ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 14 (2024) No. 2, pp. 1-20
DOI:10.33168/JSMS.2024.0201

Applying Long Short-Term Memory Algorithm for Spam Detection on Ministry Websites

Rahmi Fadillah Busyra, Abba Suganda Girsang

Computer Science Department, BINUS Graduate Program – Master of Computer Science, Bina Nusantara University, Jakarta 11480, Indonesia

rahmi.busyra@binus.ac.id; agirsang@binus.edu

Abstract. Spam refers to unsolicited messages containing harmful content such as malware, viruses, phishing, and data theft. The web form of a government ministry's website is frequently targeted by spammers, causing disruptions, database overload, hindering communication with the public, and security risks. While numerous studies have focused on spam detection, none have addressed spam detection on web form submissions and multilingual spam detection, specifically in English and Indonesian. This study developed a spam detection model to address the growing challenge of spam messages received through ministry website web forms. The proposed model employs the Long Short-Term Memory (LSTM) algorithm to detect spam in English and Indonesian effectively. The LSTM model incorporates additional stages to enhance its performance, including language detection, data augmentation, and word embedding. Evaluation results demonstrate the model's effectiveness in classifying spam and non-spam messages, particularly in datasets with balanced class distributions. This research holds practical implications for implementing the model on websites, particularly the government ministry's website, to effectively categorize incoming messages and mitigate the impact of spam. The study also contributes theoretically by showcasing the effectiveness of LSTM in spam detection and emphasizing the importance of data augmentation in handling imbalanced datasets. Overall, this study provides valuable insights and practical solutions for spam detection in web forms, applicable to government ministry websites, and expands the scope of spam detection in multiple languages, specifically English and Indonesian.

Keywords: Deep Learning, LSTM, Spam Detection, Text Classification

1. Introduction

Spam is an electronic message sent in bulk (bulk messages) without being asked and is usually unwanted by the recipient. Spam refers to messages in the form of fake advertisements, malicious URLs, fake news, phone numbers, hashtags, images with hidden URLs, etc. (Rao et al., 2021). Spam can contain harmful content such as malware, viruses, and phishing. People who create spam (spammers) spread spam for various purposes that violate the law and can harm the target, such as phishing, spying, cyberbullying, and others (Bhat & Abulaish, 2014; Bindu et al., 2018). In addition, spam can also lead to data breaches that unauthorized parties can exploit.

Spammers can distribute spam through various media. There are several categories of spam based on the distribution process, including email, mobile, web form, comment, SEO, social networking, messaging, trackback, etc. Spam emails are the most common type of spam found. Nearly 45% of the emails that sent daily are spam (Spamlaws, 2023). On average, users of global email services receive a significantly higher amount of unwanted emails every month, including unwanted commercials, viruses, Trojan Horses, worms, and phishing attempts (Šolić et al., 2013). There have been a lot of research studies discussing spam in email, but only a few have discussed spam in web form. Web form spam refers to spam that enters through the submission of forms on a website and can be done by humans (human scammers) or spambots. The information submitted through web forms is often fake or irrelevant. It can include advertisements redirecting to other websites, links leading to phishing websites, or sites that initiate malware downloads. Web forms have become targets for scammers to spread malware, steal data or personal information, create hidden/fake links, and even hijack the control of websites. This type of spam can also cause various website issues, including slow response time to non-spam messages, poor user experience, safety and security concerns, and hampered analytics.

The ministry website is a platform for sharing information and engaging with the public. These websites typically include a form where users can submit messages such as inquiries, criticisms, or suggestions to the ministry. However, the presence of this form increases the risk of spam and the infiltration of harmful content such as malware, viruses, phishing, and the leakage of important data owned by the ministry. It can lead to significant losses for the ministry and the government. Therefore, a solution is needed to prevent or filter spam on ministry websites.

One of the ways to prevent spam from entering a website is through spam detection. Spam detection is a method used to identify and detect spam that enters a platform, such as email, SMS, social media, websites, and others. Spam detection helps minimize the risks and potential harm caused by spam.

Research related to spam detection has been conducted extensively. Rachmat and Lukito (2017) conducted one of these studies using the Naïve Bayes classification method. This research focused on classifying Indonesian-language spam comments using training data consisting of spam comments on Instagram. The achieved accuracy rate was 77.25%. Another study that focused on Indonesian-language spam and used the Naïve Bayes algorithm was conducted by Vernanda et al. (2020). This research aimed to enhance spam detection using the N-gram method and implementing a REST API architecture. The study's results achieved accuracy rates ranging from 61% to 94%.

Garg and Girdhar (2021) reviewed spam filtering techniques using Natural Language Processing (NLP). They focused on spam detection in English emails and comments using various methods, resulting in different accuracies. The SVM (Support Vector Machine) algorithm achieved the highest accuracy, up to 98%. However, this model still has limitations regarding typographical errors or texts written in different languages, which may lead to undetected spam.

Jain et al. (2018) conducted a study that used the Long Short-Term Memory (LSTM) method to perform spam detection in English. Before LSTM classification, the text was transformed into

semantic word vectors using Word2Vec, WordNet, and ConceptNet. The classification results were compared with several machine learning methods, including Support Vector Machine (SVM), Naïve Bayes, Artificial Neural Network (ANN), K-Nearest Neighbor (KNN), and Random Forest. The data used for comparison are the SMS spam and Twitter datasets. The research concluded that LSTM outperformed traditional machine learning methods in detecting spam.

A study conducted by Chandra and Khatri (2019) also achieved good accuracy in detecting English spam. The research focused on Spam SMS Filtering using Deep Learning methods, specifically Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM). After comparing with the Naïve Bayes and Support Vector Machine (SVM) algorithms, this method achieved the highest accuracy of 98% with only two false negatives. The required time for this model was also relatively short, taking only 13.44 seconds.

Many spam detection models have been developed in previous research with good detection results. The Deep Learning algorithm, LSTM, has outperformed machine learning algorithms such as Naïve Bayes, SVM, KNN, ANN, and Random Forest in detecting spam. However, no research has focused on web form spam detection and can handle multilingual spam detection, especially in English and Indonesian. As messages received on ministry websites, whether spam or non-spam, are written in multiple languages rather than just a single language, it is important to consider this linguistic diversity in the spam detection process.

Therefore, this paper aims to develop a spam detection model for messages received through the web form of a ministry website using the Deep Learning method with the Long Short-Term Memory (LSTM) algorithm, capable of detecting spam in English and Indonesian. The research results show that the studied model has effectively detected spam in English and Indonesian. The main contributions of this research are as follows:

- Dataset Collection: The paper presents a dataset consisting of messages from the web form of a ministry website in both English and Indonesian languages. The dataset is labeled as spam or non-spam, which can be valuable for research related to web form spam.
- Spam Detection Model: This research proposes a model that can detect spam in two languages, English and Indonesian, using deep learning techniques, specifically the LSTM algorithm. The model is designed to identify and classify spam messages in both languages effectively.
- Comparative Evaluation: This research conducts a comparative evaluation of the proposed model using multiple datasets. The model's performance is assessed and compared across different datasets, providing insights into its effectiveness and robustness.

Overall, the paper contributes to web form spam detection by providing a labeled dataset and proposing a spam detection model that works effectively in English and Indonesian. The comparative evaluation adds further value by demonstrating the model's performance across various datasets.

