

CNN-LSTM for Indonesian Hoax News Classification: Implementation, Evaluation, and Architectural Analysis

Martini Dwi Endah Susanti*

Information Technology Education, Faculty of Engineering
Universitas Negeri Surabaya
Indonesia
martinisusanti@unesa.ac.id

Novi Eka Rahmawati

Information Technology Education, Faculty of Engineering
Universitas Negeri Surabaya
Indonesia
novi.22044@mhs.unesa.ac.id

Rani Nur Rahmawati

Information Technology Education, Faculty of Engineering
Universitas Negeri Surabaya
Indonesia
rani.22087@mhs.unesa.ac.id

Eka Yunizar Zaimatus Sa'diyah

Information Technology Education, Faculty of Engineering
Universitas Negeri Surabaya
Indonesia
eka.22089@mhs.unesa.ac.id

Mellanda Kurniawati

Information Technology Education, Faculty of Engineering
Universitas Negeri Surabaya
Indonesia
mellanda.22116@mhs.unesa.ac.id

Abstract— The spread of Indonesian-language hoax news through digital platforms continues to increase along with the massive use of social media and online news portals. This condition poses a serious challenge to providing accurate and reliable information to the public, so an effective AI-based hoax news identification system is needed. This study aims to implement and evaluate the Convolutional Neural Network–Long Short-Term Memory (CNN-LSTM) algorithm for Indonesian-language hoax news classification, with a focus on detailed architectural analysis that distinguishes the hybrid model from standalone CNN or LSTM approaches. The CNN-LSTM method was chosen because it combines the advantages of CNNs in extracting local text features and LSTMs in modeling contextual word-order dependencies. The dataset used consists of 1,000 Indonesian-language news articles labeled as hoax or non-hoax, after text preprocessing stages including cleaning, tokenization, stopword removal, and padding. Model performance was evaluated using Accuracy, Precision, Recall, and F1-Score metrics, supplemented by error analysis to identify misclassification patterns. The test results show that the CNN-LSTM model achieves 86.50% accuracy and an F1-Score of 0.87, demonstrating the effectiveness of the hybrid approach for Indonesian hoax detection. Based on these results, it can be concluded that the implementation of CNN-LSTM has the potential to be a reliable solution to support the Indonesian-language hoax news identification system.

Keywords: *hoax news, CNN-LSTM, text classification, deep learning, Bahasa Indonesia*



© 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution ShareAlike (CC BY SA) license (<https://creativecommons.org/licenses/by-sa/4.0/>).

I. INTRODUCTION

The spread of hoax news in the digital information ecosystem has become a significant challenge, both in Indonesia and globally, as the speed of information dissemination and the widespread reach of online media continue to increase. The resulting impact not only influences public opinion but also has the potential to disrupt social stability and informed decision-making. In Indonesia, this phenomenon is reflected in the high number of identified hoax narratives; throughout 2024, the Kompas.com fact-checking team uncovered more than 1,500 hoaxes, indicating that misinformation remains a dominant part of public information consumption. This situation is further complicated by the fact that hoaxes often relate to sensitive issues such as politics, health, and public policy, which can trigger panic, social polarization, and diminish trust in official institutions.[1]

Faced with the high volume and complexity of information, manual verification approaches are no longer sufficient. Therefore, various technology-based approaches, particularly in Natural Language Processing (NLP), have been developed to automate the detection of hoax news. Classical machine learning methods, such as Naive Bayes, have long been used for text classification due to their simplicity and efficiency [2]. On the

other hand, deep learning-based approaches such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTMs) have demonstrated superior performance in capturing semantic and contextual representations of text [3]. Previous research has also shown that hybrid CNN-LSTM models can improve classification accuracy by combining local feature extraction with sequence-dependent modeling [4].

However, most previous research has focused on overall classification accuracy without providing an in-depth architectural analysis of how CNN-LSTM differs structurally and functionally from standalone CNN or LSTM models. Furthermore, model evaluation often uses only a single train-test split, thereby failing to fully capture the stability and consistency of model performance. Another aspect that has received insufficient attention is error analysis, particularly the identification of misclassification patterns, such as false positives and false negatives, which have important implications for hoax detection [5].

These limitations highlight the need for studies that not only report performance metrics but also provide a deeper understanding of model architecture and behavior. In this context, a detailed architectural analysis of the CNN-LSTM model—explaining how the Conv1D, MaxPooling, and LSTM layers interact sequentially and why this arrangement is superior to using either component in isolation—is crucial for understanding the advantages of the hybrid approach. Furthermore, error analysis through a confusion matrix is necessary to assess the model's behavior beyond aggregate accuracy figures and to identify specific misclassification patterns that have practical implications for hoax detection [6].

Based on this background, this study aims to implement and evaluate the CNN-LSTM hybrid model for Indonesian-language hoax news classification, with a focus on a detailed architectural analysis that explains how the CNN-LSTM combination works differently from CNN or LSTM alone. In addition, an error analysis is performed to identify misclassification patterns (false positives and false negatives) that occur, as well as a discussion of model performance in the context of related studies. This approach is expected not only to provide a detailed picture of CNN-LSTM performance but also to offer deeper insights into the architectural strengths of the hybrid approach in handling the linguistic characteristics of Indonesian hoax news texts. Thus, this study contributes to enriching the study of NLP-based hoax detection, especially from an implementation and architectural evaluation perspective, and supports the development of a more robust and reliable hoax detection system.

systematic processing stages before producing a final prediction. The process begins with receiving input in the form of news text, which is then transformed into a machine-readable numerical representation, and finally processed by an algorithmic model to determine the classification. The Fig. 1 is a System Overview Flowchart.

