Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

Enhancing Sentiment Analysis Effectiveness with LSTM Variants, and Stratified K-Fold on Imbalanced Dataset

Rifki Andriyanto*, Kusrini

Postgraduate, Master of Informatics Engineering, Universitas Amikom Yogyakarta, Yogyakarta, Indonesia
Email: rifkiandriyanto@students.amikom.ac.id, kusrini@amikom.ac.id
Correspondence Author Email: rifkiandriyanto@students.amikom.ac.id
Submitted: **01/08/2024**; Accepted: **10/09/2024**; Published**: 11/09/2024**

Abstract−Sentiment analysis on hotel reviews often faces the challenge of class imbalance, where positive reviews are significantly more dominant than negative or neutral ones. Additionally, there is a challenge in determining the optimal combination of models such as LSTM, BiLSTM, and BiLSTM-Attention with word embedding techniques like FastText, Word2Vec, and Doc2Vec to achieve the best performance. This study aims to enhance the effectiveness of sentiment analysis on imbalanced datasets by combining these word embedding methods and model architectures. Class imbalance is addressed using SMOTE, and the models are evaluated using Stratified K-Fold cross-validation. The results show that Doc2Vec consistently outperforms FastText and Word2Vec, especially when combined with the BiLSTM-Attention architecture. The Doc2Vec-BiLSTM-Attention model achieved an accuracy, weighted average recall, and weighted average f1-score of 0.96, along with a macro average recall and macro average f1-score of 0.93. The use of SMOTE and Stratified K-Fold proved effective in improving model performance on imbalanced datasets. This research identifies the optimal combination of word embedding methods and model architectures, which, along with class imbalance handling techniques, significantly enhances the effectiveness of sentiment analysis models on hotel reviews.

Keywords: SMOTE, StratifiedKFold, Word embedding, Deep learning.

1. INTRODUCTION

Sentiment analysis often faces challenges such as class imbalance, and difficulties in determining the optimal combination of models (LSTM, BiLSTM, BiLSTM with attention) and word embedding techniques (FastText, Word2Vec, Doc2Vec) for maximum performance. This research aims to enhance sentiment analysis models by exploring various combinations of these techniques and implementing SMOTE and Stratified K Fold to address class imbalance. The main goal is to develop a more accurate and efficient model for understanding and analyzing sentiment in text, with particular emphasis on improving accuracy in minority classes, thereby achieving a balanced accuracy between minority and majority classes.

Previous research has provided valuable insights into addressing these challenges. For instance, several studies have been conducted on similar topics [1], focused on subjectivity analysis in political debates, utilizing deep learning models like LSTM and GRU. With bidirectional configurations and attention mechanisms, this research employed lexicon-based and syntactic pattern-based approaches for subjectivity annotation and explored word representation techniques like GloVe embeddings. The LSTM model with attention achieved 97.39% accuracy, demonstrating its effectiveness in complex classifications and highlighting the importance of advanced preprocessing and the use of word embeddings.

Similarly [2], introduced a novel sentiment analysis technique using a hybrid optimization algorithm that combines Taylor series expansion and Harris hawks optimization (HHO) to improve the performance of BiLSTM. This technique, named Taylor–Harris Hawks Optimization (THHO)-BiLSTM, was tested on Amazon product reviews and the Taboada dataset, achieving high accuracy. This approach employs efficient data preprocessing, the Taylor-HHO algorithm for optimal weight selection, and the effective BiLSTM structure in managing sequential data.

In another study [3], introduced Aspect-Based Latent Dirichlet Allocation Sentiment Embedding (AB-LaBSE), a novel approach for Uyghur sentiment analysis that integrates the LaBSE cross-lingual model with BiLSTM. AB-LaBSE utilizes data augmentation to enhance dataset diversity and leverages the cross-lingual representations learned from LaBSE, which is adapted for Uyghur. The BiLSTM layer enhances sentiment understanding by capturing contextual information. AB-LaBSE surpasses baseline models in precision, recall, and F1 score, showcasing the effectiveness of deep learning techniques and the transfer of knowledge across languages.

In their publication [4], developed an attention-based sentiment analysis model RU-BiLSTM for Roman Urdu, widely used on social media in Pakistan. This model combines BiLSTM with an attention mechanism to improve sentiment classification from Roman Urdu text, considering both past and future context. The research shows significant improvements in precision, recall, and F1 score, marking the potential of advanced deep learning techniques for low-resource language processing.

Another study by [5], addressed sentiment analysis of Online Chinese Buzzwords using the BERT and BiLSTM model. This model handles incomplete semantic structures and irregular features of buzzwords, generating dynamic word vector representations for downstream tasks. The BERT-BiLSTM model outperformed baseline models in precision, recall, and F1 score, marking a significant advancement in sentiment analysis of irregular texts.

Lastly [6], proposed the Self-Attention-CNN-BiLSTM (SAC-BiLSTM) method, combining BiLSTM and a Self-Attention mechanism for Chinese text sentiment analysis. This method utilizes both character and word embeddings to deepen semantic understanding and capture bidirectional semantic dependencies. Applied to the

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

onlineshopping10cats dataset, SAC-BiLSTM showed significant improvements in precision, recall, and F1 score, demonstrating its effectiveness in sentiment analysis for languages with limited data resources.