2. Literature Review

Research on text-based spam detection and classification using machine learning methods has been widely conducted. These studies employ various techniques, such as supervised learning (SL), unsupervised learning (USL), semi-supervised learning (SSL), Metaheuristic Optimization Algorithm (MOA), Ensemble Learning (EL), and Deep Learning (DL).

Gupta et al. (2018) proposed a framework using Support Vector Machine (SVM), Neural Network (NN), Random Forest (RF), and Gradient Boosted (GB) algorithms. These algorithms detected spam in user features and Twitter text (tweets) by identifying spam tweets, spammers, or suspicious accounts. Tests were carried out on 400,000 public tweets and 30 extracted words to obtain maximum information for tweet classification. The framework achieved an accuracy of 91.65%.

The study conducted by Saumya and Singh (2018) used three different classifiers, namely Gradient Boosted, Random Forest, and Support Vector Machine, to be applied to the reviews and comments. This research used a sentiment mining approach, and the testing was performed on 1332

reviews from Amazon. The F-score results showed a value of 91%. This study identified several limitations of the supervised learning method, such as manual labeling and data imbalance issues. Supervised learning methods have drawbacks in real-time scenarios, especially in the evolving form of spam.

Jain et al. (2018) implemented a Semantic Convolutional Neural Network (SCNN) model that utilizes CNN with a semantic layer. The semantic layer uses a Natural Language Processing (NLP) technique, Word2Vec, to map words to word vectors from Google's Word2Vec. The model was implemented on two datasets: a Twitter dataset with an accuracy of 94.40% and an SMS spam dataset with an accuracy of 86.5%.

Wu et al. (2017) developed a Deep Learning technique for identifying spam on Twitter using a Multi-Layer Perceptron (MLP) classifier and Word2Vec technique. The dataset in this study consists of 376,206 spam tweets and 73,836 non-spam tweets. This research was tested only on a single dataset and achieved an accuracy of 99.35%.

Murti and Naveen (2023) focused on spam detection, particularly in the context of phishing emails. They utilized a single dataset of phishing emails and evaluated various machine learning algorithms, including Naïve Bayes, Support Vector Machine (SVM), Random Forest, Decision Tree, and K-Nearest Neighbors (KNN). The results revealed that Random Forest achieved an impressive accuracy rate of 99.45%. However, it is important to note that the model was specifically designed for detecting spam phishing emails and was not tested on other types of spam.

The research conducted by Chandra and Khatri (2019) also achieved good accuracy in spam detection. This study focused on Spam SMS Filtering using Deep Learning methods, specifically Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM). After comparing with Naïve Bayes and Support Vector Machine (SVM) algorithms, this method achieved the highest accuracy of 98% with only two false negatives. The time required for this model was also relatively short, approximately 13.44 seconds.

Sinhmar et al. (2021) developed a spam detection model using a genetic algorithm to optimize the LSTM algorithm. The model was tested on several English SMS spam datasets. The results were highly promising, with an accuracy of up to 99%. These good results demonstrate that the genetic algorithm can improve the performance of LSTM.

Another study utilizing the LSTM algorithm is the spam detection model developed by Rodrigues et al. (2022). This model was used for spam detection and sentiment analysis of real-time Twitter data. Experiments were conducted using machine learning and deep learning methods such as Naïve Bayes, SVM, and LSTM. The LSTM algorithm achieved the highest accuracy rate of 98.74% for spam detection and 73.81% for sentiment analysis.

Research related to spam detection in the Indonesian language has been conducted extensively. Rachmat and Lukito (2017) conducted one of these studies using the Naïve Bayes classification method. This study classified Indonesian spam comments using training data consisting of spam comments from Instagram. The achieved accuracy rate was 77.25%.

Another study that focused on spam in Indonesian and used the Naïve Bayes algorithm was conducted by Vernanda et al. (2020). This research uses the N-gram method and REST API architecture to enhance the spam detection process. The research results have an accuracy rate ranging from 61% -94%.

Singh et al. (2019) raised the topic of spam detection research on mixed Hindi and English sentences from social media using a small dataset of Hindi-English code. This study used a combination of supervised and deep learning methods and achieved an accuracy of 73.2%.

Another study that focuses on detecting SMS spam in two different languages, especially English and Arabic, was conducted by Ghourabi et al. (2020) using the CNN-LSTM hybrid method. The results showed that the CNN-LSTM method outperformed other methods with an accuracy of 98.37%.

Based on previous research, studies on spam detection are highly diverse. Spam detection has

been extensively conducted on email, SMS, and social media datasets. Moreover, spam detection models have been developed using various methods and have achieved high-accuracy results, especially in models that utilize Deep Learning methods, including the LSTM algorithm.

However, these studies still have some limitations. There is a lack of research focusing on spam detection in web forms of a website. Most developed spam detection models are designed to detect spam in a single language. While some studies have developed models for detecting spam in multiple languages (multilingual), no research focuses on spam detection in two languages, particularly English and Indonesian.

Therefore, the contribution of the proposed research, compared to existing studies, is the proposal of a spam detection model that can handle web form spam in two languages, namely English and Indonesian, using the LSTM algorithm.

3. Methodology

In this section, the proposed model will be explained in detail. The main idea of this spam detection system is to process the text in the dataset and apply deep learning methods to classify it. In this research, spam detection is performed on a web form spam dataset consisting of two different languages, specifically English and Indonesian, using the Long Short-Term Memory (LSTM) algorithm. The proposed spam detection model can be seen in the flowchart in Figure 1.

