Retweet Prediction Using Multi-Layer Perceptron Optimized by The Swarm Intelligence Algorithm

Jondri¹, Indwiarti², Dyas Puspandari³
¹Data Science, Telkom University, Indonesia
²³Informatics, Telkom University, Indonesia

ABSTRACT
Retweets are a way to spread information on Twitter. A tweet is affected by several features which determine whether a tweet will be retweeted or not. In this research, we discuss the features that influence the spread of a tweet. These features are user-based, time-based and content-based. User-based features are related to the user who tweeted, time-based features are related to when the tweet was uploaded, while content-based features are related to the content of the tweet. The classifier used to predict whether a tweet will be retweeted is Multi Layer Perceptron (MLP) and MLP which is optimized by the swarm intelligence algorithm. In this research, data from Indonesian Twitter users with the hashtag FIFA U-20 was used. The results of this research show that the most influential feature in determining whether a tweet will be retweeted or not is the content-based feature. Furthermore, it was found that the MLP optimized with the swarm intelligence algorithm had better performance compared to the MLP.

Keywords:
Twitter
User-based
Time-based
Content-based
Multi-layer perceptron
Swarm intelligence

1. INTRODUCTION

Twitter is a social media that is widely used by people to interact with each other and share opinions about events. On Twitter, someone can upload tweets containing text, images, or videos with a maximum limit of 280 characters. Information on Twitter flows from followee to follower. Tweets uploaded by someone will appear on their followers' timelines. If the follower agrees with the content of the tweet, then he can retweet the tweet. This kind of action is called a retweet.

Many factors influence whether a tweet will be retweeted or not, including who is tweeting (user-based), the content of the tweet (content-based), when the tweet was uploaded (time-based), and the similarity of the content of previous tweets between followers and followees. The tweet uploader is the main actor in the dissemination of information on Twitter, therefore it is important to understand the influence of users on retweet predictions. The number of followers and followees is one of the factors that influence whether a tweet gets retweets or not [1][2][3][4]. Strong interaction between the uploader and his followers will influence his tweets to be retweeted by his followers [5][6]. A person's activeness in tweeting will influence the tweets they make to get retweets [7]. The similarity between the uploader and his followers, such as similar gender and similar location, also influences the possibility of the uploader's tweet being retweeted by his followers [8]. Someone tends to retweet tweets whose content is similar to the tweets they usually upload [9][8]. The content of the uploaded tweet influences whether a tweet will receive a retweet or not, such as the sentiment of the tweet [4], the arrangement of words used [10][11], the presence of words that reflect the emotions of the uploader or the presence of emoticons [12][13]. When a tweet is uploaded will influence whether the tweet will get retweets or not, tweets uploaded at midnight will get fewer retweets than tweets uploaded in the afternoon [4].
The classification method used to predict whether a tweet will get a retweet or not, among other things, Bayesian Poisson Factorization (BPF) Model [9], Log-Linear Regression [10], Artificial Neural Network [14] [11], Deep Neural Network [1] [3], Support Vector Machine (SVM) optimized by Cuckoo Search algorithm [5], XGBoost[2][13], Logistic Regression[8][11][7], MAKER-RIMER Prediction Model [12], SVM[11][7][4], Random Forest [11] [13] [7][4], Naive Bayes[11] [4], Probabilistic Matrix Factorization Method[6], Decision Tree[7], K-Nearest Neighbors (KNN) [7].

In this research, we will discuss retweet predictions based on user-based, content-based and time-based features. The contribution of this research is to compare the influence of these 3 types of features and that have a high correlation with data classes. The next contribution is that this research uses the MLP classification method with hyperparameters that are optimized using several Swarm Intelligence algorithms. These hyperparameters are the number of hidden layers, number of neurons in the hidden layer, type of an activation function, solver, alpha value and initial learning rate value.

2. THE COMPREHENSIVE THEORETICAL BASIS

2.1. Multi Layer Perceptron (MLP)

Multi-Layer Perceptron (MLP) is a learning method inspired by human brain neurons. In general, an MLP consists of an input layer, several hidden layers, and an output layer[15]. The number of neurons in the input layer depends on the number of features of the data, while the number of neurons in the output layer depends on the problem to be solved using MLP. If the problem is a regression problem, the output layer simply consists of one neuron, whereas if the problem to be solved is a classification problem, then the number of neurons depends on the number of classes of data and settings chosen. An MLP consists of 5 input neurons, 2 The hidden layer containing 3 neurons and the output layer containing 1 neuron can be seen in Figure 1.

