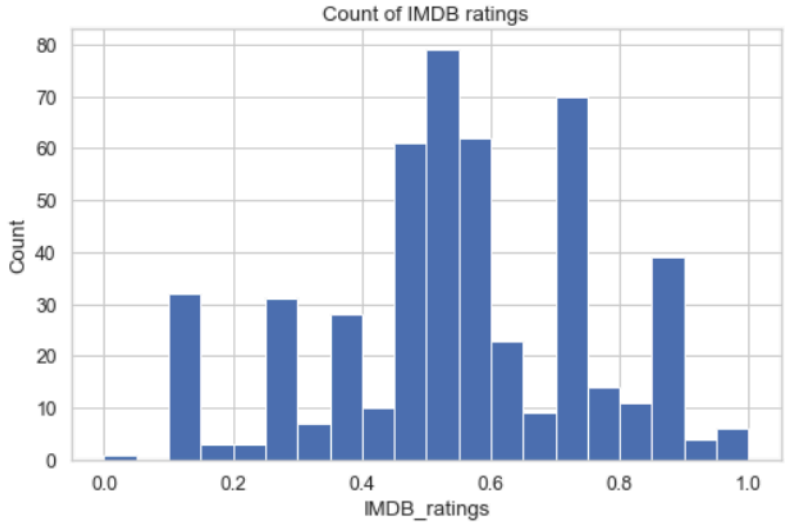

Fig. 6. Count of IMDB Ratings Graph

Figure 7 shows the distribution graph of four genres in the dataset, namely Action, Comedy, Horror, and Romance. Most movies are of the genre Action, and there are a very small number of movies in the Comedy genre.

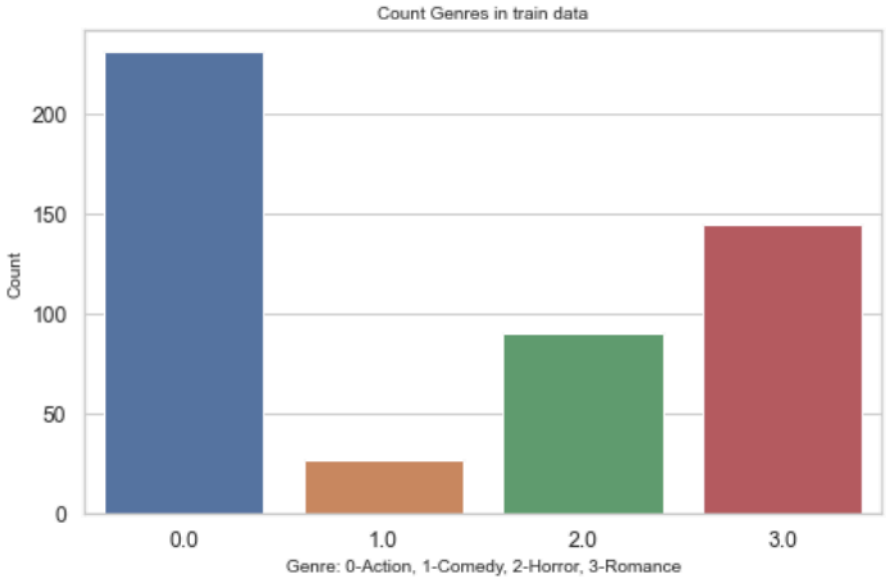

Fig. 7. Count Genres in Train Data Graph

3.2.1 Single-layer model

A single-layer fully connected model with a dropout of 50%. Show ids, user ids, and ratings are embedded to create vectors are built. The show vector and user vector are multiplied to create a user-item interaction matrix. The model is then compiled and evaluated.

A graph that plots the model loss is shown in Figure 8. It shows that the training loss is high compared to the test loss which means the model is doing a good job of reducing test loss. However, the difference between train and test loss is large, indicating that the model is overfitting.

