Leveraging LSTM Neural Networks for Advanced Harassment Detection: Insights into Insults and Defamation in the U-Tapis Module

Gerald Imanuel Wijaya[a)] and Marlinda Vasty Overbeek[b)]

Department of Engineering and Informatics, Universitas Multimedia Nusantara, Tangerang, Indonesia

[a)]Corresponding author: geraldimanuelwijaya@gmail.com
[b)]marlinda.vasty@umn.ac.id

Abstract. The prevalence of online harassment necessitates sophisticated automated systems that can accurately classify offensive content. In this work, we present a text classification system based on Long Short-Term Memory (LSTM) networks to categorize text into Neutral, Insult, and Defamation classes, thereby providing a more granular understanding of abusive behavior in digital environments. The system was evaluated using two labeled datasets—150 samples generated by ChatGPT and 1000 samples from internet sources—achieving an accuracy of 85% on both. Notably, the model demonstrated strong performance in identifying Defamation, exhibiting high precision and recall. These findings underscore the effectiveness of LSTM networks in capturing complex linguistic features, highlighting their potential for improving content moderation tools and curbing online harassment.

INTRODUCTION

In the rapidly evolving digital era, cases of cyberspace harassment are becoming increasingly complex, which requires swift and accurate handling [1]. Traditionally, the analysis of such cases has involved linguistic experts tasked with evaluating language use, including slander and defamation, under the *Undang-Undang Informasi dan Transaksi Elektronik* (UU ITE). *Badan Reserse Kriminal* (Bareskrim) and the police frequently engage linguistic experts to analyze examination reports (BAP) to determine whether a case falls under the categories of slander, defamation, or other violations.

In a private interview with M. Niknik on September 18, 2024, it was stated that, in response to technological advancements, the police have developed a website allowing the public to file complaints, including harassment cases. While this platform has facilitated the reporting process, the current system still faces limitations in terms of speed and efficiency. A primary challenge lies in the three-hour time frame set to determine whether a report can proceed. This constraint often requires linguistic experts to be available at any time, even during unconventional hours, potentially affecting the accuracy of analysis due to fatigue or time limitations [2].

Recognizing the critical role of linguistic accuracy and efficient text analysis in addressing harassment cases, efforts have been made to enhance automated systems for language processing. As cases involving language use continue to rise and technology advances, Universitas Multimedia Nusantara has strived to remain responsive by developing a system for detecting Indonesian language errors. This system aims to support automation in journalism while offering potential applications in contexts such as harassment detection [3, 4, 5]. Existing language-screening systems include modules for detecting errors in word usage, compound words, and spelling [6, 7, 8].

In forensic linguistics, texts or messages can serve as concrete evidence [9, 10, 11]. However, the exploration and development of the U-Tapis system have yet to include a module capable of detecting harassment elements within a text. Developing such a module would not only strengthen U-Tapis as a pioneer of automation in journalism, but would also contribute significantly to the prevention and resolution of harassment cases in Indonesia.

Harassment can be broadly classified into categories such as slander, abuse, insults, and defamation. According to M. Niknik in September 23, 2024, private interview, among these categories, insults and defamation are among the most frequently encountered in Indonesia. This phenomenon underscores the importance of developing applications capable of detecting insults and defamation within text [2].

Detecting text containing harassment, particularly insults and defamation, requires a comprehensive approach. A significant challenge is the accurate classification of text into harassment subcategories, given the diversity of language used, including both formal and informal styles. Insults and defamation often overlap in linguistic characteristics, necessitating models that can identify specific patterns within textual data [12, 13].

In the realm of computer science, this research is highly significant as it integrates forensic linguistics with technology-based automation. The chosen approach addresses journalistic needs for detecting harassment and supports the development of algorithms capable of processing textual data effectively and efficiently. With the ever-

increasing volume of data, employing appropriate techniques in feature engineering and classification is crucial for creating an accurate and reliable system [14, 15].

This research employs Long Short-Term Memory (LSTM), a deep learning architecture renowned for its ability to process sequential data. LSTM has demonstrated exceptional performance in identifying patterns within unstructured text, such as hate speech and negative sentiments, making it highly suitable for this research problem [16, 17, 18]. Its strength lies in its capacity to model linguistic complexities, such as informal language nuances and overlapping harassment subcategories like insults and defamation, while maintaining high accuracy [19, 20].

The objective of this research is to evaluate the accuracy of the LSTM method integrated into the U-Tapis project module for detecting insults and defamation in harassment cases, with a focus on automating text classification to improve the efficiency of online reporting systems. This approach addresses the computational demands of processing unstructured text while maintaining robustness in identifying linguistic complexities. By integrating advanced Natural Language Processing (NLP) techniques to identify specific language patterns, this research not only meets practical needs such as accelerating online reporting systems but also contributes to the development of more adaptive large-scale text analysis technologies, essential in an era of exponential digital data growth [21, 22].