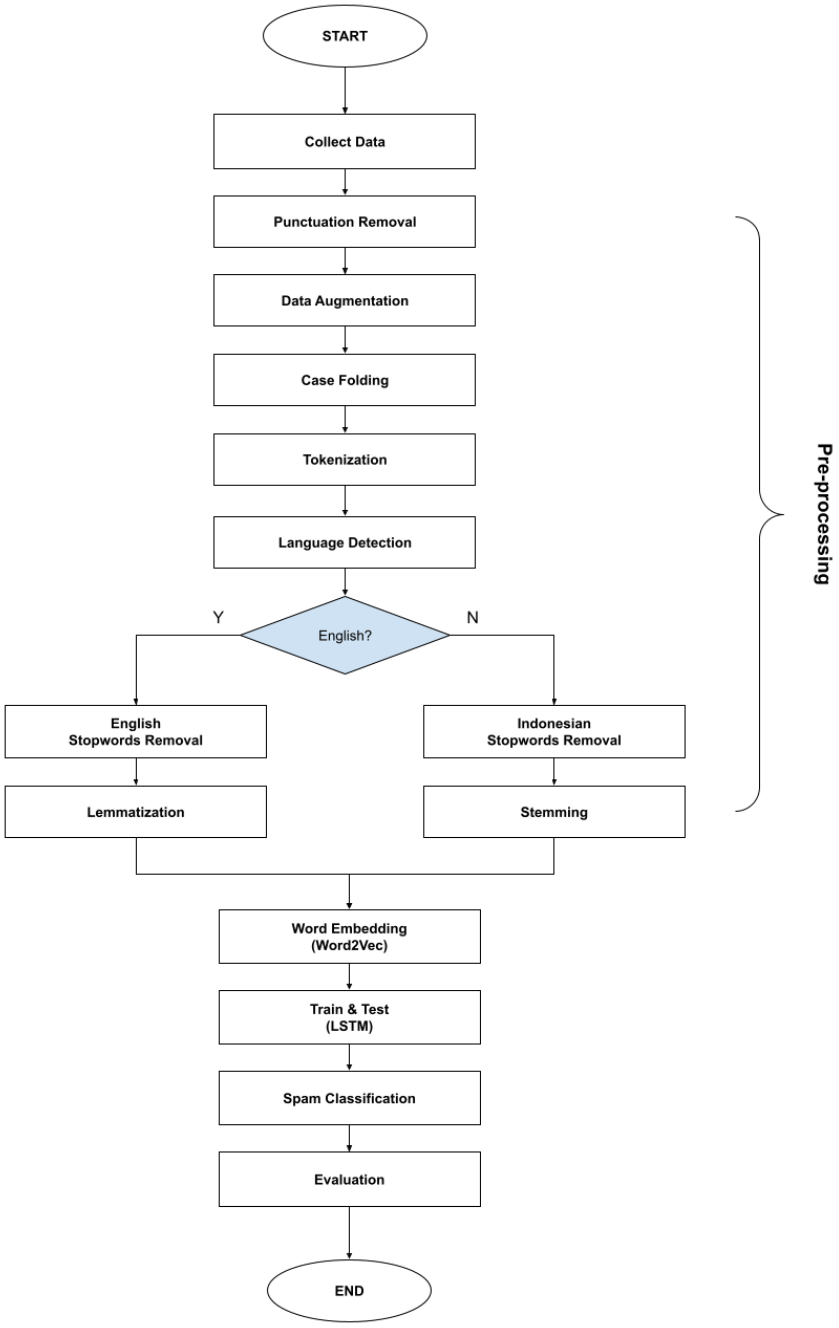

Fig.1: Spam Detection Model Flowchart

The process in the system starts with collecting data and removing empty and redundant data. It is then followed by data preprocessing, which consists of several stages. The preprocessing aims to prepare the data in a format ready to be processed. The preprocessing output is then processed using the word embeddings technique, specifically Word2Vec, to convert the text into numerical vector values. After that, the LSTM algorithm is applied to the data for classifications, aiming to distinguish between spam and non-spam text. Then, the model is evaluated by calculating accuracy, precision, recall, and f-measure scores from the obtained classification results.

3.1 Data Collection

This research uses several datasets for its training and testing processes. These include one main dataset called RIDA Web Form Spam Dataset and three public datasets: SpamAssassin Email Dataset,

SMS Spam Bahasa Indonesia Dataset, and UCI SMS Spam Collection Dataset. The public datasets are used for comparison while testing the spam detection model.

3.1.1 RIDA Web Form Spam Dataset

This dataset is obtained from the Regional Infrastructure Development Agency (RIDA), Ministry of Public Works and Housing, which is a government institution in Indonesia specializing in infrastructure development and construction. The dataset consists of messages received on the RIDA website through one of the forms, specifically from the question, critique, and suggestion form. It contains 1,228 non-spam and 3,687 spam messages in both English and Indonesian and has been labeled as spam and non-spam. The spam data in this dataset is diverse, including messages created by humans and spambots. This dataset is the main dataset used in this research.

3.1.2 SpamAssassin Email Dataset

This dataset is email data in English obtained from Apache SpamAssassin's public datasets. It includes 2,500 non-spam and 500 spam emails. In this dataset, all numbers and URLs have been replaced with the string "NUMBER" and "URL" respectively, to simplify the dataset. This dataset can be downloaded from spamassassin.apache.org.

3.1.3 SMS Spam Bahasa Indonesia Dataset

This dataset consists of 1,143 SMS messages with 569 non-spam and 574 spam data in Indonesian. It is sourced from research conducted by Rahmi and Wibisono (2016) under a Creative Commons license. In this dataset, the data is originally divided into three classes: non-spam, fraud, and advertisement/promotion. However, for this research, the data is simplified into two classes: non-spam and spam.

3.1.4 UCI SMS Spam Collection Dataset

This dataset comprises 5,574 SMS messages with spam and non-spam labels in English. There are 747 spam and 4,827 non-spam messages. The dataset was obtained from the UC Irvine Machine Learning Repository and can be downloaded from archive.ics.uci.edu.

The list of datasets can be seen in Table 1.

Table 1: List of Datasets

No.	Dataset Name	Source	Amount of data	Language
1	RIDA Web Form Spam Dataset	BPIW website	4,915	English and Indonesian
2	SpamAssassin Email Dataset	Apache SpamAssassin	3,000	English
3	SMS Spam Bahasa Indonesia Dataset	(Rahmi & Wibisono, 2016)	1,143	Indonesian
4	UCI SMS Spam Collection Dataset	UCI Machine Learning Repository	5,574	English

The dataset statistics can be seen in Table 2.

Table 2: Dataset Statistics

No.	Dataset Name	Amount of Data	Spam	Non-Spam
1	RIDA Web Form Spam Dataset	4,915	3,687	1,228
2	SpamAssassin Email Dataset	3,000	500	2,500
3	SMS Spam Bahasa Indonesia Dataset	1,143	574	569
4	UCI SMS Spam Collection Dataset	5,574	747	4,827

3.2 Data Preprocessing

Several preprocessing steps are applied to the data before classifying using LSTM for spam detection. In NLP, preprocessing refers to transforming raw data into a consistent and efficient format. The

purpose is to ensure the data quality is improved before analyzing the data. The preprocessing steps used in this research include punctuation removal, case folding, tokenization, stopword removal, stemming, and lemmatization.

In addition, data augmentation and language detection processes are also applied during the preprocessing stage. Data augmentation is a technique used to increase the amount of data by modifying existing data, which can help improve the model's performance. Language detection is the process of identifying the language of a text, which is useful when dealing with multilingual datasets to ensure that the appropriate preprocessing steps are applied for each language.

3.2.1 Punctuation Removal

In this stage, punctuation marks (such as ? ! , / = + - \ > < ; “ ( ) {} [ ] . : | and others) will be replaced with spaces. This removal is done because punctuation marks are usually ignored during training, and removing them simplifies the training process.

3.2.2 Data Augmentation

Data augmentation is a technique used to increase the amount of data by modifying the existing data to create different versions. Data augmentation refers to algorithms that generate artificial data from an available dataset (Shorten et al., 2021).

Data augmentation aims to balance the amount of data because the collected data is still imbalanced, which can affect the model's accuracy. Currently, the RIDA Web Form Spam Dataset, the main dataset, has an imbalanced distribution, with the majority class being spam and the minority class being non-spam. It can cause problems because the model training will spend more time learning the spam class than the non-spam class, making it challenging for the model to classify non-spam data due to insufficient learning of non-spam data, thus affecting the classification results. Data augmentation can increase the size of the minority class data during the training process, improving the model's performance. The more data available, the better the model's performance will be.

Data augmentation can be performed at the characters, words, or sentences level. There are several techniques in text data augmentation, one of which is Easy Data Augmentation (EDA). The techniques in EDA include Synonym Replacement, Random Insertion, Random Swapping, and Random Deletion.

In this research, the technique used for text data augmentation is Synonym Replacement. Synonym Replacement is a word-level augmentation technique that replaces words or phrases with their synonyms. It creates new data by randomly selecting words in a sentence and replacing them with synonyms of the selected words. This technique is chosen to generate effective text augmentation because it can produce diverse and different data from the original while still maintaining the context of the sentence.

