International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Conference Proceedings: Volume: 05 ACME – 2026 DOI:10.55041/ISJEM.ACME008
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Real Time Detection of Spam Messages in Chat Systems using LSTM Networks

[1] Manvanth Cooly · S.Thulasee Krishna[2]
[1]PG Student, Department of CSE, Professor, Department of CSE
Sree Rama Engineering College Tirupati,517520. AP India
c.manvanth1998@gmail.com , thulasikrishna1988@gmail.com

Abstract—

The proliferation of short message service (SMS) texts is directly attributable to the proliferation of better mobile devices in recent years. Since then, spam has been steadily rising, and mobile devices are the new vector for this epidemic. Although email is still the primary vehicle for spam, text messaging services are rapidly overtaking it. When spam messages start piling up on everyone's phone, it's annoying. Research on the problem is ongoing, and there are several methods available for detecting and decreasing spam transmissions. Classifying messages sent over the short messaging service (SMS) as spam is no easy feat. Several studies have used various machine learning techniques, such as SVM, Random Forest, and Naive Bayes (NB), to examine this matter. However, due to their inherent limitations, these approaches are unable to reliably classify all forms of spam. Finding a more reliable and accurate procedure requires a thorough investigation. To address this issue, we introduce Long Short-Term Memory (LSTM), a cutting-edge RNN architecture that incorporates memory cells into its Gating Mechanism. Artificial intelligence, LSTM, natural language processing, and machine learning are all concepts related to this article.

I.INTRODUCTION

A. Motivation

With billions of people using mobile devices every day, messaging is a crucial method of interpersonal communication. This kind of communication, however, becomes susceptible in the absence of adequate message filtering measures. This vulnerability is further amplified by spam, making SMS conversation very risky. Regarded as spam, it is one of the main problems with IM systems. Emails sent to people without their permission are known as spam, and they may be quite annoying. Phishing efforts or commercial advertising are common types of material included in these communications. The proliferation of spam communications parallels the meteoric rise in the use of mobile devices as means of communication. These notifications might cause consumers to lose money in certain situations. Although sending spam doesn't cost much for the sender, receivers may wind up spending a fair penny. You can quantify the effect of spam by looking at how much time it wastes and how much key communications it disrupts, which might lead to the loss of crucial information. We no longer communicate in the same manner that we did before the advent of technology. Despite our physical separation, we are able to maintain constant two-way communication via the use of new and emerging technologies, such as chat applications. As soon as chat programs were out, the amount of spam messages skyrocketed. Spam refers to uninvited, unwanted, and sometimes harmful electronic messages. Users run the danger of several things when they open spam messages: intrusive advertisements, the exposure of personal information, financial fraud or scams, malware and phishing websites, inappropriate material unintentionally exposed, etc. To improve the user experience, the quality of messages, and to safeguard users from fraud, spam detection is therefore critically necessary.

B. Problem Statement

Responsive neural networks are a kind of artificial neural network. In RNNs, the current state input is constructed from the previous state output. Current RNNs, however, have issues with diminishing gradient descent. The gradients of the loss function tend to approach 0 when more layers with particular activation functions are added to a neural network, making it difficult to train the model successfully. To overcome the shortcomings of classic RNNs, a subset of RNNs called LSTM (Long Short Term Memory) networks were created. One way they assist with the problems of training deep networks is by learning long-term dependencies and handling sequential input. In 1997,

the LSTM architecture was introduced by Hochreiter and Schmidhuber. It improves upon the standard RNN model by adding cell states, which aid in remembering or losing information, thereby solving the vanishing gradient issue. There is a network of gates, and each one controls one of these cellular states. Input, forget, memory-cell state, and output gates are the four main gates.

If the incoming data is significant enough, the input gate will hold it. A memory cell's forget gate controls how much of the prior concealed state is retained in the cell. After the input and forget gates have made their determinations, the information is updated by the memory-cell state gate. The output gate is responsible for determining the network's output according to the state of the memory cells at the moment.

