Received: December 23, 2023. Revised: January 15, 2024.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Movie Recommendation System Based on Tweets Using Switching Hybrid Filtering with Recurrent Neural Network

Berlian Muhammad Galin Al Awienoor[1] **Erwin Budi Setiawan[1]***

[1]*Informatics, School of Computing, Telkom University, Bandung, Indonesia*
* Corresponding author's Email: erwinbudisetiawan@telkomuniversity.ac.id

Abstract: In the current phase of technological development, Netflix has become a popular platform for entertainment. Often people feel overwhelmed when choosing a movie because of the variety of genres. To overcome this problem, a recommendation system is needed that can help people find the best movie according to their preferences. In addition, Twitter was used to collect tweets related to movies, which were then processed into rating values. The crawled dataset consisted of 855 movies and 44 user reviews (including data from IMDB, rotten tomatoes, and metacritic websites), for a total of 23,130 records. This research proposed to use the switching hybrid filtering (SHF) method combined with recurrent neural network (RNN) as classification. In SHF, the emphasis on rating prediction was initiated by the content based filtering method with RoBERTa, followed by switching to item-based collaborative filtering. This situation arose because the dataset had a sparseness of 74.46%. SHF provided accurate rating prediction with an MAE of 0.0617 and an RMSE of 0.1178. Nadam optimization with optimal learning rate in RNN classification gave the best results with an accuracy of 86.11%. The research successfully developed a method that proved effective and provided positive results, contributing to the development of a recommendation system designed to assist users in choosing movies based on their preferences.

Keywords: Recommender system, Switching hybrid filtering, Collaborative filtering, Content-based filtering, Recurrent neural network.

1. Introduction

In today's digital age, Twitter has become a popular platform for expressing movie-related opinions, reviews, and recommendations. In fact, Twitter is the most popular social microblogging service with over 336 million monthly active users and more than 500 million tweets posted daily [1]. Besides, Netflix is one of the leading platforms for enjoying movies of various genres. The availability of many movies often overwhelms users to choose a preferred movie. This is due to the many options to choose a movie including a large variety of genres, themes, actors, directors, and user reviews. Sometimes, users find it difficult to decide which movie they want to watch and get frustrated. For example, for watching Netflix movies, users may have to watch a lot of trailers before finding an interesting movie, which is a time-consuming process [2]. To solve this problem, recommendation systems are needed to find movies relevant to their preferences. Generally, recommendation systems predict personalized user preferences and identify items with the highest preferences based on previous history [3]. This research will utilize a dataset generated by crawling tweet data related to movies on Netflix.

Recommendation systems offer various model categories, including collaborative filtering (CF), content-based filtering (CBF), and hybrid filtering (HF) recommender system [3-8]. CF recommends items by identifying other users with similar preferences and using their opinions for recommendations [4, 9], while CBF recommends items based on item content information [6, 7, 10, 11]. HF is a combined single recommendation system as a sub-component [6]. HF comes to overcome some problems by using only one type of

recommendation system method. CF faces problems such as data sparsity where many users rate only a few items, so there are many empty matrices, and cold start when there are new users who have not rated [4]. CBF faces problems in providing unique recommendations because it focuses more on the similarity of features with items that have received high ratings from users.

Among the various approaches within the HF method, the method chosen for this research is switching hybrid filtering (SHF). SHF allows a system to change its method if one method proves inadequate in providing recommendations [7]. For example, if CF does not provide enough credible recommendations, the system can switch to CBF [8]. This can help system overcome the weaknesses and maximize the strengths of each method [10].

In this research, recurrent neural network (RNN) is integrated to be combined with SHF. Deep learning (DL) is becoming popular in recommendation systems because it provides high-quality performance and recommendations [13, 14]. In addition, DL is a method capable of extracting and learning complex relationships between users and items [15]. RNN is part of the DL domain, where a series of NN layers process sequentially arranged inputs [14]. RNN iteratively does the same task for each layer of sequence, with the result based on the previous calculation [15], to model the contextual relationships between elements in the input sequence [15]. RNN is used to consider both item and sequence information, allowing the model to recognize sequence patterns in the data [15]. This allows RNN to provide more personalized recommendations that match the user's preference.

This research proposes a recommendation system using the SHF method combined with RNN as classification method to improve performance and accuracy in predicting rating values. As far as we know, there is no research that combines two methods to develop a recommendation system. In addition, the utilization of various features, such as feature extraction and semantic word embedding, as well as optimization are also considered. The motivation of this research is to improve movie rating prediction by taking advantage of the combination of two methods. The main intent of developing this system is twofold. First, SHF is built to overcome the problem of sparse data and reduce the weaknesses of the CF and CBF methods. Secondly, a classification process using RNN is performed to improve the prediction accuracy of SHF and provide appropriate recommendations. By combining SHF with RNN classification, the developed recommendation system is expected to effectively provide more accurate and relevant recommendations based on user preferences and achieve better performance.

This paper is structured in several sections. Section 2 summarizes related studies that discuss similar research. Section 3 presents the proposed methodology, focusing on the development of a recommendation system with SHF and a classification model with RNN. Section 4 presents the results and discussion. Section 5 summarizes the conclusions of this research.

2. Literature review

This research was conducted according to several references from previous studies that are relevant to the methods used. The use of references aims to increase knowledge and as a guide in this research.

Research by M. Ali, et al., [7] introduced hybrid switching, which aims to improve the effectiveness of recommendation systems by incorporating Naive Bayes and support vector machine into collaborative filtering (CF). The focus is on process switching between predictions made by machine learning classifiers and collaborative filtering predictions based on confidence measures. SwitchRecNBCF uses NB predictions when confidence is high. This approach shows an MAE reduction of 8.33% compared to the simple NBCF average. Meanwhile, SwitchRecSVMCF, which uses SVM and CF methods, results in a 7.64% reduction in MAE. The developed switching hybrid exhibited lower MAE and larger coverage compared to the CF method and machine learning classification approaches that were performed independently.

Tuyet-Van T. T., and Thanh-Nhan H. L., in research [18], performed a hybrid switching approach using the Movielens dataset. The results of the research are the value of MAE of 0.73 and RMSE of 0.93, indicating that the hybrid switching method is able to provide more accurate predictions compared to user-based or item-based methods. The proposed method refers to increasing the accuracy of matrices that have 0 entries, so as to improve the system efficiency and provide more accurate prediction results. Experimental results showed that the hybrid switch method outperformed other traditional methods in terms of prediction accuracy, as indicated by lower MAE and RMSE values compared to using user-based or item-based approaches.

In research [17] Tim Donkers, et al., explained that RNN is a powerful method of modeling sequences that can combine different types of

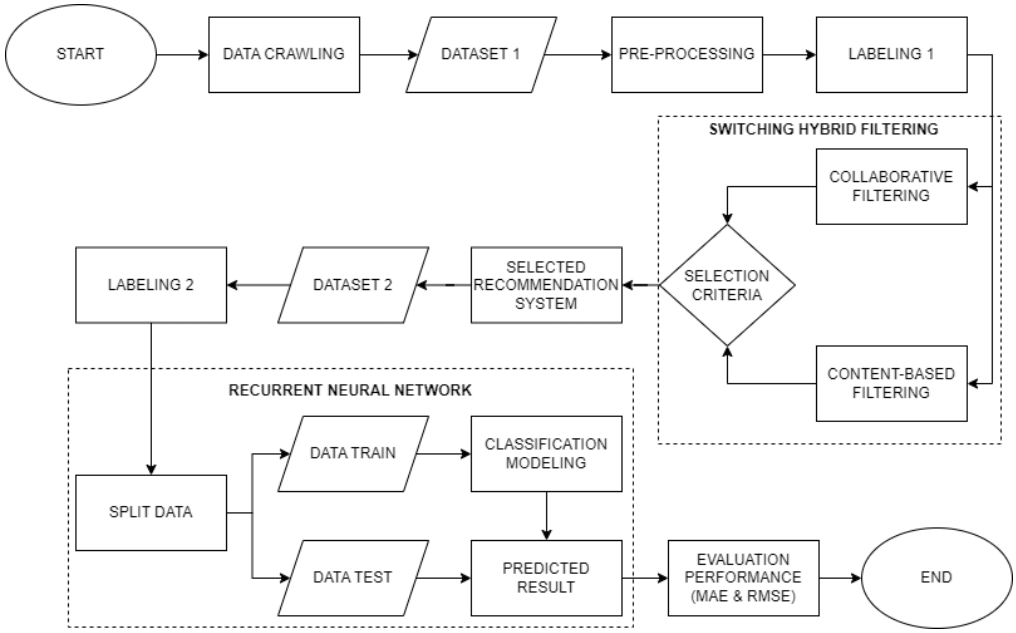

Figure. 1 Flowchart of overall system design

information. This research shows how users can be represented as additional sequences of items to produce effective personalized next-item recommendations. The results showed that RNN achieved the best overall results compared to other baselines, amounting to 0.06 for MRR and 0.20 for recall which indicates that about 20% of the total relevant items were successfully retrieved. Compared to the basic recommendation algorithm and conventional RNN) the DL approach with RNN and user-specific information achieved significant improvements in objective performance and recommendation quality.

