Available online at website: https://jurnal.iaii.or.id/index.php/RESTI

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 9 No. 4 (2025) 935 - 943 e-ISSN: 2580-0760

Language Processing for Detecting Fake News on Twitter Using a Long Short-Term Memory Architecture

Rini Sovia[1*], Dwi Andhara Valkyrie[2], Ruri Hartika Zain[3], Firdaus[4]
[1,2,3,4]Department Information Technology, Faculty of Computer Science, Universitas Putra Indonesia YPTK, Padang, Indonesia

[1]rini_sovia@upiyptk.ac.id, [2]dwiandharavalkyrie@gmail.com, [3]rurihartika_zain@upiyptk.ac.id, [4]firdaus@upiyptk ac.id

Abstract

The rapid spread of misinformation on social media platforms, particularly X (formerly Twitter), poses a significant challenge to public trust and democratic integrity. Fake news is often crafted to deceive readers and manipulate public opinion, especially in political contexts such as the 2024 Regional Head Elections (Pilkada 2024). Although various measures have been proposed to mitigate this issue, achieving an effective balance between controlling misinformation and preserving free speech remains a challenge. This study aims to address this problem by developing a fake news detection model based on Natural Language Processing (NLP) and Long Short-Term Memory (LSTM). The dataset used in this study was collected from public tweets related to Pilkada, with Kompas.com serving as the validation source to verify content authenticity. Experimental results show that the proposed LSTM model outperformed traditional classification methods, achieving a precision, recall, and F1-score of 0.95, along with an overall accuracy of 94.90%. Confusion matrix analysis further confirmed the reliability of the model by demonstrating low misclassification rates. This study contributes to the advancement of AI-driven hoax detection systems, offering an automated and scalable solution for combating misinformation in political discourse.

Keywords: Fake news detection; LSTM; NLP; political hoaxes; social media analysis

How to Cite: Rini Sovia, Firdaus, Ruri Hartika Zain, and Dwi Andhara Valkyrie, "Language Processing for Detecting Fake News on Twitter Using a Long Short-Term Memory Architecture ", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 9, no. 4, pp. 935 - 943, Aug. 2025.
Permalink/DOI: https://doi.org/10.29207/resti.v9i4.6570

Received: April 22, 2025
Accepted: August 8, 2025
Available Online: August 25, 2025

This is an open-access article under the CC BY 4.0 License
Published by Ikatan Ahli Informatika Indonesia

1. Introduction

Advancements in communication technology have significantly enhanced the way humans send and receive information. As a result, virtual communities have emerged, gradually transforming traditional social structures. One of the most prominent virtual spaces is social media, where information is rapidly disseminated through online interactions and discussions [1]. Over the past decade, social media has evolved into one of the primary sources of information for people worldwide. One of the largest and most influential social media platforms today is Twitter, now known as X. On one hand, Twitter provides easy access to fast and real-time information dissemination. On the other hand, it carries significant risks in spreading fake news—unverified or deliberately falsified information designed to mislead and manipulate readers [2]. Although Twitter provides easy access to fast, real-time information, this platform carries significant risks in terms of spreading fake news information that is either unverified or deliberately falsified with the intention of misleading and manipulating readers [3].

The spread of political hoaxes not only distorts public perception of regional leaders but also influences voter decisions and exacerbates social polarization [4]. In some cases, the circulation of false information can shift political support, potentially undermining the integrity and quality of Indonesia's democracy. Therefore, efforts to combat hoax dissemination are crucial in maintaining political stability and preserving public trust in the democratic process.

During the 2024 Pilkada campaign, hoaxes spread rapidly through social media platforms like Twitter, allowing misinformation to circulate widely in a short time, often without proper verification. As Twitter's role in information dissemination grows, managing and verifying the accuracy of circulating content

becomes increasingly challenging. In this context, technology plays a crucial role in addressing the spread of fake news [5]. One of the key technologies in the modern era is Artificial Intelligence (AI), which aims to develop machines capable of human-like thinking. With its efficiency and effectiveness, AI serves as a potential solution for addressing information challenges, including hoax detection, by enabling fast and accurate data analysis [6]. In the field of NLP, Machine Learning (Deep Learning) has become the primary solution, defining research challenges and creating high-quality evaluation methods and test collections is critical to success [7]. NLP enables machines to interpret and analyse text data, identifying linguistic patterns and clues that may indicate fake news. Meanwhile, AI particularly through machine and deep learning techniques improves the accuracy and scalability of detection systems [8]. NLP is a subfield of AI that plays a crucial role in this endeavour, enabling machines to understand, interpret and manipulate human language [9]. The main challenge lies in integrating social network analysis into the detection of fake news. Although NLP can analyse the content of news articles, it is equally important to understand how these articles spread across various social media platforms is equally important [10].

An effective AI approach to text analysis is NLP, which enables machines to understand human language. NLP is crucial in detecting social media hoaxes, as it analyzes text structure, word patterns, and contextual cues typical of fake news [11]. It can also spot features like exaggerated language and unusual dissemination patterns. To boost performance, NLP is often paired with deep learning models like LSTM, which outperform traditional RNNs by handling long text sequences and overcoming issues like vanishing gradients. In hoax detection, LSTM enhances accuracy by capturing subtle linguistic patterns and deceptive writing styles [12].

Several studies have examined information spread on social media, especially related to fake news, hate speech, and efforts to reduce their harmful effects. LSTM models, due to their inherent ability to capture temporal dependencies in sequential data, have consistently shown superior performance in text classification tasks, including fake news and hate speech detection [13]. Studies by [14] explicitly highlight LSTM's efficacy, with demonstrating an 84.39% accuracy in hate speech detection when paired with Word2Vec embeddings, significantly outperforming classical methods like Naïve Bayes. Similarly, reinforced LSTM's superiority over Naïve Bayes in fake news detection, identifying it as achieving state-of-the-art accuracy within their experimental setup. These findings collectively underscore the strength of deep learning in modeling complex textual dependencies, particularly relevant for the often-subtle cues of misinformation [15].