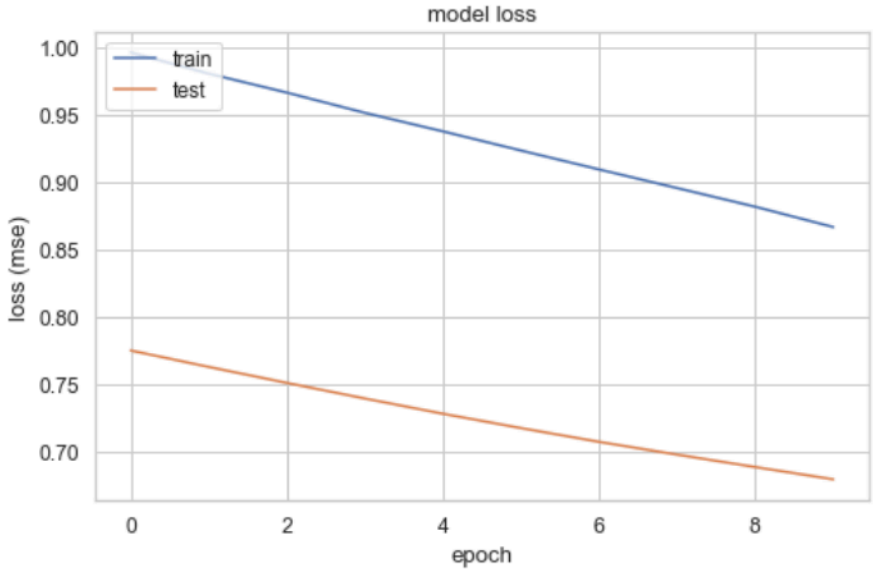

Fig. 8. Single-layer Model Loss Graph

3.2.2 Multi-layer perceptron deep learning model

A deep learning MLP model is created using Keras [23]. Keras is an open-source software library for artificial neural networks that include a Python interface. Show ids and user ids are vectorized by embedding with an embedding size of 30. The embedded vectors are then concatenated and fed into the model.

Figure 9 shows a graph that plots the model loss for the deep learning MLP model. The training and test processes' curves almost crossed each other. A good model as the training process stops when the validation error trend changes from descending to ascending [24-25]. Hence, the multi-layer perceptron model is better suited for the recommendation system as it performed better than the single-layer perceptron.

3.2.3 Recommendation of TV shows and movies to users

A Twitter user from the MLP model data is chosen for the recommendation. Table 4 shows the user's details from the analyzed data. User 12605932 tweeted, and from the natural language processing task, the extracted word from the tweet is matched to Troy's action movie in the Netflix database.

