Available online at https://journal.rescollacomm.com/index.php/ijqrm/index

International Journal of Quantitative Research and Modeling

e-ISSN 2721-477X
p-ISSN 2722-5046

Vol. 5, No. 3, pp. 295-306, 2024

Semantic Classification of Sentences Using SMOTE and BiLSTM

Irvan Tanjung[1*], Ridwan Ilyas[2], Melina[3]

[1,2,3]*Department of Informatics, Faculty of Science and Informatics, Universitas Jenderal Achmad Yani, Cimahi, Indonesia*
** Corresponding author email : irvantj99@gmail.com*

Abstract

A paraphrase is a sentence that is re-expressed with a different word arrangement without changing its meaning (semantics). To find out the semantic proximity to the pair of citation sentences in the form of paraphrases, a computational model is needed. In doing classification sometimes appears a problem called Imbalance Class, which is a situation in which the distribution of data of each class is uneven. There are class groups that have less data (minorities) and class groups that have more data (majority). Any unbalanced real data can affect and decrease the performance of classification methods. One way to deal with it is using the SMOTE method, which is an over-sampling method that generates synthesis data derived from data replication in the minority class as much as data in the majority class. The study applied SMOTE in the classification of semantic proximity of citation pairs, used Word2Vec to convert words into vectors, and used the BiLSTM model for the learning process. The research was conducted through 8 different scenarios in terms of the data used, the selection of learning models, and the influence of SMOTE. The results showed that scenarios using previous research data with BiLSTM and SMOTE models provided the best accuracy and performance.

Keywords: BiLSTM, imbalance class, semantics, SMOTE, Word2Vec

1. Introduction

Paraphrases are a form of re-disclosure of sentences or phrases with different word arrangements without changing the meaning they contain (Bhagat, 2013). In the writing of scientific works can usually be found paraphrases that are a way to cite a source of scientific work done by others indirectly containing semantics. Citation is one of the skills that a person needs in writing scientific papers to show the reader the basic theories put forward or research that has been done by others that can strengthen the newness of the research conducted (Teufel, 2006).

Measuring semantic similarities between textual items (words, sentences, paragraphs, or documents) is a very important area of research in Natural Language Processing (NLP). Semantic Textual Similarity (STS) aims to measure the degree of semantic similarity between sentence pairs based on the level of similarity that has been defined in a scale from 0 to 5, where each score range represents a given relationship, i.e. Equivalent, Similar, Specific, No Alignment, Related, and Opposite (Aguirre, 2017).

The study proposed the SMOTE and BiLSTM methods for measuring and classifying the semantic proximity of citation pairs in English scientific papers. The SMOTE method is applied in this study because the data used is imbalance class so it needs to be balanced. Then, the method for converting words into vectors uses Word2Vec as a feature extraction. The reason for choosing Word2Vec even though BERT is superior is because it requires considerable processing which makes it not enough time to complete this research so Word2Vec was selected. As for training and modeling using BiLSTM deep learning. The application of these methods in this study is used to test whether the results of accuracy and classification performance can improve or not from previous studies. The results of this study can be utilized to find out the level of closeness of a pair of citation sentences to support or fulfill the research to be built, especially in the creation of scientific papers.

2. Literature Review

Several previous studies have conducted studies to measure semantic similarity using Siamese Neural Network architectures with the GRU model (Ichida, 2018) and semantic classification of text on documents using Hierarchical Bidirectional LSTM with attention mechanisms (Shi, 2019). These methods have each provided good performance and provided ease in measuring or classifying semantic similarities. In addition, previous studies have also conducted studies to measure semantic similarities in citation sentences from English scientific papers using CNN (Nurjaman, 2020) and RNN (Besti, 2020) with six defined classes and datasets from the corpus of English written works. Based on the results of the test on the dataset both methods produce good accuracy on training data and test data. However, test results based on the entire class using F1-score both resulted in fairly low scores which is 66% for CNN and 45% for RNN. This happens because the spread of data of each class is not balanced.

The problem of unbalanced classes is called imbalance class, which is a situation in which the distribution or dissemination of data of each class is unbalanced. There are classes with less data (minorities) and class groups with more data (majority). Unbalanced classes can affect the performance of classification methods. One way to deal with it can be done by applying the Synthetic Minority Over-Sampling Technique (SMOTE) method that will generate minority data as much as majority data. Credit Card Fraud data sets have an uneven spread of class data. The data amounted to 29976 data, of which 23347 were positive data and 6629 were negative data. SMOTE can generate synthesis data for negative data twice as much as it originally was so that it amounts to 13258 data (Siringoringo, 2018).

Several other studies have applied SMOTE to the classification of emotions in Youtube (Atmadja, 2015) and Twitter (Sarakit, 2015), TV ads rating classification (Sutoyo, 2020), consumer complaint classification (Ruhyana, 2019), and online news classification (Kasanah, 2019). The application of SMOTE methods in some such cases is on average proven to improve the accuracy of classification methods well. One of them is in a study that classifies consumer complaints from Instagram using SVM classification method and Naïve Bayes. Based on test results, SVM and Naïve Bayes models without SMOTE had low accuracy of 69,98% and 51,54% respectively with significant differences compared to applying SMOTE to both methods resulting in accuracy of 76,66% and 93,86% respectively (Ruhyana, 2019).