The integration of contextual language models further refines detection capabilities. Reference notes that NLP and Neural Networks (up to 92.7% and 90.1% accuracy, respectively) generally outperform traditional models in detecting social media hoaxes, emphasizing the maturity of context-aware models in handling complex, emotion-laden misinformation [16]. Building on this, the IndoBERT-LSTM hybrid model presented in achieved an impressive 93.2% accuracy in Indonesian fake news detection [17]. This result is particularly noteworthy as it combines IndoBERT's robust contextual language understanding with LSTM's sequential modeling, effectively capturing nuanced linguistic cues indicative of disinformation and setting a high benchmark for the domain. Another approach leveraging deep learning, showcases the application of Convolutional Neural Network (CNN) algorithms, enhanced by Easy Data Augmentation (EDA) techniques (synonym replacement, random insertion, swapping, and deletion), achieving 82.81% accuracy in Indonesian fake news detection. This confirms CNN’s adaptability in linguistic pattern recognition and demonstrates how enriched training samples can improve model generalization [18].

Despite these advancements, critical research gaps persist, particularly concerning model robustness, adaptability to evolving misinformation, and generalizability. Previous studies highlight the need for larger datasets and more advanced hybrid architectures to improve generalization. Furthermore, models with fixed training windows, like the IndoBERT-LSTM in, underscore the necessity for adaptive systems capable of continuous retraining on emerging data streams to maintain relevance. Concerns also exist regarding cross-linguistic generalization and parameter optimization across diverse social media datasets, alongside the ongoing exploration of alternative augmentation strategies.

While LSTM has been widely used in text classification, few studies incorporate verified fact-checking sources like Kompas.com for manual labeling in the Indonesian political context. This research uniquely combines verified local ground truth with GloVe embeddings to improve detection accuracy in Pilkada-related misinformation. Our study uniquely integrates GloVe for word embedding to capture richer semantic relationships. Crucially, it utilizes Kompas.com as a verified news source for manual dataset labeling, providing high-quality, reliable ground truth for politically-related content. By focusing specifically on Indonesian political hoaxes related to the 2024 Pilkada, our approach develops a model directly informed by current and verified information, enhancing its adaptability and robustness within a sensitive socio-political context. This distinct combination of robust methodology and meticulously verified, context-specific data distinguishes our work, paving the way for more accurate and reliable hoax detection.

2. Methods

The this research method outlines the research design, data collection techniques, pre-processing stages and analytical procedures that were employed.

2.1 Dataset

The dataset used in this study is tweet. To collect tweets, we perform a crawling process. Crawling is the process of extracting data, whether large or small, from web pages and storing it in local storage. The data is retrieved based on specific keywords relevant to the search criteria. The crawling process utilized Twitter as the data source, leveraging the Twitter API (Application Programming Interface) provided by Twitter developers. This API serves as a bridge between the system and Twitter, enabling efficient data retrieval and processing. The crawling process was conducted in September 2024 until January 2025, as this period saw heightened discussions on the 2024 Regional Elections (Pilkada) on X (Twitter). This timeframe reflects significant public engagement and discourse surrounding the issue. A total of 10.257 tweets were successfully collected during the crawling process as seen in Table 1.

Table 1. Preview Dataset

Tweet Content	TimeStamp
Rabu 27 Nov 2024 di TPS 6 Desa Juranalas Kec Alas Kab Sumbawa NTB. Saat kotak suara dibuka kedapatan sebanyak 121 surat suara sudah dicoblos. Menurut Gakumdu Tidak ada kasus pidana. KPPS diduga langgar kode etik. Bagaimana menurut pakar hukum @mohmahfudmd	06/12/2024
NGAMUK! Pemilih Tendang Meja Petugas KPPS di Kubu Bali karena di halangi memilih paslon tertentu, Sejumlah Surat Suara Rusak	27/11/2024
Sesuai dengan jadwal yang sudah ditetapkan oleh Komisi Pemilihan Umum (KPU), Pilkada tahun ini berlangsung pada Rabu, 27 November 2024.	20/09/2024

This study uses Kompas.com, specifically the #CekFakta section, as a data validator. Kompas.com classifies news based on verified accuracy, distinguishing between hoaxes and factual reports. By leveraging #CekFakta Kompas.com, this study ensures the analyzed data comes from a credible source that has undergone thorough verification by professional journalists, enhancing reliability in the fight against misinformation.

From the 10.257 tweets collected, the data was analyzed by referencing Kompas.com as a validator and categorized into hoax and non-hoax groups. Based on Table 2, the labeling process assigned 1 for hoax news and 0 for non-hoax news, ensuring a structured classification for further analysis.

Table 1 presents the dataset distribution, showing a nearly balanced proportion between hoax and non-hoax tweets. This balanced distribution helps ensure that the classification model can learn effectively from both categories, reducing potential bias in detection performance.

Table 2. Dataset Distribution

Class	Label	Count
Hoax	1	5027
Non-Hoax	0	5230
Total:		10.257

2.2 Preprocessing

Twitter data consists of unstructured and complex text, making preprocessing a crucial step to remove irrelevant characters and retain meaningful words [19]. This process helps enhance the clarity and reliability of the data, ensuring that the extracted information is more relevant and useful for further analysis. The preprocessing stages in this study can be seen as shown in Figure 1.