Previous research has demonstrated the effectiveness of various sentiment analysis methods in different contexts, including the use of LSTM, BiLSTM, GRU, THHO-BiLSTM, AB-LaBSE pretrained BiLSTM, Attention-Based RU-BiLSTM, BERT with BiLSTM, and dual-channel BiLSTM with Self-Attention. This study aims to examine the effectiveness of BERT, LSTM, BiLSTM, and Attention BiLSTM methods in analyzing hotel review sentiment. These methods will be combined with word embedding techniques such as FastText, Doc2Vec, and Word2Vec to enrich text processing and understanding, along with SMOTE to address class imbalance in the dataset and Stratified K Fold for cross-validation. This comparison is expected to yield a sentiment analysis model that is more effective in capturing nuances and deeper context from imbalanced datasets. The research also aims to test various combinations of word embedding methods and model architectures to find the most optimal pairing, as well as to improve prediction accuracy on minority classes, which are often overlooked by models that solely focus on majority class accuracy.

LSTM was chosen for its proven effectiveness in understanding context and word order in long texts [7]. The LSTM architecture to overcome the vanishing gradient problem in regular neural networks, enabling LSTM to remember long-term information and address sequence issues in textual data [7].

BiLSTM was chosen for its ability to capture information from both directions of the word sequence, thus improving overall text understanding [8]. The BiLSTM architecture to extend the capabilities of LSTM by allowing information flow from both directions of the sequence, thereby enhancing the model's ability to understand context [8].

Additionally, BiLSTM with an attention mechanism was chosen for its ability to focus on important parts of the text, thus improving accuracy in sentiment analysis. The use of the BiLSTM-Attention model for sentiment classification through several key reasons. BiLSTM captures contextual information from both directions, enhancing text understanding. The attention mechanism allows the model to focus on the most relevant parts of the text, improving classification accuracy. Empirical results show that the combination of BiLSTM and attention provides superior performance compared to traditional models. Furthermore, BiLSTM-Attention is effective in various sentiment analysis tasks and can handle imbalanced data well, ensuring the model is not biased towards the majority class [9].

SMOTE was chosen for its proven effectiveness in addressing class imbalance in datasets. By generating synthetic samples for minority classes, it balances the class distribution, enhancing the classifier's ability to accurately identify and predict those classes [10].

Stratified K-Fold Cross-Validation was selected to ensure each data fold maintains the same class proportions as the original dataset. This approach ensures that validation results are more representative and that the model learns equally from all classes, regardless of their initial representation. By partitioning the dataset into k folds while preserving the original class distribution in each, this technique facilitates a more robust model evaluation through training and testing on multiple data subsets [11].

This research will test the combination of various word embedding techniques (FastText, Word2Vec, and Doc2Vec) with LSTM, BiLSTM, and BiLSTM-Attention models. First, the LSTM model will be combined with each word embedding technique, resulting in LSTM-FastText, LSTM-Word2Vec, and LSTM-Doc2Vec. Next, the BiLSTM model will be applied with the same word embedding variations. To improve model performance, an attention layer will be added to the BiLSTM architecture, resulting in BiLSTM-Attention-FastText, BiLSTM-Attention-Word2Vec, and BiLSTM-Attention-Doc2Vec models. Additionally, these combinations will also be tested with the implementation of SMOTE and Stratified K-Fold. By comparing the effectiveness of these models, this research aims to find the optimal combination that can accurately interpret and understand sentiment in hotel reviews, thus providing valuable insights into consumer experiences and perceptions.

2. RESEARCH METHODOLOGY

2.1 Research Stages

The research workflow begins by loading hotel review data from the Kaggle TripAdvisor dataset. The data then undergoes text preprocessing, where the reviews are cleaned and processed by removing non-alphabetic characters, tokenization, removing common meaningless words, and lemmatization. The results of this process are stored in a new column as cleaned text. Next, the review ratings are automatically converted into sentiment labels based on the Rating column, with values 1-2 considered negative, 3 neutral, and 4-5 positive. After that, the cleaned reviews are tokenized into sentences for training three different word embedding models: FastText, Word2Vec, and Doc2Vec. The word embedding models are trained using the tokenized sentences and used to convert the review texts into vectors. The review texts are converted into average vectors of word vectors using the trained word embedding models, and these vectors are then normalized. Next, three model architectures, namely LSTM, BiLSTM, and BiLSTM-Attention, are defined, each consisting of several layers and one fully connected layer. Then performs cross-validation with Stratified K-Fold to split the data into several folds. During each fold, the data is trained using SMOTE to handle class imbalance and then trained using the defined models. The best model is saved based on the highest

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

validation accuracy. The training function is used to train the LSTM, BiLSTM, and BiLSTM-Attention models using the resampled training data and the Adam optimizer, calculating the loss and accuracy for each epoch. The models are then evaluated using validation data for each fold by calculating accuracy, confusion matrix, and classification report. The best model is selected based on these evaluation results. After obtaining the best model from the cross-validation process, it is evaluated on the entire test set. The evaluation results include accuracy, confusion matrix, and classification report. The research flow can be seen in Figure 1.