The applied technique of Synonym Replacement is lexical-based replacement using WordNet. WordNet is a lexical database that can find synonyms of the tokens or words to be replaced in the original sentence. The library used for data augmentation is one of the Python libraries called Nlpaug. Nlpaug is a library for text data augmentation that can perform various data augmentation techniques at the characters, words, and sentences level. In this study, synonym replacement is performed on 30% of the words in the text, with a minimum value of 1 word and a maximum of 10 words.

3.2.3 Case Folding

Case folding is the process of converting all words to lowercase. The application of case folding aims to standardize all character types in the text, making it easier to remove specific characters or unwanted words in this research.

3.2.4 Tokenization

Tokenization is used to separate each text into small units called tokens. Tokenization divides a

sentence into separate words.

3.2.5 Language Detection

After balancing the data and performing punctuation removal, case folding, and tokenization, language detection will be performed on the data to determine whether it is in English or Indonesian. This language detection is useful for the system to determine whether the text will be processed in English or Indonesian. The Python library used for language detection, called Langdetect, is used. The Langdetect library is a port of Google's language-detection library and supports 55 languages, including English and Indonesian.

3.2.6 Stopword Removal

Stopwords are commonly used words that are often ignored in text processing. The removal of stopwords aims to reduce the number of words in a document, which can impact the speed and performance of NLP tasks. Stopword removal uses the Natural Language Toolkit (NLTK), a Python library for NLP tasks.

3.2.7 Stemming and Lemmatization

Stemming and lemmatization are preprocessing steps to convert words with affixes into their base forms. Although they have the same purpose, there are differences in the implementation of stemming and lemmatization. In stemming, the conversion is done by cutting or removing word affixes without considering the context. On the other hand, lemmatization involves a more complex process that includes using a language dictionary to find a word's base form (root). Lemmatization transforms a word into its base form while considering the context of the word. From this, it can be concluded that lemmatization has better accuracy than stemming but requires more complex processing, resulting in a longer processing time.

In this study, stemming is used for Indonesian texts. Stemming is applied using Sastrawi Stemmer, a stemmer library that transforms a word into its base form. Sastrawi Stemmer implements an algorithm based on Nazief and Adriani Stemmer, which is then enhanced with the Confix Stripping algorithm, Enhanced Confix Stripping (ECS) algorithm, and Modified ECS (Rosid et al., 2020). In this study, the results shown by the stemming process for Indonesian texts are already good, so the stemming process is sufficient without the need for lemmatization.

In contrast, lemmatization is used for English in this study. The stemming results for English were not satisfying after conducting experiments using stemming. There were many issues with the base forms of words generated by the stemming process, resulting in meaningless words that would affect the accuracy. Therefore, lemmatization was chosen to process English text. The NLTK WordNet lemmatizer library was used for its implementation. Lemmatization using this lemmatizer has shown good results.

The detailed preprocessing steps can be seen in Figure 2.

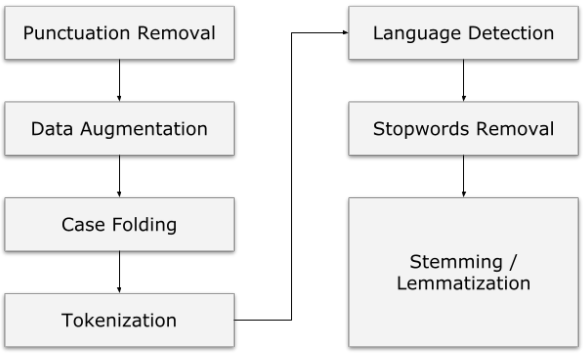

Fig.2: Preprocessing Steps

The first step of preprocessing is punctuation removal to eliminate punctuation marks in the text. After that, data augmentation is performed to balance the number of data before proceeding to the next step. Then the process followed by case folding to convert the text into lowercase. Next, tokenization is carried out to split the text into separate words or tokens. Then, language detection is performed to determine whether the text is in English or Indonesian. Based on the language detection result, stopwords removal is applied to eliminate stopwords and focus on more meaningful words. Finally, stemming (for Indonesian text) and lemmatization (for English text) are performed to convert words with affixes into their base form. The output of stemming or lemmatization is then used in the next step, Word Embedding.

3.3 Word Embedding

Word Embedding is a technique to represent words or sentences in text with real number vectors depicted in a vector space model (Ghourabi et al., 2020). In simple terms, word embedding is a technique where each word is converted into a numerical form (vector) representing that word. Vectors capture various characteristics of the word that relate to the overall text. These characteristics include semantic relationships between words, definitions, context, etc.

The problem addressed in this stage is that machine learning models can only handle numerical data and not directly process textual data. Therefore, word embedding is performed to transform text data into a format that can be understood and interpreted by machine learning algorithms.

In this research, the Word2Vec library will be used for word embedding. Word2Vec is a word embedding method used to represent words as vectors of length N. Word2Vec uses a neural network to obtain these vectors. The architecture of Word2Vec consists of three layers: input, projection (hidden layer), and output. The input of Word2Vec is a one-hot encoded vector with a length equal to the number of unique words in the training data. The steps of the word embedding process applied can be seen in Figure 3.

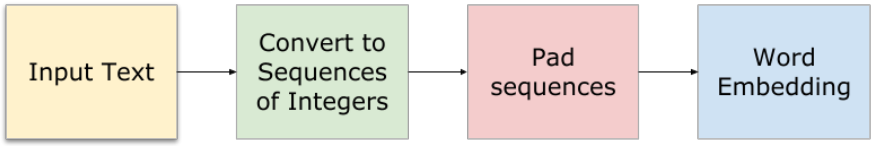

Fig.3: Word Embedding Process

First, the input data, which consists of a collection of N texts, is processed by vectorizing the text corpus. It is done by converting each text into a sequence of integers, where each integer represents the index of a token in a dictionary. The result is a list of sequences with variable sizes since each text may have a different length. Next, padding and truncating are applied to make the sequences of equal length. Padding is performed to fill the sequences with zeros until they reach the maximum length of a sentence defined beforehand. At the same time, truncating is done to cut off any extra tokens exceeding the maximum sentence length. After that, the data is trained with a word embedding layer, which will generate the input for the LSTM deep learning model.

3.4 LSTM Model

After completing the preprocessing steps, the next step is to create a spam detection model using the LSTM algorithm. This research uses a single model to detect spam in English and Indonesian. It allows the system to detect spam in texts written in either English or Indonesian.

LSTM (Long Short-Term Memory) is a Deep Learning algorithm variant of Recurrent Neural Network (RNN). RNNs face difficulties in remembering long-term dependencies due to the vanishing gradients problem. LSTM addresses this issue by allowing the network to have long-term memory. LSTM improves upon the RNN algorithm by solving the vanishing gradient problem by adding a cell state, which enables the network to remember or forget data as needed. The LSTM architecture

generally consists of three layers: the input layer, the LSTM layer (LSTM cell), and the output layer.

The input layer receives and passes the input data to the next layer. The input layer gets a sequence of input data that can be a sequence of words, characters, or other sequential data. In the input layer, an embedding layer is responsible for learning and representing the semantic meaning of words or tokens in a dense vector space.