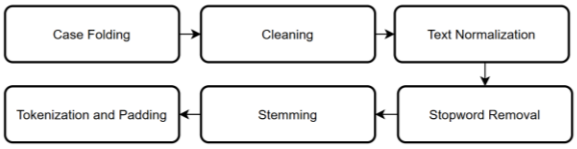


Figure 1. Preprocessing

The text pre-processing stage includes several important steps. Case folding is done by changing all letters in the text to lowercase to make text analysis more consistent [20]. Next, cleaning is performed, which is cleaning the text from special characters such as punctuation, symbols, or irrelevant numbers so that the quality of the analysis can be improved [21]. After that, text normalization is applied to replace abbreviations with full forms, change non-standard or slang words to formal forms, and unify word variations into a uniform form as a standardization effort [22]. In the next stage, stopword removal is performed, which is the removal of common words that often appear but have no significant meaning, such as "yang," "di," or "itu," to make the analysis results more accurate [23]. Next, stemming is used to return words to their basic form by removing affixes, for example the word "menggunakan" is changed to "guna," so that words with various forms can be analyzed consistently. Finally, tokenization and padding are carried out, namely breaking the text into small units called tokens (in the form of words, phrases, or characters), then standardizing the length of the text by adding special tokens (such as the number 0) to the shorter text so that all data has a uniform input length [24].

2.3 Word Embedding using GloVe

Word embedding is a technique in NLP used to represent words as numerical vectors, making them interpretable by machine learning models [25]. This approach captures word meanings and semantic relationships between words within a text, enhancing the model's understanding of linguistic context Word embedding can be implemented using various methods

such as Word2Vec, GloVe, and FastText, each employing different approaches to constructing word vector representations. In this study, GloVe is used to capture the semantic relationships between words through a co-occurrence matrix. GloVe constructs this matrix by analyzing how frequently words appear together in a given corpus, allowing it to effectively represent word relationships. This method facilitates the processing of large-scale training data while generating meaningful vector representations of words. The resulting vectors contain semantically similar words, which are further refined through an expansion process to enhance their contextual understanding [26].

2.4 LSTM Model

LSTM model is an advanced variant of the Recurrent Neural Network (RNN) designed to address RNN's limitations in capturing long-term dependencies. Unlike traditional RNNs, LSTM can retain long-term information, making it highly effective for tasks such as sentiment analysis and hoax news detection [27]. Additionally, LSTM excels at extracting features from sequential input data, where information is converted into vectors and processed efficiently to recognize complex patterns over time. Figure 2, shows the Architecture of LSTM memory [28].

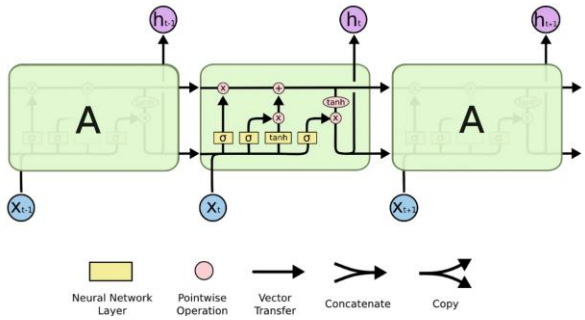


Figure 2. Architecture of Long Short-Term Memory.

LSTM achieves this capability through a unique architecture that regulates the flow of information within its network. This mechanism allows it to selectively retain important data while discarding irrelevant details, ensuring efficient learning from sequential input. At the core of LSTM's structure are the cell state and three essential gates, which control how information is stored, updated, and passed through the network [29].

The forget gate uses a sigmoid function to decide which information to remove from the cell state. It processes the previous output and input generating values between 0 and 1, where 1 means 'retain' and 0 means discard (1) [30] as shown in Equation 1.

$$f_t = \sigma\left(W_f \times [h_t, x_t] + b_f\right) \quad (1)$$

After filtering out unnecessary information, the input gate, using a sigmoid function as shown in Equation 2, determines which new data to retain. Meanwhile, the tanh layer generates candidate values to be added to the cell state (2) [30].

$$i_t = \sigma(W_i \times [h_{t-1}, x_t] + b_i)$$
$$C\bar{t} = \tanh\,(WC.[h_{t-1}, x_t] + b_C) \quad (2)$$

The cell state is updated by combining the outputs of the forget and input gates. The forget gate removes unnecessary information, while the input gate adds new data, resulting in an updated cell state (3) [31] As shown in Equation 3.

$$C_t = f_t \times C_{t-1} + i_t \times \widetilde{C_t} \quad (3)$$

The output gate uses a sigmoid function to choose which parts of the cell state are appropriate for output. The cell state is then processed by a tanh function, normalizing values between -1 and 1. The final output is obtained by multiplying it with the sigmoid gate's result, ensuring consistency with prior decisions (4) [32] as shown in Equation 4.

$$o_t = \sigma(W_o \times [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \times \tanh\,(C_t) \quad (4)$$

2.4 Validation Parameter

Ranking F1, recall, precision, and accuracy, which are derived from the confusion matrix, are used to evaluate the performance of classification algorithms. The confusion matrix provides a comparison between the system's classification results and the actual classifications [33]. The Confusion matrix is shown in Table 3.

Table 3. Confusion Matrix Table

Actual	Predicted Positive	Negatif
Positive	TP (True Positive)	FN (False Negative)
Negative	FP (False Positive)	TN (True Negative)

Accuracy is the proportion of correct predictions relative to the total dataset (5) [34] as shown in Equation 5.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

Precision is the proportion of correctly predicted positive instances relative to the total predicted positive cases (6) [27] as shown in Equation 6.

$$presicion = \frac{TP}{TP+FP} \quad (6)$$

Recall is the proportion of correctly predicted positive instances (TP) relative to the total actual positive cases (6) [27] as shown in Equation 7.

$$recall = \frac{TP}{TP+FN} \quad (7)$$

F1-Score, or F1-Measure, is derived from the weighted harmonic mean of precision and recall values (8) [35] as shown in Equation 8.

$$f1 - score = \frac{2(precision*recall)}{(precision+recall)} \quad (8)$$

3. Results and Discussions

In this section, the results of the findings obtained based on the research are explained, then an in-depth discussion is carried out to interpret the results according to the research objectives.