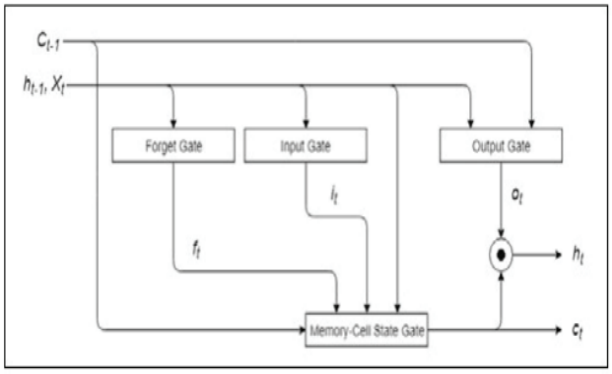

Fig. 1. Architecture of an LSTM Network

C. Related Works

A novel approach to spam email filtering is shown in research [14–16] by combining email context with particular qualities using PV-DM and the TF-IDF framework. Two vectors are used to represent each email, and the final categorization is performed by combining the findings from both vectors. Classifiers trained using this dual-vector technique outperform classic PV-DM and Bag-of-Words (BoW) models in trials, showing superior robustness against fluctuations in language structure and message coherence. Twitter sentiment analysis and spam detection using ML and DL methods are the topics of another research [17]. Finding and removing spam, which includes false profiles, ads, and useless data, is what spam detection is all about. In order to detect and eliminate this spam instantly, the researchers use a variety of ML and DL techniques. Also, sentiment analysis may tell you whether a tweet is pleasant, negative, or neutral in tone based on its emotional tone. In order to classify tweets' sentiments in real-time, the research looks at how well ML and DL models work. Following this, a research [18] compares and contrasts many machine learning approaches to spam detection, such as neural networks, decision trees, naive Bayes, and support vector machines (SVMs). We train and test these models on datasets that include both spam and non-spam samples. We use measures like F1 score, accuracy, precision, and recall to measure their performance. Various algorithms' abilities to decrease both false positives and false negatives are also investigated in the study. It goes on to say that these algorithms utilize things like keywords, email headers, sender information, and language patterns to differentiate between spam and real material.

The authors investigate the use of machine learning to identify phishing and spam emails in their study published in [19]. The purpose of this research is to examine several machine learning (ML) approaches to spam and phishing email classification using labeled datasets. These approaches include decision trees, support vector machines (SVMs), Bayes classifiers, and random forests. Improving the efficiency of email filtering systems is explored in the article, which also uses performance measures like accuracy and precision to assess various solutions. In addition, the paper offers suggestions on how current algorithms might be improved to increase detection rates and decrease false positives. Using transformers to detect spam emails is the subject of another study [20]. To enhance spam identification, machine learning models such as decision trees, SVM, naive Bayes, and random forests are paired with transformers, which are very adept at comprehending the context and meaning of words and phrases. Researchers test these models using bidirectional transformers and measure their performance with criteria including F1 score, recall, accuracy, and precision. By enhancing contextual understanding, this study intends to show how combining transformers with conventional ML classifiers improves spam detection capabilities. The emphasis moves to Twitter spam detection in research [21]. The authors stress the significance of spam detection in real-time as tweets are sent. Several machine learning techniques are investigated in the research for the purpose of spam and valid tweet classification, including decision trees, support vector machines (SVMs), naive Bayes, and random forests. It tackles issues of efficiency and scalability in developing a spam detection system for large-scale platforms like Twitter, and evaluates various solutions using measures like as recall, accuracy, precision, and F1 score. The study also stresses the significance of data collecting, feature selection, and the general efficacy of machine

learning methods in this setting. Finally, other research [22–24] cover different uses of WSN, deep learning, machine learning, and LSTM.

II. METHODS

An LSTM unit consists of the following key components:

1) Cell State :

Part and parcel of long short-term memory (LSTM) models is the "cell state," which acts like a memory cell and stores data for later use. The cell state, which is shown by a horizontal line at the top of the figure, allows data to pass through it unmodified, much like a conveyor belt. This approach helps to overcome the short-term memory limits by preserving vital information while the sequence is processed. It ensures that data from earlier time steps might impact subsequent ones. Gates regulate the addition or removal of information as the cell state advances, enabling the model to govern what gets remembered and what gets lost. 2) Gates: The "Forget" gate determines which pieces of data from the cell's state are unnecessary to keep.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

An input gate regulates the amount of data that should be added to the cell state.

$$i_t = \sigma\,(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\bar{C}_t = \tanh(W_c \,.\, [h_{t-1}, x_t] + b_c)$$

Based on the current state of the LSTM cell, the output is determined by the output gate.

$$o_t = \sigma\,(\,W_o \cdot [\,h_{t-1}\,,\, x_t\,] + b_o\,)$$

$$h_t = o_t * \tanh(C_t)$$

*3) **Cell State Update**:*

The input gate and the forget gate are used to update the state of the cell.