LITERATURE REVIEW

Harassment

Harassment refers to actions intended to demean, threaten, or embarrass individuals, either physically or verbally. In the digital context, harassment often occurs on social media, forums, and messaging applications. Common forms of this behavior include insults, threats, and the spread of defamatory statements, also known as cyberbullying. Online harassment can significantly impact mental health, increasing stress and anxiety levels [23]. Therefore, automated detection of harassment is crucial to address the rising incidents of online abuse [24].

Insults in Text Analysis

Insults are expressions intended to demean or harm the dignity of an individual. They often take the form of offensive language, derogatory remarks, or insinuations. The detection of insults in textual data poses significant challenges due to the informal, ambiguous, and context-dependent nature of language used on digital platforms [25, 26].

Early efforts in insult detection focused on rule-based approaches, leveraging predefined dictionaries of offensive words. While effective in identifying explicit insults, these methods struggled with implicit or sarcastic expressions [27]. For example, phrases disguised with humor or contextual ambiguities often go unnoticed by simplistic systems.

Defamation in Text Analysis

Defamation involves false or harmful statements aimed at damaging an individual's or an organization's reputation. It is commonly classified into libel (written or published text) and slander (spoken words) [28]. Unlike insults, which are often direct and personal, defamation can include subtle insinuations or claims that require contextual understanding. Automated detection of defamation has been successfully advanced by leveraging Natural Language Processing (NLP) techniques and machine learning models [29].

Natural Language Processing (NLP)

Natural Language Processing (NLP) is a branch of artificial intelligence focused on understanding and generating human language, enabling computers to interact with humans without relying on machine-specific languages [30].

NLP involves preprocessing steps such as tokenization, segmentation, and stemming, which significantly improve the quality of unstructured text data for information retrieval techniques [31].

Text Preprocessing

Text preprocessing is a crucial step to prepare text data for analysis by simplifying and standardizing its format. In this research, the preprocessing steps include:

1. **Slang Word Normalization**: Replacing informal or abbreviated words (e.g., "gak" to "tidak," "dgn" to "dengan") with their standard forms to enhance comprehension.
2. **Text Cleaning**: Converting all text to lowercase, removing punctuation, numbers, and extra spaces.
3. **Tokenization**: Splitting text into smaller units like words or phrases for analysis.

These steps are essential to ensure clean, semantically accurate input for subsequent Natural Language Processing (NLP) tasks. Proper preprocessing directly impacts the effectiveness of NLP applications by improving tokenization, semantic understanding, and system performance [32].

Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM)

Recurrent Neural Networks (RNNs) are a class of neural networks that excel at modeling sequential data by maintaining a hidden state across time steps. However, traditional RNNs struggle with long-term dependencies due to issues like vanishing and exploding gradients, which hinder their effectiveness for tasks requiring context over extended sequences [33].

Long Short-Term Memory (LSTM) networks address these challenges by introducing a memory cell and three key gates: the forget gate, input gate, and output gate. These mechanisms enable LSTMs to selectively store, update, or discard information, making them highly effective for tasks requiring long-term dependencies [34]. LSTMs have demonstrated robust performance in natural language processing (NLP) applications, particularly for detecting subtle patterns in textual data, such as harassment [35].

LSTM Mechanism

The operations within an LSTM cell at time step t are as follows:

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) && \text{(Forget Gate)} \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) && \text{(Input Gate)} \\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) && \text{(Candidate Memory)} \\
C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t && \text{(Memory Cell Update)} \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) && \text{(Output Gate)} \\
h_t &= o_t \odot \tanh(C_t) && \text{(Hidden State Update)}
\end{aligned}
$$

By overcoming the limitations of traditional RNNs, LSTMs can model long-range dependencies in text, which is essential for detecting subtle and context-specific harassment cues [36].

Model Implementation

The proposed model leverages an LSTM-based architecture for classifying text into harassment subcategories such as insults and defamation. The implementation involves the following steps:

1. **Embedding Layer**: Transform input text into dense vector representations using embeddings trained during the model's learning process [37].
2. **LSTM Layer**: Process the sequential data to extract contextual features and generate hidden states representing the semantic structure of the text.

3. **Fully Connected Layer**: Map the output from the LSTM layer to the target classification space.
4. **Softmax Layer**: Convert raw scores into probabilities for assigning the input to specific categories.

Objective Function

The model is trained using the categorical cross-entropy loss function:

$$L = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{1}$$

Where:

- N: Total number of samples.
- y_i: True label for the i-th sample.
- $\hat{y}_i$: Predicted probability for the true class of the i-th sample.

Optimization is performed using advanced techniques such as Adam, ensuring stable convergence during training [38].