3.1 Results

The data used in this study has undergone a preprocessing phase to refine and structure the raw text. Following this, the next step is the processing phase, which involves dataset splitting and model development using Long Short-Term Memory (LSTM) to process the preprocessed textual data. The model is built in stages to ensure it effectively learns data patterns and delivers optimal predictions. Dataset is divided into 80% training data and 20% testing data. In this study, the training set consists of 8.205 data points, while the testing set comprises 2.052 data points to evaluate the model's performance. The distribution of the dataset can be seen in Figure 3.

Following the dataset preparation and splitting, word embedding was implemented to transform words into fixed-dimensional numerical vectors. In this study, Global Vectors for Word Representation (GloVe) was used to enhance the model's ability to capture semantic relationships between words. The glove.6B.100d.txt pre-trained embedding was selected, where each word is represented by a 100-dimensional vector. This choice was made to balance semantic richness and computational efficiency. The implementation process began with loading the GloVe file, which contains word-vector pairs separated by spaces. Each word in the dataset was then mapped to its corresponding vector representation, allowing the model to retain contextual relationships within textual data. These embeddings were stored in a dictionary, ensuring that words present in both the dataset and the GloVe vocabulary could be utilized effectively. The embedding dimension was set to 100, as it provides a sufficient level of semantic representation without significantly increasing computational complexity.

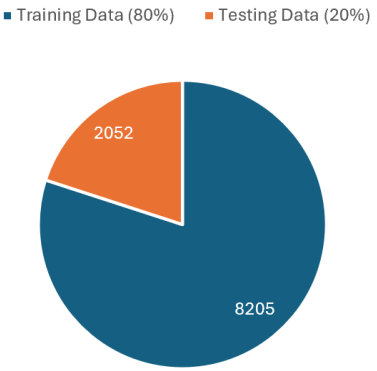


Figure 3 Training and Testing Data Comparison

Once the word embedding process was completed, the GloVe embeddings were integrated into the LSTM model to process sequential text data. The use of pre-trained embeddings enabled the model to recognize patterns in textual data more effectively, improving its ability to classify fake news. By leveraging word representations trained on large-scale datasets, the model was able to identify contextual relationships between words, ultimately enhancing classification performance. The Model Architecture can be seen on Figure 4.

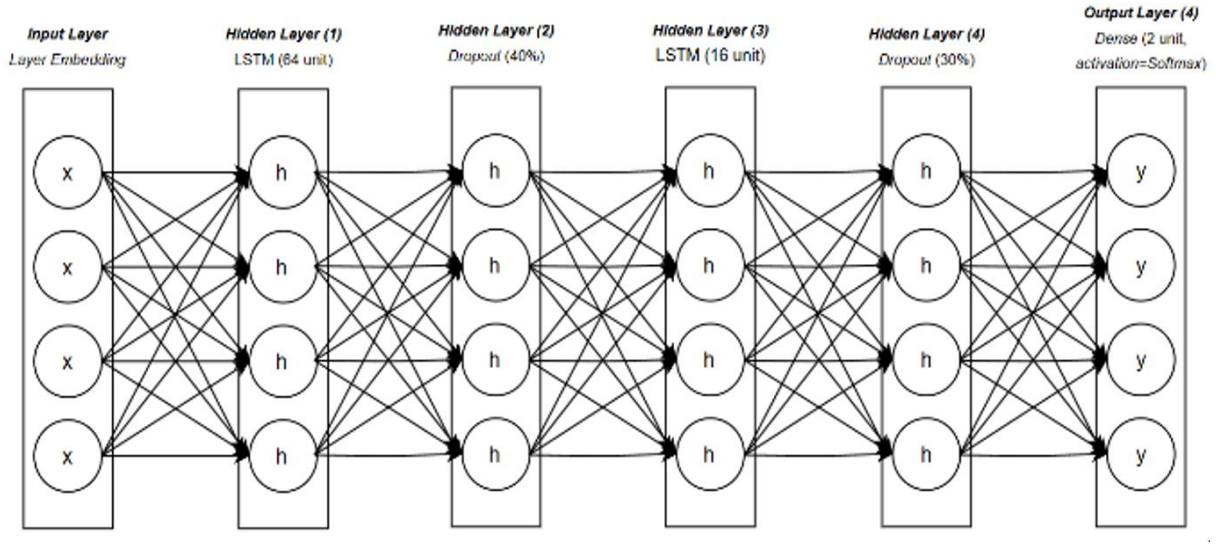


Figure 4. The Model Architecture

The input layer in this study utilizes an Embedding layer to convert numerical indices of text data into fixed-dimensional vector representations. This layer employs pre-trained embeddings from GloVe, where the embedding weights are directly extracted from the pre-existing embedding matrix without additional training. By leveraging these pre-trained word representations, the model gains a richer semantic understanding of textual data while maintaining computational efficiency. With this approach, each word in the text is transformed into a vector that captures semantic relationships between words. The

embeddings remain non-trainable (trainable=False), allowing the model to fully utilize the semantic information inherent in GloVe without additional fine-tuning. Furthermore, the mask_zero=True parameter is applied to ignore padding values (0), preventing the model from treating padding as meaningful input during training.