![Multi-Layer Perceptron (MLP)](image)

Figure 1. Multi-Layer Perceptron (MLP) with 2 hidden layers with 3 neurons in each layer

Input data \( x_1, x_2, x_3, x_4, x_5 \) will be multiplied by the weight then added, we get

\[
v_j = \sum_{i=1}^{5} w_{ij} x_i + b_j, \quad j = 1, 2, 3
\]

(1)

Where \( w_{ij} \) is the weight between \( x_i, i = 1, 2, 3, 4, 5 \) with \( h_j, j = 1, 2, 3 \) while \( b_j, j = 1, 2, 3 \) is the bias of \( j \)th neuron in the first hidden layer. The result of this sum will be input for the activation function \( f \), we get

\[
h_j = f(v_j), \quad j = 1, 2, 3
\]

(2)

The output from the first hidden layer will be input for the neurons in the second hidden layer,

\[
g_j = f(\sum_{i=1}^{5} w_{ij} h_i), \quad j = 1, 2, 3
\]

(3)

where \( w_{ij} \) is the weight between the first hidden layer neurons and the second hidden layer neurons. Next, the output from the neurons in the second hidden layer will become input for the output neurons, so that we get

\[
y = f(\sum_{i=1}^{5} w_{ij}^2 g_j), \quad j = 1, 2, 3
\]

(4)

Where \( w_{ij} \) is the weight between the second hidden layer neurons and the output layer neurons.

There are several activation functions that can be used, namely, threshold, linear, sigmoid, hyperbolic tangent, etc.

The number of neurons in the hidden layer depends on how the data is spread. If the data is separated linearly by a straight line, then there is no need for a hidden layer, as shown in Figure 2.a. If the data can be separated by \( d \) straight lines, then \( d \) neurons are needed in the hidden layer, as shown in Figure 2.b.
If the number of attributes is only 2 or 3, then a plot can be made of the data so that the MLP architecture can be decided which is appropriate for the data. In real problems, the number of attributes of the data exceeds 3, so a plot cannot be made from the data. This requires trial and error to determine the number of hidden layers and the number of neurons in each hidden layer. Besides the number of hidden layers and the number of neurons in the hidden layer, there are other hyperparameters that influence the performance of an MLP. These hyperparameters are the activation function, optimizer, algorithm, and learning rate value [16]. The range of hyperparameter values can be seen in Table 1.

Table 1. List of hyperparameters and their value limits

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Range of values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of hidden layers</td>
<td>{1,2,3,4,5}</td>
</tr>
<tr>
<td>Number of neurons in the hidden layer</td>
<td>(x</td>
</tr>
<tr>
<td>Activation function</td>
<td>{relu, tanh}</td>
</tr>
<tr>
<td>Solver</td>
<td>{adam, lbfgs}</td>
</tr>
<tr>
<td>Alpha Value</td>
<td>{0.00001, 0.001}</td>
</tr>
<tr>
<td>Initial Learning rate</td>
<td>{0.001, 0.01}</td>
</tr>
</tbody>
</table>

There are thousands of possible combinations of these hyperparameters, making it impossible to manually select the best hyperparameter combination. To overcome this, in this research, a swarm intelligence algorithm was used to select the best combination of hyperparameters.

### 2.2. Swarm Intelligence

Swarm intelligence is an artificial intelligence algorithm that imitates the intelligence of a group of animals, such as ants, fireflies, bats, gray wolves, and others. The working principle of this algorithm is to update the position of each individual so that their fitness is better. In general, the intelligent swarm algorithm is as follows [17]:

a. Initiation of initial position of particle/individual
   - A number of N individuals are selected and their positions are determined randomly in the search space.

b. Evaluate the fitness value of the particle/individual
   - At each particle/individual position, calculate the fitness value. The position of a point represents an MLP architecture. The fitness value of the individual in this position is the F1 score value from the testing data.

c. Position updates
   - The position of the particle/individual is updated with certain rules so that the fitness value of the particle/individual gets better.

d. Determines the stop condition
   - The process is repeated until the stop condition is met. The stopping condition can be that the maximum iteration is reached or if the position of the individual/particle does not change much between two consecutive iterations.