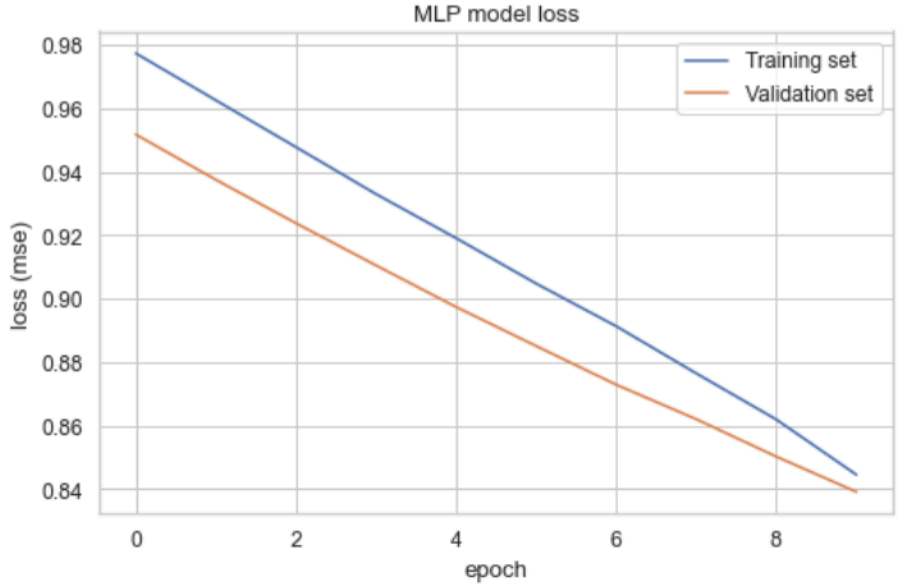

Fig. 9. Multi-Layer Perceptron Model Loss Graph

Table 4
Output sample from MLP model data of a twitter user

User ID	Title	Cleaned Text	Genre	Sentiment Score
12605932	Troy	TROY (2004) is on Netflix. \n\nI find this film to be... https://t.co/D4PrISRaiF	Action	0.122

For user 12605932, the user tweeted about the movie 'Troy', and the sentiment score for the tweet is 0.122, which the proposed approach tagged as positive. The model predicted a list of films and shows for this user as above. Among these suggestions, the model recommends the user seven action genre movies that are in the same category as the movie he tweeted about. Based on the

user's interaction and tweeted data, the user will find one of these recommendations useful. Table 5 shows the sentiment score results for Action movies based on the user 12605932's tweet.

Table 5
Sentiment score for movies based on users' tweets

Title	IMDB rating	Genre	Sentiment score
Tomorrow Never Dies	6.5	0	0.074
Remember	7.5	0	0.115
Happy!	8.2	0	0.128
The Devil Inside	4.2	2	0.130

To acquire a better value, the multi-layer perceptron can be evaluated with different dropouts, more training epochs, and more hidden layers. It is reasonable to state that deep learning algorithms have performed better than traditional machine learning algorithms. To present the recommendation results to viewers, a dashboard is proposed to display the results to the target users as they can see the movie recommendation list based on the tweets. Figure 10 shows the dashboard that displays movies recommendations based on the user's tweets.

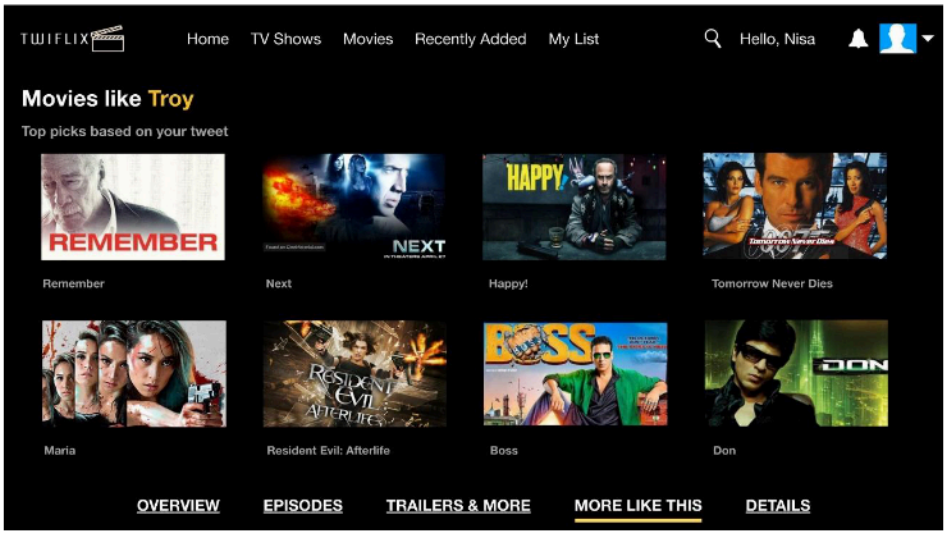

Fig. 10. Tweet-based Movie Recommendation Dashboard

4. Conclusion

As demonstrated by our proposed deep learning movie recommendation system, analyzing audience sentiment on Twitter holds significant promise for personalizing streaming service recommendations. By leveraging multi-layer perceptron to outperform traditional methods, we successfully generated recommendations based on users' tweet sentiment scores. A deep learning movie recommendation system is proposed where Twitter data is analyzed using sentiment analysis to provide information about how the audience responds to a particular movie or a TV show. The experiment result shows that the deep learning Multi-Layer Perceptron performs better than the traditional machine learning algorithm, Single Layer Perceptron. The recommender system provides recommendations to users based on their tweet sentiment score. There are several inconsistencies in the proposed approach, such as partial string matching than can be improved to produce better recommendation results by collecting more specific tweets about movies and TV shows. Since Twitter contains many user-specific unstructured data, the integrity of the time series data extracted from tweets can also be further analyzed to increase the accuracy of the recommendation result.

