Jurnal Sistem Informasi Bisnis 04(2024)

On-line: http://ejournal.undip.ac.id/index.php/jsinbis/article/view/59436

Impact Of Sarcasm Detection on Sentiment Analysis Using Bi-LSTM and FastText

Junita Amalia*, Dian Filia Matondang, Gilbert E. M. Hutajulu, Agustina Hasibuan

Dept of Information Systems, Institut Teknologi Del

Submitted: November 6th, 2023; Accepted: July 25th, 2024
DOI: 10.21456/vol14iss4pp353-362

Abstract

Sentiment analysis categorizes a collection of texts in a document as either positive or negative. However, sometimes it cannot give accurate results due to sarcastic sentences. Sarcasm involves the use of positive language to convey negative meanings, So sarcasm detection is needed for sentiment classification to provide better results. One method that can be used to perform Sentiment classification is Bidirectional Long Short-Term Memory (Bi-LSTM). However, text data cannot be processed by Bi-LSTM, so it requires word embedding to convert text data into vectors. In this study, the word embedding used is FastText because it can learn the form of words by considering subword information. The results showed that sentiment classification with sarcasm detection could improve evaluation results by 0.08 for precision, 0.07 for recall, 0.07 for F1-score, and 0.07 for accuracy. A paired sample t-test was conducted on precision, recall, F1-score, and accuracy to examine whether there is a difference between sentiment classification with and without sarcasm detection. The obtained p-values are $2.84.10^{-9}$, $4.63.10^{-7}$, and $2.40.10^{-8}$, $6.22.10^{-8}$, respectively. This indicates a difference between sentiment classification with and without sarcasm detection. Therefore, with a 95% confidence level, it can be concluded that sarcasm detection impacts sentiment classification.

Keywords: Bidirectional Long Short-Term Memory; FastText; Accuracy; Sentiment; Sarcasm; Classification

1. Introduction

Sentiments are attitudes, thoughts, or judgments driven by feelings or emotions. Sentiment analysis analyzes a text group in a document and determines that text into positive and negative categories (Samsir et al., 2021). Sentiment analysis is used to classify public opinions toward a subject or object. However, in performing this classification, inaccurate results may occur due to sentences containing sarcasm.

Sarcasm is a form of ironic expression aimed at hurting or mocking someone using positive language with a negative meaning (Alita and Isnain, 2020). Detecting sarcasm is difficult because sarcastic sentences indirectly convey their intended message. Sarcasm causes classification ambiguity, leading and sentiment analysis errors (Kumar et al., 2020). Sarcasm detection is necessary to improve sentiment analysis performance to achieve better text classification results (Muhaddisi et al., 2021). Semantic analysis of words and selecting appropriate algorithms are crucial factors in detecting and classifying sarcasm (Aritonang et al., 2022). Sarcasm detection is essential in sentiment analysis so that there are no errors in sentiment classification caused by sarcasm because sarcasm does not directly allude to the point that causes sentences that should be inculude in the negative label but instead enter the positive label.

A previous study titled "Analisis Sentimen dengan Deteksi Sarkasme pada Komentar Instagram Politikus" aimed to compare the accuracy of sentiment analysis with and without sarcasm detection. The study employed Naive Bayes and Random Forest algorithms for sentiment analysis, while sarcasm detection was performed using the Random Forest method. The research successfully improved the accuracy of sentiment analysis with the inclusion of sarcasm detection using both Naive Bayes and Random Forest methods. The Random Forest method exhibited a 0.7% increase in accuracy, from an initial accuracy of 70.4% to 71.1%. Similarly, utilizing the Naive Bayes method, the accuracy improved by 0.6%, from 66.3% to 66.9% (Muhaddisi et al., 2021).

Another study on sarcasm detection for sentiment analysis titled "Deteksi Sarkasme untuk Analisis Sentimen pada Tweet Berbahasa Indonesia" employed the Random Forest method to classify sarcasm, while sentiment analysis used the Naïve Bayes method with the TF-IDF feature. The research demonstrated a significant increase in accuracy, with a 5.5% improvement resulting in an accuracy value of 80.4%, precision of 83.2%, and recall of 91.3%. For sarcasm detection, cross-validation was utilized, achieving an accuracy of 72.2% (Yunitasari, 2018).

*) Corresponding author: junita.amalia@del.ac.id

Copyright ©2024, JSINBIS, p-ISSN: 2502-2377, e-ISSN: 2088-3587
On-line: http://ejournal.undip.ac.id/index.php/jsinbis/article/view/59436

There is another study on sarcasm detection for sentiment analysis titled "Pendeteksian Sarkasme pada Proses Analisis Sentimen Menggunakan Random Forest Classifier", in this study used a dataset of 2072 samples, with 1023 samples labeled as positive, 587 as negative, and 462 as neutral. Sentiment analysis was performed using the Support Vector Machine method, while sarcasm detection was done using the Random Forest Classifier method. The research increased accuracy by 11.27%, precision by 5.45%, recall by 9.64%, and F1-Score. The sarcasm data from the positive labels resulted in 354 sarcastic samples and 669 non-sarcastic samples (Alita and Isnain, 2020).

From the three studies described, it can be seen how sarcasm affects sentiment classification. In the first study, sentiment classification with sarcasm detection rose by 0.7% using the Random Forest and Naive Bayes methods, up 0.6%. In the second study, up 5.5%, and in the third study, up 11.27%. This suggests that detecting sarcasm in sentiment classification is quite influential and necessary because sarcasm sentences contained in the positive label class interfere with sentiment classification.