The swarm intelligence algorithms used in this research are Artificial Bee Colony (ABC) [18], Bacterial Foraging Optimization (BFO) [19], Hybrid Bat Algorithm (HBA) [20], Cat Swarm
Optimization (CSO) [21], Cuckoo Search (CS) [22], Firefly Algorithm (FA) [23], Grey Wolf Optimizer (GWO) [24], Particle Swarm Optimization (PSO) [25]. These algorithms will be used in the hyperparameter tuning of MLP.

2.3. Retweet

Retweet is a facility provided by Twitter where someone can re-upload other people's tweets. The mechanism by which retweets occur starts from a followee uploading a tweet. This tweet will appear on the follower's timeline. If the follower agrees with the content of the tweet, then he will retweet the tweet. Not all tweets get many retweets. There are even tweets that don’t get any retweets at all. There are many factors that influence a tweet to get retweets. In this research, the influence of the features of the tweet uploader (user based), the content of the tweet (content based), and when the tweet was made (time based) on the number of retweets obtained were studied. The complete features used can be seen in table 2 [4].

<table>
<thead>
<tr>
<th>Table 2. User based, content based, and time based features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature</td>
</tr>
<tr>
<td>User Based</td>
</tr>
<tr>
<td>UB_1</td>
</tr>
<tr>
<td>UB_2</td>
</tr>
<tr>
<td>UB_3</td>
</tr>
<tr>
<td>UB_4</td>
</tr>
<tr>
<td>UB_5</td>
</tr>
<tr>
<td>UB_6</td>
</tr>
<tr>
<td>UB_7</td>
</tr>
<tr>
<td>UB_8</td>
</tr>
<tr>
<td>CB_1</td>
</tr>
<tr>
<td>CB_2</td>
</tr>
<tr>
<td>CB_3</td>
</tr>
<tr>
<td>CB_4</td>
</tr>
<tr>
<td>CB_5</td>
</tr>
<tr>
<td>CB_6</td>
</tr>
<tr>
<td>CB_7</td>
</tr>
<tr>
<td>CB_8</td>
</tr>
<tr>
<td>CB_9</td>
</tr>
<tr>
<td>CB_10</td>
</tr>
<tr>
<td>CB_11</td>
</tr>
<tr>
<td>CB_12</td>
</tr>
<tr>
<td>CB_13</td>
</tr>
<tr>
<td>CB_14</td>
</tr>
<tr>
<td>TB_1</td>
</tr>
<tr>
<td>TB_2</td>
</tr>
<tr>
<td>TB_3</td>
</tr>
<tr>
<td>TB_4</td>
</tr>
</tbody>
</table>

2.4. Handling Unbalanced Data and Determining System Performance

In many two-class classification problems there are cases where the data is not balanced. The data in one class dominates the other classes. This causes the proposed machine learning method to focus on learning on the majority class. To overcome this, a process was carried out to handle this imbalanced data. There are two methods that can be used, oversampling and undersampling. The oversampling method is a way to deal with unbalanced data by adding data to the minority class so that the amount of data in both classes is equal [26]. The undersampling method is a way to deal with unbalanced data by reducing the data in the majority class so that the data for both classes is balanced [9]. Classification performance is measured by calculating accuracy, precision, recall, and f1 score [27]. In the case of imbalanced data, it is not appropriate to measure performance using accuracy, because the results are influenced by the majority class [28].
3. METHOD

Retweet prediction in this research was carried out in 4 stages, namely crawling Twitter data with the keywords FIFA U20 World Cup, getting features from tweets, tuning MLP hyperparameters, and analyzing the results as seen in figure 3.

![Diagram of the retweet prediction process](image)

Figure 3. Diagram of the retweet prediction process

The first step taken was to crawl tweet data on the topic of the FIFA U20 World Cup. The dataset was taken from the netlytic.org website from May 30, 2023 to June 8, 2023, and was limited to only Indonesian. The data obtained was 870 tweets. These tweets are then grouped into two classes, namely class 0 (tweets do not get retweets) and class 1 (tweets get retweets). From this grouping, 582 class 0 data were obtained, and 288 class 1 data were obtained. After that, we look for user-based, content-based, and time-based features of each tweet. The classification process is carried out based on 3 scenarios in the dataset and 2 scenarios in the classifier. The scenarios in the dataset are unbalanced original data, undersampling data, and oversampling data. Meanwhile, the scenario in the classifier is to compare performance results using the default MLP Python library with MLP which is hyperparameter tuned using the Swarm Intelligence algorithm.