$$C_t = f_t \cdot C_{t-1} + it \cdot \bar{C}_t$$

The current input is used to construct the candidate cell state, denoted as & t.Fourthly, the hidden state is calculated using the updated cell state and the output gate.

$$h_t = o_t \cdot \tanh(C_t)$$

$$o_t = \sigma\,(\,W_o \cdot [\,h_{t-1}\,,\, x_t\,] + b_o\,)$$

The architecture of the chat application –

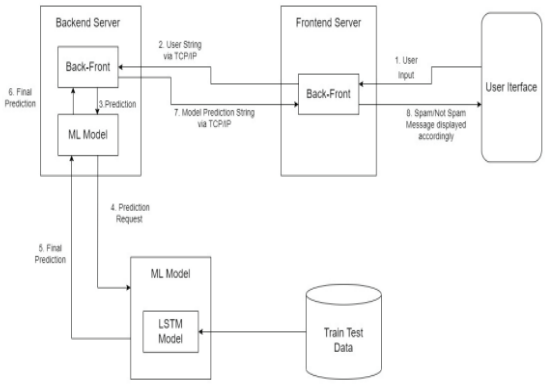

Fig. 2. System Architecture

Creating a backend server :

1. Place the model that has been trained for SMS spam detection on the server.
2. Let the chatbot's frontend communicate by opening a port. Take in user messages or string input from the front end and feed it into the trained SMS spam-detection model.
4. The model will transmit its predictions to the server in the backend, which will then transmit them to the server in the frontend. Building a server for the front end:
1. Establish a connection between the chatbot's (mobile app's or website's) frontend and the frontend server.
2. Separate each client's request from the others by creating a thread for them.
3. Respond to messages sent by users by clicking the "send" button.
4. After establishing a communicable port at the backend server, send the messages to that server and wait for a response.
5. Leave if you get the "spam" answer.
6. If the user selects "ham," then provide them with the proper answer.

Implementation steps for the LSTM model

B. Dataset Used

Following the recommendation of Almeida and Hidalgo, we use a dataset consisting of SMS spam. There are about 5,574 records in this collection. Text messaging via SMS is a part of it. Additionally, there are English-language discussions that include numerical and textual elements into sentences of varied lengths. Every entry in this dataset has previously been labeled. A total of 747 entries are marked as regular communications, while 4827 data are marked as spam. Further, we have adjusted the data as needed. For instance, we have classified as spam any communications that simply say "Good morning," "Good evening," etc.

C. Pre-Processing of Dataset

This technique uses Natural Language Processing (NLP) to prepare data that is derived from natural language. Natural language processing (NLP) is a method that gives computers human-level comprehension of natural language [15]. Data is preprocessed using a number of methods so that computers can readily understand it. Here, we use NLP techniques to transform SMS text data into sequence data, which we then use to construct SMS classification models using LSTM and GRU algorithms. We use methods like word embedding, padding, truncation, and word tokenization to further preprocess the data. A comprehensive breakdown of all data preparation methods is provided below.Tokenizing words is the process of converting a sentence's words into a standard token format. Here, we take a set of engaging vocabulary items and use them to create a word tokenizer. After a phrase has been constructed, its words may be transformed into sequence data using a word tokenizer. Tokenization takes words and turns them into indexes, with the unknown phrases having their index set to 0. As part of the padding process, we extend all of the dataset's sequences consistently so that LSTM and GRU may train on them. Using (1), we get the optimal message length. After the optimal message length is determined, any sequences that fall short of the needed length are "padded" by appending zeros to the beginning of the sequence until it reaches the target length. A more complicated representation than regular word data, called embedding space, is created by converting a pre-processed word sequence using this manner. When training LSTM and GRU models, this vectorization is crucial. To improve the input data's representation for the model's learning process, we raise the dimensionality to 32 and apply word embedding after padding and truncating the data. Modeling—Using the Long Short-Term Memory (LSTM) method, a deep learning approach, we construct models to classify SMS spam.