Previous research has also been conducted on sarcasm detection titled "Deteksi Sarkasme pada Judul Berita Berbahasa Inggris Menggunakan Algoritma Bidirectional LSTM." This research used English news headlines as the dataset since they use standardized words with correct spelling. The Bidirectional Long Short-Term Memory (LSTM) algorithm was utilized for sarcasm detection. The research achieved an accuracy of 82.55%, precision of 82.35%, recall of 79.53%, and f1-score of 80.92% (Khairi et al., 2022).

Before performing classification, the semantics of the text are extracted using word embeddings. Research "Perbandingan Kinerja Word Embedding Word2Vec, Glove, dan FastText pada Klasifikasi Teks" to compare the performance of word embeddings in text classification. These three methods were chosen because they capture semantic and word order information. The research used 20 newsgroup and router newswire datasets, and F-measure evaluated their performance. The results showed that FastText outperformed the other methods, with an F-measure of 0.979 for the 20 newsgroup dataset and 0.715 for the router dataset (Dharma et al., 2022).

As already explained, the thing behind this study is the classification of inaccurate sentiment analysis with sarcasm. On Twitter, users often give opinions that contain criticism, satire, and insult and express their feelings through tweets, both positive words and negative meanings. But Twitter users also often share their opinions or opinions sarcastically, which means that the Twitter user keeps or disguises the feelings experienced, both anger, afraid, and hurt. This can interfere with the accuracy of the sentiment analysis classification, where tweets are classified as positive but contain negative meanings and cause a decrease in the performance of sentiment analysis. This study chose to use the Bi-LSTM method because Bi-LSTM can cope well with language structures. Also, its architecture can recognize and remember relationships between temporally distant words. This feature helps capture context in tweets that can affect the presence of sarcasm (Yuliska and Syaliman, 2022).

However, text cannot be used casually in Bi-LSTM, so word embedding is needed to convert text data into vectors. In this study, the word embedding that will be used is FastText. FastText is used because it can provide a rich representation of words based on the Word2Vec method. This allows the model to understand better the meaning of words, including words often used sarcastically in tweets. This rich representation of words can improve accuracy in recognizing sarcasm.

There is another study that also discusses sarcasm detection with the title "Pengaruh Hyperparameter Pada Fasttext Terhadap Performa Model Deteksi Sarkasme Berbasis Bi-LSTM" where this study focuses more on the influence of FastText hyperparameters in the context of sarcasm detection. This study shows that fasttext hyperparameters have a different influence on evaluation results, but the increase is more apparent in the use of epoch values with a value of 100 and the use of CBOW architecture. The use of epochs with a value of 100, gives an accuracy of 77%, precision of 70%, recall of 65%, and f1-score of 66%. The use of CBOW architecture gives an accuracy of 77%, precision of 71%, recall of 64%, and f1-score of 65%.

2. Literature Review

2.1. Word Embedding FastText

Researchers use FastText to convert words into vectors. FastText is the word embedding for this research because FastText does not ignore the word order. One advantage of FastText is its ability to handle words that have never appeared before, as its weighting process uses subwords from each sentence. These subwords help break down words into small units that can facilitate the representation and generalization of words with various forms. FastText also has a different approach to representing words compared to word2vec. In FastText, words are represented by summing the values of existing n-grams. This allows words not present in the corpus to still be well represented because some subwords (n-grams) that form the word may appear in the existing n-grams in the corpus (Nurdin et al., 2020; Alfariqi et al., 2020).

The hyperparameters used to form the vector model are as follows (Zalmout and Habash, 2020):

1) Sentences: input words after text preprocessing, such as data cleaning and tokenization.
2) Vector Size: the dimension of the FastText vector model, with a value of 100 for each word.
3) Window: the distance between the current and predicted words, with a window value of 5. This sets the window size used in text feature vector formation (Dharma et al., 2022).
4) Min-count: the frequency value to ignore words that appear below the min-count value of 5. The goal is to eliminate rare or noisy words in the dataset.
5) Iter or epoch: the number of iterations (epochs) performed in model formation, with a value of 5. The amount of data used in this research is small, so using too many epochs can lead to overfitting.
6) Sg or skip-gram: Is the parameter for the FastText model, with a skip-gram value of 0. A value of 0 refers to the CBOW method in model training, where the target word is predicted based on the surrounding context words without considering the word order (Kurbatov et al., 2020).

The parameters used in this study are parameters in the gensim library, namely vector size and sg, where the default vector size value in FastText is 100. This study will use vector size 100, window size 5, min_count = 5, and sg with size 1 (Aritonang et al., 2022).

2.2. Classification using Bi-LSTM

In this study, researchers used Bi-LSTM in classifying sentiment and detecting sarcasm so that bias did not occur. In the context of sentiment classification, Bi-LSTM can study patterns and relationships between words in sentences to determine positive, negative, or neutral sentiments (Yuliska and Syaliman, 2022). In this study, researchers wanted to see if Bi-LSTM could also be used for sarcasm detection.

Sarcasm detection is challenging because sarcasm is often ambiguous and requires a deeper understanding of context. However, Bi-LSTM has a very complex architecture, so the computational load becomes high when the Bi-LSTM algorithm is high. However, this algorithm has advantages compared to other neural network algorithms. The use of word embedding in this algorithm aims to reduce the impact of words that have never appeared before and reduce the burden on the feature engineering process. For example, in text, each word is converted into a vector representing that word, and the embedding layer is placed on the first layer in the model. The bidirectional layer, namely LSTM, is one architecture often used in neural network models to process sequential data. In the Bi-LSTM model, LSTM is used on layers that have two directions. This is done so the model can process data from both directions and better represent sequential data (Kamarula and Rochmawati, 2022).