Advantages of LSTM for Harassment Detection

LSTMs are particularly well-suited for harassment detection tasks because:

- They capture long-term dependencies, allowing the detection of subtle harassment patterns spread across sentences [35].
- They handle informal and noisy text effectively, such as that found in social media [36].

By leveraging these capabilities, the proposed LSTM model provides a robust and reliable approach to detecting harassment in textual data.

METHODS

Figure 1 provides a detailed overview of the methodology applied in this research, from the initial steps to the final processes. The diagram aims to offer a clear understanding of the procedures followed in this research and how each stage is interconnected to achieve the research's objectives.

Problem Identification

In this stage, an interview was conducted with M. Niknik, a lecturer and research coordinator for the U-Tapis project. In addition to her academic role, M. Niknik is a linguistics expert frequently involved in forensic linguistic cases. During the interview, M. Niknik explained that the harassment reporting system in Indonesia has advanced to the point where individuals can report cases through a dedicated website. The system then decides whether or not the case will proceed for legal action. However, while this system was designed to streamline the process, it has inadvertently placed a burden on the professionalism of linguists [2].

One major challenge is the three-hour time limit for determining whether a case can move forward legally. Within this timeframe, the police consult linguists for an opinion on whether a given statement qualifies as harassment. Another issue is that linguists are often contacted outside of standard working hours. This situation highlights the need for a system that can automatically determine whether a sentence falls into the category of harassment, thus supporting the legal process and reducing the workload on linguistic professionals.

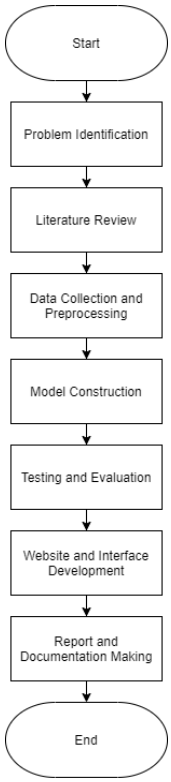

FIGURE 1. *Research Methodology Flowchart*

Literature Review

The literature review stage involves collecting both theoretical and practical knowledge to support the development of an NLP model for detecting harassment insults. The literature includes concepts such as the definition of harassment, insults, defamtaions, NLP, supervised learning, text preprocessing, and the LSTM algorithm.

Data Collection and Preprocessing

Data Collection

The data collection in this study involved 18,000 sentences generated by the ChatGPT language model using specific query methods. These sentences were then categorized into three groups: those containing insults, those containing defamation, and those that were neutral (i.e., not containing harassment). The labeling of the data was performed by ChatGPT based on predetermined rules, regulations, and definitions. These definitions were derived from relevant literature, expert-validated case examples, and the Indonesian Electronic Information and Transactions Law (UU ITE). This approach ensures that the generated data aligns with the context and boundaries of harassment, particularly in the categories of insults and defamation.

The use of synthetic data generated by ChatGPT offers several advantages that have been recognized in the literature. First, it helps address the scarcity of authentic data, which is often difficult to obtain for legal or privacy reasons. Second, the flexibility in designing query parameters enables researchers to maintain the relevance and purpose of the generated data. Studies such as those by Xu Guo and Yiqiang Chen (2024) have demonstrated that synthetic data from generative AI can effectively replace authentic data for specific tasks [39]. In the context of NLP, Ghanadian et al. (2024) utilized synthetic data generated by large models such as ChatGPT to detect suicidal ideation, thereby significantly improving model performance [40].

Text Preprocessing

Preprocessing textual data is a crucial step in preparing raw input for machine learning models. For this research on insult and defamation detection using Long Short-Term Memory (LSTM) networks, a systematic text preprocessing pipeline was implemented to ensure data consistency and facilitate effective model training. The key stages in this pipeline included text cleaning, data splitting, tokenization, conversion of text to sequences, and sequence padding.

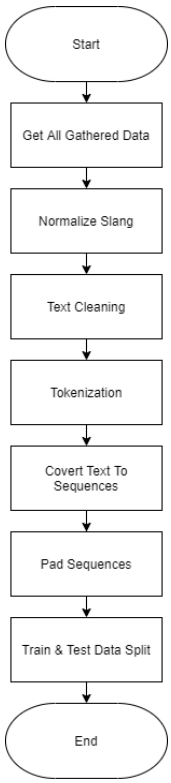

FIGURE 2. *Text Preprocessing Flowchart*

As shown in Figure 2, the preprocessing began with the normalization of slang words. Informal or abbreviated terms commonly found in Indonesian text, such as "gak" (informal for "tidak") and "dgn" (abbreviation for "dengan"), were replaced with their standard forms to ensure linguistic consistency.