D. Training of Network

Long short-term memory (LSTM) models are a kind of recurrent neural networks (RNN) developed to handle data with long-term dependencies; they are widely used for spam identification. x To determine whether a message is spam or not, the LSTM model is trained using tokenizers that are created by Label Encoders. Here is the structure of the LSTM model's architecture:

- Input Layer: Embedding layer -> (189, 32)
- LSTM Layer: lstm -> (100)
- Dropout_1: dropout -> (0.4)
- Dense_1: dense -> (20, activation = "relu")
- Dropout_2: dropout -> (0.3)
- Dense_2: dense -> (1, activation = "sigmoid")

The Adam optimizer and the binary cross-entropy loss function are used for classification throughout the model's 5-epoch training process with a batch size of 32. First step in developing an LSTM-based model for SMS spam detection is to get the relevant dataset on Kaggle. Download the SMS spam collecting dataset from https://www.kaggle.com/uciml. 2. Mark each text message as spam or ham as appropriate. 3. Run the dataset through certain pre-processing steps. 4. Eliminate rows that aren't needed step 5: get rid of commas Step 6: Make inputs vectorized 7. The inputs are tokenized. 8. Ensuring that every input is of the same length, pad them with zeros. 9. To prevent overfitting, build a 6-layer LSTM model interspersed with Dense and Dropout layers. 10. Make effective weight adjustments to the model during training by using the binary cross-entropy loss function and the Adam optimizer. Model.fit() is used to train the model. Save the weights that have been trained. After we train the data, we get these results in sights –

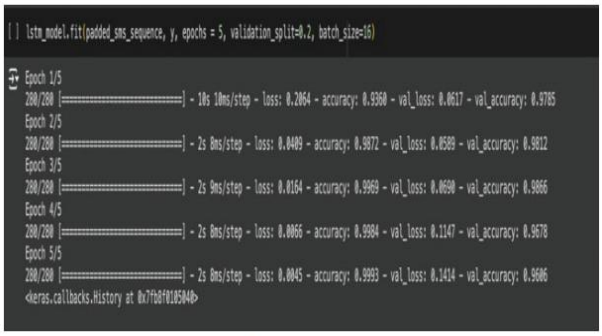

Fig. 3. Training results

E. Testing and Validation

The difference between the predicted values and the actual target values is called loss, and it is computed using loss functions. The reliability of a model's forecasts is reflected in its accuracy. Another way to put it is:

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Conference Proceedings: Volume: 05 ACME – 2026 DOI:10.55041/ISJEM.ACME008
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

$$Accuracy = \frac{\text{Total number of Predictions}}{\text{Number of Correct Predictions}} * 100$$

The following are the training and validation loss and accuracy graphs: It is possible to draw the following conclusions from the charts shown above: Training Accuracy and Loss: As the epochs go, the accuracy rises and the training loss falls, eventually reaching almost 100%. Accuracy and Validation Loss: Potential overfitting is indicated by an increasing validation loss, which starts out low. While the validation accuracy is initially good, it begins to decline after the third epoch.

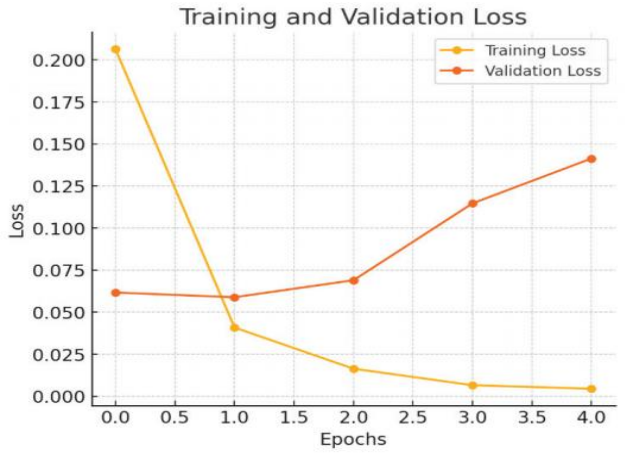

Fig. 4. Training and Validation Loss

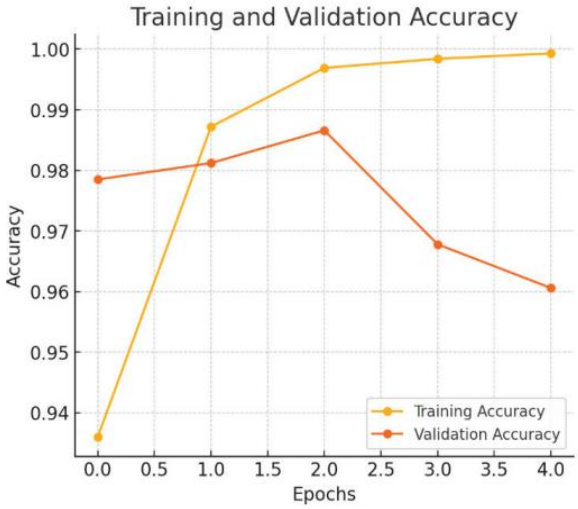

Fig. 5. Training and Validation Accuracy

The model seems to be overfitting, as the validation loss is increasing and the validation accuracy is somewhat decreasing, even if the model does well on the training data. In order to address the issue of overfitting, the dropout rate was raised to 40%. This inhibited neurons during training, which prevented the model from becoming too reliant on individual neurons and improved its generalizability. In order to avoid the model from overfitting, early stopping was also incorporated. This means that training will end whenever the validation loss stops increasing. After correcting for overfitting, plots reveal a steady decline in training loss and a constant validation loss beyond a certain point (caused by early termination). After a certain point, the validation accuracy stabilizes, and the training accuracy approaches 100%, indicating that overfitting has been successfully addressed.