The process of representing words into vectors (word embedding) is used to facilitate the process of classifying semantic proximity. There are many methods that can be used, one of which is BERT. In addition, classification can be done by using one of the deep learning models such as BiLSTM architecture to determine the class of sentence pairs. Previous research used this method for the classification of Amazon food review sentiment. In converting vectors to words use BERT as a feature extraction to be used for training and modeling processes using BiLSTM. The test results showed that with the application of BERT to BiLSTM it could result in a good accuracy of 93% based on datasets rather than using the usual word embedding methods such as Word2Vec and GloVe with CNN and LSTM deep learning models (Pasaribu, 2020).

The data used in this study is in the form of sentence pairs that will be determined by semantic proximity so that the computer needs to understand the meaning of each word and its relationship in the sentence. BERT offers an advantage over Word2Vec in word embedding because BERT makes the meaning expressed by a word in the text related to its context and the context information of a word helps improve its representation of meaning (Cai, 2020). Furthermore, BiLSTM also plays a role in understanding the meaning and relationship of each word in a sentence because it uses a two-way LSTM architecture that can consider the sequence of information forward and backward thus providing additional context for the network to get information as a whole, quickly, and obtain hidden information. Unlike LSTM which only considers information in one direction only (Song, 2020).

3. Materials and Methods

This research consists of several stages. The first stage is the collection of data in the form of pairs of English-language citation sentences from scientific papers. Second, the data labeling stage by comparing each sentence pair from the data obtained. Third, the preprocess stage is carried out in several stages, Furthermore, the data is ready to enter the training and modeling stage using BiLSTM. The last stage performs the process of measuring the level of semantic proximity of the sentence and the identification of each class of sentences that have been entered. The following is the design of a semantic classification system that can be seen in Figure 1.

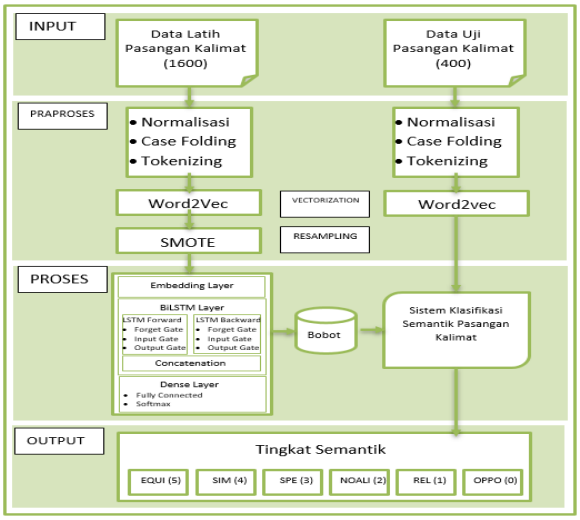

Figure 1: Semantic Classification of Sentences Using SMOTE and BiLSTM

3.1. Materials

The data used in this study is in the form of pairs of citation sentences taken from English-language papers in the field of computer science. Data is already available in a corpus in the form of a database that holds 17174 sentence pair data. The database structure of the corpus of citation pair data sets has the attributes id_case, sentence1, and sentence2. Labeling is done using a paraphrase information system that is already available. Labeling is done by comparing two sentences from the corpus dataset of sentence pairs displayed by the information system. Next, sentence pairs are labeled by choosing defined class categories, namely Equivalent, Similar, Specific, No Alignment, Related, and Opposite. The data labeled for use in the study amounted to 2000 pairs of sentences. The following is the dataset that can be seen in Table 1.

Table 1: Dataset Form

No	Sentence 1	Sentence 2	Class
1	Then we did word alignment …	In all experiments, word alignment …	Equivalent
2	The source sentences …	We trained a model …	Specific
…	…	…	…
2000	Brain images are quite noisy, …	Following the evaluation …	No Alignment

3.2. Methods

3.2.1. Preprocess

Preprocessing is the process of processing the dataset before it is ready to proceed to the training and modeling stage (Melina, 2022). This preprocess is done to eliminate noise in sentences to facilitate the machine learning process (Melina, 2024). In addition, it also produces the extraction of features in the form of vectors used as inputs in BiLSTM. Previously, the column of sentence 1 and sentence 2 in the dataset was first entered into the column "words". While the class column is converted into a binary matrix. The "words" column will enter into the preprocess stage. The following are some of the stages in the preprocess as follows:

3.2.1.1 Normalization

Normalization is the stage of removing punctuation marks and distances in each sentence. This removal aims to reduce storage usage and also eliminate noise that can affect the machine learning process. Punctuation marks or special characters are removed so that what is received is a-Z, and 0-9. Then for the distance removed is each white space at the beginning and at the end of the sentence.

