Word2Vec Optimization on Bi-LSTM in Electric Car Sentiment Classification

Siti Uswha Hasanah*, Yuliant Sibaroni, Sri Suryani Prasetyowati

School of Computing, Informatics, Telkom University, Bandung, Indonesia
Email: 1*hsnanaa@student.telkomuniversity.ac.id, Yuliant@telkomuniversity.ac.id, Srisuryani@telkomuniversity.ac.id
Correspondence Author Email: hsnanaa@student.telkomuniversity.ac.id

Abstract—The Indonesian government is actively promoting electric vehicles. This policy has generated many sentiments from the public, both positive and negative. Public sentiment can have a significant impact on the success of government policies. Therefore, it is important to understand public sentiment towards these policies. This research develops a sentiment classification model to understand public sentiment towards electric vehicles in Indonesia. Sentiment classification is the process of identifying and measuring the positive or negative sentiment in a text. This research uses a Bi-LSTM model to perform classification on a dataset of tweets related to electric vehicles. To evaluate the performance, testing was conducted through two main scenarios. In Scenario I, the focus was on finding the optimal embedding size for two Word2Vec architectures, namely CBOW and Skip-gram. Model evaluation was performed using cross-validation to gain a deeper understanding of model performance. Scenario II focused on searching for the best dropout parameters for the Bi-LSTM model. This step aimed to find the optimal configuration for the model to generate more accurate and consistent predictions in classifying tweets related to electric vehicles. The results showed that in the context of sentiment classification on tweets about electric vehicles, the combination of CBOW with an embedding size of 200 and the Bi-LSTM model with a Dropout value of 0.2 is the best choice and achieves an accuracy of 96.31%, precision of 92.57%, Recall of 98.61%, and F1-Score of 95.49%.

Keywords: Bi-LSTM; Word2Vec; Sentiment Classification; Electric Car; Tweet

1. INTRODUCTION

At present, the Indonesian automotive industry is planning a transition from internal fuel vehicles to electrification technology, often known as electric cars. This is explained in Presidential Regulation No. 55 of 2019, which accelerates the battery-based electric vehicle program to increase the number of electric vehicles and gradually replace conventional vehicles to reduce air pollution in urban centers [1]. In addition, customers’ thoughts and feelings have an impact on the adoption rate of electric cars in Indonesia; some customers said they "felt good" or "felt guilty" after driving an electric car [2]. Some people believe that electric cars can solve the oil and gas deficit, others argue that going to electric vehicles requires careful preparation, especially infrastructure plans [3]. In the context of the adoption of electric cars in Indonesia, the influence of social media in shaping public perception. Therefore, social media is very suitable to be used as a source of data for this study. Social media, especially Twitter, became an important internet platform in Indonesia. This is because social media is a place for people to express their opinions. These opinions can be analyzed using opinion mining systems, which what can be called sentiment analysis [4]. Sentiment analysis with deep learning methods has become a trend in research. Deep learning methods are gaining popularity in sentiment analysis research due to their ability to handle large and complex datasets [5]. One of the deep learning methods is Long Short-Term Memory.

Research using Bidirectional Long Short-Term Memory (Bi-LSTM) has been conducted by several researchers in the field of sentiment [6]–[8] in several studies showing the advantages of Bi-LSTM compared to other models, not only that the use of Bi-LSTM and Paragraph Vector shows a significant increase in performance that can handle the problem of phrase position faced by Bi-LSTM. Other studies show that using Doppler Domain and Bi-LSTM data achieves an average accuracy of 90%, while coverage domain data reaches 79%. The research [9] used Word2Vec and Bi-LSTM Attention to classify sentiment on IMDB movie reviews. This model produces the best accuracy of 84.12%, higher than previous models.

Other research conducted [10] shows the implementation of Word2Vec and LSTM for sentiment classification in hotel reviews. The results obtained achieved an average accuracy of 85.96%, which was produced using the skip-gram parameter as the Word2Vec architecture. Word2Vec is a vector representation model developed by Google. This model can be used to represent each word as an N-dimensional vector. Word2Vec has two distinct but similar architectures: CBOW and skip-gram [11]. These two architectures work in opposite ways because CBOW is a specific word that is predicted based on its surrounding neighbors, while skip-gram predicts a neighbor word based on a given target word [12].