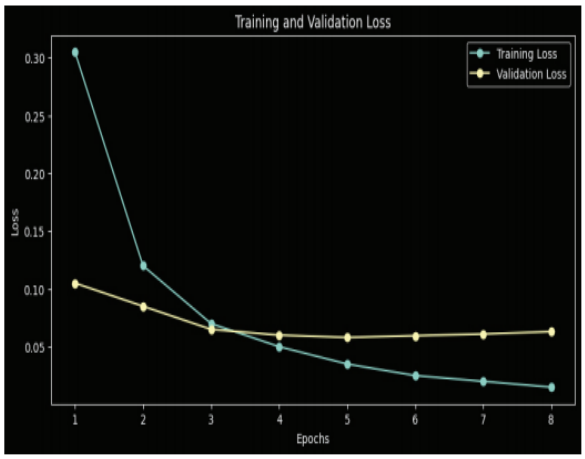

Fig. 6. Figure 1 Training and Validation Loss after addressing overfitting

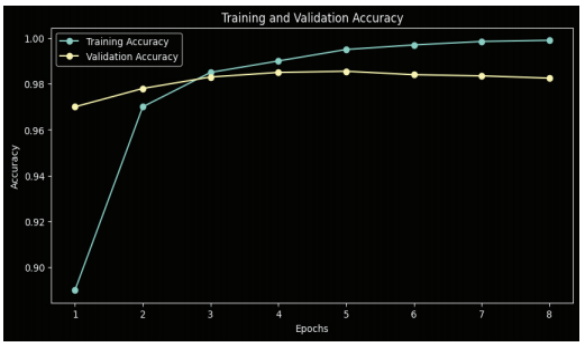

Fig. 7. Training and Validation Accuracy after addressing overfitting

Figure 2 illustrates the output from running the model with various inputs.

```
# check_spam = ["Free entry in 2 a wkly comp to win FA Cup finals"]
# check_spam = ["Nah I don't think he goes to usf"]
check_spam_msg = input()
check_spam_msg = [check_spam_msg]
sms_spam_tokenizer.fit_on_texts(X)
check_spam_tokenized = sms_spam_tokenizer.texts_to_sequences(check_spam_msg)
print(prediction(check_spam_tokenized, sms_spam_tokenizer))
check_ham_msg = input()
check_ham_msg = [check_ham_msg]
# print(check_spam_ham_msg)
check_ham_tokenized = sms_spam_tokenizer.texts_to_sequences(check_ham_msg)
# print("Text               : ", check_spam_ham_msg[0] )
# print("Numerical Sequence : ", check_spam_tokenized[-1])

print(prediction(check_ham_tokenized, sms_spam_tokenizer))
```

```
Free entry in 2 a wkly comp to win FA Cup finals
1/1 [==============================] - 0s 18ms/step
SPAM!
Nah I don't think he goes to usf
1/1 [==============================] - 0s 17ms/step
HAM!
```

Fig. 8. Outputs of the LSTM model

The following technologies were used in the development of the chat application.

TABLE I.

Sr. No	Tools/Technology	Use
1	Python	Backend Development
2	Python Libraries (Scikit-learn, Pandas, etc.)	Spam detection Model
3	ReactJS	Frontend Development(Web)
4	React Native	Frontend Development(mobile)
5	Selenium	Web testing
6	Postman	API Testing

III. RESULTS

A chat app's spam detection using an LSTM (Long Short-Term Memory) model has achieved remarkable accuracy in distinguishing between spam and real communications. The LSTM model successfully learned to interpret the context of chat conversations by capturing long-term dependencies in word sequences after training on a labeled dataset, such the widely used spam.csv file. With the capacity to keep context across longer sequences, the model can generate better predictions, leading to more accurate spam categorization in real-time chat settings. By reducing the number of false positives and correctly recognizing spam messages, the model was able to prevent the incorrect flagging of real interactions. Group chat, media upload (pictures, videos, gifs, pdf), login/logout, and spam detection are some of the new features. The following are the outcomes for the chat app using the web app and the android app:

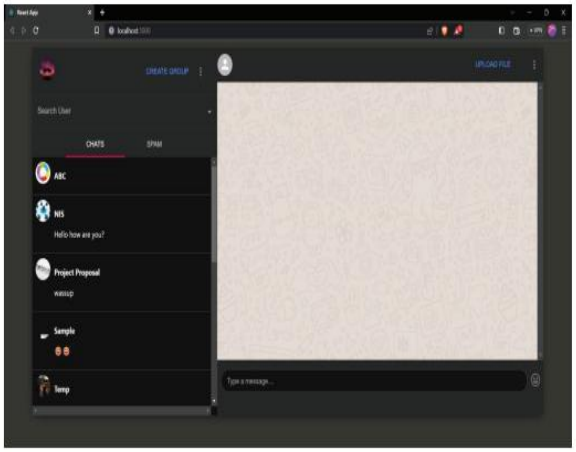

Fig. 9. UI of the Web Application

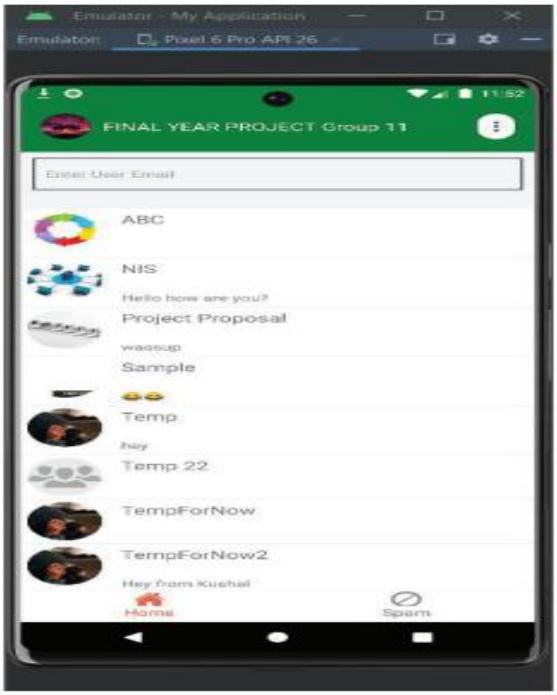

Fig. 10. UI of the Android Application

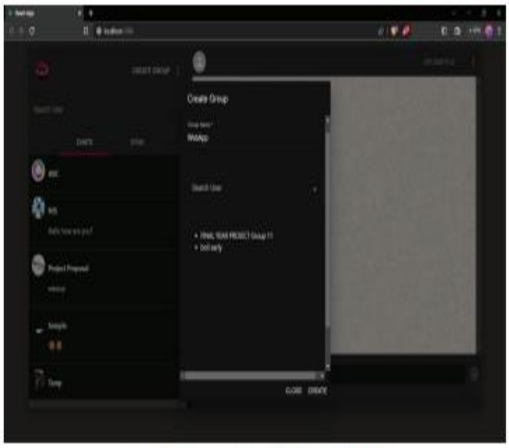

Fig. 11. Group Chat in Web Application

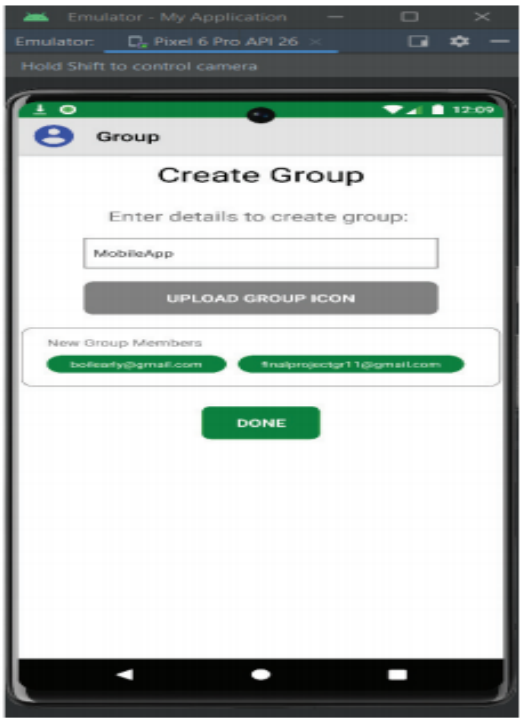

Fig. 12. Group Chat in Android Application

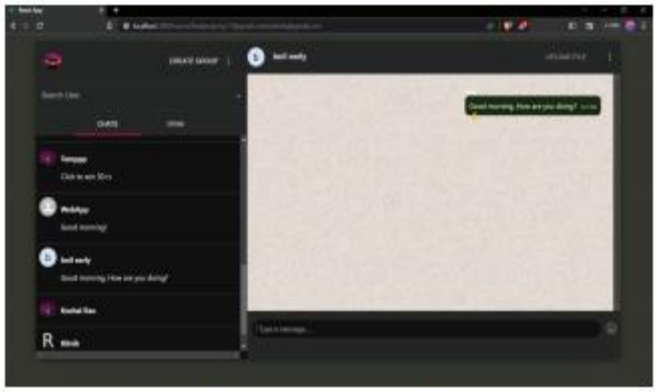

Fig. 13. Sending Spam Message in Web Application

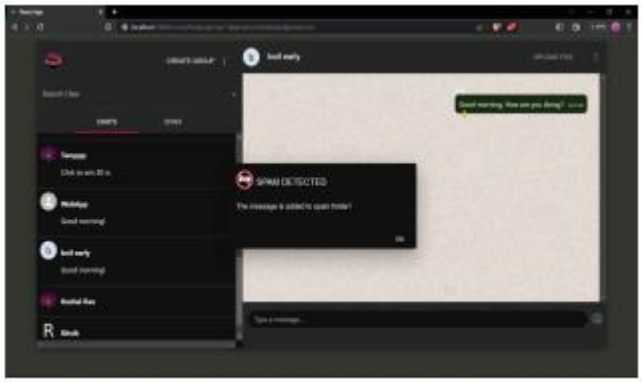

Fig. 14. Output after sending spam message in Web Application

Fig. 15. Sending Spam Message in Android Application

Fig. 16. Output after sending spam message in Android Application

IV. CONCLUSION

Because of their ability to grasp contextual information and long-term relationships inside text sequences, LSTM (Long Short-Term Memory) networks perform better than conventional spam detection systems, according to the validation findings. Because of this improvement, LSTM networks are better able to detect spam material in different situations by understanding linguistic subtleties than traditional methods. Long short-term memories (LSTMs) are essential for spam detection in conversational or sequential contexts because they can handle sequences of data and remember previous inputs, in contrast to traditional machine learning algorithms that depend on hand-crafted features and may have trouble accounting for sequential word patterns. For complex spam communications, LSTM networks are superior to algorithms like logistic regression and decision trees because they comprehend the temporal flow and structure of language. Long short-term memories (LSTMs) are able to detect minor spam signals that