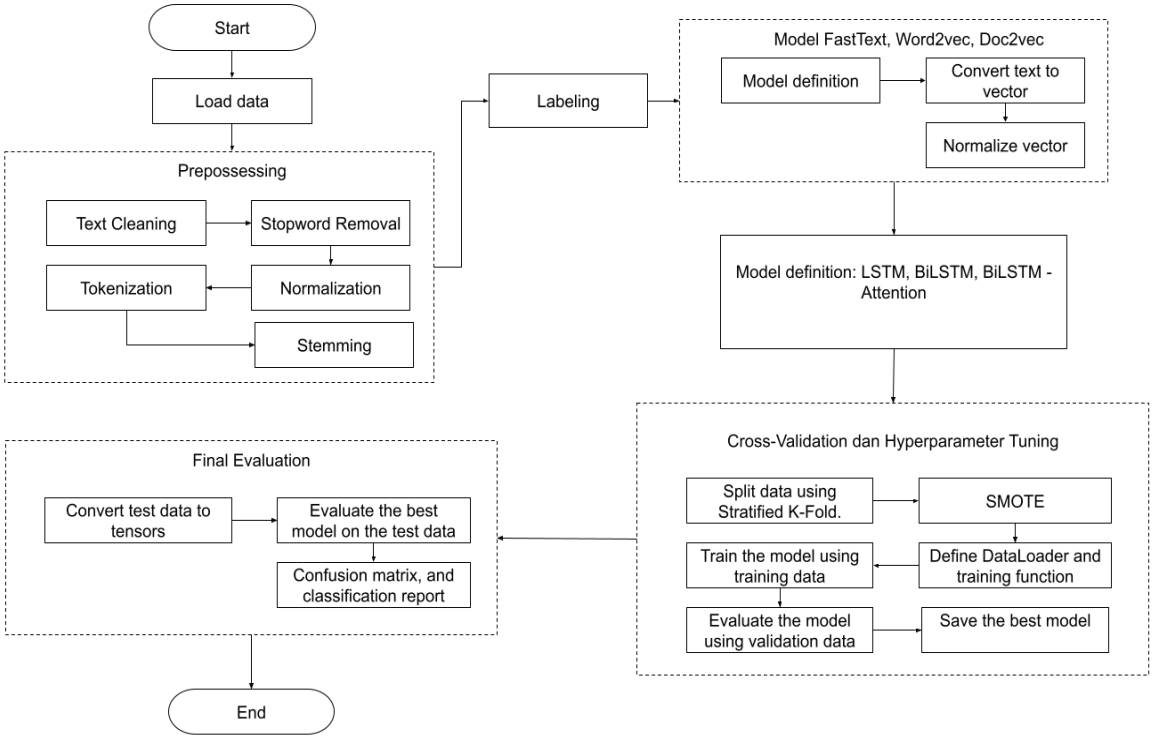

Figure 1. Research Stages

2.2 Pre-processing

Data preprocessing is an important step in data analysis, especially in natural language processing (NLP) and machine learning. This process involves various techniques to clean and prepare raw data before it is used in models. [12] explains that data preprocessing in NLP involves several important steps such as tokenization, stopwords removal, stemming, and lemmatization. They emphasize the importance of this stage to improve data quality and the effectiveness of the models used.

2.3 Word Embedding

2.3.1 FastText

FastText is a word embedding technique developed by researchers at Facebook AI Research (FAIR) aimed at enhancing the efficiency and effectiveness of text processing in various NLP applications. Unlike traditional word embedding models such as Word2Vec, FastText not only learns vector representations for whole words but also for n-grams of each word. This allows FastText to better understand the internal structure of words and produce improved representations for rare words or even previously unseen words.

The model works by breaking down each word into a series of n-grams (for example, the word "apple" with 3-gram size can be broken down into <ap, app, ppl, ple, le>). Each n-gram is represented by a vector in the embedding space, and the vector for the whole word is the average of these n-gram vectors. This is particularly useful in handling languages with rich morphology, such as Turkish or Finnish, where affixation is very common [13].

2.3.2 Word2Vec

The Word2Vec model is renowned for its ability to effectively capture both the syntactic structure and semantic nuances of natural language [14]. This model represents words as vectors within a low-dimensional space, clustering similar words together based on their shared meanings.

Word2Vec employs a neural network to transform text into vectors, capturing the meaning of words within these vector representations. Two different approaches are used within the Word2Vec model: the Skip-Gram model,

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

which predicts surrounding words based on a given word, and the Continuous Bag of Words (CBOW) model, which predicts a word based on its context.

The Skip-Gram model is designed for efficient learning of word vector representations from vast amounts of unstructured text. It predicts words within a certain range surrounding the current input word. Conversely, the CBOW model focuses on predicting the current word based on its surrounding context words. Figure 2 illustrates the architectures of the Word2Vec, CBOW, and Skip-Gram models.

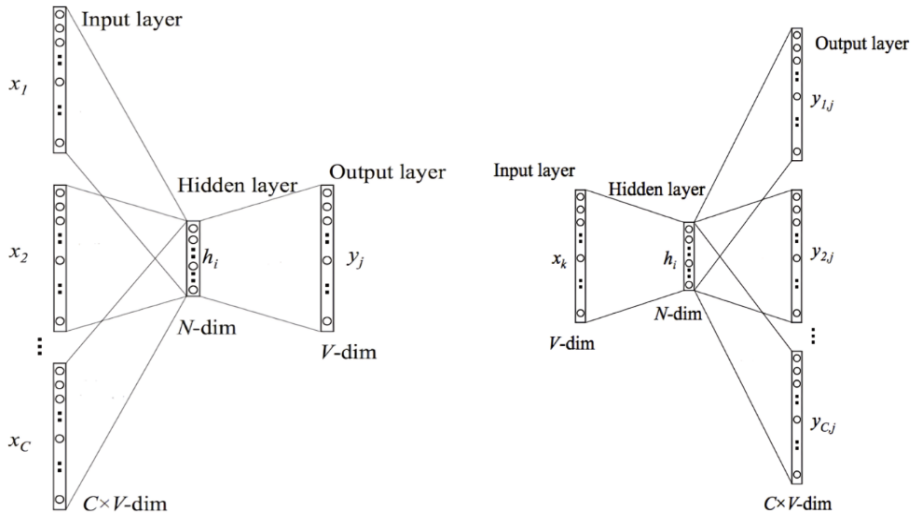

Figure 2. CBOW and Skip-Gram Architecture

In the Word2Vec model, there are two evaluation methods: Hierarchical Softmax and Negative Sampling. Hierarchical Softmax was originally introduced by [14]. It employs a binary tree to structure the output layer, placing words at the leaf nodes. Each node in the tree directly indicates the likelihood of its child nodes being selected. On the other hand, negative sampling is simpler than Hierarchical Softmax because it only updates a sample of a few output words as negative samples [15].