The LSTM layer, or LSTM cell, is the fundamental building block of the LSTM architecture where the actual processing and learning occur. It has several important components, including the cell state (c_t) and hidden state (h_t). The cell state, also known as the memory cell, is responsible for storing and maintaining long-term information. It allows the LSTM to remember important information from previous time steps. The hidden state represents the output of the LSTM cell at each time step and captures the information learned by the LSTM cell based on the input sequence at that time step.

The cell state contains a structure called cell gates. Cell gates consist of three parts: the input gate (i_t), the forget gate (f_t), and the output gate (o_t). The input gate controls which input data is relevant and should be stored. The forget gate controls the previous hidden state that will be stored in the cell memory for the current hidden state. The output gate is used to calculate the output data of the network based on the cell memory state. (Poomka et al., 2019). For each time step (t), these three gates determine how to update the current memory cell (c_t) and the current hidden state (h_t). The architecture of the LSTM algorithm can be seen in Figure 4.

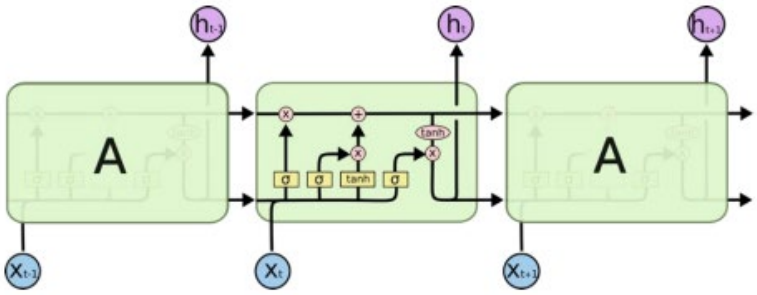

Fig.4: Architecture of Long Short-Term Memory (LSTM)

To calculate the values of the input gate, forget gate, and output gate, equations (1), (2), and (3) can be used, respectively. And to update the values of the memory cell and hidden state, equations (4) and (5) can be used, respectively. The following are the equations used in the LSTM algorithm:

- $i_t = \sigma(W_{xi} \cdot x_t + W_{hi} \cdot h_{t-1} + b_i)$ (1)
- $f_t = \sigma(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f)$ (2)
- $o_t = \sigma(W_{xo} \cdot x_t + W_{ho} \cdot h_{t-1} + b_o)$ (3)
- $c_t = f_t \odot c_{t-1} + i_t * \check{c}_t$ (4)
- $h_t = o_t \odot tanh(c_t)$ (5)

In the equation, x_t represents the input vector to the LSTM unit, σ is the sigmoid function, tanh is the hyperbolic tangent function, and $\odot$ represents the element-wise product. W represents the weight, and b is the bias vector parameter that needs to be learned during training.

The output layer, or dense layer, is the last layer of LSTM, also known as the fully connected layer. Dense is used to classify the text based on the output from the LSTM layer. This model uses a dense layer with the sigmoid activation function to provide predictions of 0 or 1 for two classes (spam and non-spam). The sigmoid function is a function that returns values between 0 and 1, as seen in the following equation (6):

$$f(x) = \frac{1}{1+e^{-x}} \quad (6)$$

In this research, several parameters are initialized. Firstly, the number of LSTM units or cells is set to 128, indicating the dimensionality of the output space and the memory capacity of the LSTM layer. Next, the dropout rate parameter is set to 0.2, which means that during training, 20% of the input units will be randomly assigned to 0 in each update. Dropout is used as a regularization parameter to prevent overfitting. The model is then compiled with the binary cross-entropy loss function and the Adam optimizer, which aims to minimize the loss value during training. Finally, the batch size is set to 32, and the number of epochs is set to 10.

4. Experiment and Results

This section will discuss the experimental setup and results obtained from testing the proposed model in this paper. First, each dataset will be divided into two parts: the training dataset and the testing dataset. The model will then be trained using the training data. After the training process is completed, the model's performance will be evaluated through the testing process using the testing data to assess the model's effectiveness. The performance of the spam detection model using the LSTM algorithm will be measured. The confusion matrix method is employed to evaluate the performance of the system. The model's performance is evaluated based on accuracy, precision, recall, and F-measure. These experiment steps are conducted for each dataset. In the end, a comparison of the model's testing results on multiple datasets will be presented. This model is implemented in Python 3.9 using the TensorFlow environment and the Keras API.

4.1 Data Splitting

The datasets are separately used for training and testing to observe the comparison of the training and testing results. In this research, each dataset is divided into two parts, with 80% for training data and 20% for testing data. This division takes 80% of each spam and non-spam data from each dataset for training data and 20% for testing data. The detailed dataset splitting can be seen in Table 3.

Table 3: Dataset Splitting

No.	Dataset Name	Number of Training Data (80%)			Number of Testing Data (20%)		
		Total	Spam	Non- Spam	Total	Spam	Non-Spam
1	RIDA Web Form Spam Dataset	3932	2949	983	983	737	246
2	SpamAssassin Email Dataset	2400	400	2000	600	100	500
3	SMS Spam Bahasa Indonesia Dataset	914	459	455	229	115	114
4	UCI SMS Spam Collection Dataset	4459	597	3862	1115	149	966

4.2 Evaluation Measures

To evaluate the performance of the proposed model, standard metrics for classification tasks are used, including Accuracy, Precision, Recall, and F1-Score through the Confusion Matrix. After training the model, testing will be conducted on the created model. The predictions generated by the system will be compared with the ground truth. From this comparison, the total values of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) will be obtained.

- Confusion Matrix consists of the following values:
  - True Positives (TP): When the actual class is 1 (spam), and the predicted result is also 1 (spam)
  - True Negatives (TN): When the actual class is 0 (non-spam), and the predicted result is also 0 (non-spam)
  - False Positives (FP): When the actual class is 0 (non-spam), but the predicted result is 1 (spam)
  - False Negatives (FN): When the actual class is 1 (spam), but the predicted result is 0 (non-spam)

The confusion matrix can be seen in Figure 5.

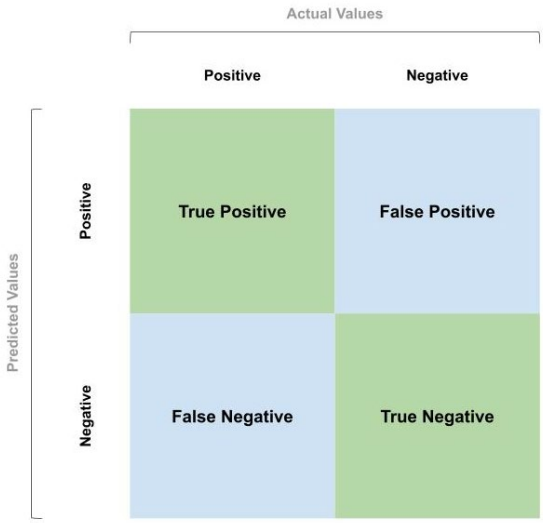

Fig.5: Confusion Matrix

- Accuracy is the proportion of correctly predicted data out of the total predicted data. Equation (7) is used to calculate the accuracy value.