3.2.1.2 Case Folding

Case folding is a stage to convert all the letters in a sentence into lowercase letters. This aims to uniformize characters on data to facilitate the machine learning process. So, each uppercase letter in the sentence (A-Z) will be converted into a lowercase form (a-z).

3.2.1.3 Tokenization

Tokenization is a stage to separate sentences into pieces that are referred to as tokens that can be words, numbers, symbols, and others. This aims to make it easier to analyze the meaning and connectedness of each word and also as an input to the word embedding process. Separation is done if a space is found between two words in a sentence.

3.2.2. Word2Vec

Word2Vec is one of the techniques of word embedding or converting every word in the context of a sentence into a vector that contains numbers with dimensions as much as N. Word2Vec implements a neural network in representing a word to perform contextual calculations and semantic similarity (contextual and semantic similarity) of each word that is input in the form of one-hot encoded vector (Mikolov, 2013). The results of calculating these similarities can represent the relationship between one word and another, such as the relationship between "Male-Female", the relationship with the verb, and also the relationship in "Country-Capital". The result of the relationship or relationship becomes a reference in representing the word into a vector (Mikolov, Zweig, 2013). The resulting vector is the result of learning from a neural network algorithm. Various types of features are taken from the word onwards on the feature is carried out in the machine learning process. Each word will have a vector that represents the meaning of the word based on features with vector shapes that vary in terms of their dimensions (Jatnika, 2019).

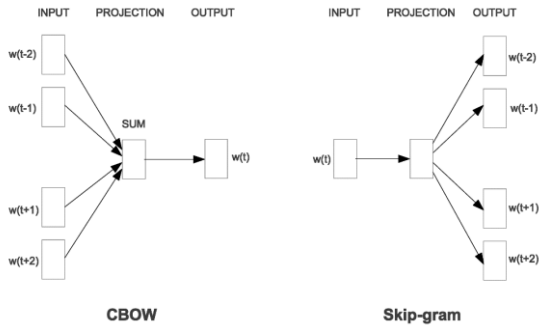

Figure 2: Two Word2Vec Architectures

Word2Vec has two architectural models, namely Continuous Bag-of-Word (CBOW) and Skip-Gram (Mikolov, Chen, 2013). Here are the CBOW and Skip-Gram architectures that can be seen in Figure 2. CBOW is a model that predicts the word used as the target of the input in the form of a given context. In contrast, Skip-Gram is a model that uses the word as input to predict a target in the form of a given context.

In the CBOW model, the context w(t)is used as input that has been converted into vectors containing 0 and 1 using one-hot encoding where t is the size of the word window. Furthermore, the multiplication of the dot product between the vector and the input weight in the input layer so as to produce a new weight in the projection (hidden layer). After that, the resulting output will be forwarded to the output layer and dot the product again between the weight of the previous result with the output weight. After that, the calculation of the error value using the cross-entropy method between the output vector and the target word vector will be continued with backpropagation to update the input weight and output weight by increasing or subtracting the gradient descent value. The process is repeated until the iteration has been established or has reached the minimum error value in cross entropy. If it is, the word representation vector can be retrieved by multiplying the vector one hot encoded of each word by the input weight.

Configurations for Word2Vec used are embedding vector size 100, window size 100, iteration 100, and Continuous Bag-of-Word (CBOW) architecture. The number of unique words/dictionary words in the dataset is 5449 words. After the Word2Vec process, each word in the data is converted to an integer according to its index. Next, padding is done to uniformize the length of each data that is 50 words.

3.2.3. Synthetic Minority Oversampling Technique (SMOTE)

Imbalance class is a state in which the distribution or dissemination of data of each class is uneven. There are class groups that have less data (minorities) and class groups that have more data (majority). Basically, any real data is unbalanced so it can make it difficult for classification methods to perform generalization functions that allow it to provide poor performance. With the advent of such problems, classification algorithms will result in higher accuracy for the majority class than minorities. Most classification methods lack the ability to overcome unbalanced classes (Sutoyo, 2020). A class is still commonly called balanced when the ratio between the minority class and the majority class is 1:100. However, if the ratio of the majority class exceeds 100, then the class can be called unbalanced.

In some cases, minority classes are more important to identify than the majority. One of them is on transactions with debit cards. In the class group, normal transactions are much more numerous than fraudulent transactions. Although few, the existence of fraud is more important to be identified than normal (Siringoringo, 2018).

SMOTE is a method of over-sampling in which data in the minority class is propagated using artificial data or synthesis derived from data replication in the minority class. In this method, instances are taken from minority classes so as to avoid excessive overfitting problems (Sutoyo, 2020).

The following are algorithms or how SMOTE works in creating data synthesis in minority classes (Chawla, 2002):

1) Determine the vector features and k of the nearest neighbors in the minority class at random.
2) Calculate the distance difference between the two.
3) Multiply the difference by a random number between 0 and 1
4) The result is added to the selected feature vector.
5) Synthesis data will be entered along the line segment between the feature vector and k.