The next step was text cleaning, which involved a series of actions to eliminate noise and irrelevant elements from the data. All text was converted to lowercase to ensure uniformity and prevent unnecessary distinctions due to capitalization. URLs, email addresses, and other unnecessary patterns were removed, followed by the elimination of special characters, numbers, and punctuation marks. Extra whitespace, including leading, trailing, and consecutive spaces, was stripped to produce a cleaner, standardized input.

After cleaning, the text was processed further without stemming or stopword removal. Unlike traditional preprocessing approaches, this research retained the original word forms and common words such as "dan," "atau," and "tetapi" to preserve contextual and linguistic patterns critical for classification. This decision was based on observations that stemming or removing stopwords could hinder the model's ability to capture important semantic and syntactic relationships. Tokenization was then performed, converting the processed text into tokens—individual words or phrases that form the building blocks of natural language processing models. To reduce computational complexity and improve model efficiency, the vocabulary was limited to the 10,000 most frequent words in the dataset, with out-of-vocabulary words replaced by a special token.

The tokenized text was subsequently transformed into sequences of integers, where each word was mapped to a unique index in the vocabulary. To address the varying lengths of text samples, all sequences were pre-padded or pre-truncated to a fixed length of 100 words. Padding augmented shorter sequences with zeros, while truncation shortened longer sequences, ensuring a uniform input shape compatible with the LSTM model.

Finally, the preprocessed data was split into two subsets: 80% for training and 20% for testing. This stratified division preserved the representativeness of the dataset across all subsets, enabling the model to learn effectively from the training data while maintaining a separate set for validation and final performance assessment.

Model Construction

The model construction and hyperparameter tuning process in this research consisted of several systematic steps to ensure robust performance in classifying insult, defamation, and neutral texts.

Model Construction

As shown on figure 3, the process began with the initialization of a sequential neural network model, enabling the addition of layers in a linear stack. The first layer was an embedding layer, configured to map the input words to dense vector representations in a 128-dimensional space. This embedding layer served to capture semantic relationships between words, ensuring that the model could learn meaningful patterns from the input text.

Following the embedding layer, the first and second LSTM (Long Short-Term Memory) layer was added to extract temporal and contextual features from the input sequences. The number of hidden units in the LSTM layer, dropout rate, and recurrent dropout rate were set as hyperparameters to be optimized during tuning. Dropout was applied to prevent overfitting by randomly deactivating a fraction of neurons during training.

The output layer was a dense layer with three units, corresponding to the three target classes: insult, defamation, and neutral. A softmax activation function was applied at this layer to normalize the outputs into probabilities, ensuring that the sum of the probabilities across the three classes equaled one. This configuration allowed the model to output class probabilities for each input text sample.

The model was then compiled with categorical crossentropy as the loss function, which is appropriate for multi-class classification tasks. The Adam optimizer was used to adjust the model's weights iteratively during training. Accuracy was chosen as the primary evaluation metric to assess the model's performance.

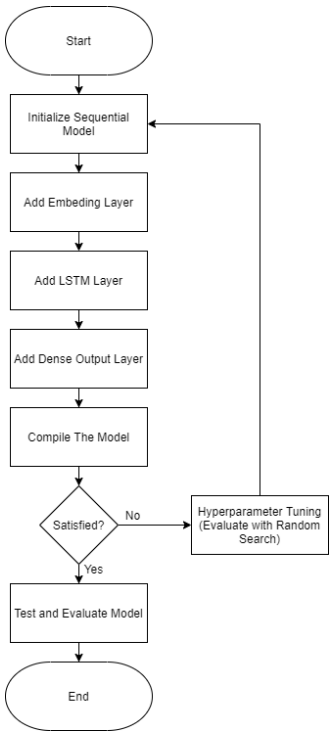

FIGURE 3. *LSTM Model Construction Flowchart*

Hyperparameter Tuning

To optimize the model's configuration, hyperparameter tuning was conducted using a grid search approach with the `keras_tuner` library. The search focused on key hyperparameters that significantly influence the model's performance. The parameters explored during the tuning process included:

- The dimensionality of the embedding layer (`embedding_dim`), tested with values of 50, 100, and 200.
- The number of units in the first and second Bidirectional LSTM layers (`lstm_units_1` and `lstm_units_2`), tested with values of 128 and 256 for each layer.
- Dropout rates for the first and second LSTM layers (`dropout_1` and `dropout_2`), as well as the dense layer (`dropout_dense`), tested with values ranging from 0.2 to 0.5 in increments of 0.1.

The grid search explored a total of 20 configurations, evaluating each based on validation accuracy. Each trial involved training a model composed of the following layers:

- An embedding layer with a vocabulary size of 10,000 and sequence length of 100.
- Two Bidirectional LSTM layers, each followed by dropout layers to prevent overfitting.
- A dense layer with 64 units and ReLU activation, followed by a dropout layer.
- A softmax output layer for classifying sentences into Neutral, Insult, and Defamation categories.