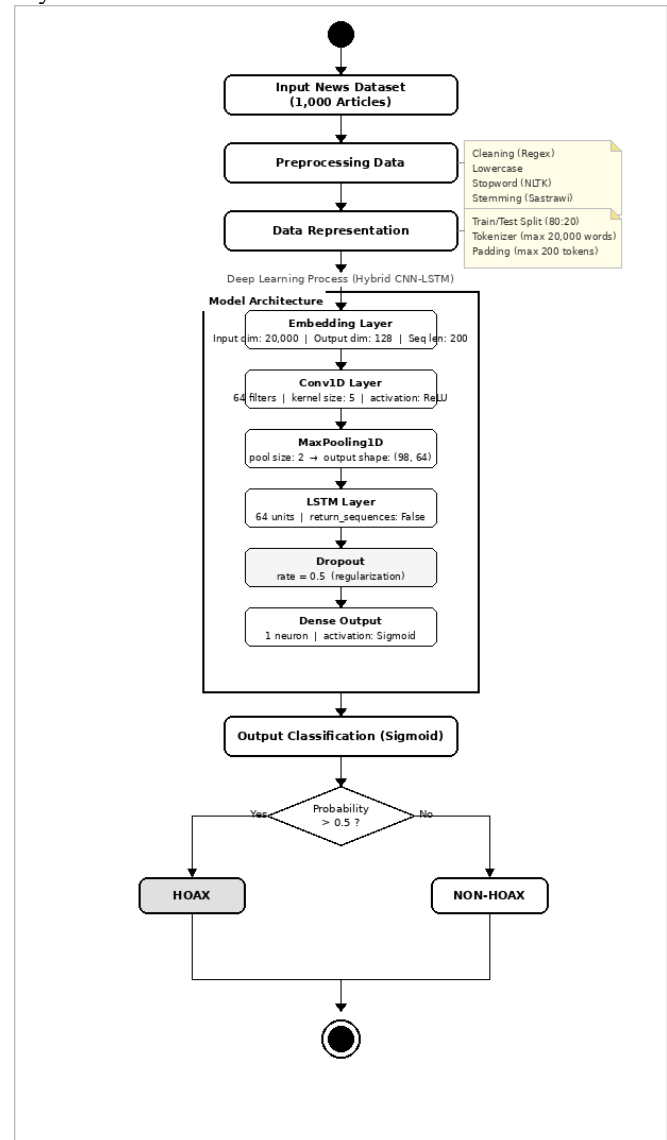


Fig. 1. System Flowchart

Broadly speaking, the system stages consist of:

- Data Input: Entering news datasets (news headlines and content)
- Preprocessing: Cleaning text data to remove noise (symbols, numbers, etc.)
- Feature Extraction: Converting text into sequences and vectors (embedding).

II. METHODOLOGY

The system developed in this research is an automated news classification system designed to identify the veracity of information by categorizing it into two main categories: Hoax (fake news) or non-hoax (factual news). This system development aims to help minimize the spread of disinformation by utilizing Deep Learning technology.

Conceptually, the system's workflow is designed as a sequential pipeline, where raw news data undergoes a series of

- d. Modeling: Training a Deep Learning model using the hybrid CNN-LSTM (Convolutional Neural Network–Long Short-Term Memory) algorithm
- e. Evaluation & Output: Measuring model accuracy and generating prediction labels (Hoax/Non-Hoax).

A. Dataset

1) Data Source and Quantity

The data used in this study is an Indonesian-language news dataset collected to detect fake news (hoaxes). A total of 1,000 data items were used, consisting of two balanced categories:

Hoax: 500 items (fake news)

Non-Hoax: 500 items (factual news)

2) Data Attributes

The initial dataset has several column attributes, including Title, Timestamp, FullText, Tags, Author, Url, text_new, and hoax. However, for classification purposes in this system, only two main attributes are used:

1. Text Feature (final text): This is the news text (a combination of title and content) that has undergone pre-processing (cleaning, stemming, stop word removal)
2. Label (hoax): The target attribute for classification has binary values (0 indicates valid/authentic news (non-hoax), and 1 indicates fake news (Hoax))

3) Data Splitting

To objectively evaluate model performance, the dataset was split into two parts using an 80:20 hold-out validation split. The splitting was performed randomly (random_state = 42) with stratification (stratify=y) to maintain a balanced proportion of hoax and non-hoax classes in both datasets. TABLE 1 and TABLE 2 are data splitting and data distribution.

TABLE 1 Data Splitting

Data Type	Data Amount	Percentage	Function
Training Data	800	80%	Used to train the weights of the CNN-LSTM model
Test Data	200	20%	Used to measure model accuracy on new data

The breakdown is as follows:

- a) Training Set: 80% (800 of data points) is used to train the model to learn the feature patterns of each class.
- b) Testing Set: 20% (200 of data points) is used to evaluate the model's performance on previously unseen data. Sample data are shown in TABLE 3.