The use of LSTM combined with Word2Vec has been carried out [13], [14] in-text classification in the fields of security and disease detection from social media posts. According to the study, LSTM Word2Vec with CBOW outperforms LSTM using the Word2Vec skip-gram feature insertion method in terms of performance. The results demonstrate that, in comparison to current methods, the suggested method yields 94% accuracy. As a result, LSTM performs better than other leading methods.

In some of the research that has been discussed, there is an important note regarding the limitations of LSTM and Word2Vec methods in the context of sentiment analysis. Previous research has highlighted that LSTM is less than optimal in capturing the context of words thoroughly and often has difficulty overcoming long distances between words. In addition, recent research highlights that Word2Vec selection has also not reached its optimality.
Although Word2Vec has great potential in representing words as N-dimensional vectors, certain parameters, such as the skip-gram embedding size and Word2Vec, have not been optimized to their full potential. The results of several studies show that the setting of these parameters affects the performance of the model in classifying sentiment. In this study, it is proposed to use Bi-LSTM and Word2Vec to improve the performance of LSTM as used by [15], in the case of electric car sentiment. Word2Vec optimisation is performed by testing various embedding sizes (100, 200, and 300) on both CBOW and Skip-gram architectures. Afterwards, the best parameters from Word2Vec will be applied to test the Dropout parameters (0.2, 0.5, and 0.7) on the Bi-LSTM model. With the ability to process information from front to back and vice versa, Bi-LSTM can effectively capture more complex contexts and improve the long-range dependency problem often faced by LSTM. This research focuses on optimising sentiment classification models using Bi-LSTM and Word2Vec to improve the accuracy and efficiency of sentiment analysis. By improving the accuracy and efficiency of sentiment analysis, this research can contribute to a better understanding of public sentiment towards electric cars in Indonesia. This can help the government and other policymakers to make more informed decisions to promote a smoother transition towards a more sustainable and environmentally-friendly transport system. The resulting model combines the best parameters from Word2vec and Bi-LSTM, with optimal tuning of the Skip-gram and Word2vec embedding sizes, as well as optimal dropout values in the Bi-LSTM layer. The model also provides better results than research conducted by [12], in the case of sentiment classification using the Bi-LSTM model which gets 91.13% accuracy.

2. RESEARCH METHODOLOGY

2.1 System Design

The system flow of this research can be seen in Figure 1. This research illustrates the system workflow that commences with the collection of Twitter tweet data. The subsequent step involves labeling this data, categorizing it into positive and negative sentiments. The labeled data undergoes a series of preprocessing procedures, including tasks such as dataset cleaning, case folding, tokenizing, and other stages. Upon completion of the preprocessing phase, the dataset is partitioned into two segments: training and testing sets. The data allocated for training proceeds to the word2vec embedding optimization stage, where optimal parameters for word2vec are tested in the subsequent phase, involving classification using the Bi-LSTM method. Model evaluation is conducted using the testing data to gauge the performance and effectiveness of the developed model.

![Figure 1. Word2Vec Classification Model on Bi-LSTM](image)

2.2 Collecting Data

This data collection uses data obtained from Twitter data, which contains a mixture of news tweets and public response tweets about electric cars. The process of collecting data is done by crawling data from Twitter using the Pandas and Node.js libraries. The tweet will be downloaded using the tweet-harvest library. This library provides various functions for crawling tweets based on specific keywords and time ranges. In this study, crawling was done with the keyword "electric car" based on tweets from 2020 to October 2023. Next, the prepared data is labeled at a later stage.

2.3 Data Labeling

Data labeling is a process in natural language processing that involves marking or labeling each word in a text based on its syntactic role in a sentence, category, aspect, or opinion. In sentiment labeling, text data will be labeled as positive or negative sentiment. Negative sentiment will be included in opinions that use harsh language or disagreeing sentences. Positive sentiment will encompass viewpoints that contain phrases or words endorsing the existence of electric vehicles [3]. Examples of labeled tweets can be seen in Table 1.