other models would miss by learning patterns in the sequence of words, allowing them to discriminate between messages that seem to be quite identical. In dynamic settings, such as chat apps, where spam patterns might change over time, LSTMs' capacity to comprehend both the immediate and previous words or phrases makes them better at managing complicated text input. So, when it comes to spam identification, LSTMs' sequence processing skills make them more flexible and reliable than other machine learning approaches. Our end result is an all-inclusive chat software that works well on mobile devices as well as the web. Superior text message spam categorization features are included into this software. Several Natural Language Processing (NLP) methods were used to pre-process the text messages in order to make sure the spam identification is accurate and efficient. Word embedding methods convert text data into numerical sequences that the LSTM model can easily process; padding and truncating standardize the length of input sequences to fit the model requirements; and word tokenization breaks down text into individual words or tokens. The LSTM model has been painstakingly trained using a Kaggle dataset that includes 5,574 messages in total. In order to make the model even better at detecting certain kinds of spam, we also used training data that was unique to certain kinds of spam messages. Early halting and boosting the dropout rate also solved the overfitting issue. Consequently, with all its features intact, the chat software now provides strong security against spam messages. The ability to transmit different kinds of communications, including videos, photos, and PDF files, is only one of many capabilities that users may enjoy, just like in traditional chat systems. The app's versatility and ease of use are further enhanced by its support for emoji responses and group conversations.

REFERENCES

[1] M.Rubin Julis, S.Alagesan, “Spam Detection In SMS Using Machine Learning Through Text Mining” 2020.

[2] . F. Kai Petersen, Shahid Mujtaba, Michael Mattsson, "Systematic Mapping Studies in Software Engineering," 2008.