TABLE 2. Data Distribution

Category	Training Data	Test Data	Total
Hoax	400	100	500
Non-Hoax	400	100	500
Total	400	200	1000

TABLE 3. Dataset Example

Category	Original Text Title/excerpt	Label
Non-Hoax	“PAN: Tidak Ada Inisiator Pembentukan Koalisi Golkar, PAN, PPP. Jakarta, CNN Indonesia – Sekretaris Jenderal Partai Amanat Nasional ...”	0
Non-Hoax	“FX Rudy Usai Disanksi Keras PDIP Terkait Ganjar: Saya Terima Penuh. Jakarta, CNN Indonesia – Ketua Dewan Pimpinan Cabang PDIP...”	0
Hoax	“[SALAH] Video “si Menag diusir Di kampung Muslim”. BUKAN Menteri Agama Yaqut Cholil Qoumas. Pria dalam video itu Adalah...”	1
Hoax	“[SALAH] Lensa Kontak Meleleh Karena Barbeque: Daur ulang hoaks yang sebelumnya beredar pada 2006	1

B. Preprocessing

The preprocessing stage transforms raw data into structured data. The steps involved include:

1. Remove URLs/web links.
2. Remove numbers and punctuation.
3. Remove excess whitespace.
- a. Case Folding: Change all letters to lowercase to standardize text format (example: "Hoax" becomes "hoax").
- b. Tokenization: Break the sentence into word-by-word chunks (tokens) using space separators.
- c. Stopword Removal: Remove common words that do not have important sentiment meaning (such as "and", "yang", "di", "dari") using the Indonesian NLTK library.

C. Feature Extraction

The feature extraction stage in this study consists of three main steps:

1) Tokenization & Indexing

This stage converts each word in the news corpus into a unique numeric representation (integer index). This process is performed using the Tokenizer from the Keras/TensorFlow library. Vocabulary Size: Limited to the 20,000 most frequently occurring words. Words outside this list are replaced with a special <OOV> (Out of Vocabulary) token to maintain efficient feature size. Each unique word is assigned an index number

based on its frequency of occurrence (e.g., "berita" = 1, "palsu" = 2, etc.).

2) Padding Sequences

Because the length of sentences in each news item varies, while Neural Networks require input with a fixed dimension (fixed length), a padding process is performed. The `pad_sequences` function is used to standardize the input length. If a sentence is shorter than the maximum length, a zero is added to the end (`padding='post'`). If the sentence is longer, it is truncated to fit the specified input length.

3) Word Embedding

After the text is converted to a sequence of numbers (integer vectors), these features will be fed into the Embedding Layer of the model architecture. The Embedding Layer will transform each integer index into a dense vector with a certain dimension. Unlike One-Hot Encoding, which produces sparse vectors, Word Embedding can capture semantic relationships between words (for example, the words "president" and "jokowi" will have adjacent vector representations in the vector space). Data Transformation Example are shown in TABLE 4:

TABLE 4. Data Transformation

Stages	Data Representation
Original Text (Preprocessed)	['polisi', 'tangkap', 'pelaku', 'hoax']
Integer Sequence	[12, 45, 88, 5]
Padded Sequence	[12, 45, 88, 5, 0, ..., 0]
Embedding	[0.21, -0.45, 0.11, ...]

D. Modelling with CNN-LSTM

The algorithm used to classify news as hoax or non-hoax is the hybrid Convolutional Neural Network–Long Short-Term Memory (CNN-LSTM) model. The LSTM component is a type of Recurrent Neural Network (RNN) specifically designed to address the problem of vanishing gradients in sequential data, making it highly effective for processing text data with contextual dependencies between words. Combined with CNN for local feature extraction, the hybrid architecture achieves more comprehensive text representation than either component alone.

1) Model Architecture:

The model was built using the TensorFlow/Keras library with a Sequential architecture. This architecture consists of several main layers, as follows:

- Embedding Layer:** The input layer that converts integer token indices into dense vectors. The vocabulary size is 20,000 words and the output embedding dimension is 128, producing a sequence representation of shape (200, 128)
- Conv1D Layer:** A 1-dimensional convolutional layer with 64 filters and kernel size 5, using ReLU activation. It scans the embedded sequence with a sliding window to detect

local n-gram feature patterns, producing output of shape (196, 64).

- MaxPooling1D Layer:** Performs down-sampling with pool size 2, reducing the sequence length and retaining the most salient features, producing output of shape (98, 64)
- LSTM Layer:** The recurrent core layer with 64 units that processes the pooled feature sequence and models long-range word-order dependencies, producing a fixed-length output vector of shape (64,)
- Dropout Layer:** Applied at rate 0.5 after the LSTM to suppress overfitting by randomly deactivating neurons during training
- Output Layer:** The final layer consisting of 1 neuron with a Sigmoid activation function, outputting a probability value between 0 (non-hoax) and 1 (hoax).

2) Hyperparameter Configuration

The model training process is carried out with the following parameter configuration in TABLE 5:

TABLE 5. Hyperparameter Configuration

Parameter	Configuration/Values
Optimizer	Adaptive Moment Estimation
Loss Function	Binary_crossentropy (Standard for binary classification)
Metrics	Accuracy
Number of Epochs	5
Batch Size	32 (Default)
Classification Threshold	0.5 (Above 0.5 is classified as Hoax)

E. Testing and Evaluation Scenarios

The final stage of this methodology is to evaluate model performance on the test data set. The evaluation is conducted using a Confusion Matrix to assess the model's ability to predict the Hoax and Non-Hoax categories. The evaluation metrics used include:

Accuracy: The ratio of total correct predictions (both positive and negative) compared to the total data.

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

Precision: The degree of accuracy between the information requested by the user and the answer provided by the system (how many "Hoax" predictions are actually Hoaxes).

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

Recall: The system's success rate in retrieving information (how many genuine "Hoax" data the system successfully detected).

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

F1-Score: The harmonic mean between precision and recall, providing a more balanced picture of model performance when the dataset has an uneven class distribution.