To ensure the model converged effectively and avoided overfitting, additional callbacks were incorporated during training:

- **EarlyStopping**: Monitored the validation loss and stopped training after three consecutive epochs without improvement, restoring the best weights to prevent overfitting.
- **ReduceLROnPlateau**: Reduced the learning rate by a factor of 0.5 if the validation loss did not improve for two consecutive epochs, facilitating smoother convergence.

After evaluating all configurations, the optimal hyperparameters were selected based on validation accuracy. The final model was trained using these hyperparameters for up to 20 epochs with a batch size of 32. This tuned model demonstrated improved performance and robustness during the evaluation phase, effectively balancing complexity and generalization.

Test and Evaluate Model

The final model was trained using the optimal hyperparameter configuration on the training dataset. The performance of the final model was evaluated using several key metrics, namely the *confusion matrix, accuracy score, precision, recall*, and *F1-score*. These metrics were chosen to provide a thorough evaluation of the model's performance, particularly in ensuring that all classes were treated equitably. By leveraging these metrics, the research ensures that the final model is both well-tuned and generalizable to unseen data, making it effective in classifying texts into the categories of insult, defamation, and neutral.

Testing and Evaluation

The evaluation of the experimental results was conducted manually by calculating and comparing the statistical metrics of the trained model. Additionally, the model was tested using a separate dataset distinct from the training data. The evaluation dataset consisted of 1150 sentences, with 150 sentences sourced from GPT 4o LLM and 1000 sentences sourced from the internet. Each dataset contained diverse examples representing the three categories: neutral, insult, and defamation.

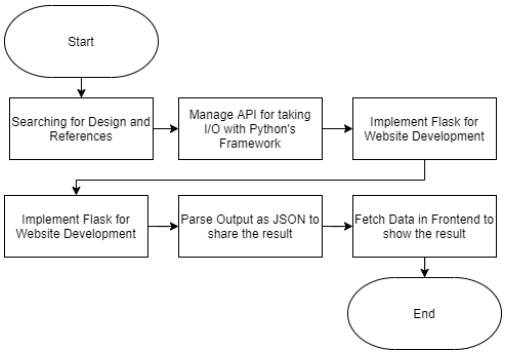

FIGURE 4. *Website Development Flowchart*

Website and Interface Development

After completing the testing and evaluation phase, the research proceeds to the creation of a web-based interface. The workflow for the web development process is illustrated in figure 4. Once the interface design or reference is finalized, the API is developed using Python. The Flask framework is employed in the web application to handle input data and generate output in JSON format. On the frontend, Vue.js is used for data processing, enabling the visualization of words and sentences. The interface displays paragraphs with identified errors alongside their corrected forms, providing an intuitive and user-friendly representation of the results.

Report and Documentation Making

At the final stage, a comprehensive report and documentation will be prepared, detailing the results of the nd the implementation of the developed model. This documentation will include information on the research findings, model implementation, and the evaluation of the model's accuracy.

RESULT AND ANALYSIS

This section is divided into four subsections: dataset collection, preprocess, the design and interface of the web-based application, and the testing and evaluation of the LSTM model. The web-based interface was developed to assist law enforcement in conducting preliminary detection of linguistically relevant cases. Following this, an in-depth analysis of the model's testing and evaluation is presented, detailing its performance and accuracy in detecting harassment cases, including insults and defamation.

Dataset

The dataset used in this research consists of 18,000 labeled text samples, evenly distributed across three categories: *neutral*, *insult*, and *defamation*, with 6,000 samples in each category. The dataset was generated using GPT, leveraging its advanced language modeling capabilities to produce linguistically diverse and contextually appropriate examples for each classification category. This approach facilitated the creation of a well-balanced dataset, enabling the LSTM model to learn the nuanced differences between categories effectively. The use of GPT-generated data also allowed for the inclusion of controlled and structured samples, which were instrumental in testing and evaluating the system's performance under various linguistic scenarios.

Preprocess

The preprocessing stage significantly enhanced the dataset's quality, ensuring optimal performance of the LSTM model. Slang normalization was a key component, where a slang-to-formal mapping was applied using a colloquial Indonesian lexicon. This step allowed the conversion of informal expressions into their standard forms, enabling the model to better capture the intended meaning of the text.

Text cleaning involved converting all characters to lowercase and systematically removing unnecessary elements such as URLs, special characters, numbers, and punctuation. By focusing on linguistically relevant content, this process reduced noise and improved data clarity.

Unlike traditional preprocessing techniques, stemming and stop-word removal were deliberately omitted in this research. This decision was based on the observation that stemming and removing stop words often eliminate critical linguistic patterns and nuances. These patterns, such as specific word choices or syntactical structures, are particularly important for distinguishing between categories like *insult* and *defamation*, where subtle variations in language play a significant role.