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (7)$$

- Precision is comparing True Positive (spam) with the overall positive predictions. Equation (8) is used to calculate the precision value.

$$Precision = \frac{TP}{TP+FP} \quad (8)$$

- Recall is the proportion of positive cases identified correctly. Equation (9) is used to calculate the recall value.

$$Recall = \frac{TP}{TP+FN} \quad (9)$$

- F1-score is the evaluation result calculated by combining the precision and recall values. Equation (10) is used to calculate the F1-score.

$$F1-Score = 2 \cdot \frac{precision \cdot recall}{precision+recall} \quad (10)$$

4.3 Training Results

For the training process, the text classification into spam and non-spam using this model experimented on several training datasets, including one main dataset, the RIDA Web Form Spam Dataset, and three public datasets, namely the SpamAssassin Email Dataset, Spam SMS Bahasa Indonesia Dataset, and SMS Spam Collection Dataset. The training results of the model trained on the RIDA Web Form Spam Dataset can be seen in the accuracy and loss curves in Figure 6 and Figure 7, respectively.

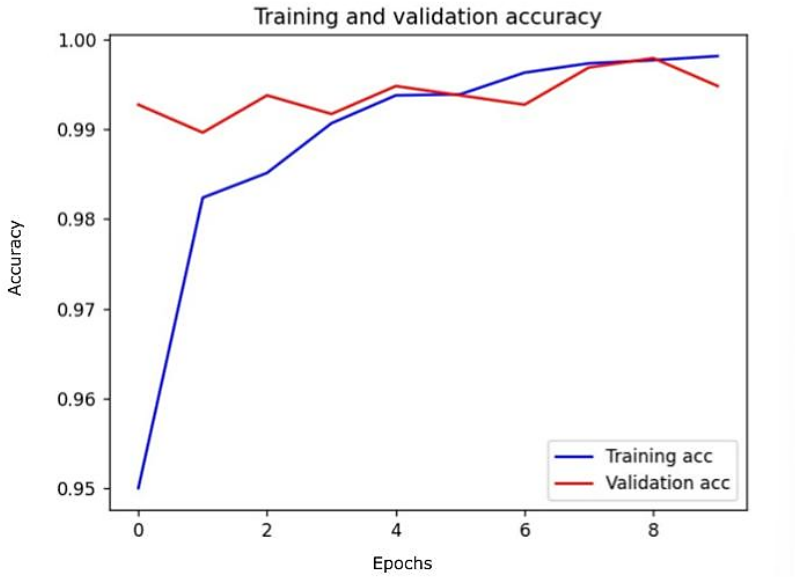

Fig.6: Training and Validation Accuracy Curve

Figure 6 shows the accuracy curve during the training and validation process. High accuracy values indicate how well the model classifies spam and non-spam. It can be observed that in almost every epoch, the training and validation accuracy values increase. The training accuracy and validation accuracy values are also very close, indicating that the built model is in optimal condition.

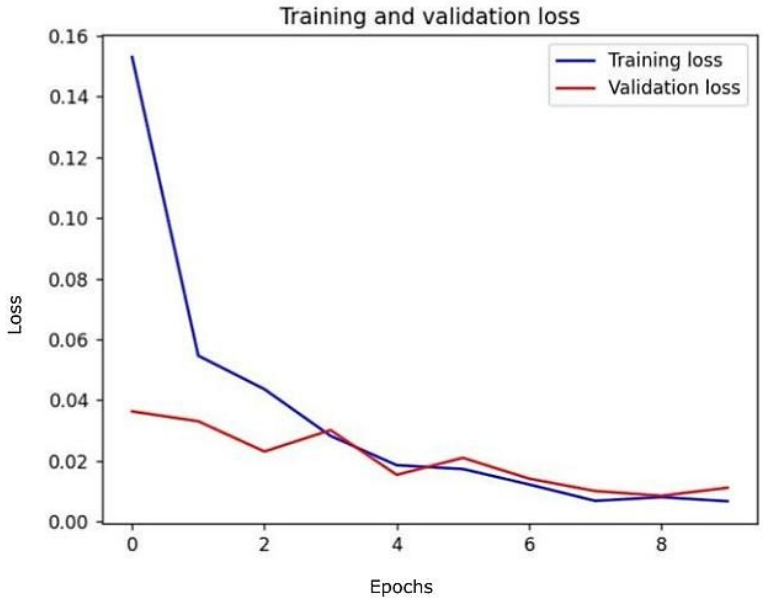

Fig.7: Training and Validation Loss Curve

Meanwhile, Figure 7 shows the loss curve in the training and validation process. The loss value shows how much error was produced by the model during the training and validation. A smaller loss value indicates better model performance in classifying spam and non-spam. It can be seen from the curve that both the training loss and validation loss decrease with each epoch. The training loss and validation loss values are not significantly different, indicating that the model is also in optimal condition without experiencing overfitting or underfitting. Figure 6 shows that the lowest loss is achieved at epoch 10 with a value of 0.0067, and the highest accuracy is achieved at epoch 10 with a value of 0.9982 or 99.82%.

Fig.8: The Accuracy and Loss Values for Each Epoch.

4.4 Testing Results

The testing process for this model was performed on the testing datasets of the RIDA Web Form Spam Dataset, SpamAssassin Email Dataset, Spam SMS Bahasa Indonesia Dataset, and SMS Spam Collection Dataset. Based on the conducted experiments, the LSTM model designed in this research has demonstrated good performance in spam detection. The evaluation results of testing the spam detection model using LSTM on multiple datasets are presented in the following Table 4.

Table 4: Evaluation Result

No.	Datasets	Accuracy	Precision	Recall	F1-Score
1	RIDA Web Form Spam Dataset	82.4%	90.1%	82.4%	84.7%
2	SpamAssassin Email Dataset	85.3%	83.2%	85.3%	83.3%
3	SMS Spam Bahasa Indonesia Dataset	96.1%	94.4%	98.0%	96.2%
4	UCI SMS Spam Collection Dataset	62.9%	81.4%	62.9%	68.7%

Table 4 lists each dataset's corresponding evaluation metrics: Accuracy, Precision, Recall, and F1-Score. The values for each metric are presented in percentage form.

In the first dataset, RIDA Web Form Spam Dataset, the model achieved an accuracy of 82.4%. It demonstrated a high precision of 90.1%, indicating a low rate of false positives. The recall value of 82.4% indicates that the model effectively identified a significant portion of the actual spam messages. The F1-score, which combines precision and recall, was calculated at 84.7%.

The second dataset, SpamAssassin Email Dataset, achieved an accuracy of 85.3% and a precision of 83.2%, indicating a balanced performance in identifying spam messages. The recall value of 85.3% suggests that the model effectively captured the majority of spam messages. The F1-score was calculated at 83.3%, indicating good overall performance.

For the third dataset, SMS Spam Bahasa Indonesia Dataset, the model achieved an impressive accuracy of 96.1%. It demonstrated a high precision of 94.4%, indicating a low rate of false positives. The recall value of 98.0% indicates that the model effectively identified almost all of the actual spam messages in this dataset. The F1-score was calculated at 96.2%, reflecting a strong overall performance.

Lastly, in the UCI SMS Spam Collection Dataset, the model achieved an accuracy of 62.9%. It demonstrated a precision of 81.4%, indicating a reasonable ability to identify spam messages. The recall value of 62.9% suggests that the model successfully captured some of the spam messages in this dataset. The F1-score was calculated at 68.7%, indicating a moderate overall performance.