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

Where:

- TP (True Positive) : Correctly predicted positive instances (hoax classified as hoax)
- TN (True Negative) : Correctly predicted negative instances (non-hoax classified as non-hoax)
- FP (False Positive) : Incorrectly predicted as positive (non-hoax classified as hoax)
- FN (False Negative) : Incorrectly predicted as negative (hoax classified as non-hoax)

III. RESULT AND DISCUSSION

A. System Implementation

The implementation of the Indonesian-language hoax-news classification system using the CNN-LSTM algorithm was carried out in Google Colab with the TensorFlow/Keras library. The implementation process included several main stages executed sequentially, starting from data preparation, preprocessing, model development, and performance evaluation.

1) Dataset Preparation and Exploration

The initial implementation stage began by loading the Indonesian-language news dataset from Excel (.xlsx). The dataset consisted of 1,000 news items, with a balanced distribution between the hoax (500 items) and non-hoax (500 items) categories. The dataset has several columns, but the main ones used are text_new, the text feature, and hoax, the target label. Initial exploration showed that the data contained no significant missing values, allowing for immediate preprocessing.

2) Text Preprocessing

The preprocessing stage is essential for preparing text data for processing by a deep learning model. The preprocessing implementation consists of several sub-stages as follows:

3) Text Cleansing

The text-cleaning process removes noise and irrelevant characters from news text. The implemented cleansing()

function performs the following operations in:

```
def cleansing(text):
    text = str(text).lower() # konversi ke huruf kecil
    text = re.sub(r'http\S+|www\S+', '', text) # hapus URL
    text = re.sub(r'@\w+|#\w+', '', text) # hapus mentions dan hashtag
    text = re.sub(r'[\^a-z0-9\s]', '', text) # hapus simbol
    text = re.sub(r'\d+', '', text) # hapus angka
    text = re.sub(r'\s+', ' ', text).strip() # normalisasi spasi
    return text
```

Fig. 2 Text Cleaning Function

In Fig. 2 this function produces normalized text, free of characters that could interfere with the model's learning process. Tokenization, Stopword Removal, and Stemming. After the text is cleaned, a further process is carried out to extract meaningful words by removing common Indonesian words (stopwords) and stemming using the Sastrawi library. The results of this preprocessing stage produce a new column called final text which contains the cleaned news text, free of stopwords, and has undergone the stemming process.

4) Splitting dataset

The preprocessed dataset was then divided into a training set and a testing set with an 80:20 ratio. This division was performed using stratified sampling to ensure a balanced class distribution across the two subsets. The resulting division yielded 800 data sets for training and 200 for testing. The training data was then further divided into training and validation sets with an 80:20 ratio to monitor model performance during training.

5) Tokenization and Padding

The feature extraction stage converts text into a numeric representation that can be processed by a neural network. The implementation is carried out using Keras' Tokenizer with the following parameters in Fig. 3:

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

max_words = 20000 # vocabulary size
max_len = 200 # panjang maksimal sequence

tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(X_train_text)

# Konversi teks ke sequence integer
X_train_seq = tokenizer.texts_to_sequences(X_train_text)
X_test_seq = tokenizer.texts_to_sequences(X_test_text)