In building a Bi-LSTM model, the Keras library is added to several layers, such as embedding, Bidirectional layers (LSTM), Dropouts, and Dense:

1) Embedding layers have a purpose in the process of building Bi-LSTM, which is to convert input data into vector form. This is done to facilitate processing by the model. Embedding layers are used on sequential data such as text or voice. For example, in text, each word is converted into a vector representing that word, and the embedding layer is placed on the first layer in the model.
2) Bi-LSTM has two layers: the forward layer and the backward layer. The forward layer captures information from the on-time direction, while the backward layer is the opposite, getting data from the backward time direction. For this study, the unidimensional size of the LSTM used was 64, and recurrrent_dropout = 0.2 (Augustyniak et al., 2019).
3) Dropout is used to prevent overfitting of the dropout layer, and the dropout used is 0.5, whereas in a study with the title "Deepfake tweets classification using stacked Bi-LSTM and words embedding" is said to reduce complexity, than the dropout value used is 0.5. In overfitting conditions, Deep Learning models will produce low errors in the data train but very high errors in the test data (Lim et al., 2019).
4) Dense Layer serves as an output layer that performs operations on the input layer and produces output. Each neuron in the previous layer is connected to a neuron in the next layer by the research to be carried out, namely classification, which uses two classes of labels so that the value of the unit argument is 2 with the activation argument, namely softmax. Softmax is used on Bi-LSTM models when classifying. The softmax function converts input values into probability distributions, where each output value indicates the likelihood that the given input data belongs to the class in question. In other words, the softmax function calculates the relative probability for each category based on a given input value (Sharma et al., 2020).

2.3. Classification Evaluation Method

The researcher utilized Accuracy, Precision, Recall, and F1-score as evaluation metrics in this study. Accuracy evaluates the model's ability to predict the correct class labels overall, representing a percentage of accurate predictions across all classes. Precision focuses on the correctness of optimistic predictions, with higher values indicating better identification of positive instances. Recall measures how well the model identifies all positive cases by calculating the proportion of correctly identified positive samples. F1-score combines precision and recall, providing a balanced measure that considers both aspects. This metric is handy when dealing with

On-line: http://ejournal.undip.ac.id/index.php/jsinbis/article/view/59436

imbalanced datasets. Before calculating these metrics, it is essential to establish the confusion matrix. The references (Mishra et al., 2019; Düntsch and Gediga, 2019; Hasnain et al., 2020) are cited as sources for these evaluation metrics:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

2.4. Paired Sample t-test

The t-test is a commonly used statistical method for comparing the means of two groups. It involves formulating statistical hypotheses to determine whether the proposed hypothesis can be accepted or rejected (Amalia et al., 2023). The t-test consists of two hypotheses: the null hypothesis assumes that the means of the two groups are equal. In contrast, the alternative hypothesis suggests that the standards are statistically different. There are three types of t-tests: one-sample t-test, independent samples t-test, and paired samples t-test. In this study, the researcher utilized a paired samples t-test. This test evaluates the statistical significance of the average difference between two paired observations. It is conducted when the same subjects are measured twice or when two different methods are used. The paired samples t-test requires the paired observations to be continuous, normally distributed variables. The mean, standard deviation, and number of pairs are used to calculate the degree of difference (Mishra et al., 2019). Before performing the paired samples t-test, two conditions must be met (Malmia et al., 2019).

1) The subject data should be interval or ratio data.
2) The two groups of data being compared should follow a normal distribution.

Therefore, before performing the paired samples t-test, it is essential to assess the normality of the data.

3. Methodology

In Figure 1., it can be seen that there are five stages. The stages consist of data preprocessing, text preprocessing, sentiment classification without sarcasm detection, sentiment classification with sarcasm detection, and difference test using a test.