In another case, research conducted by Tarana Singh, et al. [18] performed classification on movie-related tweets and provided recommendations based on positive sentiment. The RNN model analyzes the pre-processed tweet text to categorize each tweet as an expression of positive, negative, or neutral sentiment about the movie. The proposed approach achieved results with 91% accuracy, 92% precision, 90% recall, and 90% F-measure, outperforming the compared classification methods. This demonstrates the feasibility of the RNN-based movie recommendation system based on classification. This outperformed other classifiers such as Naïve Bayes and SVM, which had lower accuracy percentages, because RNN used guided looping and could process sequential data, such as a series of input vectors representing words in a sentence.

Considering the literature reviewed above, the proposed method will use switching hybrid filtering combined with deep learning recurrent neural network in building the recommendation system model. This method has consistently shown low error and high accuracy. Therefore, it is hypothesized that this model will also yield high accuracy in providing movie recommendations within the tweet dataset.

3. Proposed method

In this research, a movie recommendation system is devised through the implementation of two distinct methods. The first model uses switching hybrid filtering (SHF), while the second uses recurrent neural network (RNN) for classification. The overall system design is illustrated in Fig. 1.

3.1 Data crawling

This research uses two types of datasets, movie data from Netflix and tweet data from Twitter. The initial process of the system, as shown in Fig. 1, involves collecting the dataset through crawling using the Twitter API, where information such as tweets, user IDs, and movie titles are extracted. The retrieved tweet data specifically includes user reviews or opinions of movies on Netflix, and the crawling results are saved in comma separated values (CSV) format.

3.2 Preprocessing data

This is the stage where data is processed to prepare it for modeling. Raw data will be processed into clean and informative data for use in the next stages. Data preprocessing involves the following processes.

1. **Data cleaning:** The process of deleting or repairing irrelevant data in order to become cleaner and more structured data.
2. **Case folding:** The process of converting all letters in a sentence into lowercase letters, to overcome the problem of irrelevant writing differences.
3. **Stop word removal:** The process of removing common words that do not provide important information or specific meaning.
4. **Stemming:** The process of converting a word to its base form by removing any affixes or suffixes.
5. **Tokenization:** The process of dividing sentences into smaller units (tokens) for ease of further processing.

3.3 Labeling

Labeling refers to the process of assigning labels or markers to data. As shown in Fig. 1, labeling is performed 2 times. Labeling 1 is the process of assigning rating values to tweets containing user reviews of movies. In the polarity method, Labeling 1 on Dataset 1 is assigned a value between 0 and 5, where values close to 0 indicate negative sentiment and values close to 5 indicate positive sentiment. Meanwhile, Labeling 2 changes the value of Dataset 2 to 0 and 1, after the SHF stage to focus on developing a recommendation system that labels whether a movie is recommended (1) or not recommended (0).

3.4 Collaborative filtering

Collaborative filtering (CF) is a recommendation method that predicts user preferences based on similarities with other users who have similar preferences on a particular item [3-5, 9, 21]. CF makes recommendations based on the preferences of users with similar patterns by focusing on the relationship between users and items. There are two types of recommendation systems based on CF, namely user-based and item-based [20, 22, 23].

3.4.1. Data normalization

Data normalization is a data transformation technique that aims to improve the efficiency and accuracy of algorithms [23]. This process organizes data values so that they exist on a uniform scale. The formula is:

$$nr_{i,u} = r_{i,u} - \tilde{r}_u \quad (1)$$

The normalized rating for item i by user u, denoted as $nr_{i,u}$, is derived from the actual rating $r_{i,u}$. Additionally, $\tilde{r}_u$ represents the average rating given by user u to the rated items.

3.4.2. Calculated similarity

The process of calculating similarity in this research involves using the Cosine similarity method to evaluate the similarity between users or items. This method considers specific preferences or attributes, allowing the similarity of users to be evaluated based on their significance [24]:

User-based: Measuring similarity between users a and b based on their preference for item.

$$\text{Sim}(\overrightarrow{r_a}, \overrightarrow{r_b}) = \frac{\overrightarrow{r_a} \cdot \overrightarrow{r_b}}{|\overrightarrow{r_a}||\overrightarrow{r_b}|} \quad (2)$$

Item-based: Measuring similarity between items x and y based on the user's preference for item.

$$\text{Sim}(\overrightarrow{r_x}, \overrightarrow{r_y}) = \frac{\overrightarrow{r_x} \cdot \overrightarrow{r_y}}{|\overrightarrow{r_x}||\overrightarrow{r_y}|} \quad (3)$$

3.4.3. Rating prediction

Rating prediction is the process of making predictions of ratings that may be given by users to an item. The selection of the n in the topN method is based on the criteria of optimal MAE and RMSE value. The formula can be written as follows [22]:

User-based:

$$P_{a,x} = \bar{R}_a + \frac{\sum_{b=1}^{N}(R_{b,x} - \bar{R}_b) \cdot Sim_{a,b}}{\sum_{b=1}^{N} Sim_{a,b}} \quad (4)$$

The average rating provided by user a is represented as $\bar{R}_a$. While $R_{b,x}$ denotes rating assigned by user b to item x. For each user a and b will be calculated similarity with $Sim_{a,b}$.

Item-Based:

$$P_{a,y} = \bar{R}_y + \frac{\sum_{x=1}^{N}(R_{a,x} - \bar{R}_x) \cdot Sim_{x,y}}{\sum_{b=1}^{N} Sim_{x,y}} \quad (5)$$

$\bar{R}_y$ is the average rating given for item x. $R_{a,x}$ represents the rating given by user a for item x. For each item x and y will be calculated similarity with $Sim_{x,y}$.

3.5 Content-based filtering

Content-based filtering (CBF) is a recommendation method that relies on the similarity

of items based on users' previous preferences. It is based on a comparison between items and additional user information [5, 13]. CBF classifies all items into item profiles based on their features, without requiring data from other users as CF does. Therefore, CBF will use feature extraction and word embedding to improve semantic representation to provide more accurate recommendations.

3.5.1. Feature extraction TF-IDF

Term frequency-inverse document frequency (TF-IDF) is a popular text mining method that uses the frequency of words in documents and document collections to calculate weights [7, 15]. It is a method to assess the importance of the word in the document, determined by assigning a score that considers both the frequency of words within that specific document (term frequency) and its rarity across the entire collection of documents (inverse document frequency). The TF-IDF formula is as follows [7]:

$$\mathrm{TF}(i,z) = \frac{\mathrm{freq}(i,z)}{\mathrm{maxOthers}(i,z)} \tag{6}$$

$$\mathrm{IDF}(i) = \log\left(\frac{X}{DF}\right) \tag{7}$$

$$\text{TF-IDF}(i,z) = \mathrm{TF}(i,z) \times \mathrm{IDF}(i) \tag{8}$$

TF in formula (6), $\mathrm{freq(i,z)}$ indicates the count of occurrences of a term in the document z, and $\mathrm{maxOthers(i,z)}$ is the count of occurrences for most other terms within the same document. Meanwhile, IDF in formula (7) measures the informativeness of a term by calculating the logarithm of the count of documents in the collection (X) divided by the count of documents containing term I (DF). Combining these two metrics, $TF - IDF$ provides a score that takes into consideration the frequency of a term in one document and the informativeness of that term in the overall document collection.