<table>
<thead>
<tr>
<th>Table 1. Electric Car Tweet Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tweets</td>
</tr>
<tr>
<td>setuju si kalo ini jadi trend, gw jga kmrin mudik pakke mobil listrik emang hemat dan nyaman bgt, di tambah lagi bisa cek lokasi SPKLU pakai aplikasi PLN Mobile, jadi praktis banget</td>
</tr>
</tbody>
</table>
2.4 Preprocessing

Pre-processing is a very important stage in the process of classifying text data [16]. Pre-processing is the initial stage in data processing that aims to clean, change format, and reduce the complexity of text data before it is used in analysis. Pre-processing is the process of converting raw data into data that is ready for analysis. This process is typically done to improve data quality, reduce noise, and prepare data for use by machine learning models [4]. The stages of preprocessing carried out in this study are explained as follows:

a) Dataset cleanup

In the dataset cleaning stage, specific steps were taken to clean the data. These include the removal of HTML tags to ensure the data is not contaminated with markup elements, replacing each URL with `<url>`, removing usernames with `<user>`, as well as removing mentions that start with '@'. In addition, this process also includes the removal of non-alphabetic characters such as punctuation marks and other symbols. Table 2 shows an example of the dataset cleaning process.

Table 2. Dataset Cleanup

<table>
<thead>
<tr>
<th>Tweets</th>
<th>Sentiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>@piotrj @nafasidn Apapun itu yg jelas solusinya adalah mobil listrik. Pasti nanti jadi menengkang kualitas udaranya.</td>
<td>Positive</td>
</tr>
<tr>
<td>Ngurangi subsidi transportasi publik tapi ngasih subsidi ke mobil dan motor listrik</td>
<td>Negative</td>
</tr>
<tr>
<td>@BagusBimantara2 @pejaburt Sumpah ini kebijakan tolol banget, gua heran sama pemikiran generasi boomers ini, mobil listrik di subsidi kendaraan umum malah di mahalin hahaha</td>
<td>Negative</td>
</tr>
</tbody>
</table>

b) Case folding

The process of changing every letter in a text to lowercase is known as case folding. This process is carried out to homogenize the shape of letters in the text so that words with the same meaning are considered to have the same features.

c) Tokenizing

Tokenizing is the process of breaking down text or data into smaller pieces, called tokens. A token can be a word, phrase, or other piece of text that has meaning. The main purpose of tokenization is to facilitate text processing, analysis, or data processing.

d) Stopword removal

Stopwords are general words that often appear in a text but lack specific meaning in text analysis or natural language processing. The process of removing stopwords is carried out to eliminate these words from the text thereby improving the quality of text analysis by paying attention to more important words. In this study, researchers used the corpus provided by NLTK.

e) Steaming

Steaming is the process of reducing words to their basic form or root word ("stem") by eliminating word inflection or affixation. In this study, researchers used PorterStemmer() from the NLTK module. An illustration of the stemming process can be seen in Table 3.

Table 3. Stemming

<table>
<thead>
<tr>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mengurangi</td>
<td>Kurang</td>
</tr>
<tr>
<td>terbaru</td>
<td>Terbaru</td>
</tr>
<tr>
<td>menyediakan</td>
<td>sedia</td>
</tr>
</tbody>
</table>

2.5 Word2Vec Embedding Optimization

In this study, the formation of word vectors was carried out using the Word2Vec model, which is a vector representation model developed by Google. Word2Vec can be used to represent any word in vector form with a specific N dimension size [5]. This technique has been widely applied as an initial step towards predictive models in semantic and information retrieval tasks because it can capture contextual word-to-word relationships in multidimensional spaces [9]. The Word2Vec model has two different but similar architectures viz: CBOW and Skip-gram. In this study, both architectures are evaluated to find the best embedding size.
is part of the optimisation as it has a significant impact on the quality of word representation in the Word2Vec model. Illustrations of CBOW and skip-gram architecture can be seen in Figure 2.

These two architectures work in opposite ways because CBOW is a specific word that is predicted based on its surrounding neighbors, while skip-gram predicts a neighbor word based on a given target word [14]. Here is an explanation of each architecture:

1. **Continuous Bag of Words (CBOW)**

   The CBOW architecture works more efficiently on training data in small or medium sizes [14]. This is because CBOW only needs to predict one word at a time. For example, in the sentence “Electric cars are the future”, CBOW will try to predict the word “electricity” based on other words in this sentence.

2. **Skip-gram**

   Skip-gram works the opposite of CBOW. The skip-Gram architecture works efficiently on large amounts of training data. This is because the skip-gram can capture more complex word relationships [14]. For example, if the target word is “electric”, the skip-gram will try to predict words like “car”, “is”, and “future” based on the word “electric”.