Based on the steps already mentioned above, the following is the Equation (1) in the calculation process:

$$Xnew = Xi + (Xknn - Xi) \times \delta \quad (1)$$

Information Xi : vector of features in minority classes, $Xknn$: data that has the closest distance to Xi, and δ: random numbers between 0 and 1

Previously, the dataset was first divided into 80% for training data (1600) and 20% for test data (400). The data that will be done by the SMOTE process is training data only. The SMOTE configuration used is the value of the nearest neighbor k = 3 and the resampling strategy is a re-sample of the entire class except the majority class. The results of SMOTE made the training data increase from 1600 to 2844.

3.2.4. Bidirectional Long Short Term Memory (BiLSTM)

BiLSTM is a development of the LSTM model in which there are two layers of LSTM whose processes are in reverse direction. This model is very good at recognizing patterns in sentences because every word in a sentence is processed sequentially. The top layer moves forward to understand and process from the first word to the last word. While the bottom layer moves backward to understand and process from the last word to the first word. With the existence of these two-way opposite layers, the model can understand and take perspective from the previous word and the leading word so that the learning process will deepen which has an impact on the model will better understand the context of the sentence.

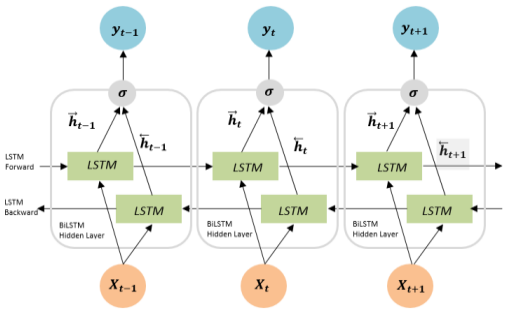

Figure 3: BiLSTM Architecture

In Figure 3 it can be seen that each hidden unit of ht output at the top and bottom layers is combined which forms the feature value of the word with a larger size than using regular LSTM so that the information processed at a later stage will classify more accurately. In combining the two units, there are several ways that are done, namely Sum output summed together, Multiply output multiplied together, Concatenation output combined together, Average average output taken. Usually used is Concatenation, the following is Equation (2) to combine output units.

$$h_t = [\vec{h}_t, \overleftarrow{h}_t] \quad (2)$$

After that, the result of merging the output units goes into the Softmax activation function that serves to convert the output into probability for each class. The following is Equation (3) for Softmax where yt is the output value, i is the unit index of output, m is the number of units at the output layer, eht is the exponential value of the unit, and ehi is the exponential value of the unit.

$$y_t = \frac{e^{h_t}}{\sum_{i=1}^{m} e^{h_i}} \tag{3}$$

Finally, perform an error calculation on the output value of a Softmax result using cross entropy that can be seen in Equation (4) where S is the Softmax value and L is the class label.

$$D(S, L) = -\sum_i L_i \, log \, log \, (S_i) \tag{4}$$

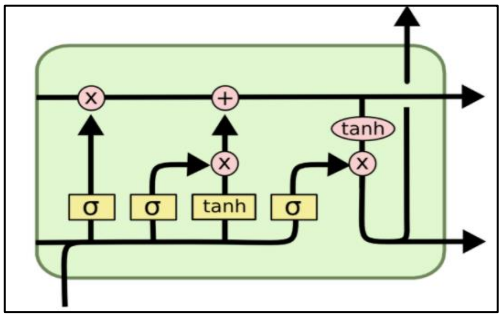

Figure 4: LSTM Architecture

Figure 4 is an architecture of LSTM that becomes forward and backward in BiLSTM. There is a lower path called cell gates that serves to regulate the information to be issued to the next unit, which is the upper path called the cell state to send information to other units without obstruction. In general, LSTM has three gates, namely input gate, forget gate, and output gate. The following are some equations for processing each gate in LSTM, namely Equation (5) for forget gate, Equation (6) and (7) for input gate, Equation (8) to update cell state value, and Equation (9) and (10) for gate output.

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \tag{5}$$

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \tag{6}$$

$$\tilde{C}_t = tanh(W_c . [h_{t-1}, x_t] + b_c) \tag{7}$$

$$\tilde{C}_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{8}$$

$$o_t = \sigma(W_o . [h_{t-1}, x_t] + b_o) \tag{9}$$

$$h_t = o_t * tanh(C_t) \tag{10}$$

Information:

i, f, o	= *input, forget, output gate*
ct	= internal memory unit
c_{t-1}	= previous memory
x_t	= input at each time *step time* t currently
σ	= sigmoid activation function $(0, 1)$
$tanh$	= tangent activation function $(-1, 1)$
W_f, W_t, W_c, W_o	= weight matrix
h_{t-1}	= previous *hidden state*
b_f, b_i, b_c, b_o	= vector bias

Berikut ini adalah model yang dibagun untuk melakukan pelatihan data seperti pada Tabel 2.