3.5.2. Word embedding RoBERTa

The robustly optimized BERT approach (RoBERTa) is an upgraded iteration of the BERT model, featuring enhanced training methodologies [25]. Through intensive training, RoBERTa successfully overcomes the weaknesses of BERT by better-handling document context and overcoming text length limitations. In contrast to BERT's use of a static masking pattern, an innovative dynamic masking pattern has been implemented [25, 26]. RoBERTa can capture relationships between words in item descriptions and is suitable for more complex documents. The RoBERTa method uses a transformer approach to understand sentence context and relationships between words, resulting in more contextualized word embedding. Therefore, RoBERTa is the choice in this research for word embedding in content-based filtering, where a deep understanding of the text content can improve the accuracy of recommending items based on the suitability of user preferences.

3.5.3. Rating prediction

Rating prediction in content-based filtering uses the vector representation values of TF-IDF and RoBERTa. By using the TF-IDF value, the importance of a word within a document can be assessed and the relevance of the document to the user's preferences can be measured. Words that have a high TF-IDF value reflect the important characteristics of an item. In addition, RoBERTa provides a deeper dimension of contextual understanding by representing word semantics in relation to a wider context. Both methods involve evaluating the relevance of an item to the user's preferences through content analysis. Items with increasingly higher values are recommended.

3.6 Switching hybrid filtering

Hybrid filtering is a method used in recommendation systems, which is a combination of a single recommendation system as a sub-component [6], such as combining collaborative filtering (CF) and content-based filtering (CBF) [21]. As shown in Fig. 1, this research uses switching

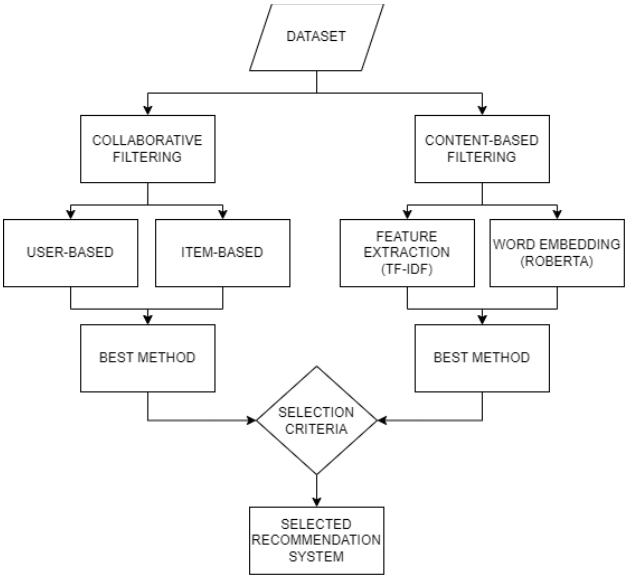

Figure. 2 Flowchart of switching hybrid filtering

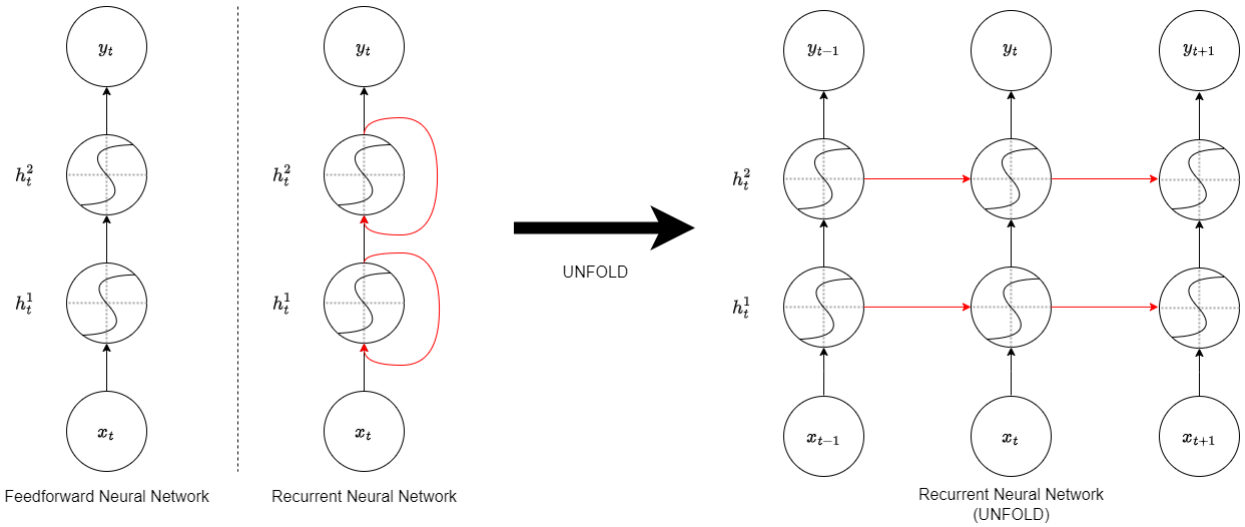

Figure. 3 Recurrent neural network illustration

hybrid filtering (SHF) method. SHF selects a single recommendation system based on a specific context or situation. The model is designed to handle sensitive datasets at the item level, and the determination of recommendation selection criteria is based on user preferences or other characteristics. The selection of a recommendation method is determined by several criteria or can be switched between recommendation techniques [7, 12, 18].

Based on Fig. 2, this research will use the best method from each CF and CBF which will then be combined in the SHF method. Both can be measured based on confidence measures, specifically MAE and RMSE. First, CF and CBF independently predict ratings using methods such as CF with user-based and item-based, CBF with TF-IDF, and RoBERTA. The system then evaluates the predictions and switches methods if necessary, aiming to optimize accuracy by combining CF and CBF while considering the confidence level [18]. The SHF methodology introduced by Ghazanfar [7], can switch from one method to another. This adaptive approach ensures effective performance of the recommendation system that is customized to the dataset characteristics and user preferences.

3.7 Performance evaluation of switching hybrid filtering model

In the performance evaluation of the SHF stages, the accuracy of recommendations is measured using the mean absolute error (MAE) and the root mean square error (RMSE), as shown in formulas (9) and (10), respectively [7]. The metrics evaluate the variance between predicted and actual user ratings, considering data with accurate label values. Both MAE and RMSE serve as metrics that provide confidence in the SHF method. Here are the MAE and RMSE formulas [12, 18]:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|r_i - \hat{r}_i| \tag{9}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_i - \hat{r}_i)^2} \tag{10}$$

MAE measures the average of the absolute difference between the predicted value ($\hat{r}_i$) and the actual value (r_i) for a set of n data. On the other hand, RMSE is the square root of the average of the squares of the difference between the predicted value ($\hat{r}_i$) and the actual value (r_i) for a set of n data.

3.8 Recurrent neural network

Recurrent neural network (RNN) is one type of method in deep learning (DL) that can classify sequence data. RNNs learn from the sequence of items consumed by users over time [2, 3]. RNN-based recommendation system utilizes user profile data along with session information to predict the user's preferred items [3]. RNN provides better time-order patterns and overall performance. Fig. 3 shows the architecture of the RNN [3]. RNN is a changed version of a Feedforward Neural Network (FNN). In Fig. 3, the hidden layers in RNN depend on previous information obtained through a recurrent mechanism, which is represented by red-colored edges in the RNN structure. These recurrent edges begin from the hidden layer and connect back to the same hidden layer. When given a series of input data, the RNN can update the state of the hidden unit at each iteration by utilizing the previous (recurrent) state of the hidden unit and the new input [3]. This process allows the RNN to establish time

dependencies on sequential data. In summary, the formula for updating the hidden state (h) and generating the output (y) in the RNN structure is as follows [3]:

$$h_t = f(U_t x_t + W_t h_{t-1} + b_h) \quad (11)$$

$$y_t = g(W_y h_t + b_y) \quad (12)$$

Where the parameters of the weight are given by U and W, and the bias parameter is given by b. h_t is the hidden state actualization process, while y_t is the output. f and g are the activation functions applied to the summation result. Specifically, RNNs play a role in classifying time series data due to their ability to store and transfer information from the previous step to the next. This means that previous inputs have an influence on the prediction of the current output. This ability allows RNN to model temporal relationships and patterns, providing more personalized recommendations according to the user's interest sequence patterns in the recommendation system. By considering time as a key factor in modeling user preferences, RNN has become a powerful method in the context of recommendation systems [17].