The proposed approach focused on the social media platform Twitter with promising results. Future works should look into user opinions created on other popular platforms such as Facebook, Instagram, and TikTok. The target users of the proposed recommendation are social media platform users who also subscribe to streaming services. The recommendation system approach based on sentiment analysis can be further applied in other domains such as e-commerce systems, music preferences, political preferences, education preferences, and many more.

Acknowledgement
This research is not funded by any grant.

References

[1] Son, Jieun, and Seoung Bum Kim. "Content-based filtering for recommendation systems using multiattribute networks." *Expert Systems with Applications*, vol. 89 (2017): 404–412. https://doi.org/10.1016/j.eswa.2017.08.008
[2] Anantha, N. L., and B. P. Bathula. "Comparative study on traditional recommender systems and deep learning-based recommender systems." *Advances in Modelling and Analysis B*, vol. 61, no. 2 (2018): 64–69. https://doi.org/10.18280/ama_b.610202
[3] 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. https://doi.org/10.1016/j.jksuci.2021.09.014
[4] He, Xiangnan, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. "Neural collaborative filtering." In *Proceedings of the 26th international conference on world wide web*, pp. 173-182. 2017. https://doi.org/10.1145/3038912.3052569
[5] Ullah, Farhan, Bofeng Zhang, Rehan Ullah Khan, Tae-Sun Chung, Muhammad Attique, Khalil Khan, Salim El Khediri, and Sadeeq Jan. "Deep edu: a deep neural collaborative filtering for educational services recommendation." *IEEE access* 8 (2020): 110915-110928. https://doi.org/10.1109/ACCESS.2020.3002544
[6] Rendle, Steffen, Walid Krichene, Li Zhang, and John Anderson. "Neural collaborative filtering vs. matrix factorization revisited." In *Proceedings of the 14th ACM Conference on Recommender Systems*, pp. 240-248. 2020. https://doi.org/10.1145/3383313.3412488
[7] Al Fararni, Khalid, Fouad Nafis, Badraddine Aghoutane, Ali Yahyaouy, Jamal Riffi, and Abdelouahed Sabri. "Hybrid recommender system for tourism based on big data and AI: A conceptual framework." *Big Data Mining and Analytics* 4, no. 1 (2021): 47–55. https://doi.org/10.26599/BDMA.2020.9020015
[8] Kbaier, Mohamed Elyes Ben Haj, Hela Masri, and Saoussen Krichen. "A personalized hybrid tourism recommender system." In *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, pp. 244-250. Ieee, 2017. https://doi.org/10.1109/AICCSA.2017.12
[9] Trampuš, Mitja, Flavio Fuart, Daniele Pighin, Tadej Štajner, Jan Berčič, Blaz Novak, Delia Rusu, Luka Stopar, and Marko Grobelnik. "DiversiNews: Surfacing Diversity in Online News." *AI Magazine* 36, no. 4 (2015): 87–104. https://doi.org/10.1609/aimag.v36i4.2528
[10] Kakulapati, V., D. Vasumathi, and G. Suryanarayana. "User-Query Processing through Dynamic Tweets Status Recommender System." *PSYCHOLOGY AND EDUCATION* 58, no. 1 (2021): 5600-5606. https://doi.org/10.17762/pae.v58i1.2180
[11] Kouki, Pigi, James Schaffer, Jay Pujara, John O'Donovan, and Lise Getoor. "User preferences for hybrid explanations." In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pp. 84-88. 2017. https://doi.org/10.1145/3109859.3109915
[12] Kumar, Sudhanshu, Kanjar De, and Partha Pratim Roy. "Movie Recommendation System Using Sentiment Analysis from Microblogging Data." *IEEE Transactions on Computational Social Systems* 7, no. 4 (2020): 915–923. https://doi.org/10.1109/TCSS.2020.2993585
[13] Selmene, Safa, and Zahra Kodia. "Recommender System Based on User's Tweets Sentiment Analysis." In *Proceedings of the 4th International Conference on E-Commerce, E-Business and E-Government*, pp. 96-102. 2020. https://doi.org/10.1145/3409929.3414744
[14] Rakshitha, Kakuthota, H. M. Ramalingam, M. Pavithra, H. D. Advi, and Maithri Hegde. "Sentimental analysis of Indian regional languages on social media." *Global Transitions Proceedings* 2, no. 2 (2021): 414-420. https://doi.org/10.1016/j.gltp.2021.08.039