Table 2: Developed Model

Layer (type)	Output Shape	Parameter
Embedding	(None, 50, 100)	544900
Bidirectional	(None, 50, 200)	160800
Bidirectional_1	(None, 128)	135680
Dense	(None, 32)	4128
Dense_1	(None, 6)	198

3.2.4.1 Embedding Layer

This layer serves as an input from the preprased and vector-shaped data of the word, i.e. each sentence in the data has a length of 50 word sequences and each word in the sentence has a vector length of 100. The input dimension is the number of unique words with a vector length so that there are 544900 parameters. The output of this layer is a vector with a length of 100.

3.2.4.2 Bidirectional Layer

This layer performs feature extraction consisting of two LSTM layers that process words sequentially through two directions, forward and backward. The BiLSTM layer used is two with each output there is a dropout layer to reduce the number of neurons. Each LSTM has 4 gates, meaning that Each BiLSTM has 8 gates. The formula is gate (input + output)+ output). So for the first BiLSTM, $8(100(100 + 100) + 100) = 160800$ parameters. As for the second BiLSTM, $8(64(64 + 200) + 64) = 135680$ parameters. Each output from BiLSTM is combined with advanced LSTM and backward LSTM results.

3.2.4.3 Dense Layer

This layer is an output that will generate class probabilities. The first dense is a fully connected layer that has 32 nodes that have $128 \text{ x } 32 + 32 = 4128$ parameters. The second dense is entered for the Softmax activation function which has 6 nodes according to the number of classes. This dense has $32 \text{ x } 6 + 6 = 198$ parameters.

Next, the model is evaluated by calculating errors using cross entropy. In addition, optimization is also done using Adam, SGD, and Adadelta, as well as model performance measurements using Confusion Matrix.

3.2.5. Optimization Model

In machine learning or machine learning, Gradient Descent is usually used as one of the optimization methods. This method will look for the most optimal weight by doing learning that moves forward repeatedly(iterative). This aims to get a cost function that has a minimum value that represents the level of error that appears when predicting the target. There are three learning methods used in this study.

3.2.5.1 Adaptive Moment Optimization (Adam)

Adaptive Moment Optimization (Adam) is one of the adaptive learning methods by calculating individual learning levels on each parameter. Through these calculations, the error value can be minimized when the prediction process takes place.

3.2.5.2 Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) is one of the learning methods that derives by taking measurements of changes in functions along with changes in input values. In evaluating gradients, SGD uses randomly taken data samples from multiple parts of training data processed in one iteration. While finding local optimum values, the Gradient Descent method can take a long time because it uses all the training data in one process.

3.2.5.3 Adaptive Learning Rate (Adadelta)

Adaptive Learning Rate (AdaDelta) is a development of SGD that uses adaptive techniques in improving the weight and level of learning. This optimization method will accumulate gradients that can help increase probability in finding solutions in subsequent iterations so that convergence can be achieved in a fast way.

3.2.6. Confusion Matrix

Confusion Matrix is a performance measurement for models used in machine learning classification problems visualized in the form of tables where the output can be two or more classes. Each input in the confusion matrix indicates the number of predictions made by the model in which it classifies correctly or incorrectly. The confusion matrix has four metrics that are a combination of predictive and actual results, namely TP, TN, FP, and FN that can be seen in Table 3. True Positive (TP), the sum of the predictions in which the classifier correctly predicts a positive class as "positive". True Negative (TN), the sum of the predictions in which the classer correctly predicts a negative class as "negative". False Positive (FP), the sum of the predictions in which the classer incorrectly predicts a negative

class as "positive". False Negative (FN), the sum of predictions in which the classer incorrectly predicts a positive class as "negative".

Table 3: Confusion Matrix

		Actual Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

The following are some of the most common ways to take performance measurements in the confusion matrix, among others:

- *Precision,* informs how many of all classes are predicted as positive classes that are actually positive. Here's Equation (11) for *precision.*

$$\text{Precision} = \left(\frac{TP}{TP+FP}\right) \times 100\% \quad (11)$$

- *Recall*, informs how many of all positive classes are correctly predicted as positive by the classer. Here's Equation (12) for *recall.*

$$\text{Recall} = \left(\frac{TP}{TP+FN}\right) \times 100\% \quad (12)$$

- *Accuracy,* informs the overall accuracy of the model which means how much of all classes (positive and negative) is correctly predicted by the classer. Here's Equation (13) for *accuracy.*

$$\text{Accuracy} = \left(\frac{TP+TN}{TP+TN+FP+FN}\right) \times 100\% \quad (13)$$

- *F-Measure,* combines *precision* and *recall* values into a single measurement. It is mathematically referred to as the *harmonic* average of *precision* and *recall.* Here is Equation (14) for *f-measure.*

$$\text{F-Measure} = 2 \times \frac{Precision \times Recall}{Precision+Recall} \times 100\% \quad (14)$$

4. Results and Discussion

In this study, there were 8 scenarios based on the dataset used and the model used. There are two datasets, namely data from this study (A) and data from previous related research (B) (Besti, 2020). As for the 4 models, namely the model used in this study (BiLSTM + SMOTE), the model used in previous related research (LSTM) (Besti, 2020), the LSTM model, and the LSTM + SMOTE model. The following are the scenarios of testing that can be seen in Table 4. The SMOTE parameter has a choice sampling strategy, which is how to re-sample each class. The selected method is "not majority", meaning smote will re-sample the entire class except the majority class.