After the text data is transformed into word embeddings through the input layer, the next step involves processing sequential information using hidden layers. In this model, the hidden layers consist of two LSTM layers to capture word sequences and semantic relationships, along with Dropout regularization mechanisms to mitigate overfitting risks.

The first LSTM layer, consisting of 64 units, is responsible for capturing patterns within word sequences. The return_sequences=True parameter ensures that outputs from all timesteps are passed to the next layer, allowing the model to retain richer contextual information. Following this, a Dropout layer with a 40% dropout rate is applied to prevent overfitting by randomly deactivating neurons during training, ensuring that the model does not overly rely on specific features in the training data.

A second LSTM layer with 16 units is then used to refine the extracted features and produce a more compact representation of the text data. Unlike the first LSTM layer, this layer utilizes return_sequences=False, meaning only the final timestep's output is passed to the next stage. Another Dropout layer, with a 30% dropout rate, is applied after this LSTM layer to further control model complexity and reduce overfitting. By randomly deactivating neurons during training, Dropout enhances the model's generalization ability, preventing it from memorizing patterns specific to the training data while improving its performance on unseen data.

This architecture enables the LSTM model to effectively capture word relationships, retain critical textual information, and maintain a balance between accuracy and robustness against overfitting. The sequential nature of the LSTM model allows it to process text while preserving contextual dependencies over long distances, enhancing its ability to understand complex linguistic patterns. Meanwhile, Dropout regularization ensures stability throughout the training process, preventing overfitting and making the model more adaptable to real-world data [36]. After the data is processed through the hidden layers, the final step is generating predictions using a Dense layer with softmax activation. This output layer consists of two units, corresponding to the two target classes: Hoax and Non-Hoax. The softmax activation function converts the output into probability values, ensuring that the total probability always sums to 1, allowing the model to effectively classify the input into one of the two categories [37]. The summary of the result from the model show on Table 4.

Table 4. Model Summary

Layer	Output Shape	Parameter
embedding (Embedding)	(None, 206, 100)	1,200,000
lstm (LSTM)	(None, 206, 64)	42,240
Dropout (Dropout)	(None, 206, 64)	0
lstm_1 (LSTM)	(None, 16)	5,184
Dropout_1 (Dropout)	(None, 16)	0
dense (Dense)	(None, 2)	34

At this stage, the LSTM model is trained using the training dataset, optimizing weights through backpropagation and the Adam optimizer. The model minimizes the categorical crossentropy loss function to improve prediction accuracy on validation data. The training process is iterative, adjusting weights based on prediction errors measured by the loss function. To address class imbalance, class weighting is applied to prevent the model from favoring the majority class. The compute_class_weight method from Scikit-Learn automatically assigns higher weights to minority class samples, ensuring a more balanced classification.

For training, the model.fit() function is used with configurations designed for stability and efficiency. The input data, processed through tokenization and padding, is fed into the model as fixed-length sequences, while labels are converted to one-hot encoding for accurate comparison. Class weights are incorporated into the loss function, ensuring fair learning across classes. This training setup ensures that the LSTM model learns effectively, remains stable, and avoids overfitting, leading to optimal performance in hoax news classification. The result of training based on the model shown on Table 5.

Table 5. Model Training Results

Epoch	Accuracy	Loss
1	0.7286	0.5321
2	0.9181	0.1982
3	0.9404	0.1376
4	0.9471	0.1266
5	0.9536	0.1173
6	0.9500	0.1139
7	0.9696	0.0855
8	0.9664	0.0847
9	0.9672	0.0776
10	0.9765	0.0616
11	0.9795	0.0567
12	0.9792	0.0499
...	...	...
19	0.9867	0.0312
20	0.9899	0.0272

After completing the training process, the model was evaluated on the test dataset to measure its generalization performance. The final test results indicate a test loss of 0.1791 and a test accuracy of 94.90%, demonstrating that the LSTM model maintains strong predictive performance while effectively minimizing errors in hoax news classification. The relatively low test loss suggests that the model has successfully learned meaningful patterns from the data without significant overfitting. Additionally, the high accuracy score indicates that the model is capable of distinguishing between hoax and

non-hoax news with a high degree of confidence. These results confirm that the training strategy, including tokenization, padding, and class weighting, contributed to the stability and efficiency of the model in real-world scenarios. The result of evaluation between Training Accuracy and Validation shown on Figure.5.

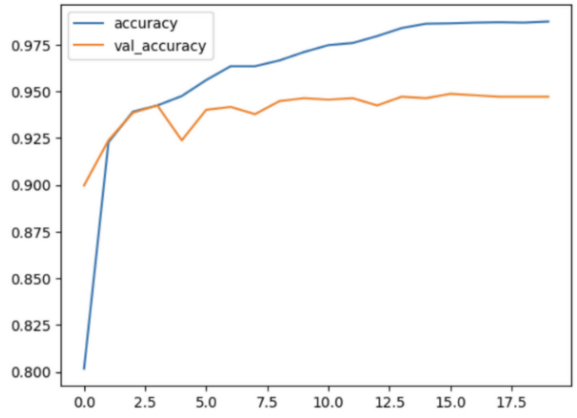


Figure 5. Training Accuracy vs. Validation Accuracy

In addition, evaluation of training losses and validation losses was also carried out as shown in Figure 6.