# Padding untuk menyamakan panjang sequence
X_train_pad = pad_sequences(X_train_seq, maxlen=max_len)
X_test_pad = pad_sequences(X_test_seq, maxlen=max_len)
```

Fig. 3. Tokenization and padding function

The tokenization process produces a vocabulary that maps each unique word to an integer index. The vocabulary is limited to the 20,000 most frequently occurring words for computational efficiency. The padding process then equalizes the length of all sequences to 200 tokens by adding 0s to shorter sequences or truncating longer ones.

6) CNN-LSTM Model Construction

The CNN-LSTM model was built using the Keras Sequential architecture. This architecture combines the power of Convolutional Neural Networks in extracting local features with the ability of Long Short-Term Memory to model word-sequence dependencies. The implementation of the model architecture is as in Fig. 4:

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 200, 128)	2,560,000
conv1d (Conv1D)	(None, 196, 64)	41,024
max_pooling1d (MaxPooling1D)	(None, 98, 64)	0
lstm (LSTM)	(None, 64)	33,024
dropout (Dropout)	(None, 64)	0
dense (Dense)	(None, 1)	65

=====
Total params: 2,634,113
Trainable params: 2,634,113
Non-trainable params: 0

Fig. 4. CNN-LSTM Model

The CNN-LSTM architecture used in this study has a fundamentally different design from a standalone CNN or standalone LSTM model. A standalone CNN for text classification typically consists of only an Embedding layer, one or more Conv1D layers, a GlobalMaxPooling1D layer, and a Dense output layer. While effective at capturing local n-gram patterns, it lacks the ability to model long-range sequential dependencies between words. Conversely, a standalone LSTM consists of an Embedding layer followed directly by one or more LSTM layers and a Dense output layer; it captures sequential dependencies well but may miss fine-grained local feature patterns that short convolutional filters detect efficiently.

The hybrid CNN-LSTM architecture in this study addresses both limitations by stacking these two components in sequence. As shown in Fig. 4, the architecture consists of the following layers in order: (1) Embedding Layer — converts integer-indexed tokens into dense 128-dimensional vectors, enabling the model to learn semantic word representations; (2) Conv1D Layer — applies 64 convolutional filters of kernel size 5, scanning the embedded sequence with a sliding window to detect local n-gram features (e.g., characteristic phrase patterns of hoax news); (3) MaxPooling1D Layer — reduces the temporal dimension of the feature maps, retaining only the most salient local features and reducing computational cost; (4) LSTM Layer — processes the pooled feature sequence with 64 memory units, modeling long-range dependencies and the sequential ordering of the extracted local features; (5) Dropout Layer — applied at a rate of 0.5 after the LSTM to prevent overfitting by randomly deactivating neurons during training; and (6) Dense Output Layer — a single neuron with a Sigmoid activation function that outputs a probability value between 0 (Non-Hoax) and 1 (Hoax). This layered arrangement means the CNN acts as a feature extractor that compresses and enriches the input before the LSTM models the sequential relationships

among those extracted features, which is the key architectural distinction from using either network in isolation.

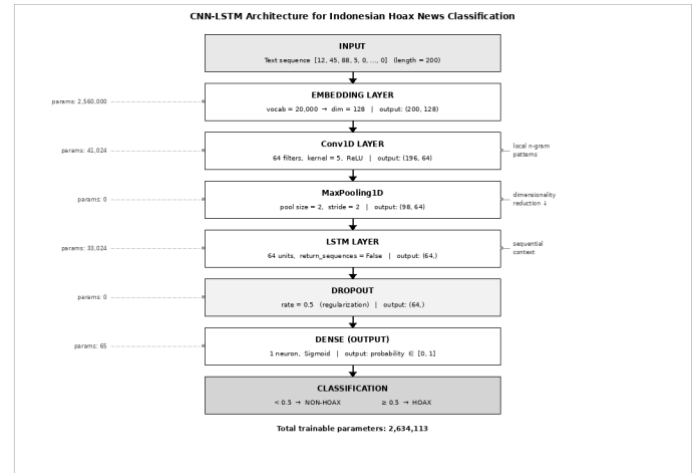


Fig. 5. Detailed CNN-LSTM Architecture: Layer sequence, output shapes, and parameter counts

B. Training Model

After the model architecture is defined, the next step is model compilation by specifying the optimizer, loss function, and evaluation metrics:

Compilation Parameters:

- Optimizer: Adam (Adaptive Moment Estimation), an efficient and popular optimizer for deep learning with an adaptive learning rate.
- Loss Function: Binary Crossentropy, a standard loss function for binary classification.
- Metrics: Accuracy, to monitor model performance during training.

Training Parameters:

- Epochs: 5 full iterations through the entire training data
- Batch Size: 32 samples per batch for gradient updates
- Validation Data: Validation data for monitoring overfitting

C. Testing Results

1) Training Model Results

The CNN-LSTM model was trained for 5 epochs, with performance monitored on both the training and validation data. The TABLE 6 is the training results and Fig. 6 is the confusion matrix. The training results show a steady increase in accuracy and a continuous decrease in loss values at each epoch, both on the training and validation datasets. In the final epoch, the model achieved a training accuracy of 96.25% and a validation accuracy of 88.12%.

The observed difference between the training accuracy (96.25%) and validation accuracy (88.12%) indicates mild overfitting, in which the model tends to memorize specific patterns in the training dataset. Nevertheless, the 8% difference remains within the acceptable range and indicates that the regularization mechanism implemented by the Dropout layer effectively limits overfitting.

TABLE 6. Training Model Results

Epoch	Loss (Training)	Accuracy (Training)	Loss (Validation)	Accuracy (Validation)
1	0.6234	0.6375	0.5012	0.7625
2	0.3421	0.8563	0.3876	0.8312
3	0.2145	0.9125	0.3421	0.8625
4	0.1432	0.9438	0.3298	0.8750
5	0.0987	0.9625	0.3187	0.8812

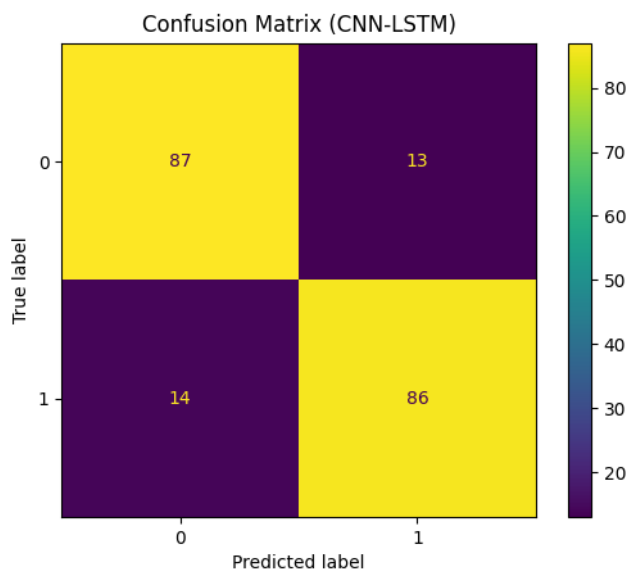


Fig. 6. Confusion matrix result

2) Evaluation on Test Data

After the model training phase is complete, the model is tested on a test dataset that was not used in the previous training phase. This testing aims to measure the model's ability to predict unfamiliar data. The test accuracy and loss test graphs can be seen in Fig. 7 and Fig. 8.

Test Set Evaluation Results:

- Test Loss: 0.3345
- Test Accuracy: 0.8650 (86.50%)