Table 4: Testing Scenario

No.	Data	Model
1	A	LSTM
2	A	LSTM+SMOTE
3	A	BiLSTM
4	A	BiLSTM+SMOTE
5	B	LSTM
6	B	LSTM+SMOTE
7	B	BiLSTM
8	B	BiLSTM+SMOTE

In Table 5, the "Similar" class's "Similar" class is the majority class so the other classes are re-displayed as many as 474 for the training data. After SMOTE, the total amount of data became 3244 data, which is 2844 training data and 400 test data.

Table 5: Dataset Current Research

Class	Sum	Before SMOTE		After SMOTE	
		Train	*Test*	*Train*	*Test*
Equivalent	423	338	85	474	85
Similar	591	474	117	474	117
Specific	514	411	103	474	103
No Alignment	379	303	76	474	76
Related	89	71	18	474	18
Opposite	4	3	1	474	1
Total	2000	1600	400	2844	400

While in Table 6, the "Equivalent" class B training data is the majority class so the other classes will have 657 data for the training data. After SMOTE, the total amount of data became 4348 data, which is 3942 training data and 406 test data.

Table 6: Dataset Previous Research

Class	Sum	Before SMOTE		After SMOTE	
		Train	*Test*	*Train*	*Test*
Equivalent	822	657	165	657	165
Similar	714	573	141	657	141
Specific	451	362	90	657	90
No Alignment	32	25	7	657	7
Related	4	3	1	657	1
Opposite	6	4	2	657	2
Total	2030	1624	406	3942	406

All scenarios use as many as 50 epochs in the training process against training data. The optimization models used are Adam, SGD, and Adadelta. Results showed that scenario 8 (previous research data + BiLSTM + SMOTE) produced the best accuracy of all scenarios on Adam optimization model, which was 93.28% for training data and 88.92% for test data. In contrast, scenario 2 (current research data + LSTM + SMOTE) resulted in the lowest accuracy on the SGD optimization model, at 29.57% for training data and 23.25% for test data. The average BiLSTM was able to improve the accuracy of scenarios using the LSTM model. Likewise, SMOTE on average can produce better accuracy compared to scenarios without applying SMOTE, but not for scenarios that use current research data. For datasets used, scenarios that used previous research data resulted in higher accuracy than scenarios that used current research data. Adam's model is able to provide the best accuracy because on the graph it always reaches a stable point faster than SGD and Adadelta. The following is the accuracy of each scenario in percent that can be seen in Table 7.

Table 7: Result of Accuration Model Optimization

No	Adam		SGD		Adadelta	
	Train	*Test*	*Train*	*Test*	*Train*	*Test*
1	77.44	26.75	43.63	31.25	70.56	28.50
2	70.29	33.25	**29.57**	**23.25**	64.63	34.75
3	77.44	32.50	54.00	34.50	77.13	28.00
4	74.89	32.50	45.99	31.75	73.45	27.25
5	88.05	68.47	65.33	48.52	83.74	67.00
6	93.20	85.47	57.41	37.93	78.72	78.82
7	88.55	72.91	76.85	59.11	89.47	72.66
8	**93.28**	**88.92**	74.35	67.49	93.02	87.44

Based on the accuracy graph in Figure 5, the left graph (scenario 5) that uses the model in previous research, namely LSTM, it appears that for training data has reached a stable point in the 20th epoch which means the model can perform classification well. Same with test data, but not too high. While the right graph (scenario 8) that uses the model in this study, namely BiLSTM + SMOTE, shows that the training data has stabilized faster at epoch 10. Likewise with test data that looks higher than the left graph.

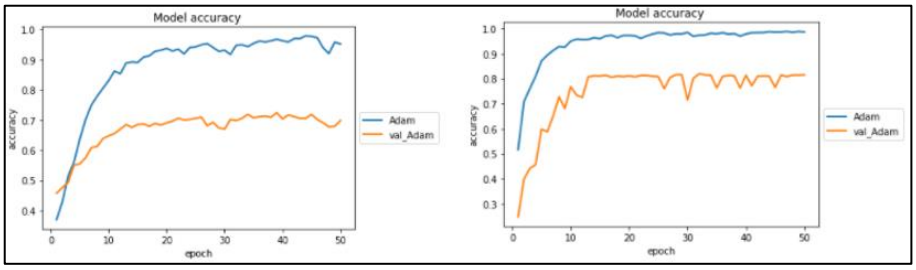

Figure 5: Adam Scenario Optimization Accuracy Graph 5 and 8

As for the loss chart in Figure 6, the left chart (scenario 5)shows that the graph is decreasing until it reaches a stable state, which means the model is able to recognize the training data well and can handle the architecture and parameters of the built model. However, for test data the graph looks a little increased which means the model is still not able to recognize the test data and handle complex models. While the right graph (scenario 8) shows that both training data and test data decreased the value of loss to stable, which means the model is able to recognize the data well.