3.9 Optimization

As the advancement of deep learning is followed by various model designs, the importance of the role of optimizers in the learning process is illustrated in the effort to improve the performance and efficiency of neural networks [27]. Neural network optimization is the process of improving the performance and efficiency of the model by making various adjustments and changes to achieve higher accuracy. There are many types of optimization algorithms for neural network models, such as Adam, Nadam, Adamax, Adagrad, Adadelta, SGD, RMSprop, and others. These algorithms utilize the weights and learning speed of the model in the training process, to minimize the loss function and maximize the accuracy [26].

The learning rate is a hyperparameter in the optimizer that controls how much the model weights are changed each time the model is updated during training. If the learning rate is too small, training can become very slow or even struggle to converge. Conversely, if the learning rate is increased during training to maintain a wide search range, bad results can be avoided, although convergence becomes more difficult [28]. Therefore, choosing an appropriate learning rate is crucial to ensure efficient training and optimal model convergence.

Table 1. Confusion matrix

Confusion Matrix		**Predict Values**	
		Positive	Negative
Actual Values	Positive	TP	FN
	Negative	FP	TN

3.10 Performance evaluation of classification model

In the evaluation stage, classification performance metrics are calculated to assess how often the decisions made by the system match good recommendations for users [7]. The evaluation method uses the confusion matrix, a table that visualizes the performance of a classification algorithm by comparing the predicted results with the actual classification, providing a comprehensive picture of the system's performance [28].

Based on Table 1, TP (true positive) is a result that is correctly identified as positive, TN (true negative) is a result that is correctly identified as negative, FP (false positive) is a result that is incorrectly identified as positive, and FN (false negative) is a result that is incorrectly identified as negative [28]. Metrics that can be used to evaluate the performance of different recommender methods include accuracy, precision, recall, and f1-score [4]. The formula for the performance metric is as follows [4, 11, 16, 29]:

1. **Accuracy:** is a metric that measures the overall correctness of the recommendation system.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \quad (13)$$

2. **Precision:** is a metric that measures the extent to which the recommended items are relevant.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (14)$$

3. **Recall:** is a metric that measures the extent to which relevant items are successfully recommended.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (15)$$

4. **F1-Score:** is a metric that combines Precision and Recall. It provides a balance between these two metrics and is useful when we want to pay attention to both simultaneously.

Table 2. Movies dataset

type	name	…	genre	datePublished
Movie	14 Cameras	…	['Crime', 'Horror', 'Thriller']	2018-07-27
Movie	Dolphin Tale 2	…	['Drama', 'Family']	2014-09-12
…	…	…	…	…
Movie	The Smurfs	…	['Adventure', 'Comedy', 'Family']	2011-10-07
Movie	Özel Ders	…	['Comedy', 'Romance']	2022-12-16

$$\text{F1} - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (16)$$

4. Result and discussion

4.1 Result

This research was primarily divided into three phases, which are data preparation, recommendation system, and classification. The data preparation phase involved crawling data and preprocessing it to ensure that it was ready for use. In the recommendation system phase, the objective was to predict ratings, and this was accomplished through three steps, namely collaborative filtering (CF), content-based filtering (CBF), and switching hybrid filtering (SHF). The results of this phase included rating predictions, which were then evaluated using MAE and RMSE. The classification phase was focused on labeling each rating and was modeled using a recurrent neural network (RNN). The results of this phase were evaluated based on accuracy, precision, recall, and f1-score. Each of these phases is explained in detail below:

4.1.1. Data preparation result

This research used two types of data obtained from different sources. The first dataset, namely "Movie Dataset" is movie information obtained through the crawling process on the IMDB website with features such as name, description, and keywords. The second dataset, namely “Movie Reviews Dataset” (tweets) obtained from Twitter by crawling based on movie title keywords matched to the first data. Both datasets were pre-processed before being integrated into the recommendation system stages.

This research involved two data crawling steps. First, movie titles were retrieved from the IMDB site by applying the Netflix filter, and relevant features were extracted using the PyMovieDb library. Data was first retrieved from IMDB with a Netflix filter, resulting in 265 movies. The results were merged with previous research data, with the format of the "Movies Dataset" measuring 854 x 18, as shown in Table 2.

Next was a second crawl using tweet-harvest, which aimed to obtain movie reviews from Twitter. This process was carried out on 39 movie reviewer accounts and then combined with movie reviews that had been collected in previous studies. Tweets reviewing the movie were crawled in Bahasa (Indonesian Language) and translated into English

Table 3. Movie reviews dataset

username	movie	…	tweet
AnakNonton	Drive	…	"Drive" was awesome. Recommendd! ;) RT @wilfredcull n: @AnakNonton drive Isn't the movie okay? :)
CenayangFilm	Wind River	…	Wind River is cool too. A drama movie but it's a thrill to watch
…	…	…	…
moviemenfes	The Machine	…	mvs those of you who haven't watched it, let's try it. it's exciting, funny, touched 😍😍 title : The Mitchells vs the machine https://t.co/jWjecis85k
winseulbear	The Social Dilemma	…	@leejaeropak @collegemenfess which I think is good: the king's speech and the social dilemma😀👍

Table 4. Cleaned dataset

username	movie	…	cleaned_text
AnakNonton	Drive	…	drive was awesome recommended drive is the film okay
CenayangFilm	Wind River	…	wind river is cool too drama film but excited to watch
…	…	…	…
moviemenfes	The Machine	…	those of you who havent watched lets try it this is exciting funny moving title the mitchells vs the machine
winseulbear	The Social Dilemma	…	which i think is good the king speech and the social dilemma

Table 5. Ratings dataset

username	movie	polarity_score
AnakNonton	Drive	5.0
CenayangFilm	Wind River	3.3437
…	…	…
moviemenfes	The Machine	3.8
winseulbear	The Social Dilemma	3.4166

Table 6. Final dataset

Movie	AnakNonton	…	IMDB	…	zavvi
14 Cameras	0.00	…	2.30	…	0.00
17 Again	0.00	…	3.20	…	0.00
…	…	…	…	…	…
Zodiac	0.00	…	3.85	…	0.00
Zombieland	0.00	…	3.80	…	0.00

using the GoogleTranslator library. The number of movie reviews obtained reached 34,086, with a size of 4138 x 7 as shown in Table 3.

In this research, the process of converting reviews into ratings was conducted through the data-cleaning phase, which included the removal of irrelevant information such as emoticons, tags, mentions, and links. As a result, this process helped clean up reviews, remove unimportant items, and convert text to lowercase, as shown in Table 4.

The calculation of the polarity score used the TextBlob library, producing a score in the range of 0-5 representing the user's rating of the movie. After the preprocessing stage, the movie review data became a rating dataset and had a size of 6479 x 4, as shown in Table 5.

Next, the rating dataset was merged and converted into a 514 x 45 matrix representing the number of movies and users. The columns of the dataset indicated users (including IMDB, rotten tomatoes, and metacritic websites), the rows of dataset indicated movie names, and the values of dataset were the result of polarity and rating values on websites. This dataset was the result of data preparation named "Final Dataset".

As shown in Table 6, there were still many ratings that had a value of 0, known as sparse data. Sparse data reflects the state where most items do not receive ratings from users [4]. The sparseness of data in the rating matrix reached 74.46%, which was only a small part of the total data. The Final Dataset was used for the next phase.

4.1.2. Collaborative filtering result

Dataset 1 (final dataset) was utilized in the collaborative filtering (CF) phase. Next, the similarity value between users (UBCF) and between movies (IBCF) was calculated using the cosine similarity method with formulas (2) and (3). In the topN method, the similarity value plays a key role in the selection of the best parameter (n). This approach involves rating prediction, which is evaluated by the MAE and RMSE metrics. By analyzing the relationship between the value of n and MAE-RMSE, the optimal setting for n can be identified [23]. This ensured accurate rating prediction. The results of finding the optimal value of n with each MAE and RMSE generated from UBCF and IBCF were presented in Table 7.