4. RESULT AND DISCUSSION

4.1. Analysis of Features that Influence Retweets

To see the features that influence retweets, experiments were carried out using all features, namely user-based features, content-based features, and time-based features. There are two classifiers used, namely the default MLP in the Sklearn Python library and MLP with hyperparameters optimized using swarm intelligence. MLP optimized with swarm intelligence used is MLP optimized with Artificial Bee Colony (MLP-ABC), MLP optimized with Bacterial Foraging Optimization (MLP-BFO), MLP optimized with Hybrid Bat Algorithm (MLP-HBA), MLP optimized with Cat Swarm Optimization (MLP-CSO), MLP optimized with Cuckoo Search (MLP-CS), MLP optimized with Firefly Algorithm (MLP-FA), MLP optimized with Gray Wolf Optimizer (MLP-GWO), and MLP which is optimized with Particle Swarm Optimization (MLP-PSO). In Table 3, the F1 score values for each of these classifiers are visible. The data used is original data.

<table>
<thead>
<tr>
<th>Feature</th>
<th>MLP</th>
<th>MLP-ABC</th>
<th>MLP-BFO</th>
<th>MLP-HBA</th>
<th>MLP-CSO</th>
<th>MLP-CS</th>
<th>MLP-FA</th>
<th>MLP-GWO</th>
<th>MLP-PSO</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Feature</td>
<td>0.54</td>
<td>0.66</td>
<td>0.7</td>
<td>0.65</td>
<td>0.71</td>
<td>0.74</td>
<td>0.7</td>
<td>0.71</td>
<td>0.64</td>
<td>0.67</td>
</tr>
<tr>
<td>User-Based</td>
<td>0.55</td>
<td>0.63</td>
<td>0.66</td>
<td>0.65</td>
<td>0.65</td>
<td>0.65</td>
<td>0.75</td>
<td>0.71</td>
<td>0.72</td>
<td>0.66</td>
</tr>
<tr>
<td>Time-Based</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
</tr>
<tr>
<td>Content-based</td>
<td>0.67</td>
<td>0.74</td>
<td>0.72</td>
<td>0.69</td>
<td>0.75</td>
<td>0.69</td>
<td>0.74</td>
<td>0.72</td>
<td>0.72</td>
<td>0.72</td>
</tr>
<tr>
<td>Average</td>
<td>0.5425</td>
<td>0.61</td>
<td>0.6275</td>
<td>0.6075</td>
<td>0.615</td>
<td>0.6375</td>
<td>0.6375</td>
<td>0.6425</td>
<td>0.6225</td>
<td></td>
</tr>
</tbody>
</table>

Based on table 3, it can be seen that the best feature used to predict whether a tweet will get a retweet or not is the content based feature with an average F1 score of 0.72. Meanwhile, the best classifier when using all features is MLP-CS with an F1 score of 0.74, MLP-FA when using user based features with an F1 score of 0.75, MLP-CS when using content based features with an F1 score of 0.75. Meanwhile, MLP-GWO is a classifier with the highest average F1 score for various feature combinations with an average F1 score of 0.6425.

4.2. The Effect of Undersampling and Oversampling

The distribution of the data used in this research is not balanced, where class 0 is more numerous than class 1, so undersampling and oversampling methods are used on the training data. The results of the F1 score for handling unbalanced data using the undersampling method can be seen in Table 4, while using the oversampling method can be seen in Table 5.

<table>
<thead>
<tr>
<th>Feature</th>
<th>MLP</th>
<th>MLP-ABC</th>
<th>MLP-BFO</th>
<th>MLP-HBA</th>
<th>MLP-CSO</th>
<th>MLP-CS</th>
<th>MLP-FA</th>
<th>MLP-GWO</th>
<th>MLP-PSO</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Feature</td>
<td>0.49</td>
<td>0.66</td>
<td>0.58</td>
<td>0.67</td>
<td>0.64</td>
<td>0.67</td>
<td>0.59</td>
<td>0.64</td>
<td>0.63</td>
<td>0.62</td>
</tr>
<tr>
<td>User-Based</td>
<td>0.48</td>
<td>0.64</td>
<td>0.63</td>
<td>0.57</td>
<td>0.64</td>
<td>0.65</td>
<td>0.61</td>
<td>0.64</td>
<td>0.64</td>
<td>0.61</td>
</tr>
</tbody>
</table>
Based on Table 4, it can be seen that after undersampling, the best feature used to predict whether a tweet will get a retweet or not is the content-based feature with an average F1 score of 0.67. In fact, the average F1 score when using content-based is better than using all features. Meanwhile, the best classifier if using all features is MLP-CS and MLP-HBA with an F1 score of 0.67, MLP-CS if using user-based features with an F1 score of 0.65, MLP-ABC if using content-based features with an F1 score of 0.72. Meantime, MLP-CS is a classifier with the highest average F1 score for various feature combinations with an average F1 score of 0.645.