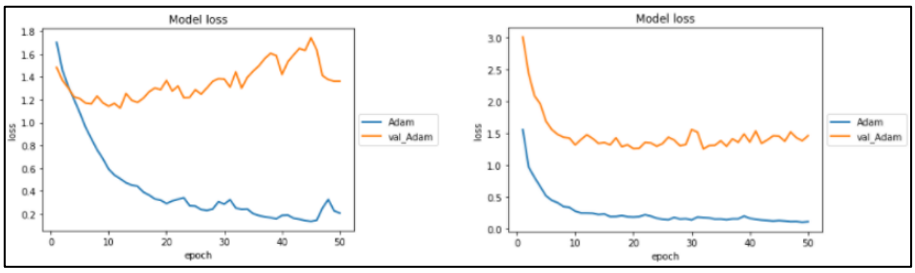

Figure 6: Adam Scenario Optimization Loss Graph 5 and 8

Based on confusion matrix measurements, scenario 8 also produces the best performance than other scenarios, namely accuracy 89%,precision 74,3%,recall 69,5%, and f-measure 70,5%. As for the lowest occurred in scenario 1, namely accuracy 26,8%,precision 19,7%,recall 19%, and f-measure 19%. Both BiLSTM and SMOTE models improve the performance of LSTM models and without SMOTE. However, SMOTE's cap on scenarios using current research data and the BiLSTM model has decreased. The following are the results of each confusion matrix measurement of a class of each scenario in percent that can be seen in Table 8.

Table 8: Result of Confusion Matrix Measurement

No.	Accuracy	Precision	Recall	F-Measure
1	**26.8**	**19.7**	**19**	**19**
2	33.3	24.5	23.8	24.2
3	32.5	24.7	24.5	24.3
4	32.5	22.7	22.5	22.5
5	68.5	50.1	43.7	46.2
6	85.5	72.3	65.7	67.5
7	73	52.2	45.2	47.7
8	**89**	**74.3**	**69.5**	**70.5**

In Table 9, you can see a comparison of the measurement results of each class using the confusion matrix between scenario 5 using the LSTM (S-5) model and scenario 8 that uses the BiLSTM + SMOTE (S-8) model in percent. In

accuracy, it increased by 20.5%. In precision, each class increases, especially in class 1 which initially has no precision value due to the small amount of data so that the model cannot recognize it and is unable to classify the class. The same goes for recalls and f-measures.

Table 9: Result of Performance Measurement Scenario 5 and 8

No.	Accuracy	Precision	Recall	F-Measure
1	**26.8**	**19.7**	**19**	**19**
2	33.3	24.5	23.8	24.2
3	32.5	24.7	24.5	24.3
4	32.5	22.7	22.5	22.5
5	68.5	50.1	43.7	46.2
6	85.5	72.3	65.7	67.5
7	73	52.2	45.2	47.7
8	**89**	**74.3**	**69.5**	**70.5**

5. Conclusion

Based on the results of the tests and discussions that have been done in this study, it can be concluded that the BiLSTM model was able to improve the results of scenario accuracy in the LSTM model because the model processes words forward and backward so that it can recognize patterns and capture information better. Applying SMOTE in this study resulted in better accuracy compared to scenarios without applying SMOTE. This is because the distribution of data of each class becomes balanced which affects the model's ability to classify. In addition, the Adam optimization model used in this study produces good performance because the graph always reaches a stable state in classification earlier than SGD and Adadelta. When compared to the results of previous research related, both in terms of the testing of each data and each class turned out to be improved which means that the model used in this study can provide better accuracy and performance results. In addition, among BiLSTM and SMOTE the most dominant in improving accuracy is SMOTE.

Testing with optimization and confusion matrix models has been done through 8 different scenarios based on the data and models used. In testing each data using the optimization model, it produces the best accuracy in Adam optimization model, which is 93,28%for training data and 88,92% for test data. While in terms of testing each class using confusion matrix, resulting in accuracy 89%, precision 74,3%, recall 69,5%, and f-measure 70,5% for the entire class. Datasets in this study (2000), it turns out to always produce low accuracy and performance even though it is almost the same as the previous research (2030). In addition, the average use of BiLSTM models or the application of SMOTE with such data even decreased accuracy and performance. This may be because the dataset in this study is less suitable for use on the proposed model so that the selection of other models is more precise.

Suggestion, SMOTE has several ways to re-sample data, namely *minority, not minority, not majority,* and *all.* For the future, maybe you can try one of these methods or use different *nearest neighbor k* values as a comparison. In addition, you can also try other methods for the *word embedding* process such as the recent BERT *(Bidirectional Encoder Representation from Transformers)* and other learning models for its classification.