Table 7. Optimal value of n

n value	User-Based CF		Item-Based CF	
	MAE	RMSE	MAE	RMSE
2	0.1748	0.2306	0.1522	0.2133
3	0.1562	0.2031	0.1332	0.1832
4	**0.1503**	**0.1942**	0.1274	0.1731
5	0.1522	0.1978	0.1248	0.1706
6	0.1540	0.1986	0.1230	0.1679
7	0.1587	0.2032	0.1225	0.1669
8	0.1636	0.2080	0.1218	0.1660
9	0.1692	0.2136	**0.1216**	**0.1658**
10	0.1741	0.2182	0.1222	0.1662
11	0.1796	0.2239	0.1227	0.1664
12	0.1847	0.2292	0.1236	0.1671

Table 8. User-based CF predictions

Movie	AnakNonton	…	zavvi
14 Cameras	0.0400	…	0.0400
17 Again	0.1000	…	0.1000
…	…	…	…
Zodiac	0.4427	…	0.1900
Zombieland	0.5879	…	0.3300

Table 9. Item-based CF predictions

Movie	AnakNonton	…	zavvi
14 Cameras	0.0700	…	0.0500
17 Again	0.0700	…	0.0500
…	…	…	…
Zodiac	0.0700	…	0.0500
Zombieland	0.3037	…	0.0500

Table 7 shows the UBCF evaluation, which achieved the lowest RMSE and MAE values at n = 4 to be applied to the topN method. The same process is applied to the IBCF method, where the lowest RMSE and MAE values are at n = 9. These values were applied to the TopN method to predict user ratings. The prediction result tables for both are shown in Tables 8 and 9.

As shown in Tables 8 and 9, the resulting prediction rating had a value range between 0 and 1.

Table 10. CBF TF-IDF predictions

Movie	AnakNonton	…	zavvi
14 Cameras	0.0421	…	0.0336
17 Again	0.0896	…	0.0495
…	…	…	…
Zodiac	0.0478	…	0.0351
Zombieland	0.0779	…	0.0356

Table 11. CBF RoBERTa predictions

Movie	AnakNonton	…	zavvi
14 Cameras	0.0701	…	0.0483
17 Again	0.0701	…	0.0483
…	…	…	…
Zodiac	0.0703	…	0.0484
Zombieland	0.0703	…	0.0484

A rating value close to 0 indicated that the movie was not recommended, while if rating value was close to 1, the movie was recommended. UBCF with MAE of 0.2355 and RMSE of 0.3211, while IBCF with MAE of 0.1083 and RMSE of 0.1696. This shows that IBCF produced 54% more accurate predictions than user-based. So, the collaborative filtering that will be used in switching hybrid filtering is the item-based CF approach.

4.1.3. Content-based filtering result

Content-based filtering (CBF) used Dataset 1 (final dataset) and combined the description, genre, and keyword columns from "Movie Dataset" into a "content" column as a feature that described the movie. This feature was cleaned up, such as removing 0 value entries, and used to build item profiles. In this research, the CBF model used TF-IDF and RoBERTA to improve the recommender system's ability to understand and analyze the content of items [5, 21]. The results of rating prediction using TF-IDF and RoBERTa are shown in Tables 10 and 11.

As seen in Tables 10 and 11, the resulting prediction rating also had a value range between 0 and 1. A rating value close to 0 (less than or equal to 0.5) indicated that the movie was not recommended, while if rating value close to 1 (more than 0.5), indicated that the movie was recommended. CBF using TFIDF produced MAE of 0.0823 and RMSE of 0.1237. Meanwhile, CBF that used RoBERTa produced MAE of 0.0800 and RMSE of 0.1221. This shows that both worked very well, but RoBERTa can produce a slightly lower error. This was proven because RoBERTa used the understanding of context and relationships between words by processing the text contextually, resulting in a better semantic representation. Therefore, the CBF that will be used is the CBF that used RoBERTa approach.

Table 12. Switching hybrid filtering predictions

Movie	AnakNonton	…	zavvi
14 Cameras	0.0893	…	0.0813
17 Again	0.0847	…	0.0671
…	…	…	…
Zodiac	0.0904	…	0.0741
Zombieland	0.2181	…	0.0394

4.1.4. Switching hybrid filtering result

In the data preparation results, the final dataset showed a relatively high amount of data sparsity of 74.46%. Based on related research [8], collaborative filtering (CF) faced challenges in scenarios with minimal user interaction, while content-based filtering (CBF) overcame this limitation by providing recommendations based on item characteristics or content. Therefore, the combination of switching hybrid filtering was done with CBF as the first method and CF as the second method. As seen in Fig. 2, each of the CF and CBF was selected based on the best method. CBF RoBERTa was used as the initial method, if it produced enough confidence, then Item-Based CF was not required. The confidence of prediction with CBF was considered high, which meant that the recommendation must be accurate [7]. So, if the RoBERTa prediction result still had a rating value of 0, the item-based CF process will be performed. Therefore, sparse data can be reduced through the SHF process. The rating prediction results generated by the SHF process are shown in Table 12.

From the final dataset, 74.46% of the sparse data was filled, ensuring that the dataset had no 0 entries (0% sparse data), as shown in Table 11. This confirmed that the proposed SHF method showed good performance in providing recommendations. Graphs illustrating the differences in mean absolute error (MAE) and root mean square error (RMSE) between collaborative filtering method, content-based filtering method, and switching hybrid filtering method are shown in Fig. 4. Lower MAE and RMSE values indicated better performance.

As shown in Fig. 4, the error generated by user-based collaborative filtering (UBCF) was the highest, indicating its limited ability to handle sparse data. In contrast, the switching hybrid filtering (SHF) method produced the lowest error. The resulting MAE was 0.0617, and RMSE was 0.1178, indicating that the rating prediction was more accurate compared to the other methods. However,

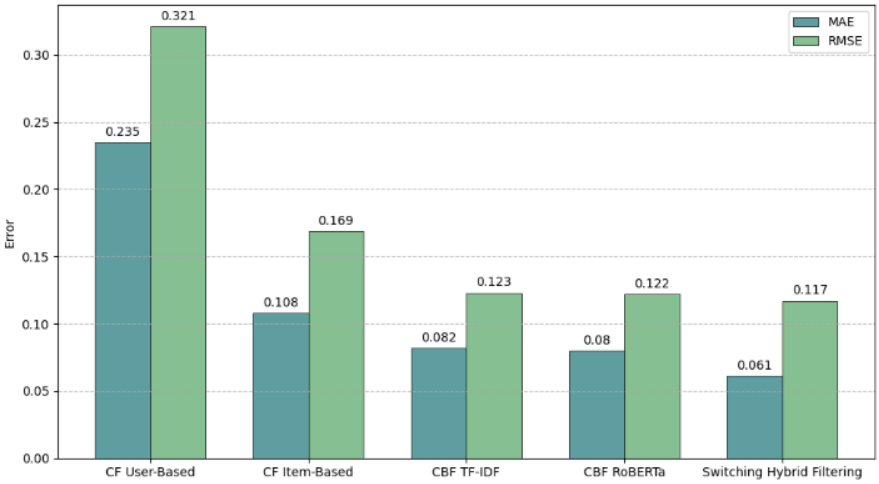

Figure. 4 Comparison RMSE and MAE for each method

Table 13. Labeling result

Movie	AnakNonton	…	zavvi
14 Cameras	0.0	…	0.0
17 Again	0.0	…	0.0
…	…	…	…
Zodiac	0.0	…	0.0
Zombieland	0.0	…	0.0

Table 14. Performance analysis of baseline model

Test Size (%)	Performance Metrics			
	Accuracy	Precision	Recall	F1-Score
10	84%	82.44%	82.98%	80%
20	**84.23%**	83%	84.38%	82.6%
30	83.96%	84%	85%	82.58%
40	83.7%	84.21%	85.25%	82.59%

the difference was not very significant compared to content-based filtering (CBF) with RoBERTa, which operated independently. This was because the CBF method did not rely on user and item interactions, focusing only on item attributes. The independence from user and item interactions might have contributed to its competitive performance. These results show that the proposed SHF method showed excellent performance in predicting ratings.