Overall, the model's performance varied across different datasets. The SMS Spam Bahasa Indonesia dataset achieved the highest values in all evaluation metrics, including accuracy, precision, recall, and F1-score. On the other hand, the UCI SMS Spam Collection dataset had the lowest values in each evaluation metric. These results demonstrate the effectiveness of the spam detection model in distinguishing between spam and non-spam messages across various datasets. Based on the analysis, these different outcomes could be influenced by multiple factors.

The results comparison between LSTM and other methods, specifically K-Nearest Neighbors, are presented in Table 5.

Table 5: Comparison of Result Between LSTM and K-Nearest Neighbors (KNN)

No.	Datasets	LSTM				KNN			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
1	RIDA Web Form Spam Dataset	**82.4%**	90.1%	**82.4%**	**84.7%**	74.4%	**99.0%**	66.5%	79.8%
2	SpamAssassin Email Dataset	**85.3%**	**83.2%**	85.3%	**83.3%**	71.5%	72.1%	**91,8%**	80,9%
3	SMS Spam Bahasa Indonesia Dataset	**96.1%**	94.4%	**98.0%**	**96.2%**	47.0%	**99.8%**	46.9%	63.6%
4	UCI SMS Spam Collection Dataset	62.9%	81.4%	62.9%	68.7%	**89.7%**	**99,9%**	**89.49%**	**94.4%**

The results show that the performance of LSTM and KNN algorithms varies depending on the dataset. In some cases, LSTM performs better in accuracy, precision, recall, and F1-score, while in other cases, KNN outperforms LSTM.

4.5 Discussion and Analysis

Based on the evaluation results, it can be observed that the SMS Spam Bahasa Indonesia Dataset has relatively high evaluation scores for accuracy, precision, recall, and F1-Score. On the other hand, the RIDA Web Form Spam Dataset and the SpamAssassin Email Dataset have moderate evaluation scores. Lastly, the UCI SMS Spam Collection Dataset has relatively low evaluation scores compared to the other three datasets. The variations in performance across the different datasets can be attributed to several factors.

One factor that may affect this LSTM model's performance is the imbalanced data. Imbalanced data can significantly impact a model's accuracy, precision, recall, and F1-score. The imbalance in the datasets may have influenced the performance, as the model might be biased towards the majority class and struggle to identify the minority class properly. In addressing this issue, data augmentation techniques were applied to balance the datasets before training the model. Experiments have been conducted to test the impact of data augmentation techniques in this model. The results show that the applied data augmentation technique can increase the model's accuracy compared to the accuracy before the data augmentation was applied. However, although data augmentation techniques were applied to help balance the class distribution by generating synthetic samples for the minority class, the effectiveness of data augmentation depends on various factors. If the class imbalance is extremely severe, generating a few additional synthetic samples through data augmentation may not be sufficient to address the imbalance adequately. The quality and diversity of the augmented samples can impact the model's ability to learn and generalize well. Suppose the augmented samples are too similar to the original minority class samples or do not adequately capture the underlying patterns. In that case, the model may still struggle to classify instances from the minority class accurately.

The SMS Spam Bahasa Indonesia Dataset has the highest evaluation results among all datasets. It

has a balanced class distribution, where the number of spam and non-spam instances is relatively equal. This proportional representation enables the model to learn effectively from both classes, resulting in better performance in spam detection. On the other hand, The RIDA Web Form Spam Dataset and SpamAssassin Email Dataset have achieved moderate evaluation results. It may be caused by the imbalanced class distributions observed in these datasets, meaning there is a significant difference in the number of spam and non-spam instances. Imbalance data can impact the model's ability to learn and generalize effectively, leading to relatively lower performance metrics. But overall, the model still performs well classifying spam on those datasets. Lastly, The UCI SMS Spam Collection Dataset has obtained relatively lower results than the other datasets. It appears to suffer from an extremely severe class imbalance issue.

The other factor that can affect the model's performance is dataset characteristics. Each dataset has its unique characteristics, such as the distribution of spam and non-spam instances, the quality of data, and the complexity of patterns exhibited by spam messages. For example, SMS Spam Bahasa Indonesia Dataset may contain informative and discriminative features that help distinguish spam messages from non-spam messages accurately. These features could capture specific characteristics or patterns unique to SMS spam in the Indonesian language, enabling the model to make more accurate predictions. On the other hand, RIDA Web Form Spam Dataset, SpamAssassin Email Dataset, and UCI SMS Spam Collection Dataset might contain more complex and diverse spam patterns that are challenging to detect accurately. Some spam messages may employ sophisticated techniques to evade detection, making it more difficult for the model to classify them correctly.

Based on the comparison results between LSTM and KNN, it can be observed that the LSTM algorithm generally achieves higher accuracy and F1-score compared to KNN in most of the datasets. This indicates that LSTM is more effective in accurately classifying spam and non-spam instances. However, considering other factors, such as precision and recall, is also important. For example, in some cases, KNN may have higher precision, indicating a lower rate of false positives. Still, it may have lower recall, implying it may miss some spam instances. The results suggest that the LSTM algorithm shows promise in spam detection tasks, particularly for the datasets mentioned.

Returning to the objective of this research, which is to develop a spam detection model using LSTM for messages received through the web form of a government ministry's website, we can focus on the RIDA Web Form Spam Dataset. The testing results of the model on the RIDA Web Form Spam Dataset indicate that the model performs well in classifying spam and non-spam messages in both English and Indonesian languages received through the web form. Implementing this model on the ministry's website will assist in categorizing incoming messages as either spam or non-spam. Implementing this model can minimize the potential impact of spam-related losses on the ministry's website. Because government ministry websites serve as crucial communication platforms for the ministry and the public, rendering them vulnerable to attacks.

The LSTM approach applied in the spam detection model in this research has advantages compared to previous studies. This model can detect spam in two languages, namely English and Indonesian. Additionally, the model effectively classifies spam and non-spam across various datasets. The LSTM approach is used to develop this spam detection model in this research due to its several strengths in the case of spam detection. LSTM is very efficient at capturing long-range dependencies and sequential patterns in data which is particularly useful for detecting patterns in spam messages that may span multiple words or phrases. LSTM also can understand the context of a message throughout the processing of a sequence and make informed decisions about its spam or non-spam classification. LSTM can handle variable-length sequences, which is important in spam detection, where messages can vary in length.

However, in addition to that, there are also some limitations associated with LSTM. LSTM requires more training data to learn effectively. This research applies a data augmentation technique by utilizing existing data to generate new synthetic data, specifically the Synonym Replacement

technique, to mitigate this limitation. In addition, LSTM models tend to experience overfitting issues. Overfitting is a condition where the network performs well on the training data but obtains suboptimal results when using other data. This research also addresses overfitting by increasing the dataset size through data augmentation. Additionally, the technique of dropout is employed to handle overfitting issues. During the neural network training, the dropout technique randomly deactivates neurons in the hidden layers (along with their connections).

The spam detection model developed in this research is highly applicable for implementation on other websites. This model will be effective, especially for websites with web forms and receiving messages written in English, Indonesian, or both languages. However, it is important to note that additional training data from the specific website is needed to enhance the model's knowledge and improve its effectiveness.