2.3.4 Doc2vec

Doc2Vec is a further development of Word2Vec. While Word2Vec is used to classify words or text, Doc2Vec is useful for classifying groups of words or sentences, commonly referred to as documents. The goal of Doc2Vec is to create vector representations for documents. While words in a sentence have a structure (grammar), a document does not have a logical structure. To overcome this problem, an additional vector (Paragraph ID) needs to be added to fit the Word2Vec modeling. This is the main difference between Word2Vec and Doc2Vec [16].

In its application, Doc2Vec has two main approaches: PV-DM (Paragraph Vector-Distributed Memory) and PV-DBOW (Paragraph Vector-Distributed Bag of Words).

a. Paragraph Vector-Distributed Memory
   The core concept of the PV-DM model draws inspiration from the Continuous Bag of Words (CBOW) approach in Word2Vec. In CBOW, the model predicts a target word based on its surrounding context words. For instance, given the sentence "The cat sat on the sofa," CBOW would predict the word "sat" using "The," "cat," "on," and "sofa" as context. Similarly, PV-DM randomly selects a sequence of words from a paragraph and predicts a target word within that sequence. However, PV-DM incorporates both word context and the paragraph ID as input [16].
b. Paragraph Vector-Distributed Bag of Words
   The PV-DBOW model diverges slightly from the PV-DM model. In PV-DBOW, context words are ignored in the input, and instead, the model is tasked with predicting words randomly chosen from the paragraph. For instance, given {The, cat, sat, on, sofa}, the model might learn to predict two random samples like "cat" and "sofa." Essentially, to learn the document vector, it samples multiple words from that paragraph. The key distinction between skip-gram and PV-DBOW lies in the input: PV-DBOW utilizes the document ID (Paragraph ID) and aims to predict multiple random word samples from the associated document [16].

2.4. Long Short Term Memory

LSTM, a specialized type of Recurrent Neural Network, excels at learning long-term dependencies [17]. These dependencies significantly impact the meaning and overall sentiment of a document. LSTM networks include gates that regulate the flow of inputs and outputs, making sure that essential information is kept or discarded as required. An LSTM cell typically consists of three gates: the input gate, the forget gate, and the output gate. The input gate controls what new information should be added to the memory. The forget gate determines how long the stored

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

information should be retained and when to reset the memory. Finally, the output gate decides when to output the stored information [1].

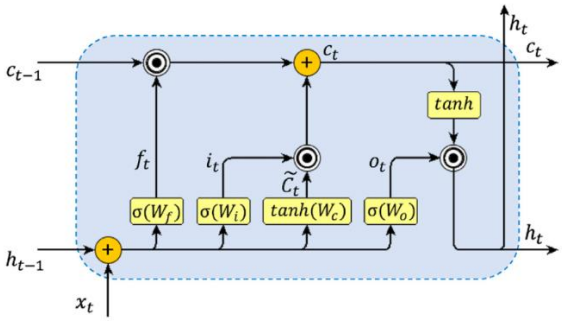

Figure 3. Structure of the Long Short-Term Memory (LSTM) neural network

These gates govern the flow of information at each timestep, and their function is described by a series of mathematical equations [18] [19] as follow:

$$i_t\& = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (1)$$

$$f_t\& = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \quad (2)$$

$$o_t\& = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$

$$c_t\& = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (4)$$

$$\tilde{c}_t\& = \tanh\left(W_c x_t + U_c h_{t-1} + b_c\right) \quad (5)$$

$$h_t\& = o_t \odot \tanh\left(c_t\right) \quad (6)$$

Where σ,$\odot$ refer to the logistic sigmoid function and the elementwise multiplication operator, respectively. [1] Assume that n is the batch size, h is the number of hidden units, and d is the number of inputs; $x_t \in \mathbb{R}^{n\times d}$ is the input vector (i.e., word embedding) at time t. i_t, f_t, and $o_t \in \mathbb{R}^{n\times h}$ correspond to the input gate, the forget gate, and the output gate of the LSTM at time t respectively. c_t is the memory cell state at timestep t. $\tilde{c}_t$ represents the candidate memory cell vector derived from the current input and the previous hidden state, containing potential candidates to be incorporated into the state, while $h_t \in \mathbb{R}^{n\times h}$ is the hidden state [1]. The weights in the LSTM layers are learned during the training phase [1]. W_i, W_f, and $W_o \in \mathbb{R}^{d\times h}$ are denoted weight matrices for hidden state h_t. U_i, U_f, and $U_o \in \mathbb{R}^{h\times h}$ are weight parameters of different gates for input x_t. b_i, b_f, b_o, and $b_c \in \mathbb{R}^{1\times h}$ are bias parameters [20]. $h_{t-1} \in \mathbb{R}^{n\times h}$ is the hidden state at time $t-1$ [1].