3. **Window Size**

   The window size is a parameter that determines the number of neighboring words that will be used to predict a particular word or neighbor words. For example, if window size = 5, then CBOW will use neighboring words that are in positions 0, 1, 2, 3, and 4 of a particular word. Window sizes that are too small can cause Word2Vec models to be unable to capture complex word relationships. Window sizes that are too large can cause Word2Vec models to become too complex and take a long time to train.

4. **Embedding Size**

   The embedding size is a parameter that determines the number of dimensions on the vector of the word representation. For example, if the embedding size = 100, then each word will be represented in vector form with 100 dimensions. Embedding sizes that are too small can cause Word2Vec models to not capture enough information about Word. Embedding sizes that are too large can cause Word2Vec models to become too complex and take a long time to train.

   Previous research, conducted [10] also showed that the selection of Word2Vec was not optimal. The study used Word2Vec with skip-gram architecture to classify sentiment on hotel reviews. The results obtained reached an average accuracy of 85.96%. This shows that there is still potential to improve accuracy by using a different Word2Vec architecture. In this study, researchers implemented Word2Vec using the Gensim library, window size = 5, using CBOW and skip-gram architectures as a comparison at each embedding size of 100, 200, and 300 for each architecture tested.

### 2.6 Bi-LSTM Training Model

An advancement on Long Short-Term Memory (LSTM) is Bidirectional Long Short-Term Memory (Bi-LSTM). A front layer and a back layer, which are accessible both before and after, are combined in Bi-LSTM. LSTM solely utilizes context history, in contrast to Bi-LSTM. As a result, it performs better than LSTM in Bi-LSTM when solving sequential modeling tasks. At the moment, several advancements have been made in the classification of texts using LSTM and Bi-LSTM [18].

Bi-LSTM is one variant of the Long Short-Term Memory (LSTM) architecture used in the field of natural language processing Bi-LSTM can overcome the weaknesses of the LSTM architecture which is only able to process sequences of data in one direction only (forward) [19]. To create a Bi-LSTM network, LSTM neurons are divided into two directions: one to process the sequence of data from beginning to end (forward), and another to process the sequence of data from end to beginning (backward). LSTM is a repeatable neural network architecture designed to solve the vanishing gradient problem in RNNs [20]. Vanishing gradient is a problem that occurs when the value of the learning gradient becomes very small, thus making the training process difficult or even impossible. LSTM solves this problem by using a memory-cell block consisting of an input gate, a forget gate, and an output gate. These memory-cell blocks serve to store information for a long time, thus allowing LSTM to study long-term dependencies [21].
LSTM has the disadvantage of only processing one-way words, namely from start to finish. Figure 3 shows the LSTM and Bidirectional LSTM configurations. Thus, Bi-LSTM can combine information from both directions of the data sequence and produce a richer representation of the input data.

![LSTM and Bi-LSTM configuration](image)

**Figure 3.** LSTM and Bi-LSTM configuration [22]

The parameters in Figure 3 play a key role in this neural network architecture. They serve as important components that facilitate the flow and processing of information from the input layer to the output layer. The following is an explanation of these parameters in the context of the Bi-LSTM network:

a) Input layer (X): This is the input layer that receives the data to be processed by the neural network.

b) Forward layer and Backward layer: These two layers are an integral part of the Bi-LSTM architecture. The forward layer is in charge of processing information sequentially from beginning to end, while the backward layer processes information in reverse, from end to beginning. Both layers consist of Long Short-Term Memory (LSTM) units that receive input from the previous layer or in parallel.

c) Output layer: This layer produces the final output of the network after information processing by both LSTM layers.

d) h (Hidden State): Each LSTM unit has a hidden state that serves as a representation of the information that has been processed at a time interval. This supports the understanding and extraction of patterns from the processed data.

e) C (cell state): Records the information in the LSTM cell during a certain time interval.

f) f, i, o (forgot gate, input gate, output gate): These components are the key elements of each LSTM unit that manage the flow of information within the LSTM cells. Forgot gate is responsible for determining the information that needs to be deleted, the input gate determines the information that needs to be updated, and the output gate determines the information to be revealed.

g) tanh Function: This function generates a candidate vector for updating the cell state on each LSTM unit.