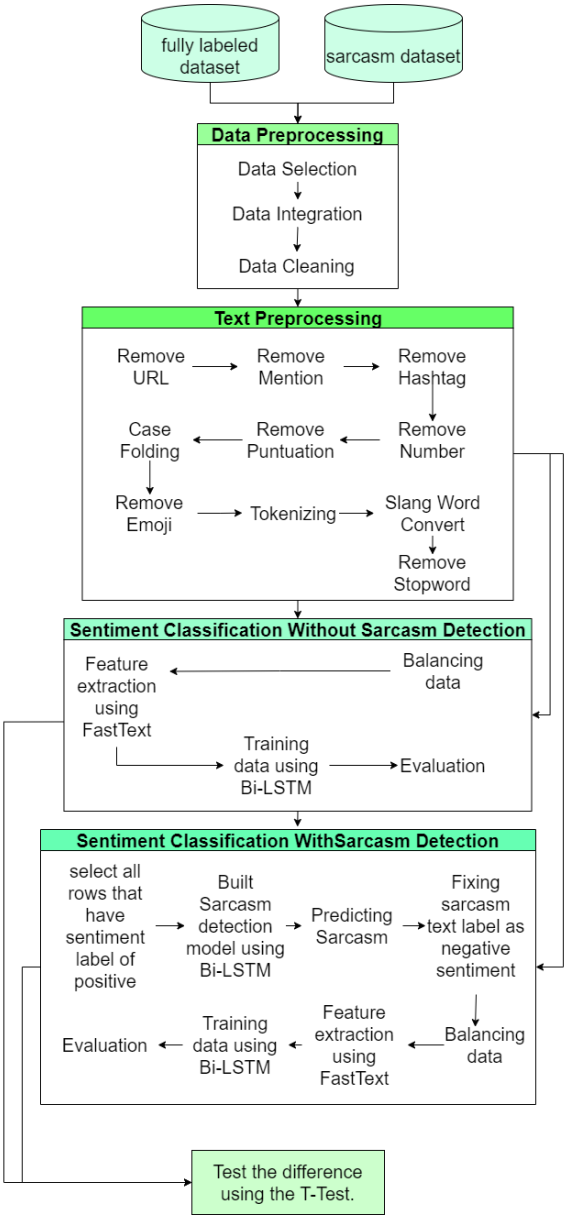

Figure 1. General Research Flow

The following is an explanation of the general research process:

1) The data used in this study is sourced from two datasets: the fully-labeled dataset and the sarcasm dataset. The dataset contains a collection of Indonesian language tweets with pre-labeled information. The total number of rows in the dataset is 4,477 for the fully-labeled dataset and 601 for the sarcasm dataset. Each row in the dataset consists of three attributes: tweet_text, sentiment_label, and sarcasm_label. The sentiment_label has two labels: positive and negative. The positive label is further divided into two categories: sarcasm and non-sarcasm.

On-line: http://ejournal.undip.ac.id/index.php/jsinbis/article/view/59436

2) The next step involves data preprocessing, including data selection, integration, and cleaning.
3) The dataset will go through the following preprocessing steps: removing URLs, removing punctuation, removing hashtags, removing numbers, performing case folding, removing emoticons, tokenizing, converting words, and removing stopwords. These preprocessing steps will result in clean text, which will be used in the two classification stages.
4) The first classification stage is sentiment classification without sarcasm detection. Before performing the classification, the oversampling technique is applied to balance the imbalanced data. After this step, the data will be split, and feature extraction will be performed using FastText. Subsequently, training will be conducted using Bi-LSTM, followed by evaluation.
5) Before proceeding with the second classification, the sarcasm detection model is built first.
6) Then, sentiment classification will be performed using the previously built sarcasm detection model. In this step, the dataset will be filtered by selecting positive sentiment label data, which will be used to detect sarcasm. After sarcasm detection, the detected sarcastic data will be changed to a negative sentiment label. After changing the label of the data, it will be combined with the previous data for sentiment classification. Similar to the previous classification, oversampling technique will be applied to balance the dataset. Data splitting will then be conducted, and feature extraction using FastText will be performed. After vectorization, the data will undergo training using Bi-LSTM, followed by evaluation.
7) After the evaluation stage, a t-test will be conducted to compare the results before and after sarcasm detection. The t-test will be performed for each classification, and 10 evaluation results will be collected. This process is carried out to assess whether sarcasm detection significantly affects the sentiment classification results.

3.1 Data Preprocessing

The data preprocessing process on both datasets to be studied is carried out with the aim of obtaining clean data as input in sarcasm detection research. The stages include data integration, data cleaning, and data selection.

1) Data Selection, in this stage the drop column command removes attributes not used in research during data selection. In this stage, researchers remove the data's features not used in the study, namely id, text, emojis, and emoji_label (Garcia et al., 2015).
2) Data Integration, in this study researchers used data integration to aggregate two datasets: fully-labeled-dataset and sarcasm_dataset (Garcia et al., 2015).
3) Data Cleaning, aims to facilitate models use of cleaner data. If used in modeling, duplicate data will lead to a more complex need to execute the same word. For this reason, the data cleaning stage will be helpful to check whether there is repeated data and take just one sentence to improve model performance (Han et al., 2012).

Table 1. displays the data selection, integration, and cleaning results. In the selection data, the attributes selected for use in research are tweet_text, sentiment_label, and sarcasm_label. It displays the results of combining data from both datasets for data integration. The data cleaning team has cleaned the data of duplicate and null values.

Table 1. Results of Data Preprocessing

Dataset	**Sentiment Classification Without Sarcasm Detection**		
	Data Selection	**Data integration**	**Data Cleaning**
fully-labeled-dataset	tweet_text, sentiment_label, sarcasm_label	tweet_text, sentiment_label, sarcasm_label	Missing values = 0.0%
sarcasm_dataset	tweet_text, sentiment_label, sarcasm_label	tweet_text, sentiment_label, sarcasm_label	Duplicate values = 0

3.2 Text Preprocessing

Text in user posts on Twitter's social media platform often uses non-standard language, containing new words, URLs, abbreviations, and emojis. Before using data, text preprocessing is done first. In this study, text preprocessing aims to clean data from noise, such as checking text and spelling or reducing text from repeated words. This study will use Several techniques to perform text preprocessing, such as remove URL, remove mention, remove hashtag, remove number, remove punctuation, case folding, remove emoticon, tokenizing, convert word, and remove stopwords aiming to enhance the dataset's quality.