The LSTM modules contain computational blocks that control the flow of information, allowing the LSTM cells to track information during timestep t [19]. The forget gate f_t may or may not pass the previous memory h_{t-1} and decides what information needs to be removed from the LSTM memory. The input gate i_t decides whether to add new information to the LSTM memory. This gate has two layers. The sigmoid function determines how much information passes through the gate, ranging from nothing (0) to everything (1). The tanh layer generates new candidate vector values $\tilde{c}_t$ between -1 and 1 . The input gate function allows the LSTM to selectively add or remove information from its cell state . The output gate o_t determines whether or not to pass the output of the memory cell [21] [19]. Here is the architecture of LSTM as seen in Figure 4

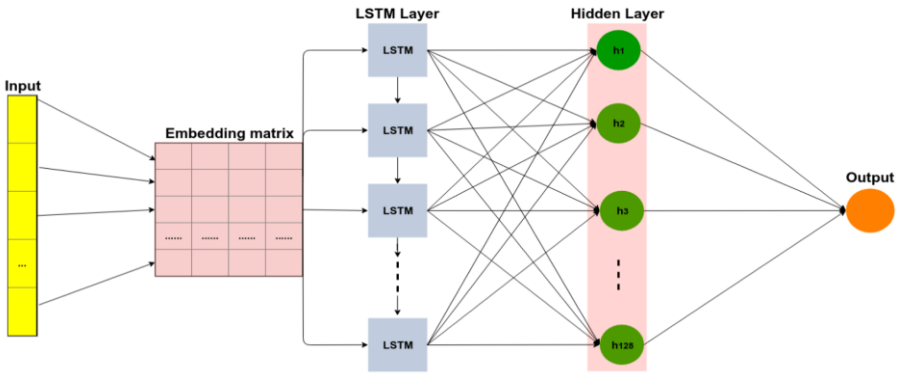

Figure 4. LSTM Architecture

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

Figure 4 illustrates an example of LSTM architecture applied to text classification. The input data is first transformed into an embedding matrix. This matrix then feeds into the LSTM layer, which consists of 200 cells. The output from the LSTM layer is passed to a fully connected layer with 128 cells. This final layer employs a sigmoid activation function to reduce the 128-dimensional vector into a single output vector, corresponding to the two classes to be predicted (positive and negative) [22].

2.5 Bidirectional Long Short Term Memory – Attention

a. Bidirectional Long Short Term Memory
BiLSTM enhances the traditional LSTM model by utilizing two distinct LSTM hidden layers. These layers process the input sequence in opposite directions, with one layer analyzing the sequence from beginning to end and the other from end to beginning.

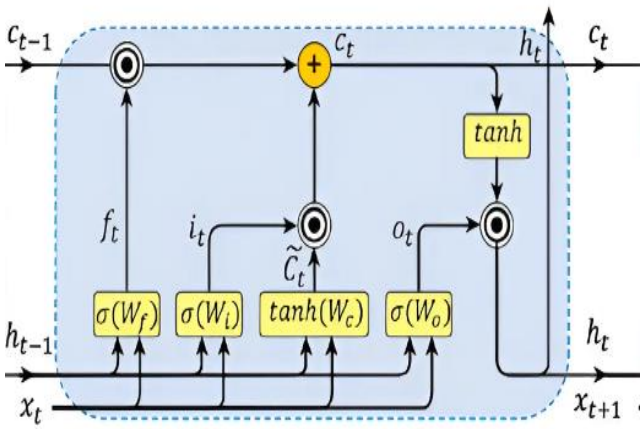

Figure 5. The structure of the BiLSTM neural network diagram

This allows for a more comprehensive understanding of the context surrounding each element in the sequence, overcoming the limitation of standard LSTM models that only consider past information [21]. For a specific time t, the input is $x_t = (x^1, x^2, \ldots, x_n) \in R^{nxd}$. The internal representations for both the forward and backward directions are $h_t \in \mathbb{R}^{n \times d}$ and $h_t \in \mathbb{R}^{n \times d}$ sequentially. We can calculate the forward and backward hidden states as follows:

$$\vec{h}_t = \sigma\left(W_{\vec{h}} x_t + W_{\vec{h}\vec{h}} \vec{h}_{t-1} + b_{\vec{h}}\right) \quad (7)$$

$$\overleftarrow{h}_t = \sigma\left(W_{\overleftarrow{h}} x_t + W_{\overleftarrow{h}} \overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}\right) \quad (8)$$

$$y_t = W_{\vec{h}y} \vec{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y \quad (9)$$

The output layer integrates the hidden states from both the forward and backward layers. BiLSTM produces a series of hidden states represented as:

$$y_t = \sigma\left[\vec{h}_t, \overleftarrow{h}_t\right] \quad (10)$$

Where the function $\sigma$is used to merge the two output sequences. The final output is calculated by merging the outputs from the forward and backward layers, drawing upon the internal representations of both LSTMs. The final hidden state h_t represents the entire sentence, where h_t equals $\left[\vec{h}_t, \overleftarrow{h}_t\right]$

b. Attention Layer
The attention layer is a mechanism that enables the model to concentrate on specific parts of the input data that are most pertinent to the given task. Similar to how humans naturally focus on key words or phrases to grasp the overall meaning of a sentence, the attention layer emulates this selective focus within a neural network model. It emphasizes the most informative words, integrating their representations to form a comprehensive sentence vector. Essentially, the attention mechanism computes the context vector within a sentence [23].
In more technical terms, an LSTM or BiLSTM network generates a hidden state h_t state at each time step. This vector h_t is first processed through a single-layer MLP (Multilayer Perceptron) to derive the hidden representation u_t. Subsequently, a scalar importance value fo h_t s calculated based on u_t and the word-level context vector u_w. The model then employs the softmax function to compute the weighted average of the states h_t. The context vector u_w acts as a high-level representation [24], aiding in distinguishing the significance of different words within the word set. The pertinent formulas are expressed as follows:

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

$$u_t = \tanh\left(W_w h_t + b_w\right) \quad (11)$$

$$a_t = \frac{\exp(u_t^T u_w)}{\sum_t \exp\left(u_t^T u_w\right)} \quad (12)$$

$$c = \sum_t a_t h_t \quad (13)$$

2.7 SMOTE

SMOTE, or Synthetic Minority Oversampling Technique, is a popular strategy to tackle the problem of uneven class distribution in datasets. Class imbalance is a condition where one or more classes have significantly fewer samples than other classes. This often occurs in various applications, such as fraud detection, medical diagnosis, and text classification. SMOTE achieves balance between classes by generating additional samples for the minority class, effectively boosting its size to match that of the majority class. [10] Deep SMOTE, a new method to address imbalanced datasets in deep learning. This innovative approach combines the strengths of the well-established SMOTE algorithm with the capabilities of deep neural networks.