5. Conclusion

This paper aims to create a spam detection model for messages received through the web form of a ministry website using the Deep Learning approach with the Long Short-Term Memory (LSTM) algorithm. The model aims to detect spam in both English and Indonesian languages. Spam detection for two or more different languages is a challenging problem due to the differences in language structure.

The main contribution of this paper is this research successfully developed a model for spam detection in two languages, English and Indonesian, using the LSTM algorithm on the ministry website. This model has been tested on several datasets, and evaluations have been conducted on the testing results. This model effectively distinguishes between spam and non-spam messages, achieving an accuracy parameter value of 82.4% in the main dataset, namely RIDA Web Form Spam Dataset, 85.3% in the SpamAssassin Email Dataset, 96.1% in the Spam SMS Bahasa Indonesia dataset, and 62.9% in the UCI SMS Spam Collection Dataset. These varied results are influenced by the balance between the number of spam and non-spam data in the datasets. Implementing this model on the ministry's website helps classify incoming messages as spam or non-spam, making it easier for the ministry to follow up on them. And most importantly, implementing this model can minimize the potential impact of spam-related losses on the ministry's website.

A limitation of this research is that the developed model is ineffective when the dataset used has an extreme imbalance between the spam and non-spam classes. The applied data augmentation technique can handle data imbalance issues that are not too significant. Still, if the imbalance is extremely large, the data augmentation may not work effectively, resulting in poor spam classification results.

Based on the limitations of this research, there are several suggestions and recommendations for future studies. First, future research can explore and implement more advanced techniques or algorithms specifically designed to handle extreme class imbalances. Second, future studies can consider alternative algorithms for spam detection. While LSTM has shown good results in spam detection, future studies could explore ensemble methods of LSTM or combine multiple models to leverage their strengths and mitigate their weaknesses.

References

Almeida, T., & Hidalgo, J. (2012). *SMS Spam Collection.* UCI Machine Learning Repository.

Bhat, S. Y., & Abulaish, M. (2014). Using communities against deception in online social networks. *Computer Fraud & Security 2014*, 8-16. doi:10.1016/s1361-3723(14)70462-2

Bindu, P., Mishra, R., & Thilagam, P. S. (2018). Discovering spammer communities in twitter. *Journal of Intelligent Information Systems, 51*(3), pp 503-527. doi:10.1007/s10844-017-0494-z

Chandra, A., & Khatri, S. K. (2019). Spam SMS Filtering using Recurrent Neural Network and Long Short Term Memory. *2019 4th International Conference on Information Systems and Computer Networks (ISCON).* India: IEEE. doi:10.1109/ISCON47742.2019.9036269

Garg, P., & Girdhar, N. (2020, December). A Systematic Review on Spam Filtering Techniques based on Natural Language Processing Framework. *11th International Conference on Cloud Computing, Data Science & Engineering (Confluence).* IEEE. doi:10.1109/confluence51648.2021.9377042

Ghourabi, A., Mahmood, M. A., & Alzubi, Q. M. (2020). A Hiybrid CNN-LSTM Model for SMS Spam Detection in Arabic and English Messages. *Future Internet 2020, 12*. doi:10.3390/fi12090156

Gupta, H., Jamal, M. S., Madisetty, S., & Desarkar, M. S. (2018). A Framework for Real-Time Spam Detection in Twitter. *10th International Conference on Communication Systems and Networks.* Bangalore, India: COMSNETS. doi:10.1109/COMSNETS.2018.8328222

Jain, G., Sharma, M., & Agarwal, B. (2018). Spam Detection on Social Media Using Semantic Convolutional Neural Network. *International Journal of Knowledge Discovery in Bioinformatics, 8*(1). doi:10.4018/IJKDB.2018010102

Murti, Y. S., & Naveen, P. (2024). Machine Learning Algorithms for Phishing Email Detection. *Journal of Logistics, Informatics and Service Science, Vol. 10 (2023) No.2*, 249-261. doi:10.33168/JLISS.2023.0217

Poomka, P., Pongsena, W., Kerdprasop, N., & Kerdprasop, K. (2019). SMS Spam Detection Based on Long Short Term Memory and Gated Recurrent Unit. *International Journal of Future Computer and Communication, 8*(1), 11-15. doi:10.18178/ijfcc.2019.8.1.532

Rachmat, A., & Lukito, Y. (2017). Deteksi Komentar Spam Bahasa Indonesia Pada Instagram Menggunakan Naive Bayes. *ULTIMATICS, ISSN 2085-4552, IX*(1).

Rahmi, F., & Wibisono, Y. (2016). Aplikasi SMS SPam Filtering pada Android Menggunakan Algoritma Naive Bayes.

Rao, S., Verma, A. K., & Bhatia, T. (2021, December 30). A review on social spam detection: Challenges, open issues, and future directions. *Expert Systems with Applications, 186*. doi:10.1016/j.eswa.2021.115742

Rodrigues, A. P., Fernandes, R., A, A., B, A., Shetty, A., K, A., . . . Shafi, R. M. (2022). Real-Time Twitter Spam Detection and Sentiment Analysis using Machine Learning and Deep Learning Techniques. *Computational Intelligence and Neuroscience*. doi:10.1155/2022/5211949

Rosid, M. A., Fitrani, A. S., Astutik, I. R., Mulloh, N. I., & Gozali, H. A. (2020). Improving Text Preprocessing For Student Complaint Document Classification Using Sastrawi. *IOP Conference Series: Materials Science and Engineering 874*. doi:doi:10.1088/1757-899X/874/1/012017

Saumya, S., & Singh, J. P. (2018). Detection of spam reviews: a sentiment analysis approach. *CSI Transactions on ICT, 6*, 137-148. doi:10.1007/s40012-018-0193-0

Shorten, C., Khoshgoftaar, T. M., & Furht, b. (2021). Text Data Augmentation for Deep Learning. *Journal of Big Data 8*, 101. doi:https://doi.org/10.1186/s40537-021-00492-0

Singh, V., Varshney, A., Akhtar, S. S., Vijay, D., & Shrivastava, M. (2019). Aggression Detection on Social Media Text Using Deep Neural Networks. *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, 43-50. doi:10.18653/v1/W18-5106

Sinhmar, A., Malhotra, V., Yadav, R., & Kumar, M. (2022). Spam Detection Using Genetic Algorithm Optimized LSTM Model. *Computer Networks and Inventive Communication Technologies, Lecture Notes on Data Engineering and Communications Technologies, 75*. doi:10.1007/978-981-16-3728-5_5

Šolić, K., Ilakovac, V., & Galić, D. (2013). Amount of spam in global free e mail services - simulation. *Technical gazette, Vol. 20 No.2*, 311-313.

Spam Statistics and Facts. (2023). Retrieved May 2, 2023, from SpamLaws.com: https://www.spamlaws.com/spam-stats.html

Vernanda, Y., Hansun, S., & Kristanda, M. B. (2020). Indonesian language email spam detection using N-gram and Naive Bayes algorithm. *Bulletin of Electrical Engineering and Informatics, Vol. 9 No.5, ISSN 2302-9285*, pp 2012-2019. doi:10.11591/eei.v9i5.2444

Wu, T., Liu, S., Zhang, J., & Xiang, Y. (2017). Twitter Spam Detection based on Deep Learning. *ACSW '17: Proceedings of the Australasian Computer Science Week Multiconference*, (pp. 1-8). Geelong, Australia. doi:10.1145/3014812.3014815