1) Remove URL, in text classification URLs cannot be used as test data because they do not convey the sentiment of Twitter text. Entering URLs in the classification process can cause errors in classifying data during the training phase. Therefore, URLs must be removed to avoid impacting the model's performance in Twitter's text sentiment classification process (Saragih et al., 2022).
2) Remove Mention, when performing sentiment classification on Twitter text, the username role is no longer required for training. Therefore, the words that are shown as user names are removed. Many rows of data containing symbols can interfere with the classification process in the dataset used for this study (Saragih et al., 2022).
3) Remove Hashtag can help remove metadata that may be unwanted or irrelevant for a particular

purpose. In this scenario, removing hashtags can help make the text easier to read and understand (Aritonang et al., 2022).

4) Remove Number, the process of removing numbers is carried out to eliminate numeric values in the data. In this classification of studies, the text plays an important role. Therefore, it is necessary to remove the numberes from the existing data (Saragih et al., 2022).
5) Remove Punctuation, the process of removing punctuation is carried out to eliminate punctuation marks from the data. Punctuation marks can cause errors in classification by the model because they introduce additional characters that need to be processed but do not significantly affect the classification. Therefore, removing punctuation marks is necessary to streamline the classification process (Saragih et al., 2022).
6) Case Folding converts text characters into the same format to facilitate text processing and analysis. In sentiment and sarcasm analysis, every word and sentence in the text is processed and analyzed to determine the polarity of sentiment or tone and the use of sarcastic language. To facilitate word processing, it is necessary to standardize the use of upper or lower case letters in the first letter (Saragih et al., 2022).
7) Remove emoticon, emoticons represent facial expressions such as smiling, crying, laughing, and whatnot. Removing emoticons can facilitate text processing and can improve accuracy in determining sentence sentiment (Aritonang et al., 2022).
8) Tokenizing is the process of converting data into individual words or tokens. At this stage, the method divides each sentence into independent words. Without tokenization, the model cannot process the text and cannot serve weighting for each term. Consequently, models must be more effectively trained to detect sentiment and sarcasm (Saragih et al., 2022).
9) Slang word converts, using KBBI, serve to convert informal speech into formal terms in the data set. This conversion aims to ensure that the colloquial words in the sentence carry their intended meaning. Researchers used Slang word conversion because the dataset contained many informal words. KBBI lists everyday terms and their standard equivalents used in this conversion process (Aritonang et al., 2022).
10) Remove Stopwords is one of the processes that will be used to select the right words to represent each critical piece of information in a text document. This process involves removing irrelevant words from the text by checking the words in the sentence against a list of unimportant words (Aritonang et al., 2022).

This is Table 2., as result of the text preprocessing process.

Table 2. Results of Text Preprocessing

No	Text Preprocessing	Result
1	Initial text	@semproelz Alergi sama arab tapi ngarep investasinya, bgitu investasinya kecil...eeh nyesel udah mayungin Raja Salman😂😂 #RadikalNutupIsuEkonomi #RadikalNutupIsuEkonomi
2	Remove URL	@semproelz Alergi sama arab tapi ngarep investasinya, bgitu investasinya kecil...eeh nyesel udah mayungin Raja Salman😂😂 #RadikalNutupIsuEkonomi #RadikalNutupIsuEkonomi
3	Remove mention	Alergi sama arab tapi ngarep investasinya, bgitu investasinya kecil...eeh nyesel udah mayungin Raja Salman😂😂 #RadikalNutupIsuEkonomi #RadikalNutupIsuEkonomi
4	Remove hashtag	Alergi sama arab tapi ngarep investasinya, bgitu investasinya kecil...eeh nyesel udah mayungin Raja Salman😂😂
5	Remove numbering	Alergi sama arab tapi ngarep investasinya, bgitu investasinya kecil...eeh nyesel udah mayungin Raja Salman😂😂
6	Remove punctuation	Alergi sama arab tapi ngarep investasinya bgitu investasinya kecil eeh nyesel udah mayungin Raja Salman😂😂
7	Case Folding	alergi sama arab tapi ngarep investasinya bgitu investasinya kecil eeh nyesel udah mayungin raja salman😂😂
8	Remove emoticon	alergi sama arab tapi ngarep investasinya bgitu investasinya kecil eeh nyesel udah mayungin raja salman
9	Tokenization	[‘alergi’, ‘sama’, ‘arab’, ‘tapi’, ‘ngarep’, ‘investasinya’, ‘bgitu’, ‘investasinya’, ‘kecil’, ‘eeh’, ‘nyesel’, ‘udah’, ‘mayungin’, ‘raja’, ‘salman’]
10	Slang word convert	[‘alergi’, ‘sama’, ‘arab’, ‘tapi’, ‘berharap’, ‘investasinya’, ‘begitu’, ‘investasinya’, ‘kecil’, ‘eeh’, ‘menyesal’, ‘sudah’, ‘memayungi’, ‘raja’, ‘salman’]
11	Remove stopwords	[‘alergi’, ‘sama’, ‘arab’, ‘tapi’, ‘berharap’, ‘investasinya’, ‘begitu’, ‘investasinya’, ‘kecil’, ‘menyesal’, ‘memayungi’, ‘raja’, ‘salman’]

3.3 Oversampling

Oversampling aims to balance the dataset so that it is better and produces more accurate predictions. Dataset imbalance occurs when the number of datasets in a positive and negative class is out of balance. Table 3. shows that the number of positive classes is 1928 and 3141 for negative classes, indicating that the datasets in this study need to be balanced. To overcome the data imbalance, oversampling

On-line: http://ejournal.undip.ac.id/index.php/jsinbis/article/view/59436

techniques are performed. Oversampling was chosen because the data used in the study was not balanced, which caused the model evaluation results on Bi-LSTM to be less good. To overcome this, researchers apply oversampling to balance the data, and the model can provide good results. Oversampling is done to correct data imbalances by increasing the number of samples in minority classes.