Through the use of these parameters, the Bi-LSTM network can incorporate information from both directions of the data sequence, overcome the limitations of LSTMs that only work in one direction, and produce a richer representation of the input data.

### 2.7 Evaluation

In this study, researchers evaluated the model using cross-validation. Cross-Validation is a model evaluation parameter that indicates the system's ability to make accurate predictions [23]. Researchers evaluated the model using k-fold cross validation to get a more complete picture of the model's performance and reduce the risk of overfitting. In k-fold cross validation, each subset is then used as a test dataset once, while the other k-1 subsets are used as training datasets, as well as removing noise or bias in the words so that a high level of accuracy can be obtained [24]. This process is repeated k times, so each subset is used as a test dataset once. An illustration of K-fold Cross Validation can be seen in Figure 4.

![K-fold Cross Validation](image)

**Figure 4.** 5-Fold Cross Validation illustration

Researchers utilize confusion matrix for evaluation parameters, such as accuracy which measures the model's ability to classify data correctly, precision which assesses the accuracy of positive predictions, recall which measures the model's ability to identify all positive examples, and F1-Score which provides a balanced measure between precision and recall. This confusion matrix provides the number of true and false predictions based on...
actual values. There are 4 values, namely, True Positives (TP), True Negatives (TN), False Positives (FP), and False Negative (FN). An illustration of the confusion matrix can be seen in Figure 5. The following is an explanation of each evaluation parameter in the confusion matrix.

Explanation:

a) True Positive (TP): Indicates the amount of data correctly classified as positive (positive class) by the model.

b) True Negative (TN): Indicates the amount of data correctly classified as negative (negative class) by the model.

c) False Positive (FP): Indicates the amount of data that is incorrectly classified as positive by the model, when it should be negative.

d) False Negative (FN): Indicates the amount of data that is incorrectly classified as negative by the model, when it should be positive.

1. Accuracy
Accuracy is a measure of how accurately a model can classify data. High accuracy indicates that the model rarely makes mistakes in classifying data.

\[
\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative}}
\]

(1)

2. Precision
A metric used to evaluate classification models, precision quantifies how well the model predicts positive classes. The number of accurate positive predictions divided by the total number of positive predictions the model makes yields the precision.

\[
\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}
\]

(2)

3. Recall
Recall is a measure of how well a model can identify positive examples. High recall indicates that models rarely miss positive examples. Recall is an important metric for classification models that aim to detect all positive instances.

\[
\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}
\]

(3)

4. F1-Score
F1-Score is the harmonic mean of recall and precision. This metric provides a balanced measure between the two metrics.

\[
\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]

(4)

![Figure 5. Illustration and Confusion Matrix](image)

3. RESULTS AND DISCUSSION

The study used the Bi-LSTM classification method to analyze a dataset containing tweets about electric cars, with two labels: positive and negative. Testing was conducted in two scenarios:


2. Scenario II: Find the best dropout parameter for the Bi-LSTM model.

This study tested embedding sizes on two types of Word2Vec architecture, CBOW (100, 200, 300), and skip-gram (100, 200, 300), as well as Dropout parameters (0.2, 0.5, 0.7) in the Bi-LSTM model. The main focus of the study was to analyze the impact of parameter changes in Word2Vec and Bi-LSTM on the accuracy of sentiment classification models. The purpose of the second experiment was to assess how differences in Bi-LSTM
parameters impact sentiment classification model performance. The results suggest that the best scheme for sentiment classification can be identified from the results of both scenarios explored.

3.1 Scenario I

Scenario I involves testing on two Word2Vec architectures: Continuous Bag of Words (CBOW) and skip-gram. Within each architecture, variations in embedding size (100, 200, and 300) were evaluated using 5-fold cross validation to see how they affected sentiment classification. Table 1 shows the test results on two Word2Vec architectures.

From the results of the evaluation carried out, the best architecture that emerged was CBOW with an embedding size of 200. In the context of sentiment classification, CBOW with an embedding size of 200 showed consistency and high performance with 88.38% accuracy, 88.14% precision, 87.57% recall, and 87.85% F1-Score. A richer word representation of an embedding size of 200 provides an advantage in understanding sentiment context which results in better results in classifying sentiments in text. Therefore, CBOW with an embedding size of 200 is the best choice for sentiment classification purposes in this study. Details of Word2vec architecture testing can be seen in Table 4.