Tokenization transformed the cleaned text into sequences of integer tokens, limiting the vocabulary to the 10,000 most frequent words for computational efficiency. Padding and truncation standardized the length of input sequences to 100 tokens, ensuring uniformity across the dataset. Additionally, class labels were encoded numerically, facilitating seamless integration with the LSTM model.

These preprocessing steps not only refined the input data but also directly contributed to the model's ability to distinguish between subtle linguistic differences in the categories of *neutral*, *insult*, and *defamation*, as demonstrated in subsequent performance evaluations.

Website Implementation

FIGURE 5. Home Page Section 1

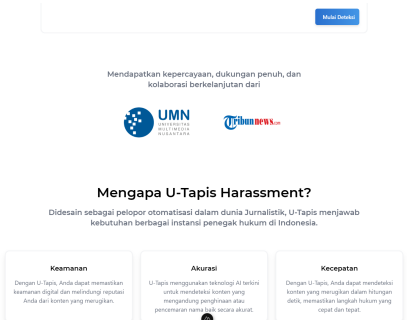

FIGURE 6. Home Page Section 2

FIGURE 7. Home Page Footer

The developed web-based interface provides an intuitive and user-friendly platform for text classification, specifically designed to assist in the detection of linguistic harassment. Figure 7 illustrates the initial interface displayed when users access the website. To begin with, users are required to input a sentence in the designated input field provided on the homepage. Once the input is complete, they can proceed by clicking the *"Mulai Deteksi"* button, which triggers the system to process the text and classify it accordingly.

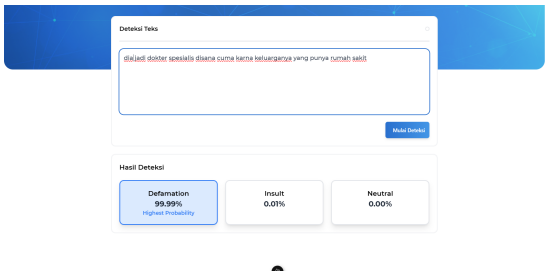

FIGURE 8. Classification Result

Figure 8 shows the output generated by the system after processing the user's input. The results include the probabilities associated with each classification category—*neutral*, *insult*, and *defamation*—displayed as percentages below the input field. This clear visualization enables users to interpret the results effectively. Additionally, if users wish to analyze another sentence or recheck their input, they can simply click the *"Mulai Deteksi"* button again. This action resets the interface and prompts the system to perform a new calculation based on the updated input.

The interface is designed with simplicity and efficiency in mind, ensuring that it meets the practical needs of law enforcement officers who may require rapid and accurate preliminary assessments.

Testing and Evaluation

The purpose of this section is to assess the performance and effectiveness of the developed system in accurately classifying text into neutral, insult, and defamation categories. To achieve this, the system was evaluated using a dataset specifically designed to test its capabilities under both controlled and real-world scenarios. This evaluation utilized a total of 1150 data samples, comprising 1000 labeled data samples sourced from the internet to reflect everyday language use and 150 labeled data from LLM ChatGPT 4o.

By conducting these tests, this section aims to validate the accuracy and reliability of the system while identifying areas for potential improvement. This rigorous evaluation process is critical to ensuring the system's readiness for practical applications, such as its integration into a web-based interface for law enforcement.

Testing with 150 GPT Datasets

The evaluation of the system using the *LSTM* model with 150 labeled data samples generated by ChatGPT aims to assess the model's ability to accurately classify text into the categories of Neutral, Insult, and Defamation. The evaluation results are presented in Table 1.

TABLE 1. Confusion Matrix for the LSTM Algorithm (ChatGPT Data)

True Label	Neutral (0)	Insult (1)	Defamation (2)
Neutral (0)	54	0	0
Insult (1)	15	32	2
Defamation (2)	5	0	42

The confusion matrix demonstrates that the *LSTM* model classified most samples accurately, achieving an overall accuracy of 85%. The detailed metrics of precision, recall, and F1-score for each category are provided in Table 2.

TABLE 2. Classification Report for the LSTM Algorithm (ChatGPT Data)

Label	Precision	Recall	F1-Score	Support
Neutral (0)	0.73	1.00	0.84	54
Insult (1)	1.00	0.65	0.79	49
Defamation (2)	0.95	0.89	0.92	47
Accuracy		**0.85 (85%)**		

The confusion matrix and classification report reveal several performance trends:

- Neutral (True Label: 0): All 54 Neutral samples were correctly classified as Neutral, resulting in a recall of 100% and an F1-score of 84%. The precision of 73% suggests a modest possibility of false positives for this category when considering the other labels.
- Insult (True Label: 1): Out of 49 Insult samples, 32 were correctly classified, while 15 were misclassified as Neutral and 2 were misclassified as Defamation. This category achieved an F1-score of 79%, reflecting a perfect precision of 100%, but a recall of 65% indicates further room for improvement in detecting Insult content.
- Defamation (True Label: 2): Of the 47 Defamation samples, 42 were correctly identified, while 5 were misclassified as Neutral. This resulted in an F1-score of 92%, with precision of 95% and recall of 89%.