4.1.5. Classification result

The classification phase used Dataset 2, which contained the rating matrix resulting from switching hybrid filtering phase. The matrix was converted into binary labels 0 and 1. The value 0.5 was the separator of the two labels because it was the midpoint between the highest rating value (1) and the lowest rating (0). 0 represents a rating from 0 to 0.5, indicating a dislike for the movie (not recommended), while 1 represents a rating from 0.6 to 1, indicating a preference or like for the movie (recommended). Table 13 shows the labeling results (classification dataset).

After labeling, the total ratings converted to 0 was 18,284, while 1 was 4,846. The results were continued to the RNN phase. In the classification process, experiments were performed with three scenarios. In the first scenario, the classification process was performed on s basic RNN (baseline) model without any optimizations or features. In the second scenario, the classification process was performed on the RNN model by applying several optimization algorithms. Finally, in the third scenario, the most optimal learning rate parameter value, and the addition of features in each optimization were determined with the aim of improving the accuracy.

4.1.5.1. RNN baseline model

The baseline RNN model was run in the first scenario for the classification dataset with a sequence length of 44. This model was tested with sigmoid activation, 50 epochs, batch size 64, and dropout 0.3. In addition, this model was run without applying any optimizations and features. The model was run 5 times, and the average value was taken. This scenario was performed to find the best number of test sizes based on the highest accuracy. The results of the metric performance comparison of each test size are shown in Table 14.

Based on Table 14, the highest accuracy was achieved at a test size of 20%, which was 84.23%. This indicated that the baseline model could predict well with an appropriate sample size for training of 80% and testing of 20%. In addition, the split data had a good representation of the different classes for

Table 15. Performance analysis of each optimizer

Optimizer	Performance Metrics (%)			
	Accuracy	**Precision**	**Recall**	**F1-Score**
Adam	85.27 (+1.24%)	83.58 (+0.7%)	84.81 (+0.51%)	82.86 (+0.31%)
Nadam	**85.75 (+1.81%)**	**84.47 (+1.78%)**	**85.37 (+1.17%)**	**83.55 (+1.15%)**
Adamax	84.97 (+0.88%)	83.23 (+0.27%)	84.77 (+0.46%)	82.67 (+0.08%)
Adagrad	84.79 (+0.66%)	83.15 (+0.18%)	84.22 (-0.19%)	81.77 (-1.00%)
Adadelta	84.29 (+0.07%)	83.69 (+0.83%)	84.49 (+0.13%)	82.58 (-0.02%)
SGD	79.39 (-5.57%)	68.24 (-18%)	79.16 (-6.19%)	72.23 (-12.59%)
RMSprop	84.87 (+0.76%)	84.4 (+1.45%)	84.58 (+0.24%)	82.12 (-0.58%)

the model to perform the classification process. The baseline model could also be considered optimal because accuracy was calculated by dividing the number of correct predictions by the total number of predictions. In addition, other performance metrics were calculated in Scenario 1 to measure the overall performance of the model in classifying the dataset. The results showed an average of more than 80%, especially with f1-score reaching 82.6%, indicated that the provided dataset was balanced enough for effective model training. The next step in Scenario 2 was to apply various optimization algorithms aimed at increasing the performance metric value.

4.1.5.2. RNN model optimization

In the second scenario, criteria such as sigmoid activation with a consistent epoch of 50, batch size 64, and dropout 0.3 will be used for the classification with a sequence length of 44. The test size used was 20% according to the most optimal results in Scenario 1. The model was run 5 times, and the average value was taken. The scenario applied several optimization algorithms to differentiate the results of all experimental scenarios. Some of the optimizations that were used included Adam, Nadam, Adamax, Adagrad, Adadelta, Stochastic Gradient Descent (SGD), and Root Mean Square Propagation (RMSprop). The optimization buffer was set with a consistent learning rate parameter of 0.001 (default). The use of optimization aimed at increasing the results of the performance metrics and reducing the losses during the classification process. The results are shown in Table 15.

After performing all the experiments in Scenario 2, several optimizations showed consistent and not significantly different results. However, the model optimized with SGD tended to produce lower performance compared to the baseline and other optimizations, with the metric value not reaching 80%. This indicates that the default parameters were not well defined. On the other hand, the model optimized with Nadam was the best, showing improvements in all metrics compared to the baseline model. This happened because Nadam with the default parameters could define it well. Nadam, as an optimization method that combines nesterov accelerated gradient (NAG) and adaptive moment estimation (Adam), had the advantage of handling gradient problems involving momentum, thus providing good performance. As shown in Table 15, nadam produced an accuracy of 85.75%, precision of 84.47%, Recall of 85.37%, and F1-Score of 83.55%, where the values of these results were higher than other optimizations.

Table 16. The best learning rate of each optimizer

Optimizer	Learning Rate	Accuracy (%)
Adam	0.004498432668969454	85.29 (+1.26%)
Nadam	**0.0010985411419875597**	**86.11 (+2.23%)**
Adamax	0.0035564803062231283	85.53 (+1.54%)
Adagrad	0.04714866363457394	84.81 (+0.69%)
Adadelta	0.30888435964774846	84.35 (+0.14%)
SGD	0.037275937203149416	84.41 (+0.21%)
RMSprop	0.0010985411419875597	85.36 (+1.34%)

The decrease and increase in metric performance results lie in the parameter values and features used by each optimizer. Each optimizer depends on different parameters and feature values. Therefore, if all of them are run on a model with default parameters, they will produce different results. Therefore, Scenario 3 was run to determine the most optimal value of parameters such as learning rate along with features to achieve better accuracy [28].

4.1.5.3. Optimization of learning rate and features for each optimizer

In the third scenario, the focus was on finding the optimal value of the learning rate parameter for

each optimization. Learning rate, a hyperparameter determining the number of steps taken in each learning iteration, is widely employed to enhance testing accuracy [27]. The learning rate can affect the stability of the model, so we decided to conduct experiments to find the best learning rate.

This process was carried out by testing the learning rate value in the range of 1e-10 to 1e0. Iterations were performed during testing with 50 epochs and 5 periods consistently to get better information about the performance of the learning rate. In addition, parameter features were added to control the performance of each optimizer, such as exponential moving average, beta, epsilon, and momentum. These parameter features helped in model performance, especially in convergence. The results of the best learning rate and accuracy obtained by each optimizer are shown in Table 16.

Table 16 shows that all optimizers showed an increase. Based on these results, this approach proved effective in achieving improved test accuracy under certain conditions [28]. This occurred because each optimizer was configured with the appropriate parameter values and features to train the RNN model. SGD, which had its learning rate parameter set to 0.037275937203149416, showed a significant increase in accuracy, reaching 84.41%. In Scenario 2, with the default learning rate value, SGD only achieved an accuracy of 79.39%, indicating that with appropriate settings, it could provide a significant performance boost in improving test accuracy, by 0.21% from baseline. Besides, the model optimized with Nadam still showed the highest accuracy. The learning rate was initially set at 0.01 and then adjusted to 0.0010985411419875597, identified as the optimal learning rate. Nadam, utilizing an optimal learning rate prevented the model from diverging and made the model converge faster during the training process. This happened due to the effectiveness of Nesterov Momentum and the adaptability of Adam, thus providing an increase in accuracy. In addition, Nadam implemented features such as exponential moving average, which contributed to an increase in accuracy of 2.23%.

4.2 Discussion

In this research, several test scenarios were conducted to determine the model with the best combination of optimization and learning rate. The focus of this research was the calculation of accuracy, which served as a reference for each scenario. Other performance metrics were calculated to determine if the model was good enough to classify the dataset.