[15] Bonta, V., Kumaresh, N., & Janardhan, N. (2019). A comprehensive study on lexicon based approaches for sentiment analysis. *Asian Journal of Computer Science and Technology*, *8*(S2), 1-6. https://doi.org/10.51983/ajcst-2019.8.S2.2037

[16] Elbagir, Shihab, and Jing Yang. "Twitter sentiment analysis using natural language toolkit and VADER sentiment." In *Proceedings of the international multiconference of engineers and computer scientists*, vol. 122, p. 16. 2019. https://doi.org/10.1142/9789811215094_0005

[17] Adarsh, R., Ashwin Patil, Shubham Rayar, and K. M. Veena. "Comparison of VADER and LSTM for sentiment analysis." *International Journal of Recent Technology and Engineering* 7, no. 6 (2019): 540-543.

[18] Bennett, James, and Stan Lanning. "The netflix prize." In *Proceedings of KDD cup and workshop*, vol. 2007, p. 35. 2007. https://doi.org/10.1145/1345448.1345459

[19] Elbagir, Shihab, and Jing Yang. "Sentiment analysis on Twitter with Python's natural language toolkit and VADER sentiment analyzer." In *IAENG Transactions on Engineering Sciences: Special Issue for the International Association of Engineers Conferences 2019*, pp. 63-80. 2020. https://doi.org/10.1142/9789811215094_0005

[20] Narayan, S. "Multilayer Perceptron with Autoencoder enabled Deep Learning model for recommender Systems." *Future Computing and Informatics Journal.* (2020). https://doi.org/10.54623/fue.fcij.5.2.3

[21] Octaria, Orissa, Danny Manongga, Ade Iriani, Hindriyanto Dwi Purnomo, and Iwan Setyawan. "Mining Opinion Based on Tweets about Student Exchange with Tweepy and TextBlob." In *2022 9th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pp. 102-106. IEEE, 2022. https://doi.org/10.1109/ICITACEE55701.2022.9924013

[22] Yujian, Li, and Liu Bo. "A Normalized Levenshtein Distance Metric." In *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, no. 6 (2007): 1091-1095. https://doi.org/10.1109/TPAMI.2007.1078

[23] Hall, Patrick AV, and Geoff R. Dowling. "Approximate string matching." *ACM computing surveys (CSUR)* 12, no. 4 (1980): 381-402. https://doi.org/10.1145/356827.356830

[24] Kandel, Sean, Ravi Parikh, Andreas Paepcke, Joseph M. Hellerstein, and Jeffrey Heer. "Profiler: Integrated statistical analysis and visualization for data quality assessment." In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 547-554. 2012. https://doi.org/10.1145/2254556.2254659

[25] Anzanello, Michel Jose, and Flavio Sanson Fogliatto. "Learning curve models and applications: Literature review and research directions." *International Journal of Industrial Ergonomics* 41, no. 5 (2011): 573-583. https://doi.org/10.1016/j.ergon.2011.05.001