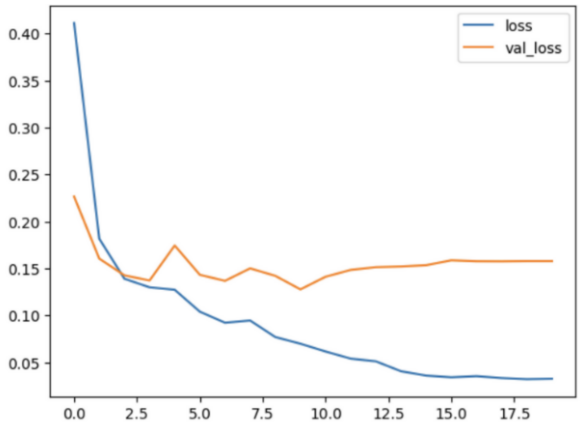


Figure 6. Training Loss vs. Validation Loss

The model is evaluated using a classification report, which provides key metrics such as precision, recall, and F1-score for each class: hoax and non-hoax. Table 6. explains the metrics in assessing model performance to distinguish two categories.

Meanwhile, the confusion matrix table can be seen based on Figure 7. The results indicate that the model consistently distinguishes between hoax and non-hoax classes. The high precision, recall, and F1-score values demonstrate the model's ability to make accurate predictions with minimal errors while effectively recognizing samples from both classes. The overall accuracy further confirms the model's strong generalization capability on the testing data. Additionally, the macro and weighted averages show that the model maintains stable performance despite class sample size differences, ensuring balanced classification outcomes.

Table 6. Model Evaluation

Class	*Precision*	*Recall*	*F1-Score*
1	0.96	0.93	0.95
0	0.94	0.96	0.95
Accuracy	-	-	0.95
Macro Avg	0.95	0.95	0.95
Weighted Avg	0.95	0.95	0.95

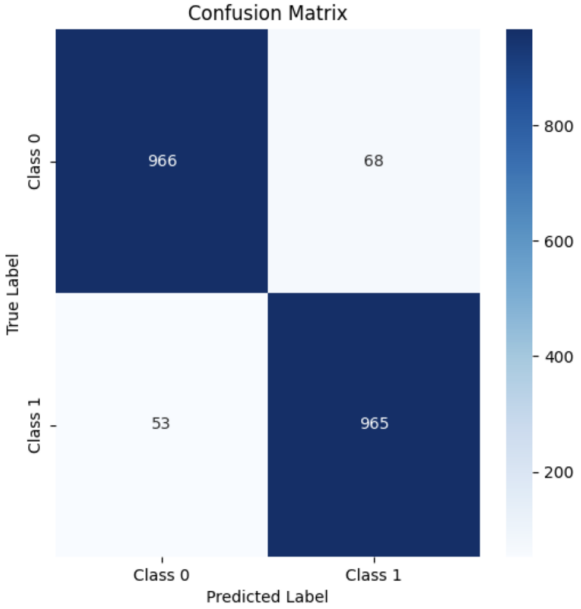


Figure 7. Confusion Matrix

3.2 Discussions

The results obtained from this study demonstrate the effectiveness of the LSTM model in detecting fake news on social media (X/Twitter), particularly in the context of Pilkada 2024. The model achieved a high accuracy of 94.90%, with precision, recall, and F1-score values reaching 0.95, indicating strong classification performance across both hoax and non-hoax categories. One of the key factors contributing to the model's performance is the use of GloVe word embeddings, which allowed the model to capture semantic relationships between words. This helped improve the understanding of contextual meaning, making the model more reliable in identifying misleading information. Additionally, the class weighting technique was applied to address class imbalance, ensuring that both categories were learned effectively and preventing the model from favoring one class over the other.

To further enhance performance and prevent overfitting, Dropout regularization was incorporated, allowing the model to generalize better on unseen data. The use of early stopping and adaptive learning rate adjustments also contributed to a more stable training process, ensuring that the model did not continue training unnecessarily once validation performance plateaued.

Despite these strengths, some challenges remain. Misinformation on social media is constantly evolving, and the language used in fake news can change over time. A static model may struggle to adapt to these

changes, requiring periodic retraining with updated datasets. Additionally, while Kompas.com was used as a trusted validator, expanding the dataset with multiple verified sources could further enhance the model's reliability.

Overall, this study highlights that LSTM combined with NLP techniques and verified datasets provides a strong foundation for automated fake news detection. However, continuous improvements, dataset expansion, and regular model updates will be necessary to maintain its effectiveness in real-world applications.

4. Conclusions

This study implements a fake news model tailored for political misinformation on Twitter (X) in relation to Indonesia's 2024 Pilkada. The model utilizes GloVe word embeddings and LSTM architecture, achieving strong recall and precision benchmarks, corroborated with manually labeled data from Kompas.com. The dataset comes form local media ground-truthing provides context, enhancing trustworthiness. The model demonstrated high classification performance, achieving strong metrics across accuracy, precision, recall, and F1-score, thereby effectively distinguishing between hoax and non-hoax news. The integration of GloVe word embeddings was crucial, significantly enhancing the model's ability to capture nuanced semantic relationships and improve contextual understanding of the text. Furthermore, the strategic application of class weighting and Dropout regularization effectively addressed challenges posed by class imbalance and mitigated overfitting, ensuring the model's robust generalization capabilities on unseen data. However, some limitations remain.

The model could struggle with misinformation from different political contexts or novel linguistic framings. Real-time deployment and scalability remain primary unsolved problems. Broadening the dataset with vetted news outlets, exploring hybrid or transformer architectures, and investigating continual learning methods could effectively enhance fake news detection systems.

References

[1] N. I. Saputri, Y. Sibaroni, and S. S. Prasetiyowati, "Covid-19 Fake News Detection on Twitter Based on Author Credibility Using Information Gain and KNN Methods," *Jurnal RESTI*, vol. 7, no. 1, pp. 185–192, 2023, doi: 10.29207/resti.v7i1.4871.

[2] M. Nazari, L. D. Jiménez-Franco, M. Schroeder, A. Kluge, M. Bronzel, and S. Kimiaei, "Automated and robust organ segmentation for 3D-based internal dose calculation," *EJNMMI Res*, vol. 11, no. 1, pp. 1–13, 2021, doi: 10.1186/s13550-021-00796-5.