Table 3. Total Combined Amount of Fully Labelled Dataset and Sarcasm

Description	Total
Positive	1928
Negative	3141

4. Result and Discussion

In this chapter, the results of sentiment classification with sarcasm detection and without sarcasm detection will be displayed. The results of the Paired Sample T-Test statistical test will also be shown in this chapter.

4.1. Sentiment Classification Without Sarcasm Detection

This classification was done to test sentiment classification without a model to detect sarcasm. The results of this classification can be seen in Table 4.:

Table 4. Sentiment Classification Without Sarcasm Detection

Running	Sentiment Classification Without Sarcasm Detection			
	Precision	Recall	F1-score	Accuracy
1	0.80	0.80	0.80	0.80
2	0.82	0.82	0.82	0.82
3	0.82	0.83	0.81	0.81
4	0.81	0.81	0.83	0.83
5	0.81	0.82	0.81	0.83
6	0.82	0.81	0.83	0.81
7	0.80	0.83	0.80	0.80
8	0.80	0.80	0.80	0.80
9	0.83	0.82	0.82	0.82
10	0.83	0.81	0.81	0.82
Average	0.81	0.81	0.81	0.81

Good evaluation results were obtained after running ten times on the sentiment classification algorithm without sarcasm detection. Average precision of 0.81, recall of 0.81, f1-score of 0.81, and accuracy of 0.81. These results show that our model can recognize sentiment well, with a relatively low error rate.

4.2. Sentiment Classification With Sarcasm Detection

This classification is done to test sentiment classification with models to detect sarcasm. The results of this classification can be seen in Table 5.:

Table 5. Sentiment Classification With Sarcasm Detection

Running	Sentiment Classification With Sarcasm Detection			
	Precision	Recall	F1-score	Accuracy
1	0.90	0.90	0.90	0.90
2	0.91	0.88	0.88	0.89
3	0.88	0.89	0.89	0.89
4	0.90	0.88	0.89	0.88
5	0.89	0.87	0.88	0.88
6	0.90	0.89	0.89	0.89
7	0.89	0.87	0.87	0.87
8	0.89	0.89	0.89	0.89
9	0.89	0.89	0.89	0.89
10	0.90	0.90	0.90	0.90
Average	0.89	0.88	0.88	0.88

After running the sentiment classification algorithm with sarcasm detection for ten times, the evaluation results obtained showed improvement compared to the previous classification. The average precision achieved was 0.89, recall was 0.89, f1-score was 0.88, and accuracy reached 0.88. These results indicate that the developed model can detect sarcasm sentiments well, with a relatively low error rate.

Previous research conducted by Aritonang et al., (2022) recorded an accuracy of 0.77, precision of 0.70, recall of 0.65, and f1-score of 0.66 using an epochs value of 100. However, through this study, there was a significant increase in evaluation values, namely an increase in accuracy by 0.11, precision by 0.19, recall by 0.24, and f1-score by 0.22. This change can be attributed to the use of a lower epochs value of 5. By reducing the epochs value, the model converges faster, thereby increasing efficiency in the training process. This can result in improved performance of the model in detecting sarcasm sentiment.

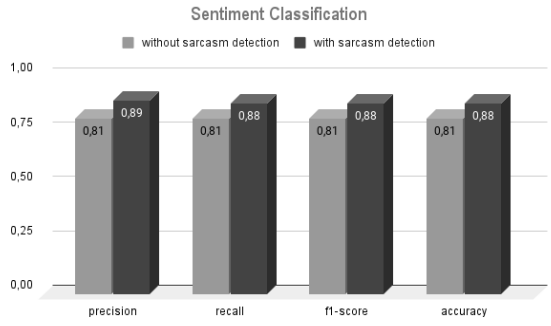

Figure 2. Sentiment Classification

Figure 2. represents the evaluation results of sentiment classification without sarcasm detection and with sarcasm detection displayed in data visualization to demonstrate significant differences. It can be observed that the average sentiment classification results without and with sarcasm detection are shown for each evaluation. Sentiment classification performed with sarcasm detection is superior to sentiment classification without sarcasm detection. The evaluation results are better because, in each

classification performed, oversampling has been applied to balance the data. For sentiment classification without sarcasm detection, the data with negative sentiment labels amounted to 3,112, while positive labels amounted to 1,918. The same thing also happens in sentiment classification with sarcasm detection, where changing labels makes the number of positive sentiment labels smaller, namely 934, and the number of positive labels is 4,096. If unresolved, data imbalance occurs, making the classification evaluation results poor. By applying oversampling techniques, researchers can produce better evaluations than if they do not apply oversampling techniques. Another thing that also affects the evaluation results to be better is that the model has recognized sarcasm and can label negative sentiments in texts that detect sarcasm. This indicates that sarcasm detection is necessary for sentiment classification to improve sentiment classification results.

4.3. Paired sample t-test

In this study, a difference test was carried out to evaluate each classification using paired samples t-test to assess the effect of sarcasm detection on sentiment classification. There are two conditions for the classification to be carried out: sentiment classification without sarcasm detection and sentiment classification without sarcasm detection. Each classification will run ten times; paired samples t-test analysis will be carried out, we can see in the Table 6.