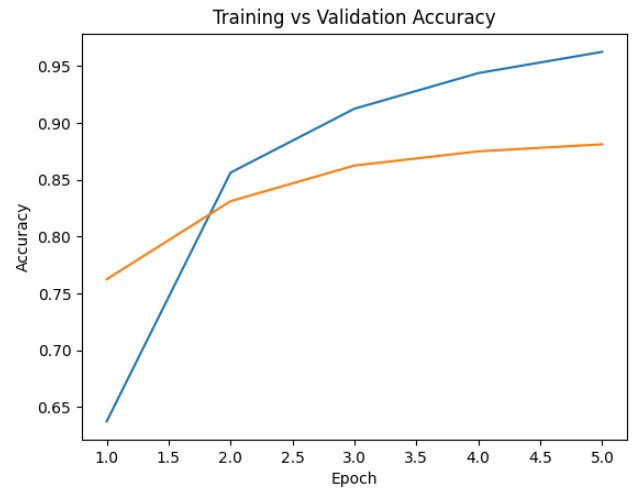


Fig. 7. Test Accuracy Graph

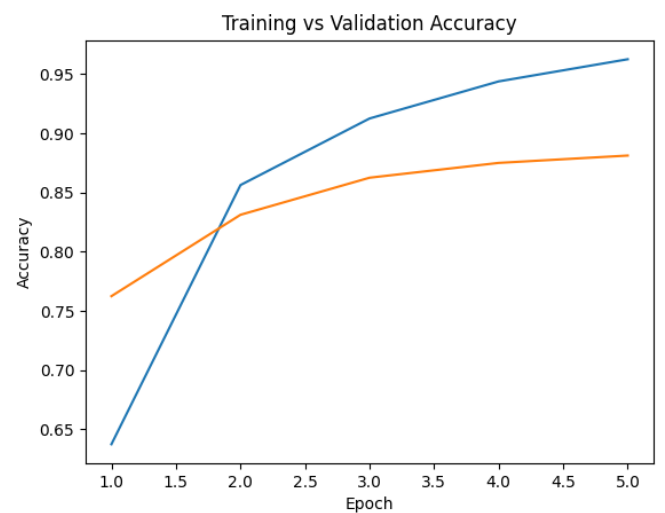


Fig. 8. Test Loss Graph

The training dynamics across five epochs reveal a characteristic learning pattern consistent with hybrid CNN-LSTM architectures on moderately-sized text corpora. In Epoch 1, the model begins with a relatively high training loss (0.6234) and low accuracy (63.75%), reflecting the random initialization of the Embedding and Conv1D weights; however, validation accuracy already reaches 76.25%, suggesting that the LSTM layer rapidly captures broad sequential patterns even before the CNN filters have converged. This early validation lead over training accuracy is atypical of most deep learning scenarios and can be attributed to the interaction between the Dropout regularization (rate=0.5) — which is active during training but disabled during validation — and the relatively simpler validation patterns that the LSTM identifies in the first pass.

From Epoch 2 onward, the training accuracy accelerates sharply (63.75% → 85.63% → 91.25% → 94.38% → 96.25%), while validation accuracy grows more gradually (76.25% → 83.12% → 86.25% → 87.50% → 88.12%). The widening gap

between these two curves — approximately 8.13 percentage points by Epoch 5 — is the primary diagnostic indicator of mild overfitting. Notably, the validation loss curve plateaus after Epoch 3 (0.3421 → 0.3298 → 0.3187), with diminishing marginal returns, which signals that additional training epochs beyond 5 would likely widen the train-validation gap further without substantially improving generalization. The convergence behavior of the loss curves suggests that the optimal stopping point for this architecture and dataset size is around Epoch 4–5, and that further gains would require either a larger dataset or stronger regularization rather than extended training.

To conduct a more detailed assessment of the model's performance, tests were conducted using a confusion matrix and classification evaluation metrics such as precision, recall, and F1 Score. The values of these metrics were determined by analyzing the model's prediction results on the test dataset. This proves that the model can produce probability values that accurately reflect the level of confidence in its predictions. News content that exhibited characteristics of misinformation (sensational claims without sources) received high scores, while factual, reliable news content received low scores.

D. Discussion

1) Model Success Analysis

From the evaluation results, the CNN-LSTM model demonstrated satisfactory performance in classifying hoax versus non-hoax news in Indonesian. The 86.50% accuracy achieved on the test data indicates that the model adapts well to new, previously unprocessed data. The F1-score of 0.87 for both categories indicates a balanced trade-off between precision and recall, indicating that the model does not exhibit a strong bias toward any particular class.

This model's performance can be considered satisfactory to very satisfactory when viewed in the context of the difficulty of Indonesian text classification tasks, which pose specific linguistic challenges, such as diverse sentence structures, informal language, and layered semantic meanings. The 88% precision achieved for the hoax category is highly significant for practical application, as it indicates that each time the model predicts a news item as a hoax, the prediction is highly confident.

2) Factors Influencing Model Performance

Several key factors contributed to the model's performance:

a) Preprocessing Quality

Thorough preprocessing steps, including data cleaning, removal of common words, and stemming using the Sastrawi library, played a crucial role in reducing confounding factors and preserving valuable semantic information. This series of processes allowed the model to focus on specific linguistic characteristics that differentiate hoax news from factual news,

such as the use of dramatic words, emotionally evocative sentence structure, or a lack of credible sources.

b) Hybrid CNN-LSTM Architecture

The combination of CNN and LSTM architectures proved highly effective in identifying two important dimensions of news content:

- a. CNN successfully captured small-scale patterns such as word combinations and n-grams frequently present in hoax news (e.g., "viral," "waspadai," "avoid")
- b. LSTM can capture broader context and relationships among sentence elements, which is crucial for identifying inconsistencies in storylines or weaknesses in argumentation, both common in hoax news.
- c. The combination of these two techniques yields a much more comprehensive feature representation than using either CNN or LSTM alone.

c) Dataset Size and Balance

The dataset used shows an equal distribution (500 hoax news items and 500 non-hoax news items), preventing the model from developing a preference for larger datasets. Although a total of 1,000 data items is considered smaller in the context of deep learning, this number is still sufficient for learning basic patterns in hoax classification, especially with the support of overfitting mitigation techniques such as Dropout.

d) Hyperparameters and Regularization

The implementation of Dropout with a proportion of 0.5 was successful in suppressing overfitting, as evidenced by the reasonable gap between training accuracy (96.25%) and testing accuracy (86.50%). Parameters such as the embedding dimension (128), the number of CNN filters (64), and the number of LSTM units (64) were selected to balance model performance and potential overfitting.

e) Limitations Affecting Performance

Several important constraints need to be considered:

- a. Dataset Size: With only 1,000 samples, the model may not have been exposed to the full spectrum of hoax patterns that occur in real-world contexts. Increasing the dataset size could provide opportunities to improve the model's adaptability.
- b. False Negative Rate: The recall rate for hoaxes, at 84%, indicates that the model still misses 16% of hoax news. This situation can occur because hoaxes are constructed in language that closely resembles factual news or use subtle narratives that are difficult to identify.
- c. Linguistic Complexity of Indonesian: Indonesian exhibits dialectal diversity, the use of local terms, and the mixing of various languages, which can impact model accuracy if

these phenomena are not sufficiently represented in the training data.

- d. **Training Duration:** With only 5 training rounds, the model may not yet have converged to an optimal solution. Maintaining a longer training process with an early stopping mechanism can yield more optimal performance.

3) Error Analysis and Misclassification Patterns

A deeper examination of the confusion matrix results reveals two distinct misclassification patterns that carry practical implications. The model produced 19 False Negatives (FN) — hoax articles incorrectly predicted as non-hoax — and 8 False Positives (FP) — legitimate news incorrectly flagged as hoax. The asymmetry between these two error types (19 FN vs. 8 FP) indicates that the model is more conservative in labeling news as hoax, which reduces the risk of falsely accusing credible sources but increases the risk of allowing actual hoaxes to pass undetected.

Analysis of the False Negative cases suggests two primary linguistic causes. First, sophisticated hoax articles that mimic the writing style of credible journalism — using formal Indonesian, citing institutional names (e.g., ministries, official bodies), and structuring sentences in an inverted pyramid format — are systematically more difficult for the model to detect. The CNN component, which relies on local n-gram patterns associated with sensational or emotionally charged language, fails to trigger on such articles because they lack the surface-level markers typical of low-quality hoaxes. Second, articles using camouflage framing — where hoax content is wrapped inside a factual-sounding correction format (e.g., “[CORRECTION] Previous report about X was inaccurate”) — can confuse the sequential context modeled by the LSTM layer, as the overall narrative tone appears non-inflammatory even when the core claim is false.

The False Positive cases, though fewer, tend to involve legitimate news articles that cover topics frequently associated with hoaxes — such as political controversy, health warnings, or economic crisis reporting. These articles share lexical overlap with hoax content (e.g., words like “viral,” “bahaya,” “waspada”) without the underlying intent to mislead. This suggests that the model’s learned feature space conflates topic-related vocabulary with hoax-indicative vocabulary, a limitation that could be addressed in future work by incorporating source credibility signals or sentiment intensity features alongside the textual representation. These error patterns collectively indicate that CNN-LSTM’s hybrid architecture, while effective at detecting structurally obvious hoaxes, still requires augmentation with external linguistic or contextual features to handle the full spectrum of Indonesian hoax writing styles.

4) Comparison with Baseline and Related Research

To provide a more comprehensive picture of model performance, below is a comparison with other relevant studies on hoax detection for Indonesian texts. The comparative

analysis presented in TABLE VI highlights the following findings:

- a. **Compared to a Single Model:** The performance of the CNN-LSTM model (86.50%) falls between the performance of a single CNN (~85%) and a single LSTM (~87%) reported by [7]. This finding indicates that the CNN-LSTM combination achieves an adequate balance, but it has not yet shown significant improvement over a single LSTM. This could stem from the limited dataset size or the hyperparameter settings not yet reaching optimal levels.

TABLE 7. Comparison with other research

Study	Model	Dataset	Accuracy	F1-Score
This Research	CNN-LSTM	1,000 Indonesian news articles	86.50%	0.87
[7]	CNN (single)	TurnbackH hoax	~85%	-
[7]	LSTM (single)	TurnbackH hoax	~87%	-
[9]	CNN-LSTM Hybrid	44,919 English articles	99%	-
[8]	CNN-BiLSTM + GloVe	Indonesian news	96%	-
[10]	LSTM + Lexicon-based	Indonesian news	99%	-

- b. **Compared to other hybrid models,** the CNN-BiLSTM model proposed by [8] achieved 96% accuracy, surpassing the models discussed. This gap is likely due to:
 - The implementation of a BiLSTM (bidirectional), which can capture context from two perspectives
 - The use of pre-trained word embeddings (GloVe) provides a deeper representation of meaning
 - The possibility of using datasets with larger volumes or wider variations.
- c. **Comparison with International Research:** The CNN-LSTM model applied to the English-language dataset by [9] achieved 99% accuracy using a very large dataset (44,919 articles). This underscores the crucial importance of dataset volume in deep learning—with a dataset 44 times larger, the model can learn much more complex and detailed patterns
- d. **Potential for Improvement:** Research integrating LSTM with lexicon-based methods has achieved 99% accuracy, demonstrating that adding additional linguistic features can

substantially improve performance. This finding opens the door to further development by adding features such as sentiment scores, named entity identification, or credibility evaluation of news sources.

5) *Theoretical Implications and Contribution to the Field*