[3] A. S. Zamani, A. H. A. Hashim, S. S. I. Mohamed, and N. Alam, "Optimized Deep Learning Techniques to Identify Rumors and Fake News in Online Social Networks," *Journal of Computational and Cognitive Engineering*, vol. 4, no. 2, pp. 142–150, 2025, doi: 10.47852/bonviewJCCE52023348.

[4] C. Jocelynne, L. Tobing, I. Lanang Wijayakusuma, L. Putu, and I. Harini, "Detection of Political Hoax News Using Fine-Tuning IndoBERT," *Journal of Applied Informatics and Computing (JAIC)*, vol. 9, no. 2, pp. 354–360, 2025, [Online]. Available: http://jurnal.polibatam.ac.id/index.php/JAIC

[5] Crisanadenta Wintang Kencana, Erwin Budi Setiawan, and Isman Kurniawan, "Hoax Detection System on Twitter using Feed-Forward and Back-Propagation Neural Networks Classification Method," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 4, no. 4, pp. 655–663, 2020, doi: 10.29207/resti.v4i4.2038.

[6] W. S. Sayed, A. G. Radwan, H. A. H. Fahmy, and A. L. Elsedeek, "Trajectory control and image encryption using affine transformation of lorenz system," *Egyptian Informatics Journal*, vol. 22, no. 2, pp. 155–166, 2021, doi: 10.1016/j.eij.2020.07.002.

[7] L. Chiruzzo, S. María Jiménez-Zafra, and F. Rangel, "Overview of IberLEF 2024: Natural Language Processing Challenges for Spanish and other Iberian Languages," in *CEUR Workshop Proceedings*, 2024, pp. 1–14. [Online]. Available: https://www.fing.edu.uy/index.php/es/node/40865

[8] T. Sowmya and E. A. Mary Anita, "A comprehensive review of AI based intrusion detection system," *Measurement: Sensors*, vol. 28, pp. 1–13, Aug. 2023, doi: 10.1016/j.measen.2023.100827.

[9] M. S. Khan, Z. Shah, M. Roman, W. Khan, N. Vrinceanu, and M. H. Alshehri, "Entropy generation in magneto couple stress bionanofluid flow containing gyrotactic microorganisms towards a stagnation point on a stretching/shrinking sheet," *Sci Rep*, vol. 13, no. 1, pp. 1–26, 2023, doi: 10.1038/s41598-023-48676-3.

[10] A. Nistor and E. Zadobrischi, "The Influence of Fake News on Social Media: Analysis and Verification of Web Content during the COVID-19 Pandemic by Advanced Machine Learning Methods and Natural Language Processing," *Sustainability (Switzerland)*, vol. 14, no. 17, pp. 1–24, Sep. 2022, doi: 10.3390/su141710466.

[11] X. Yang, Y. Wu, M. Yang, H. Chen, and X. Geng, "Exploring Diverse In-Context Configurations for Image Captioning," in *Conference on Neural Information Processing Systems*, Jan. 2024, pp. 1–20. doi: 10.48550/arXiv.2305.14800.

[12] M. K. Anam, S. Defit, Haviluddin, L. Efrizoni, and M. B. Firdaus, "Early Stopping on CNN-LSTM Development to Improve Classification Performance," *Journal of Applied Data Sciences*, vol. 5, no. 3, pp. 1175–1188, 2024, doi: 10.47738/jads.v5i3.312.

[13] U. Gawande, K. Hajari, and Y. Golhar, "SIRA: Scale illumination rotation affine invariant mask R-CNN for pedestrian detection," *Applied Intelligence*, vol. 52, no. 9, pp. 10398–10416, Jul. 2022, doi: 10.1007/s10489-021-03073-z.

[14] F. Sriyono and A. Nasiri, "Detecting Hate Speech In Twitter Using Long Short-Term Memory And Naïve Bayes Method," *Syntax Literate*, vol. 7, no. 2, pp. 685–695, 2022, doi: 10.36418/syntax-literate.v7i2.6313.

[15] D. Mouratidis, A. Kanavos, and K. Kermanidis, "From Misinformation to Insight: Machine Learning Strategies for Fake News Detection," *Information (Switzerland)*, vol. 16, no. 3, pp. 1–30, Mar. 2025, doi: 10.3390/info16030189.

[16] S. Yin *et al.*, "Automatic kidney segmentation in ultrasound images using subsequent boundary distance regression and pixelwise classification networks," *Med Image Anal*, vol. 60, pp. 1–14, Feb. 2020, doi: 10.1016/j.media.2019.101602.

[17] D. Y. Yefferson, V. Lawijaya, and A. S. Girsang, "Hybrid model: IndoBERT and long short-term memory for detecting Indonesian hoax news," *IAES International Journal of Artificial Intelligence*, vol. 13, no. 2, pp. 1911–1922, 2024, doi: 10.11591/ijai.v13.i2.pp1913-1924.

[18] P. A. Cicalese, A. Mobiny, Z. Shahmoradi, X. Yi, C. Mohan, and H. Van Nguyen, "Kidney Level Lupus Nephritis Classification Using Uncertainty Guided Bayesian Convolutional Neural Networks," *IEEE J Biomed Health Inform*, vol. 25, no. 2, pp. 315–324, 2021, doi: 10.1109/JBHI.2020.3039162.

[19] S. Muzamil, T. Hussain, A. Haider, U. Waraich, U. Ashiq, and E. Ayguadé, "An intelligent iris based chronic kidney identification system," *Symmetry (Basel)*, vol. 12, no. 12, pp. 1–14, 2020, doi: 10.3390/sym12122066.