Table 6. Sentiment Classification Without and With Sarcasm Detection

Evaluation	Sentiment Classification Without and With Sarcasm Detection		
	T Stat	**P Value**	**T Table**
Precision	21.094	21.094×10^{-9}	1.833
Recall	11.743	4.630×10^{-7}	1.833
F1-Score	16.542	2.400×10^{-8}	1.833
Accuracy	14.833	6.220×10^{-8}	1.833

The following hypothesis is used to see the difference in sentiment classification with and without sarcasm detection.

H_0: There is no difference in evaluation values without and with sarcasm detection.

H_1: The evaluation value with sarcasm detection is more excellent than without.

The basis for decision making (Djam'an et al., 2021).

- If the p-value < 0.05, then H_0 is rejected
- If the p-value > 0.05, then H_0 is accepted

The p-value for all evaluation results inTable 6. is smaller than 0.05, indicating that we can learn the following:

1) Reject H_0 on the accuracy hypothesis, meaning that the accuracy value with sarcasm detection is more excellent than without sarcasm detection.
2) Reject H_0 on the precision hypothesis, which means that the precision with sarcasm detection is greater than the precision without sarcasm detection value.
3) Reject H_0 on the recall hypothesis, meaning that the recall value with sarcasm detection is more excellent than without sarcasm detection.
4) Reject H_0 on the f1-score hypothesis, which means that the value of the f1-score with sarcasm detection is greater than the f1-score value without sarcasm detection

So there is a significant difference between the results of sentiment classification without sarcasm detection and with sarcasm detection. This can also happen because the model has recognized sarcasm and can label negative sentiments on a text that detects sarcasm. This indicates that sarcasm detection is needed in sentiment classification to improve the classification results on sentiment.

5. Conclusion

This study aimed to investigate the impact of sarcasm detection on sentiment classification using the Bi-LSTM and FastText methods, focusing on the results of classification evaluation. The findings reveal that sentiment classification with sarcasm detection outperforms sentiment classification without sarcasm detection. The evaluation results for sentiment classification without sarcasm detection yielded an average accuracy, precision, recall, and f1-score of 0.81. In contrast, sentiment classification with sarcasm detection achieved accuracy, precision, recall, and f1-score of 0.88, 0.89, 0.88, 0.88 respectively. There was an increase in evaluation results on sentiment classification with and without sarcasm detection, namely 0.08 on precision, 0.07 on recall, 0.07 on F1-score, and 0.07 on accuracy. Additionally, the hypothesis tests conducted to assess the accuracy, precision, recall, and f1-score differences, as described in the classification difference test subchapter, further support the superiority of sentiment classification with sarcasm detection. The test results rejected the null hypothesis (H_0) for all evaluation metrics, indicating a significant distinction between sentiment classification without sarcasm detection and sentiment classification with sarcasm detection. Another factor contributing to the improved evaluation metrics in sentiment classification with sarcasm detection is the model's ability to recognize sarcasm. Positive sentiment texts are identified as sarcasm and converted into negative labels. Hence, sarcasm detection is crucial for enhancing the accuracy of sentiment classification results.

Acknowledgments

Thank you for being so supportive in implementing the "Impact Of Sarcasm Detection on

Sentiment Analysis Using Bi-LSTM and FastText" research to the Lembaga Penelitian dan Pengabdian kepada Masyarakat (LPPM) Institut Teknologi Del so that this paper can be written well.

References

Alfariqi, F., Maharani, W., Husen, J.H., 2020. Klasifikasi Sentimen pada Twitter dalam Membantu Pemilihan Kandidat Karyawan dengan Menggunakan Convolutional Neural Network dan Fasttext Embeddings. *E-Proceeding of Engineering*, *7*(2), 8052-8062.

Alita, D., Isnain, A.R., 2020. Pendeteksian Sarkasme pada Proses Analisis Sentimen Menggunakan Random Forest Classifier. *Jurnal Komputasi*, *8*(2), 50-58. https://doi.org/10.23960/komputasi.v8i2.2615

Amalia, J., Fitriyaningsih, I., Agnesia, Y., 2023. *Buku Ajar Probabilitas dan Statistika*. Makasar: PT. Nas Media Indonesia.

Aritonang, Y.V., Napitupulu, D.P., Sinaga, M.H., Amalia, J., 2022. Pengaruh Hyperparameter pada Fasttext terhadap Performa Model Deteksi Sarkasme Berbasis Bi-LSTM. *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, 9(3), 2612-2625. https://doi.org/10.35957/jatisi.v9i3.1331

Augustyniak, L., Kajdanowicz, T., Kazienko, P., 2019. Aspect detection using word and char embeddings with (Bi) LSTM and CRF. *Proceedings - IEEE 2nd International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2019*, 43-50. https://doi.org/10.1109/AIKE.2019.00016

Dharma, E.M., Gaol, F.L., Warnars, H.L.H.S., Soewito, B., 2022. The Accuracy Comparison Among Word2Vec, Glove, and Fasttext Towards Convolution Neural Network (CNN) Text Classification. *Journal of Theoretical and Applied Information Technology*, 100(2), 349-359.

Djam'an, N., Asdar, Nasullah, Djadir, Fauzan, M., 2021. The Effect of Online Learning using Zoom on Students' Learning Outcomes. *International Conference on Educational Studies in Mathematics (ICoESM 2021)*, 611, 92-96.