References

Atmadja, A. R., & Purwarianti, A. (2015). Comparison on the rule based method and statistical based method on emotion classification for Indonesian twitter text. 2015 International Conference on Information Technology Systems and Innovation (ICITSI), 2, 1–6. https://doi.org/10.1109/icitsi.2015.7437692

Besti, A., Ilyas, R., Kasyidi, F., & Djamal, E. C. (2020). Semantic classification of scientific sentence pair using recurrent neural network. 2020 7th International Conference on Electrical Engineering, Computer Sciences and Informatics (EECSI), 1, 150–155. https://doi.org/10.23919/eecsi50503.2020.9251897

Bhagat, R., & Hovy, E. (2013). What is a paraphrase? Computational Linguistics, 39(3), 463–472. https://doi.org/10.1162/coli_a_00166

Cai, R., Qin, B., Chen, Y., Zhang, L., Yang, R., Chen, S., & Wang, W. (2020). Sentiment analysis about investors and consumers in energy market based on Bert-BILSTM. IEEE Access, 8, 171408–171415. https://doi.org/10.1109/access.2020.3024750

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321–357. https://doi.org/10.1613/jair.953

González Aguirre, A. (2017). Computational models for semantic textual similarity.

Ichida, A. Y., Meneguzzi, F., & Ruiz, D. D. (2018). Measuring semantic similarity between sentences using a Siamese neural network. 2018 International Joint Conference on Neural Networks (IJCNN), 1, 1–7. https://doi.org/10.1109/ijcnn.2018.8489433

Jatnika, D., Bijaksana, M. A., & Suryani, A. A. (2019). Word2Vec model analysis for semantic similarities in English words. Procedia Computer Science, 157, 160–167. https://doi.org/10.1016/j.procs.2019.08.153

Kasanah, A. N., Muladi, M., & Pujianto, U. (2019). Application of SMOTE Technique to Overcome Class Imbalance in Online News Objectivity Classification Using KNN Algorithm. RESTI Journal (Systems Engineering and Information Technology), 3(2), 196-201.

Melina, M., Sukono, Napitupulu, H., & Mohamed, N. (2024). Modeling of machine learning-based extreme value theory in stock investment risk prediction: A systematic literature review. Big Data. https://doi.org/10.1089/big.2023.0004.

Melina, Sukono, Napitupulu, H., Sambas, A., Murniati, A., & Kusumaningtyas, V. A. (2022). Artificial Neural Network-Based Machine Learning Approach to Stock Market Prediction Model on the Indonesia Stock Exchange During the COVID-19. *Engineering Letters*, *30*(3).

Mikolov, T. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 26.

Mikolov, T., Yih, W. T., & Zweig, G. (2013, June). Linguistic regularities in continuous space word representations. In Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies (pp. 746-751).

Nurjaman, J., Ilyas, R., & Kasyidi, F. (2020, September). Measuring Semantic Similarity of Citation Sentence Pairs Using Convolutional Neural Network. In Proceedings of Industrial Research Workshop and National Seminar (Vol. 11, No. 1, pp. 510-516).

Pasaribu, D. J. M., Kusrini, K., & Sudarmawan, S. (2020). Improving Amazon Food Review Sentiment Classification Accuracy with Bidirectional LSTM and Bert Embedding. Inspiration: Journal of Information and Communication Technology, 10(1), 9-20.

Ruhyana, N. A. N. A. N. G., & Rosiyadi, D. I. D. I. (2019). Classification of Instagram Comments to Identify Customer Complaints for Freight Forwarding Services with Smote Technique. Faktor Exacta, 12(4), 280-290.

Sarakit, P., Theeramunkong, T., & Haruechaiyasak, C. (2015). Improving emotion classification in imbalanced YouTube dataset using smote algorithm. 2015 2nd International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA), 3, 1–5. https://doi.org/10.1109/icaicta.2015.7335373

Shi, X., & Lu, R. (2019). Attention-based bidirectional hierarchical LSTM networks for text semantic classification. 2019 10th International Conference on Information Technology in Medicine and Education (ITME). https://doi.org/10.1109/itme.2019.00143

Siringoringo, R. (2018). Classification of unbalanced data using SMOTE and k-nearest neighbour algorithms. Journal Information System Development (ISD), 3(1).

Song, Y., Tian, S., & Yu, L. (2020). A method for identifying local drug names in Xinjiang based on Bert-BILSTM-CRF. Automatic Control and Computer Sciences, 54(3), 179–190. https://doi.org/10.3103/s0146411620030098

Sutoyo, E., & Fadlurrahman, M. A. (2020). Application of SMOTE to Overcome Class Imbalance in Television Advertisement Performance Rating Classification Using Artificial Neural Network. JEPIN (Jurnal Edukasi dan Penelitian Informatika), 6(3), 379-385.

Teufel, S., Siddharthan, A., & Tidhar, D. (2006). An annotation scheme for citation function. Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue - SigDIAL '06, 80. https://doi.org/10.3115/1654595.1654612