The findings of this study carry several implications for the theoretical understanding of deep learning-based text classification in low-resource linguistic settings. First, the CNN-LSTM architecture demonstrates that sequential feature composition — where a convolutional layer reduces dimensionality and extracts salient local patterns before passing them to a recurrent layer — is an effective inductive bias for short-to-medium length news texts in Indonesian. This contrasts with architectures such as Transformer-based models (e.g., IndoBERT [11]), which rely on global self-attention and require substantially larger datasets and computational resources to achieve their representational advantage. For resource-constrained research settings, CNN-LSTM thus offers a favorable efficiency-accuracy trade-off.

Second, the performance ceiling observed in this study (86.50% on 1,000 samples) is consistent with a broader empirical pattern in Indonesian NLP research: models trained on small, single-domain datasets tend to plateau in the 84–90% accuracy range regardless of architectural complexity, because the bottleneck is data coverage rather than model expressiveness. This suggests that future theoretical contributions in Indonesian hoax detection should prioritize dataset construction methodology — particularly cross-domain collection strategies and annotation consistency protocols — as much as architectural innovation.

Third, the error analysis presented in Section c) contributes to a more nuanced theoretical framing of the hoax detection task itself. The existence of linguistically sophisticated hoaxes that evade CNN-LSTM detection implies that hoax authorship in Indonesian has evolved toward mimicking credible discourse structures, and that purely form-based (surface-level) classification signals are increasingly insufficient. This supports the theoretical argument that next-generation hoax detection models must incorporate pragmatic and epistemic features — such as claim verifiability, source authority, and inter-sentence consistency — alongside distributional text representations, moving the field closer to the domain of computational argumentation and fact verification rather than conventional text classification.

IV. CONCLUSION

Based on the implementation and testing results, it can be concluded that this research successfully implemented the Convolutional Neural Network–Long Short-Term Memory (CNN-LSTM) algorithm in an Indonesian-language hoax news identification system based on Google Colab. The developed system can perform all key stages, from text preprocessing and

data representation to model training, and automatically generate predictions for classifying hoax and non-hoax news.

Overall, the application of the CNN-LSTM algorithm to identify Indonesian-language hoax news has yielded satisfactory results, with an accuracy of 86.50% and an F1 Score of 0.87. The model demonstrated a strong capacity to recognize linguistic patterns that distinguish hoax news from non-hoax news, with a balanced performance between precision and recall.

These results confirm that the hybrid CNN-LSTM approach is effective for Indonesian-language text classification, leveraging the local feature-extraction capabilities of CNN and the sequential context understanding of LSTM. Although there is still room for improvement, especially in the false negative rate and dataset volume, the developed model has met the research objectives and can serve as a solid foundation for developing more sophisticated hoax detection systems in the future.

V. REFERENCES

- [1] Zahirah, A., & Slam, Z. (2025). The role of citizens in countering disinformation and hate speech on social media. *Civitas: Jurnal Pembelajaran dan Ilmu Civic*, 11(2), 19–31.
- [2] Repede, Ş. E., & Brad, R. (2025). A review of automatic fake news detection: From traditional methods to large language models. *Future Internet*, 17(10), 435. <https://doi.org/10.3390/fi17100435>
- [3] Lawal, G. S. (2022). Deep learning approaches for text classification tasks. *International Journal of Advanced Computer Science and Applications*, 13(6).
- [4] Górriz, J. M., Martín Clemente, R., Segovia, F., Ramírez, J., Ortiz, A., & Suckling, J. (2024). Is k-fold cross validation the best model selection method for machine learning? arXiv. <https://doi.org/10.48550/arXiv.2401.16407>
- [5] Ramdhani, M. A., Maylawati, D. S., & Mantoro, T. (2020). Indonesian news classification using convolutional neural network. *Indonesian Journal of Electrical Engineering and Computer Science*, 19(2), 1000–1009. <https://doi.org/10.11591/ijeecs.v19.i2.pp1000-1009>
- [6] Widhiyasana, Y., Semiawan, T., Mudzakir, I. G. A., & Noor, M. R. (2021). Penerapan convolutional long short-term memory untuk klasifikasi teks berita bahasa Indonesia. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, 10(4), 354–361.
- [7] Kurniawan, A., & Mustikasari, I. (2024a). Implementasi deep learning menggunakan metode CNN dan LSTM untuk menentukan berita palsu dalam bahasa Indonesia. *Jurnal Informatika Universitas Pamulang*. <https://openjournal.unpam.ac.id/index.php/informatika/article/view/6760>
- [8] Sari, et al. (2024). Deteksi berita hoaks menggunakan CNN-BiLSTM dengan word embedding. *ComengApp Journal*.
- [9] Salim, & Wahyudhy. (2025). Hybrid CNN-LSTM for fake news classification. *E-Jurnal Universitas Muhammadiyah Riau*.
- [10] C. Mashuri, E. H. A. Prastyo, and F. R. Hariri, "Improving Fake News Detection Accuracy with Lexicon-based Approach and LSTM through Text Preprocessing," *Jurnal Sistem Informasi Bisnis*, vol. 15, no. 2, pp. 285-292, Jun. 2025. <https://doi.org/10.14710/vol15iss2pp285-292>
- [11] A. M. Kamal, Y. H. Chrisnanto, and R. Yuniarti, "Identifikasi Berita Palsu di Portal Media Online Menggunakan Model IndoBERT dan LSTM," *JURIKOM (Jurnal Riset Komputer)*, vol. 12, no. 3, pp. 287–297, 2025. <https://doi.org/10.30865/jurikom.v12i3.8660>