Based on Table 5, it can be seen that after oversampling, the best feature used to predict whether a tweet will get a retweet or not is the content-based feature with an average F1 score of 0.687. In fact, the average F1 score when using content-based is better than using all features with an F1 score of 0.65. Meanwhile the best classifier if using all features is MLP-PSO with an f1 score of 0.71, MLP-PSO if using user-based features with an f1 score of 0.69, MLP-GWO and MLP-PSO if using content-based features with an f1 score of 0.71. Meantime, MLP-PSO is a classifier with the highest average F1 score for various feature combinations with an average F1 score of 0.6625.

4.3. Analyze the effect of Hyperparameter Tuning on MLP

The average F1 score value of the default MLP classifier and the MLP whose architecture was optimized using swarm intelligence can be seen in Figure 4.

![Figure 4. Comparison of the average F1 score of the default MLP classifier with MLP optimized with the Swarm Intelligence algorithm for cases of a) raw data, b) undersampling data, c) oversampling data.](image-url)
4.4. Correlation Analysis between Features and Retweet Classes

The correlation value between attributes and output (retweet status) can be seen in Figure 5. Figure 5a shows the correlation between time-based feature attributes and output, Figure 5b shows the correlation between user-based feature attributes and output, while Figure 5c shows the correlation between content-based feature attributes and output. The features with the highest correlation with output (retweet status) are contain_url, len_of_text, con_user_mentioned, contain_excl, contain_location, and contain_org.

![Figure 5. Correlation between output and features a) time-based, b) user-based, c) content-based](image)

The features that have the highest correlation with this output are then used as features for retweet prediction. The classifier used is the classifier with the highest average value for raw data (MLP-GWO), undersampling data (MLP-CS), and oversampling data (MLP-PSO). The F1 score values for these three classifiers can be seen in Table 6.

<table>
<thead>
<tr>
<th>Fitur</th>
<th>MLP-CS</th>
<th>MLP-GWO</th>
<th>MLP-PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 Best features</td>
<td>0.74</td>
<td>0.67</td>
<td>0.66</td>
</tr>
</tbody>
</table>

The best F1 score value is obtained when using the MLP classifier with hyperparameter tuning using the Cuckoo Search algorithm (MLP-CS).

From Figure 4, it can be seen that the performance of the MLP optimized with swarm intelligence is better than the default MLP. These results are consistent for raw data, data that has undergone oversampling and undersampling processes. However, there are two problems that arise when using MLP optimized with swarm intelligence in retweet prediction. The first problem is that the hyperparameters obtained are not global optimum solutions, but local optimum values. This can be seen
from the hyperparameter results obtained which will be different if the experiment is carried out repeatedly. The second problem is in terms of program execution time, MLP optimized with swarm intelligence requires much longer execution time compared to the default MLP F1 score. These two problems can be material for further research studies.

5. CONCLUSION

Hyperparameter tuning on the MLP was proven to improve the performance of the MLP. This is indicated by the F1 score value of the MLP classifier optimized with the swarm intelligence algorithm being greater than the default MLP F1 score. Furthermore, the feature that has the most influence on whether a tweet gets a retweet or not is the content-based feature. The content-based feature is related to the content of the tweet.

ACKNOWLEDGEMENTS

This research was funded by the Telkom University Center for Research and Community Service (PPM) through the Basic Research scheme. We would like to thank the Chancellor of Telkom University for his support for this research.

REFERENCES

Retweet Prediction Using Multi-Layer Perceptron Optimized by The Swarm Intelligence Algorithm

Jondri¹, Indwiarti², Dyas Puspandari²