Based on the results of Scenario 1, it was shown that the baseline RNN model with 80:20 split data (test size 20%) produced the highest accuracy, which was 84.23%. The model was tested with a sequence length of 44, in the last layer using sigmoid activation for binary classifiers, 50 epochs, a batch size of 64, and a dropout of 0.3. The model was run 5 times, and the average of each run was taken to avoid skewed results. This model served as a reference for Scenario 2 by applying a test size of 20% and implementing 7 optimizations, namely Adam, Nadam, Adamax, Adagrad, Adadelta, stochastic gradient descent (SGD), and root mean square propagation (RMSprop). The learning rate was set to 0.001 and the model was run 5 times for averaging.

In Scenario 2, the model was run with the same criteria 5 times and the average was taken. The model optimized with Nadam was found to have higher accuracy than the compared optimizers. This is because Nadam is a derivative of Adam that incorporates elements of the nesterov accelerated gradient (NAG) algorithm. NAG is an algorithm that adjusts the position of the gradient before updating the model parameters, thus affecting the speed at which the model converges. This was demonstrated by Nadam achieving higher metric performance, with an accuracy of 85.75%, which was 0.56% better than Adam. However, the model optimized with SGD showed the opposite result. This was because the parameters set by default could not be well defined, resulting in a decrease in performance.

Therefore, Scenario 3 aimed to identify the best learning rate for each optimizer, focusing on achieving the highest accuracy in model training. The learning rate finder algorithm was used for this purpose, testing several learning rate values within a certain range to identify the upper and lower bounds obtained from significant changes in the loss graph during the learning rate search. The reference criterion for considering the learning rate as the best was the range of values that resulted in the lowest loss. In this process, learning rate values in the range of 1e-10 to 1e0 were tested in iterations, with the test criteria being 50 epochs and 5 periods consistently. Parameters like exponential moving average, beta, epsilon, and momentum were fine-tuned for optimization. The focus in this scenario was to find the optimal learning rate that gave an increase in accuracy. SGD, with an optimal learning rate, exhibited a notable accuracy increase from 79.39% to 84.41%, showcasing a 6.29% improvement. Besides, the model optimized with

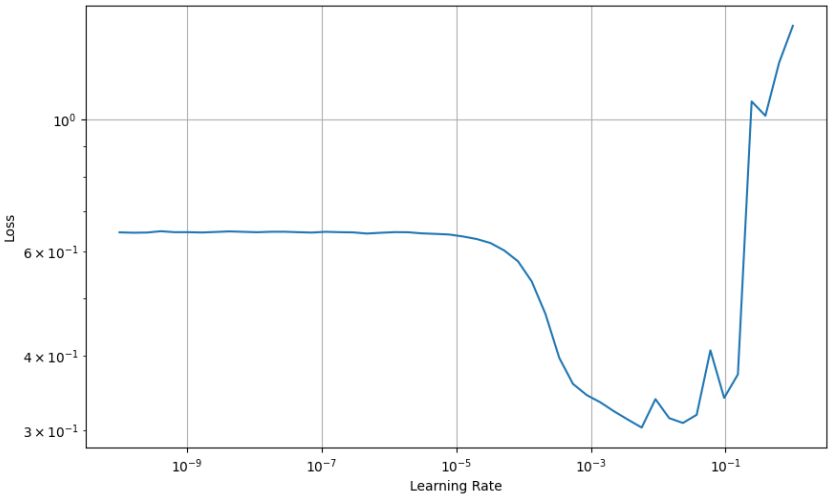

Figure. 5 Optimal learning rate of Nadam optimizer

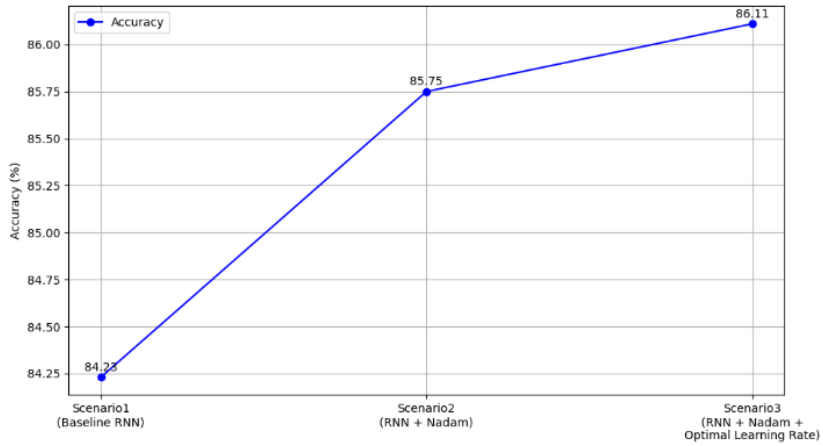

Figure. 6 Graph of the increase in accuracy

Nadam still showed the highest accuracy. The learning rate was initially set at 0.01 and later adjusted to the optimal value for Nadam. Moreover, Nadam implemented features such as exponential moving average, contributing to an increase in accuracy of 0.36%. A graph of the learning rate search process of the Nadam optimizer, determined based on the resulting loss, is shown in Fig. 5.

As seen in Fig. 5, the upper bound lay at 2.12e-04, and the lower bound lay at 5.69e-03, where after the lower bound, there was an increase in learning speed that led to high losses. The most optimal learning rate ranged from 2.12e-04 to less than 5.69e-03, and it was found that 0.0010985411419875597 was the best learning rate for Nadam optimization. With a predefined learning rate, the RNN model is tuned to optimize Nadam with the parameters of the learning rate value plus predefined features. The result shows an increase of 0.36% over the default tuning. Based on scenarios 2 and 3, Nadam demonstrated suitable performance for classification in this research. Overall, the use of an appropriate learning rate value facilitated an optimal balance between convergence speed and accuracy on the dataset, resulting in a significant improvement in test accuracy. Fig. 6 shows the graph of the increase in accuracy for each scenario for the RNN model with Nadam optimizer.

The results of this study showed that the use of switching hybrid filtering (SHF), which utilized item-based collaborative filtering and RoBERTa word embedding for content-based filtering, provided good performance. Given that RoBERTa could understand tweets context, handle language variations, and had a strong hierarchical representation, which allowed the model to understand text meaning at the word, phrase, and sentence levels. This was evident in the low mean absolute error (MAE) and root mean square error (RMSE) values. Additionally, the baseline recurrent neural network (RNN) using the rating prediction result dataset from SHF showed high metric

performance, as the RNN model could represent the classes well. In addition, these results also proved that combining the advantages of SHF and RNN provided high accuracy values. The use of Nadam for RNN model optimization with an optimal learning rate setting provided higher accuracy compared to the baseline model. Overall, this research achieved the goal of achieving high accuracy performance so that movie recommendations can be provided to users accurately according to their preferences.

5. Conclusion

In this research, a switching hybrid filtering (SHF) method was developed by integrating collaborative filtering (CF) and content-based filtering (CBF) combined with a recurrent neural network (RNN) for classification. The dataset consists of 23.130 ratings containing 514 movies and 44 users (including IMDB, rotten tomatoes, and metacritic websites). The dataset had sparse data with 74.46% of the total data. In the SHF stage, the initialization method used CBF with RoBERTa, and a switch was made to use the item-based CF method. The rating prediction produced less error with MAE of 0.0617 and RMSE of 0.1178, which is lower than other methods that independently predicted ratings. The results indicated that the proposed SHF method predicted more accurate ratings and was able to handle the sparse data problem. Afterward, the classification process was performed with RNN to improve accuracy and provide higher metric performance to recommend a movie. The RNN model provided optimal results by applying Nadam optimization tuned for learning rate and features. The RNN model optimized by Nadam with a learning rate of 0.0010985411419875597 and the addition of exponential moving average, beta, epsilon, and momentum parameter features achieved 86.11% accuracy, which is a higher result than the compared optimizers. This result also increased the accuracy by 2.23% over the baseline model. The results of this study illustrate the satisfactory performance of the system with the proposed method. Future research is expected to develop a recommendation system by integrating different types of hybrid filtering, as well as using other deep learning models with optimization and enhanced features.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, B. M. G. Al Awienoor and E.B. Setiawan; methodology, B. M. G. Al Awienoor; analysis and validation, B. M. G. Al Awienoor and E.B. Setiawan; resources, data curation and writing—original draft preparation, B. M. G. Al Awienoor; writing—review, editing, and visualization B. M. G. Al Awienoor; supervision, project administration, and funding acquisition E. B. Setiawan.