[20] R. S. Putra, W. Agustin, M. K. Anam, L. Lusiana, and S. Yaakub, "The Application of Naïve Bayes Classifier Based Feature Selection on Analysis of Online Learning Sentiment in Online Media," *Jurnal Transformatika*, vol. 20, no. 1, pp. 44–56, Jul. 2022, doi: 10.26623/transformatika.v20i1.5144.

[21] P. P. Putra, M. K. Anam, S. Defit, and A. Yunianta, "Enhancing the Decision Tree Algorithm to Improve Performance Across Various Datasets," *INTENSIF: Jurnal Ilmiah Penelitian dan Penerapan Teknologi Sistem Informasi*, vol. 8, no. 2, pp. 200–212, Aug. 2024, doi: 10.29407/intensif.v8i2.22280.

[22] A. N. Ulfah, M. K. Anam, N. Y. S. Munti, S. Yaakub, and M. B. Firdaus, "Sentiment Analysis of the Convict Assimilation Program on Handling Covid-19," *JUITA: Jurnal Informatika*, vol. 10, no. 2, pp. 209–216, 2022, doi: 10.30595/juita.v10i2.12308.

[23] M. K. Anam, M. B. Firdaus, F. Suandi, Lathifah, T. Nasution, and S. Fadly, "Performance Improvement of Machine Learning Algorithm Using Ensemble Method on Text Mining," in *ICFTSS 2024 - International Conference on Future Technologies for Smart Society*, Kuala Lumpur: Institute of Electrical and Electronics Engineers Inc., Sep. 2024, pp. 90–95. doi: 10.1109/ICFTSS61109.2024.10691363.

[24] T. Dang, Y. Sakai, T. Tabaru, and A. Kasagi, "Regularizing Data for Improving Execution Time of NLP Model," in *The International FLAIRS Conference Proceedings*, 2022, pp. 1–4. doi: 10.32473/flairs.v35i.130672.

[25] M. K. Anam *et al.*, "Sara Detection on Social Media Using Deep Learning Algorithm Development," *Journal of Applied Engineering and Technological Science*, vol. 6, no. 1, pp. 225–237, Dec. 2024, doi: 10.37385/jaets.v6i1.5390.

[26] L. Xiaoyan, R. C. Raga, and S. Xuemei, "GloVe-CNN-BiLSTM Model for Sentiment Analysis on Text Reviews," *J Sens*, vol. 2022, pp. 1–12, 2022, doi: 10.1155/2022/7212366.

[27] F. Wundari, M. N. A. Syaiba Amien, and D. H. Irtsa, "Identifying Fake News Using Long-Short Term Memory Model," *Journal of Dinda : Data Science, Information Technology, and Data Analytics*, vol. 4, no. 1, pp. 28–34, 2024, doi: 10.20895/dinda.v4i1.1424.

[28] D. K. Talapula, A. Kumar, K. K. Ravulakollu, and M. Kumar, "A hybrid deep learning classifier and Optimized Key Windowing approach for drift detection and adaption," *Decision Analytics Journal*, vol. 6, 2023, doi: 10.1016/j.dajour.2023.100178.

[29] F. E. Laghrissi, S. Douzi, K. Douzi, and B. Hssina, "Intrusion detection systems using long short-term memory (LSTM)," *J Big Data*, vol. 8, no. 1, 2021, doi: 10.1186/s40537-021-00448-4.

[30] Y. Zhang, X. Hao, and Y. Liu, "Simplifying Long Short-Term Memory for Fast Training and Time Series Prediction," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Jun. 2019. doi: 10.1088/1742-6596/1213/4/042039.

[31] M. Cheon, D. Hee Lee, H. Seon Joo, and O. Lee, "Deep Learning Based Hybrid Approach of Detecting Fraudulent Transactions," *J Theor Appl Inf Technol*, vol. 31, no. 16, pp. 4044–4054, 2021, [Online]. Available: https://www.kaggle.com/mlg-

[32] A. P. Savant, "an Analysis of Deep Learning and Attention Models for Natural Language Processing Tasks," *International Research Journal of Computer Science*, vol. 7, no. 9, pp. 227–232, 2020, doi: 10.26562/irjcs.2020.v0709.001.

[33] M. K. Anam, T. P. Lestari, H. Yenni, T. Nasution, and M. B. Firdaus, "Enhancement of Machine Learning Algorithm in Fine-grained Sentiment Analysis Using the Ensemble," *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, vol. 19, no. 2, pp. 159–167, Mar. 2025, doi: 10.37936/ecti-cit.2025192.257815.

[34] F. Suandi *et al.*, "Enhancing Sentiment Analysis Performance Using SMOTE and Majority Voting in Machine Learning Algorithms," in *International Conference on Applied Engineering*, Atlantis Press, 2024, pp. 126–138. doi: 10.2991/978-94-6463-620-8_10.

[35] M. K. Anam *et al.*, "Enhancing the Performance of Machine Learning Algorithm for Intent Sentiment Analysis on Village Fund Topic," *Journal of Applied Data Sciences*, vol. 6, no. 2, pp. 1102–1115, 2025, doi: 10.47738/jads.v6i2.637.

[36] A. C. M. V. Srinivas, Ch. Satyanarayana, Ch. Divakar, and K. P. Sirisha, "Sentiment Analysis using Neural Network and LSTM," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, Feb. 2021, pp. 1–7. doi: 10.1088/1757-899x/1074/1/012007.

[37] M. K. Anam *et al.*, "Improved Performance of Hybrid GRU-BiLSTM for Detection Emotion on Twitter Dataset," *Journal of Applied Data Sciences*, vol. 6, no. 1, pp. 354–365, Jan. 2025, doi: 10.47738/jads.v6i1.459.