Table 4. CBOB and Skip-gram Architecture Testing

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBOW 100</td>
<td>87.25%</td>
<td>86.24%</td>
<td>87.69%</td>
<td>86.96%</td>
</tr>
<tr>
<td>CBOW 200</td>
<td>88.38%</td>
<td>88.14%</td>
<td>87.57%</td>
<td>87.85%</td>
</tr>
<tr>
<td>CBOW 300</td>
<td>87.44%</td>
<td>90.73%</td>
<td>84.05%</td>
<td>87.26%</td>
</tr>
<tr>
<td>Skip-gram 100</td>
<td>85.24%</td>
<td>86.71%</td>
<td>85.60%</td>
<td>86.15%</td>
</tr>
<tr>
<td>Skip-gram 200</td>
<td>88.25%</td>
<td>86.05%</td>
<td>91.16%</td>
<td>88.53%</td>
</tr>
<tr>
<td>Skip-gram 300</td>
<td>87.50%</td>
<td>88.50%</td>
<td>87.10%</td>
<td>87.80%</td>
</tr>
</tbody>
</table>

Figure 6 shows graphs of CBOW and skip-gram architectural test results at three embedding sizes, namely 100, 200, and 300. At an embedding size of 200, the CBOW line rises sharply, while the skip-gram line rises only slightly. This shows that CBOW provides a significant performance improvement compared to skip-gram. Thus, the results of this test show that CBOW with an embedding size of 200 gets the best accuracy for sentiment classification.

3.1.1 Scenario II

Scenario II involves testing on a Bi-LSTM model with variations in Dropout values (0.2, 0.5, 0.7) to evaluate model performance on test data. In the results you show in the Table 5 below:

Table 5. Dropout Parameter Test Results

<table>
<thead>
<tr>
<th>Dropout</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>96.31%</td>
<td>95.91%</td>
<td>95.36%</td>
<td></td>
</tr>
<tr>
<td>0.5</td>
<td>92.57%</td>
<td>86.28%</td>
<td>84.74%</td>
<td></td>
</tr>
<tr>
<td>0.7</td>
<td>98.61%</td>
<td>99.62%</td>
<td>99.12%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>95.49%</td>
<td>92.47%</td>
<td>91.37%</td>
<td></td>
</tr>
</tbody>
</table>

These results show that varying the Dropout value has an impact on model performance. Dropout is used to prevent overfitting in neural networks, where the Dropout value controls the proportion of units that are randomly "dropped out" or deactivated at each iteration during the training process. Dropout 0.2 gives the best results with the highest accuracy (96.31%). Whereas in [10], using the LSTM and Word2Vec method, an average accuracy of 85.96% was obtained, which was generated by using the Skip-gram parameter as the Word2Vec architecture testing can be seen in Table 4.
architecture. This proves that Word2Vec optimisation using CBOW architecture with an embedding size of 200 and a dropout parameter of 0.2 in Bi-LSTM provides better results compared to previous research.

![Figure 7. Dropout Parameter Test Results](image)

Based on the accuracy illustration obtained in each dropout value tested in Figure 7, this could indicate that at a dropout rate of 0.2, the model tends to be better at generalizing and reducing overfitting, as more information is dropped out with each iteration, forcing the model to learn more general features and rely less on specific training data.

4. CONCLUSION

This study focuses on classifying sentiments related to electric cars using the Bi-LSTM classification method on a dataset of tweets categorized into positive and negative labels. Two scenarios were run to explore key parameters. Results show in the context of sentiment classification on tweets about electric cars, the combination of CBOW with an embedding size of 200 and a Bi-LSTM model with a Dropout value of 0.2 is the best choice and achieves 96.31% accuracy, 92.57% precision, 98.61% Recall, 95.49% F1-Score. Based on the results obtained, it has proven that Word2Vec optimisation using CBOW architecture with an embedding size of 200 and a dropout parameter of 0.2 in Bi-LSTM provides better results compared to previous research. The conclusions of these two scenarios provide important insights into the influence of parameters in Word2Vec and Bi-LSTM in improving the model’s ability to interpret sentiment in the context of tweets about electric cars. In future studies, researchers are expected to use more than one representation architecture, such as GloVe and FastText.

REFERENCES