2.8 Stratified K-Fold

Stratified K-Fold Cross-Validation is a validation technique that ensures each data fold contains a proportional representation of classes, mirroring the overall dataset distribution. This is crucial when dealing with imbalanced data, as it ensures the trained model can effectively recognize and predict all classes. By dividing the dataset into k sections, making sure each section maintains the same ratio of classes as the original dataset, this technique enables a more robust model evaluation through training and testing on multiple data subsets [11].

3. RESULT AND DISCUSSION

This section delves into the outcomes and analysis of the research, with a particular focus on the practical application of the employed methodology. This can be achieved through a straightforward presentation of the study's data. Additionally, the section encompasses various illustrative elements such as explanations, images, tables, and other relevant visuals.

3.1 Data Collection and Preprocessing

In this experiment, the dataset consists of 20,000 hotel reviews from Kaggle, which is publicly available. The dataset is sourced from TripAdvisor and stored in a CSV file. This dataset contains two columns: Review and Rating column. The following steps were taken for data collection and preprocessing.

First, text preprocessing was performed, which included text cleaning, tokenization, lemmatization, and removal of stop words. Text cleaning involved removing non-alphabet characters and extra spaces, and converting the text to lowercase. After that, the cleaned text was tokenized into separate words, then the words were lemmatized, and stop words were removed using NLTK.

Next, the Rating column was transformed into sentiment labels based on the following rules: ratings 1 and 2 were categorized as 'negative', rating 3 as 'neutral', and ratings 4 and 5 as 'positive'. These sentiment labels were then converted into categories.

The following step was to load word embedding models (FastText, Word2vec, and Doc2vec). These were pre-trained to convert review text into vector representations. Each review was transformed into the average vector of the words present in the word embedding model. The resulting text vectors were then normalized to ensure a uniform scale.

With these steps, the hotel review data was prepared for further model training processes, including text cleaning and normalization, as well as converting the reviews into appropriate vector representations.

3.2 Experiment Parameter Settings

This section describes the parameters and hyperparameters used in the implementation of sentiment analysis models based on LSTM, BiLSTM, and BiLSTM-Attention. These models are designed to classify hotel review sentiments into three categories: negative, neutral, and positive. The parameters and hyperparameters were carefully chosen and adjusted through experiments to optimize the models' performance in sentiment classification tasks.

a. Converting Text to Vectors
   Each review text was converted into vectors using FastText, Word2vec, and Doc2vec models. These vectors were obtained by calculating the average word vector for each token in the text corresponding to the FastText, Word2vec, and Doc2vec models. The resulting vectors were then normalized to ensure a consistent scale.
b. Definition of Sentiment Analysis Models
   Sentiment analysis models such as LSTM, BiLSTM, and BiLSTM-Attention are defined by specific parameters. These parameters include an input size of 300, which corresponds to the vector size of FastText, Word2vec, and

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

Doc2vec. The hidden size is set to 128 units, with 2 layers, and a dropout rate of 0.3. These models include a sentiment analysis layer to process word vector sequences and a fully connected layer to classify sentiment.

c. Cross-Validation and Hyperparameter Tuning
Cross-validation was performed using Stratified K-Fold with 5 folds to ensure balanced class distribution. In each fold, the training data was augmented using SMOTE to address class imbalance. The data was then converted into tensors and fed into a DataLoader for training. The LSTM model was trained using the Adam optimizer and Cross-Entropy loss function weighted according to class distribution. Each model was trained for 20 epochs, and its performance was evaluated on the validation data.

Several key parameters and hyperparameters were set to optimize model performance. The parameters set include learning rate, number of epochs, and batch size. Additionally, hyperparameters such as the number of hidden layers in LSTM/BiLSTM/BiLSTM-Attention, the number of units in each layer, and dropout rate were optimized for each combination of word embedding methods and model architectures. The following table summarizes the parameters and hyperparameters used:

Table 1. Parameters and Hyperparameters

Parameter	Value	Description
Device	cuda	Performance by GPU
input_size	300	Word vector size
hidden_size	128	LSTM, BiLSTM, BiLSTM-Att units
output_size	3	Number of sentiment categories
num_layers	2	LSTM, BiLSTM, BiLSTM-Att layer
dropout	0.3	Dropout rate
Optimizer	Adam	Optimizer used for model training
learning rate	0.001	Learning rate for the optimizer
loss function	Cross Entropy Loss	Loss function used
class weights	Based on class distribution	SMOTE class weights
num_folds	5	Number of folds in cross-validation
num_epochs	20	Number of epochs in model training

d. Final Evaluation on Test Set
The best model from the cross-validation process was evaluated on the entire test set. This evaluation included calculating accuracy, confusion matrix, and a classification report covering precision, recall, and F1 score metrics for each sentiment class. These final results provide a comprehensive overview of the model's performance on unseen data during training. With this approach, the models are effectively trained and evaluated to classify sentiments from hotel reviews, enhancing the effectiveness of sentiment analysis and word embedding models on imbalanced datasets

3.2 Experimental Result and Analysis

In this study, the models were tested on a test dataset. The experimental effect was evaluated using three indicators: Accuracy (A), Recall (R), and F1 (F1-score). Accuracy represents the percentage of accurately classified reviews in the review text. Recall rate represents the percentage of all actual reviews that are correctly classified in the sample.