Düntsch, I., Gediga, G., 2019. Confusion Matrices and Rough Set Data Analysis. *Journal of Physics: Conference Series*, 1229(1). https://doi.org/10.1088/1742-6596/1229/1/012055

Garcia, S., Luengo, J., Herrera, F., 2015. Data Preprocessing in Data Mining. I*ntelligent Systems Reference Library*, (10). https://doi.org/10.1007/978-3-319-10247-4

Han, J., Kamber, M., Pei, J., 2012. Third Edition : *Data Mining Concepts and Techniques*. San Fransisco: Morgan Kaufmann Publishers (an imprint of Elsevier).

Hasnain, M., Pasha, M.F., Ghani, I., Imran, M., Alzahrani, M.Y., Budiarto, R., 2020. Evaluating Trust Prediction and Confusion Matrix Measures for Web Services Ranking. *IEEE Access*, 8, 90847-90861. https://doi.org/10.1109/ACCESS.2020.2994222

Kamarula, M.R.F., Rochmawati, N., 2022. Perbandingan CNN dan Bi-LSTM pada Analisis Sentimen dan Emosi Masyarakat Indonesia Di Media Sosial Twitter Selama Pandemik Covid-19 yang Menggunakan Metode Word2vec. *Journal of Informatics and Computer Science*, 4(2), 219-228. https://doi.org/10.26740/jinacs.v4n02.p219-228

Kumar, A., Narapareddy, V.T., Srikanth, V.A., Malapati, A., Neti, L.B.M., 2020. Sarcasm Detection using Multi-Head Attention Based Bidirectional LSTM. *IEEE Access*, 8, 6388-6397. https://doi.org/10.1109/ACCESS.2019.2963630

Kurbatov, Y., Rytsarev, I., Kupriyanov, A., 2020. Research of Text Data Processing Algorithms in Social Networks. *Proceedings of ITNT 2020 - 6th IEEE International Conference on Information Technology and Nanotechnology*, 1. https://doi.org/10.1109/ITNT49337.2020.9253271

Lim, E., Setiawan, E. I., Santoso, J., 2019. Stance Classification Post Kesehatan di Media Sosial Dengan FastText Embedding dan Deep Learning. *Journal of Intelligent System and Computation*, 1(2), 65-73. https://doi.org/10.52985/insyst.v1i2.86

Malmia, W., Makatita, S.H., Lisaholit, S., Azwan, A., Magfirah, I., Tinggapi, H., Umanailo, M.C.B. 2019. Problem-Based Learning as an Effort to Improve Student Learning Outcomes. *International Journal of Scientific and Technology Research*, 8(9), 1140-1143.

Mishra, P., Singh, U., Pandey, C.M., Mishra, P., Pandey, G., 2019. Application of student's t-test, analysis of variance, and covariance. *Annals of Cardiac Anaesthesia*, 22(4), 407-411. https://doi.org/10.4103%2Faca.ACA_94_19

Khairi, M.A., Munandar, T.A., Setiawati, S., 2022. Implementasi Augmented Reality untuk Pengembangan Aplikasi Pengenalan Senjata Tradisional Kujang. *Journal of Dinda*, 2(2), 82-89.

Muhaddisi, A., Prastowo, B.N., Putri, D.U.K., 2021. Analisis Sentimen dengan Deteksi Sarkasme pada Komentar Instagram Politikus. *Skripsi*. Yogyakarta: Universitas Gadjah Mada.

Nurdin, A., Aji, B.A.S., Bustamin, A., Abidin, Z., 2020. Perbandingan Kinerja Word Embedding Word2vec, Glove, dan Fasttext pada Klasifikasi Teks. *Jurnal TEKNOKOMPAK*, 14(2), 74-79. https://doi.org/10.33365/jtk.v14i2.732

Samsir, Ambiyar, Verawardina, U., Edi, F., Watrianthos, R., 2021. Analisis Sentimen Pembelajaran Daring pada Twitter di Masa Pandemi COVID-19 Menggunakan Metode Naïve Bayes. *JURNAL MEDIA INFORMATIKA*

BUDIDARMA, 5(1), 157-163.

Saragih, L., Nababan, M., Simatupang, Y., Amalia, J., 2022. Analisis Self-Attention Pada Bi-Directional Lstm Dengan Fasttext dalam Mendeteksi Emosi Berdasarkan Text. *ZONAsi: Jurnal Sistem Informasi*, 4(2), 144-156. https://doi.org/10.31849/zn.v4i2.10846

Sharma, S., Sharma, S., & Anidhya, A. (2020). Activation Functions in Neural Networks. *International Journal of Engineering Applied Sciences and Technology*, 4(12), 310-316. http://dx.doi.org/10.33564/IJEAST.2020.v04i12.054

Yuliska, Syaliman, K.U., 2022. Peringkasan Dokumen Teks Otomatis Berdasarkan Sebuah Kueri Menggunakan Bidirectional Long Short Term Memory Network. Journal of Information Technology and Computer Science (INTECOMS), 5(2), 65-71. http://dx.doi.org/10.30865/mib.v5i1.2580 https://doi.org/10.31539/intecoms.v5i2.4729

Yunitasari, Y., Musdholifah, A., Sari, A.K., 2018. Deteksi Sarkasme untuk Analisis Sentimen pada Tweet Berbahasa Indonesia. *Tesis*. Yogyakarta: Universitas Gadjah Mada.

Zalmout, N., Habash, N., 2020. Utilizing Subword Entities in Character-Level Sequence-to-Sequence Lemmatization Models. *COLING 2020-28th International Conference on Computational Linguistics, Proceedings of the Conference*, 4676-4682. https://doi.org/10.18653/v1/2020.coling-main.412