References

[1] N. B. Lhachemi, E. H. Nfaoui, and J. Boumhidi, "Hashtag Recommender System Based on LSTM Neural Reccurent Network", In: *Proc. of 2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS)*, 2019, pp. 1-6, doi: 10.1109/ICDS47004.2019.8942380.

[2] S. Bansal and N. Baliyan, "Remembering past and predicting future: a hybrid recurrent neural network based recommender system", *J Ambient Intell Humaniz Comput*, 2022, doi: 10.1007/s12652-022-04375-x.

[3] B. Choe, T. Kang, and K. Jung, "Recommendation System with Hierarchical Recurrent Neural Network for Long-Term Time Series", *IEEE Access*, Vol. 9, pp. 72033–72039, 2021, doi: 10.1109/ACCESS.2021.3079922.

[4] Y. M. Arif, H. Nurhayati, S. M. S. Nugroho, and M. Hariadi, "Destinations Ratings Based Multi-Criteria Recommender System for Indonesian Halal Tourism Game", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 1, pp. 282–294, 2022, doi: 10.22266/IJIES2022.0228.26.

[5] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives", *ACM Computing Surveys*, Vol. 52, No. 1, pp. 1–38, 2019, doi: 10.1145/3285029.

[6] M. Khalaji, C. Dadkhah, and J. Gharibshah, "Hybrid Movie Recommender System Based on Resource Allocation", *arXiv (Cornell University)*, 2021, doi: 10.48550/arXiv.2105.11678.

[7] M. A. Ghazanfar and A. P. Bennett, "Building Switching Hybrid Recommender System Using Machine Learning Classifiers and Collaborative Filtering", *IAENG International Journal of Computer Science*, 2010.

[8] S. Natarajan, S. Vairavasundaram, S. Natarajan, and A. H. Gandomi, "Resolving data sparsity

and cold start problem in collaborative filtering recommender system using Linked Open Data", *Expert Syst Appl*, Vol. 149, 2020, doi: 10.1016/j.eswa.2020.113248.

[9] Z. Romadhon, E. Sediyono, and C. E. Widodo, "Various Implementation of Collaborative Filtering-Based Approach on Recommendation Systems using Similarity", *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, pp. 179–186, 2020, doi: 10.22219/kinetik.v5i3.1062.

[10] E. Çano and M. Morisio, "Hybrid recommender systems: A systematic literature review", *Intelligent Data Analysis*, Vol. 21, No. 6. IOS Press, pp. 1487–1524, 2017, doi: 10.3233/IDA-163209.

[11] H. S. Ramadhan and E. B. Setiawan, "Social Media Based Film Recommender System (Twitter) on Disney+ with Hybrid Filtering Using Support Vector Machine", *Sinkron*, Vol. 8, No. 4, pp. 2215–2225, 2023, doi: 10.33395/sinkron.v8i4.12876.

[12] M. E. B. H. Kbaier, H. Masri, and S. Krichen, "A personalized hybrid tourism recommender system", In: *Proc of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA, IEEE Computer Society*, Jul. 2017, pp. 244–250. doi: 10.1109/AICCSA.2017.12.

[13] S. Manonmani, T. Dharshana, and N. Ragaanasuya, "Movie recommendation system with hybrid collaborative and content-based filtering using convolutional neural network", *Int J Health Sci (Qassim)*, Vol. 6, No. S8, pp. 5357–5372, 2022.

[14] Hanafi, E. Pujastuti, A. Laksito, R. Hardi, R. Perwira, A. Arfriandi, and Asroni, "Handling Sparse Rating Matrix for E-commerce Recommender System Using Hybrid Deep Learning Based on LSTM, SDAE and Latent Factor", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 2, pp. 379–393, 2022, doi: 10.22266/ijies2022.0430.35.

[15] R. Mu, X, Zeng, and L. Han, "A Survey of Recommender Systems Based on Deep Learning", *IEEE Access*, Vol. 6, pp. 69009–69022, 2018, doi: 10.1109/ACCESS.2018.2880197.

[16] A. Ramadhan and E. Setiawan, "Aspect-based Sentiment Analysis on Social Media Using Convolutional Neural Network (CNN) Method", *Building of Informatics, Technology and Science (BITS)*, Vol. 4, No. 4, p. 1828−1836, 2023, doi: 10.47065/bits.v4i4.3103.

[17] H. I. Lee, I. Y. Choi, H. S. Moon, and J. K. Kim, "A multi-period product recommender system in online food market based on recurrent neural networks", *Sustainability (Switzerland)*, Vol. 12, No. 3, 2020, doi: 10.3390/su12030969.

[18] T. T. T. Van and T. N. H. Ly, "A Switching Hybrid Approach to Improve Sparse Data Problem of Collaborative Filtering Recommender System", *Int J Res Appl Sci Eng Technol*, Vol. 8, No. 8, pp. 1060–1064, 2020, doi: 10.22214/ijraset.2020.31095.

[19] T. Donkers, B. Loepp, and J. Ziegler, "Sequential user-based recurrent neural network recommendations", In: *Proc. of the 11th ACM Conference on Recommender Systems, Association for Computing Machinery, Inc*, Aug. 2017, pp. 152–160. doi: 10.1145/3109859.3109877.

[20] T. Singh, A. Nayyar, and A. Solanki, "Multilingual Opinion Mining Movie Recommendation System Using RNN", In: *Proc. of First International Conference on Computing, Communications, and Cyber-Security (IC4S 2019). Lecture Notes in Networks and Systems*, Vol 121, 2019, doi: 10.1007/978-981-15-3369-3_44.

[21] D. Das, H. T. Chidananda, and L. Sahoo, "Personalized movie recommendation system using twitter data", *Advances in Intelligent Systems and Computing, Springer Verlag*, pp. 339–347, 2018, doi: 10.1007/978-981-10-7871-2_33.

[22] A. A. Fakhri, Z. K. A. Baizal, and E. B. Setiawan, "Restaurant Recommender System Using User-Based Collaborative Filtering Approach: A Case Study at Bandung Raya Region", *Journal of Physics: Conference Series, Institute of Physics Publishing*, 2019, doi: 10.1088/1742-6596/1192/1/012023.

[23] G. Ramadhan, "Collaborative Filtering Recommender System Based on Memory Based in Twitter Using Decision Tree Learning Classification (Case Study : Movie on Netflix)", In: *Proc. of 2022 International Conference on Advanced Creative Networks and Intelligent Systems (ICACNIS), Bandung, Indonesia*, pp. 1-6, 2022, doi: 10.1109/ICACNIS57039.2022.10055248.

[24] H. Liu, Z. Hu, A. Mian, H. Tian, and X. Zhu, "A new user similarity model to improve the accuracy of collaborative filtering", *Knowl Based Syst*, Vol. 56, pp. 156–166, 2014, doi: 10.1016/j.knosys.2013.11.006.

[25] S. Pericherla and E. Ilavarasan, "Performance analysis of Word Embeddings for Cyberbullying Detection", *IOP Conf Ser Mater Sci Eng*, Vol. 1085, No. 1, p. 012008, 2021, doi: 10.1088/1757-899x/1085/1/012008.

[26] D. T. Putra and E. B. Setiawan, "Sentiment Analysis on Social Media with Glove Using Combination CNN and RoBERTa", *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, Vol. 7, No. 3, pp. 457-563, 2023, doi: 10.29207/resti.v7iX3.4892.

[27] N. Fatima, "Enhancing Performance of a Deep Neural Network: A Comparative Analysis of Optimization Algorithms", *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, Vol. 9, No. 2, pp. 79–90, 2020, doi: 10.14201/adcaij2020927990.

[28] T. Takase, S. Oyama, and M. Kurihara, "Effective neural network training with adaptive learning rate based on training loss", *Neural Networks*, Vol. 101, pp. 68–78, 2018, doi: 10.1016/j.neunet.2018.01.016.

[29] M. Awad and R. Khanna, "Efficient learning machines: Theories, concepts, and applications for engineers and system designers", *Apress Media LLC*, 2015, doi: 10.1007/978-1-4302-5990-9.