Overall, the *LSTM* model attained an accuracy of 85%. The confusion matrix indicates solid performance for Defamation and good performance for Neutral samples, with comparatively more confusion between the Neutral and Insult categories. These findings highlight the model's ability to handle text generated by ChatGPT effectively, while still underscoring opportunities to refine Insult detection accuracy.

Testing with 1000 Internet-Sourced Datasets

The evaluation of the system using the *LSTM* model with 1000 labeled data samples collected from various Internet sources aims to assess the model's ability to accurately classify text into the categories of Neutral, Insult, and Defamation. The evaluation results are presented in Table 3.

The confusion matrix demonstrates that the *LSTM* model correctly classified most samples, achieving an overall accuracy of 85%. The detailed metrics of precision, recall, and F1-score for each category are provided in Table 4.

- Neutral (True Label: 0): Of the 412 Neutral samples, 356 were correctly classified, yielding a recall of 86% and an F1-score of 89%. The precision of 91% indicates strong performance in correctly identifying Neutral instances.

- Insult (True Label: 1): Among the 320 Insult samples, 274 were correctly classified, while 15 were labeled as Neutral and 31 as Defamation. This category achieved an F1-score of 85% with a recall of 86%, suggesting reliable but not perfect detection of Insult cases.
- Defamation (True Label: 2): Of the 268 Defamation samples, 222 were correctly identified, while 19 were misclassified as Neutral and 27 as Insult. This resulted in an F1-score of 80%, underlining the category's slightly greater confusion compared to Insult and Neutral.

TABLE 3. Confusion Matrix for the LSTM Algorithm (Internet-Sourced Data)

True Label	Neutral (0)	Insult (1)	Defamation (2)
Neutral (0)	356	24	32
Insult (1)	15	274	31
Defamation (2)	19	27	222

TABLE 4. Classification Report for the LSTM Algorithm (Internet-Sourced Data)

Label	Precision	Recall	F1-Score	Support
Neutral (0)	0.91	0.86	0.89	412
Insult (1)	0.84	0.86	0.85	320
Defamation (2)	0.78	0.83	0.80	268
Accuracy		**0.85 (85%)**		

Overall, the *LSTM* model attained an accuracy of 85%. While performance across categories is generally strong, future improvements may focus on refining the detection of Defamation instances and reducing the confusion among related categories.

CONCLUSION

This research evaluated a text classification system using Long Short-Term Memory (LSTM) networks to categorize text into Neutral, Insult, and Defamation categories. The system was tested on two datasets: a 150-sample set generated by ChatGPT and a 1000-sample set collected from the internet, achieving accuracies of 85% and 85%, respectively. Notably, the model demonstrated strong performance in recognizing Defamation, showing high precision and recall, and effectively handled diverse linguistic expressions.

Challenges did emerge in distinguishing Insult from Neutral in some cases, as well as in correctly identifying Defamation when text was subtle or ambiguous. These observations underscore the need for enhanced contextual understanding and refined classification in borderline instances. Future work could explore integrating large language models (LLMs)—such as Indonesia's Sahabat AI or BLOOM—to improve semantic representations and overall accuracy. This integration can help the system better adapt to domain-specific nuances and underrepresented language patterns, ultimately strengthening its performance in real-world scenarios.

ACKNOWLEDGMENTS

The authors would like to express their gratitude to Universitas Multimedia Nusantara for their support of this research, as well as to TribunNews for partnering with us in this research.

REFERENCES

1. J. Nelson, "Digital harassment and its implications in the modern era," Journal of Cyber Ethics **15**, 120–134 (2019).