In this experiment, positive and negative categories were used as positive and negative samples to assess the model's precision in review categorization. The neutral category was used to assess the model's ability to distinguish sentiments that do not lean towards positive or negative.

In this study, three different word embedding models, namely FastText, Word2Vec, and Doc2Vec, were used to generate word vector representations. Each word embedding model was trained on the same dataset. Subsequently, the following sentiment analysis models were built and compared:

a. LSTM: LSTM-FastText, LSTM-Word2Vec, LSTM-Doc2Vec
b. BiLSTM: BiLSTM-FastText, BiLSTM-Word2Vec, BiLSTM-Doc2Vec
c. BiLSTM-Attention: BiLSTM-Attention-FastText, BiLSTM-Attention-Word2Vec, BiLSTM-Attention-Doc2Vec

To ensure accurate model evaluation and address potential class imbalances, we employed Stratified K-Fold cross-validation with 5 folds. In each fold, the training data was oversampled using the SMOTE technique to balance the class distribution. The best model was selected based on the average validation accuracy across all folds, and then evaluated on a separate test set to obtain the final performance.

Furthermore, the results of this study, presented in the table and figure, provide valuable insights into the performance of each combination.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

Table 2. Comparison of Model Performance

Model	Accuracy	Weighted Avg Recall	Weighted Avg F1-Score	Macro Avg Recall	Macro Avg F1-Score
FastText-LSTM	0.83	0.83	0.85	0.75	0.73
Word2Vec-LSTM	0.93	0.93	0.93	0.9	0.87
Doc2Vec-LSTM	0.96	0.96	0.96	0.93	0.93
FastText-BiLSTM	0.85	0.85	0.86	0.79	0.76
Word2Vec-BiLSTM	0.95	0.95	0.95	0.88	0.9
Doc2Vec-BiLSTM	0.96	0.96	0.96	0.92	0.93
FastText-BiLSTM-Attention	0.85	0.85	0.86	0.76	0.74
Word2Vec-BiLSTM-Attention	0.95	0.95	0.95	0.92	0.92
Doc2Vec-BiLSTM-Attention	0.96	0.96	0.96	0.93	0.93

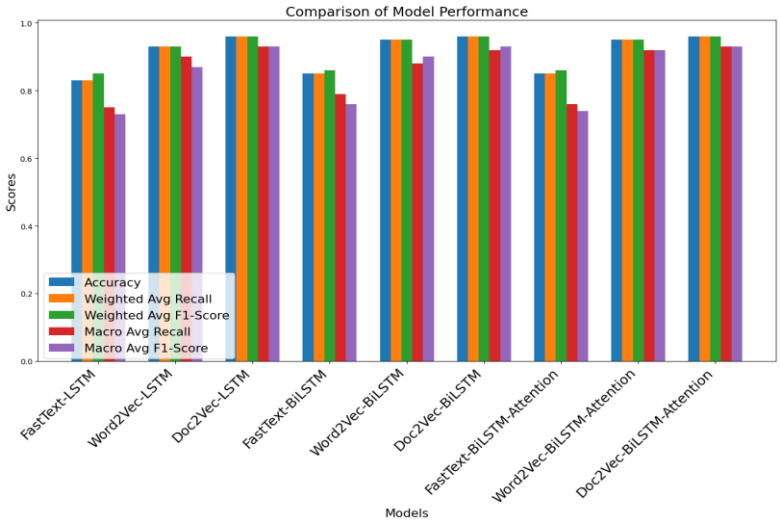

Figure 7. Comparison of Model Performance

Overall, it is evident that the use of Doc2Vec as the word embedding method consistently yields the best performance across all evaluation metrics (accuracy, weighted average recall, weighted average F1-score, macro average recall, and macro average F1-score). This indicates that Doc2Vec is able to capture better semantic representations of words compared to FastText and Word2Vec, thus contributing to improved accuracy and generalization capability of the model.

The BiLSTM architecture, especially when combined with the attention mechanism, also shows a significant performance improvement compared to the standard LSTM architecture. This demonstrates that BiLSTM can capture contextual information from both directions (forward and backward), while the attention mechanism allows the model to focus on the most relevant parts of the text for sentiment classification tasks.

Although FastText shows the lowest overall performance, the combination of FastText with BiLSTM and attention architecture yields a significant improvement compared to the standard FastText-LSTM. This suggests that a more complex model architecture can help mitigate the limitations of a less optimal word embedding method.

This study successfully achieves its goal of enhancing the effectiveness of sentiment analysis models. The combination of Doc2Vec with BiLSTM-Attention, supported by the application of SMOTE and Stratified K Fold, proves to be the most effective, achieving the highest accuracy and best performance across all evaluation metrics. However, the results also indicate that the choice of word embedding method and model architecture should be tailored to the characteristics of the dataset and the goals of the sentiment analysis.

4. CONCLUSION

This study successfully identified that Doc2Vec is the most superior word embedding method. By combining Doc2Vec with SMOTE techniques to address class imbalance and Stratified K-Fold cross-validation for more robust model evaluation, the Doc2Vec-LSTM and Doc2Vec-BiLSTM models achieved an accuracy, weighted average recall, and weighted average f1-score of 0.96, with a macro average recall and macro average f1-score of 0.93. The combination of Doc2Vec with BiLSTM-Attention demonstrated the best performance in understanding text sentiment, achieving the same results with an accuracy, weighted average recall, and weighted average f1-score of 0.96, as well as a macro average recall and macro average f1-score of 0.93.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 2, September 2024, Page: 981−990
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i2.5721

These findings have broad practical implications, particularly in applications requiring a deep understanding of text sentiment, such as social media sentiment analysis and public opinion surveys. However, it should be acknowledged that the generalizability of the model may be limited to domains and types of texts similar to those in the dataset used in this study.

Future research is expected to expand the scope by testing models on larger and more diverse datasets, as well as exploring a wider range of architectures. Additionally, focusing on developing more effective techniques for handling class imbalance and addressing overfitting issues is also important. Thus, sentiment analysis models can become increasingly accurate and reliable.

ACKNOWLEDGMENT

I want to thank the University of Amikom Yogyakarta for their support and excellent environment. Special thanks to Prof. Dr. Kusrini M.Kom for your invaluable guidance. To my family, your love and support have been my strength. To my wife, your faith and sacrifices inspire me. Finally, I dedicate this work to my daughter, Wafiza, with the hope that she will surpass my achievements.

REFERENCES

[1] A. Al Hamoud, A. Hoenig, and K. Roy, "Sentence subjectivity analysis of a political and ideological debate dataset using LSTM and BiLSTM with attention and GRU models," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 7974–7987, Aug. 2022, doi: 10.1016/j.jksuci.2022.07.014.
[2] J. Sangeetha and U. Kumaran, "A hybrid optimization algorithm using BiLSTM structure for sentiment analysis," *Measurement: Sensors*, vol. 25, Aug. 2023, doi: 10.1016/j.measen.2022.100619.
[3] Y. Pei, S. Chen, Z. Ke, W. Silamu, and Q. Guo, "AB-LaBSE: Uyghur Sentiment Analysis via the Pre-Training Model with BiLSTM," *Applied Sciences (Switzerland)*, vol. 12, no. 3, Aug. 2022, doi: 10.3390/app12031182.
[4] B. A. Chandio, A. S. Imran, M. Bakhtyar, S. M. Daudpota, and J. Baber, "Attention-Based RU-BiLSTM Sentiment Analysis Model for Roman Urdu," *Applied Sciences (Switzerland)*, vol. 12, no. 7, Aug. 2022, doi: 10.3390/app12073641.
[5] X. Li, Y. Lei, and S. Ji, "BERT- and BiLSTM-Based Sentiment Analysis of Online Chinese Buzzwords," *Future Internet*, vol. 14, no. 11, Aug. 2022, doi: 10.3390/fi14110332.
[6] Y. Yuan, W. Wang, G. Wen, Z. Zheng, and Z. Zhuang, "Sentiment Analysis of Chinese Product Reviews Based on Fusion of DUAL-Channel BiLSTM and Self-Attention," *Future Internet*, vol. 15, no. 11, Aug. 2023, doi: 10.3390/fi15110364.
[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.
[8] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
[9] P. Wu, X. Li, C. Ling, S. Ding, and S. Shen, "Sentiment classification using attention mechanism and bidirectional long short-term memory network," *Appl Soft Comput*, vol. 112, p. 107792, Aug. 2021, doi: 10.1016/J.ASOC.2021.107792.
[10] D. Dablain, B. Krawczyk, and N. V Chawla, "DeepSMOTE: Fusing deep learning and SMOTE for imbalanced data," *IEEE Trans Neural Netw Learn Syst*, vol. 34, no. 9, pp. 6390–6404, 2022.
[11] T. R. Mahesh, O. Geman, M. Margala, M. Guduri, and others, "The stratified K-folds cross-validation and class-balancing methods with high-performance ensemble classifiers for breast cancer classification," *Healthcare Analytics*, vol. 4, p. 100247, 2023.
[12] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.
[13] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," Aug. 2016.
[14] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," in *International workshop on artificial intelligence and statistics*, 2005, pp. 246–252.
[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Adv Neural Inf Process Syst*, vol. 26, 2013.
[16] K. I. Gunawan and J. Santoso, "Multilabel text classification menggunakan svm dan doc2vec classification pada dokumen berita bahasa indonesia," *Journal of Information System, Graphics, Hospitality and Technology*, vol. 3, no. 01, pp. 29–38, 2021.
[17] S. Khotijah, J. Tirtawangsa, and A. A. Suryani, "Using lstm for context based approach of sarcasm detection in twitter," in *Proceedings of the 11th international conference on advances in information technology*, 2020, pp. 1–7.
[18] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, 2013, pp. 6645–6649.
[19] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.
[20] K. Wang, J. He, and L. Zhang, "Sequential weakly labeled multiactivity localization and recognition on wearable sensors using recurrent attention networks," *IEEE Trans Hum Mach Syst*, vol. 51, no. 4, pp. 355–364, 2021.
[21] X. Ma and E. H. Hovy, "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF," *CoRR*, vol. abs/1603.01354, 2016, [Online]. Available: http://arxiv.org/abs/1603.01354
[22] N. C. Dang, M. N. Moreno-Garc\'\ia, and F. la Prieta, "Sentiment analysis based on deep learning: A comparative study," *Electronics (Basel)*, vol. 9, no. 3, p. 483, 2020.
[23] X. Yao, "Attention-based BiLSTM neural networks for sentiment classification of short texts," in *Proc. Int. Conf. Inf. Sci. Cloud Comput*, 2017, pp. 110–117.
[24] S. Sukhbaatar, J. Weston, R. Fergus, and others, "End-to-end memory networks," *Adv Neural Inf Process Syst*, vol. 28, 2015.