2. M. Niknik, “private communication,” (2024).
3. M. Niknik, *Automation in Journalism: Challenges and Opportunities* (Media Nusantara Press, 2021).
4. N. Mediyawati, “Developing automated linguistic tools for journalism in indonesia,” Indonesian Journal of Media Studies **10**, 45–60 (2021).
5. A. Sutomo, “Advancements in indonesian language error detection systems,” Journal of Computational Linguistics **18**, 78–95 (2024).
6. R. Saputra, “Improving word usage detection in indonesian texts,” Language Technology Journal **22**, 120–145 (2024).
7. H. Siswanto, “Analyzing compound word errors in indonesian journalism,” Journal of Automated Linguistics **16**, 88–101 (2024).
8. E. Dwitya, “Spelling error detection in indonesian texts using nlp,” Indonesian Computational Journal **12**, 65–80 (2024).
9. A. Ramezani, “Forensic linguistics and its role in textual evidence,” International Journal of Linguistic Evidence **7**, 15–30 (2016).
10. S. Ahmed, “The evolution of forensic linguistics in digital investigations,” Journal of Digital Criminology **9**, 200–220 (2021).
11. M. Coulthard, *An Introduction to Forensic Linguistics* (Oxford University Press, 1998).
12. V. Balakrishnan and S. Khan, “Hate speech detection using deep learning approaches,” Journal of Artificial Intelligence Research **65**, 377–392 (2020).
13. Z. Zhang and L. Luo, “Hate speech detection: Challenges and opportunities,” Journal of Computational Linguistics **44**, 1–20 (2018).
14. T. Chen and C. Guestrin, “Text classification using advanced lstm networks,” Proceedings of the Neural Information Processing Symposium , 1280–1290 (2018).
15. H. Yao and M. Sun, “Feature engineering for lstm-based text classification,” Neural Networks Journal **123**, 90–102 (2019).
16. S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation **9**, 1735–1780 (1997).
17. D. Yin and R. Roscher, “Sentiment and hate speech detection using lstm,” Journal of Machine Learning Applications **33**, 45–60 (2017).
18. P. Badjatiya and S. Gupta, “Deep learning for hate speech detection in tweets,” Proceedings of the International Conference on Web and Social Media , 759–760 (2017).
19. J. Kim and K. Lee, “Lstm in hate speech detection for multi-lingual contexts,” Journal of Applied Deep Learning **28**, 212–228 (2020).
20. B. Barzegar and H. Soleimani, “Text classification and linguistic nuances detection using lstm,” Neural Language Processing Journal **47**, 158–170 (2021).
21. X. Yang and Y. Li, “Lstm applications in online hate speech classification,” Digital Communication Research Journal **53**, 89–110 (2021).
22. A. Schmidt and M. Wiegand, “Survey on hate speech detection using deep learning methods,” Journal of Computational Ethics **48**, 1–20 (2019).
23. A. S. Baker, “Cyber harassment and its impact on mental health: A literature review,” Journal of Digital Safety and Wellbeing **4**, 45–61 (2020).
24. J. Cheng, C. Danescu-Niculescu-Mizil, and J. Leskovec, “Antisocial behavior in online discussions,” in *Proceedings of the Ninth International AAAI Conference on Web and Social Media* (AAAI, 2015) pp. 61–70.
25. K. Dinakar, B. Jones, C. Havasi, H. Lieberman, and R. Picard, “Common sense reasoning for detection, prevention, and mitigation of cyberbullying,” ACM Transactions on Interactive Intelligent Systems **2**, 18 (2012).
26. T. Davidson, D. Warmsley, M. Macy, and I. Weber, “Automated hate speech detection and the problem of offensive language,” Proceedings of the International AAAI Conference on Web and Social Media **11**, 512–515 (2017).
27. P. Burnap and M. Williams, “Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making,” Policy and Internet **7**, 223–242 (2015).
28. D. McLean, *Digital Defamation: Law and Policy* (Routledge, 2012).
29. S. Hewett, “Detecting defamatory statements in social media content,” Journal of Digital Safety **8**, 22–35 (2016).
30. R. V and S. Naik, “Natural language processing,” International Journal of Computing Algorithm **12** (2023), 10.20894/IJCOA.101.012.002.001.
31. X. Sun, X. Liu, J. Hu, and J. Zhu, “Empirical studies on the nlp techniques for source code data preprocessing,” in *Proceedings of the 2014 3rd International Workshop on Evidential Assessment of Software Technologies* (ACM, 2014) pp. 32–39.
32. A. Tabassum and R. R. Patil, “A survey on text pre-processing & feature extraction techniques in natural language processing,” International Research Journal of Engineering and Technology (IRJET) **7**, 4864–4867 (2020).
33. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” Advances in neural information processing systems **30** (2020).
34. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” Proceedings of the 2019 NAACL-HLT **1**, 4171–4186 (2019).
35. Z. Zhang, L. Luo, and J. Zeng, “Deep learning for hate speech detection on social media: A review,” Online Information Review **43**, 329–351 (2019).
36. J. Y. Choi, J. S. Lee, and J. Kim, “Detecting offensive content on social media to protect adolescent online safety,” Multimedia Tools and Applications **78**, 10529–10552 (2019).
37. Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” Advances in neural information processing systems **32** (2019).
38. T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, *et al.*, “Transformers: State-of-the-art natural language processing,” arXiv preprint arXiv:1910.03771 (2020).
39. X. Guo and Y. Chen, “Generative ai for synthetic data generation: Methods, challenges and the future,” ArXiv **abs/2403.04190** (2024), 10.48550/arXiv.2403.04190.
40. H. Ghanadian, I. Nejadgholi, and H. A. Osman, “Socially aware synthetic data generation for suicidal ideation detection using large language models,” IEEE Access **12**, 14350–14363 (2024).