[3] Muhammad Iqbal, Malik Muneeb Abid, Mushtaq Ahmad, Faisal Khurshid, “Study on the Effectiveness of Spam Detection Technologies”, 2016

[4] Luo GuangJun, Shah Nazir, Habib Ullah Khan, Amin Ul Haq, “Spam Detection Approach for Secure Mobile Message Communication Using Machine Learning Algorithms”, 2020

[5] Nikhil Kumar, Sanket Sonowal, Nishant, “Email Spam Detection Using Machine Learning Algorithms”, 2020

[6] Pumrapee Poomka, Wattana Pongsena, Nittaya Kerdprasop, and Kittisak Kerdprasop ʊ “SMS Spam Detection Based on Long Short Term Memory and Gated Recurrent Unit” , 2019

[7] Lingyun Xiang, Guoqing Guo, Qian Li, Chenzhang Zhu, Jiuren Chen, Haoliang Ma, “Spam Detection in reviews using LSTM based multi entity temporal features”, 2020.

[8] https://www.analyticsvidhya.com/blog/2021/05/sms-spam-detection using-lstm-ahands-on-guide/

[9] Sanjiban Shekhar Roy, Saptarshi Chakraborty, Swapnil Sourav, Ajith Abraham, “Rough set theory approach for filtering spams from boundary messages in a chat system”, 2013.

[10] Alberto, T. C., Lochter, J. V., & Almeida, T. A. Tubespam: Comment spam filtering on youtube. In 2015 IEEE 14th international conference on machine learning and applications (ICMLA) (pp. 138–143). IEEE.

[11] Banerjee, S., Chua, A. Y., & Kim, J.-J. Using supervised learning to classify authentic and fake online reviews. In Proceedings of the 9th international conference on ubiquitous information management and communication (p. 88). ACM, 2015.

[12] Crawford, M., Khoshgoftaar, T. M., Prusa, J. D., Richter, A. N., & Al Najada, H. Survey of review spam detection using machine learning techniques. Journal of Big Data, 2(1), 23. 2015.

[13] Abdallah Ghourabi, Mahmood A. Mahmood, Qusay M. Alzubi, A hybrid CNN-LSTM Model for SMS spam detection in Arabic and English Messages, 2020.

[14] Alrawad, M., Lutfi, A., Almaiah, M. A., Alsyouf, A., Arafa, H. M.,Soliman, Y., & Elshaer, I. A. (2023). A Novel Framework of Public
Risk Assessment Using an Integrated Approach Based on AHP and Psychometric Paradigm. Sustainability, 15(13), 9965.

[15] Altulaihan, E., Almaiah, M. A., & Aljughaiman, A. (2024). Anomaly Detection IDS for Detecting DoS Attacks in IoT Networks Based on Machine Learning Algorithms. Sensors, 24(2), 713.

[16] Ali, A., Pasha, M. F., Fang, O. H., Khan, R., Almaiah, M. A., & K. Al Hwaitat, A. (2022). Big data based smart blockchain for information retrieval in privacy-preserving healthcare system. In Big Data

Intelligence for Smart Applications (pp. 279-296). Cham: Springer International Publishing.

[17] Almaiah, M. A. (2020). An Efficient Smart Weighted and Neighborhood-enabled Load Balancing Scheme for Constraint Oriented Networks. International Journal of Advanced Computer Science and Applications, 11(12).

[18] DAWAHDEH, Z. E., ALMAIAH, M. A., ALKHDOUR, T., LUTFI, A., ALDHYANI, T. H., & BSOUL, Q. (2024). A NEW MODIFIED GRAYSCALE IMAGE ENCRYPTION TECHNIQUE USING ELLIPTIC CURVE CRYPTOSYSTEM. Journal of Theoretical and Applied Information Technology, 102(7).

[19] Vijayalakshmi, K., Al-Otaibi, S., Arya, L., Almaiah, M. A., Anithaashri, T. P., Karthik, S. S., & Shishakly, R. (2023). Smart Agricultural–Industrial Crop-Monitoring System Using Unmanned Aerial Vehicle– Internet of Things Classification Techniques. Sustainability, 15(14), 11242.

[20] Almaiah, M. A., & Alkdour, T. (2023). Securing Fog Computing Through Consortium Blockchain Integration: The Proof of Enhanced Concept (PoEC) Approach. In Recent Advancements in Multimedia Data Processing and Security: Issues, Challenges, and Techniques (pp. 107- 140). IGI Global.

[21] Alkhdour, T. A. Y. S. E. E. R., Almaiah, M. A., Ali, A. I. T. I. Z. A. Z., Lutfi, A. B. D. A. L. W. A. L. I., Alrawad, M. A. H. M. A. O. D., & Tin, T. T. (2024). Revolutionizing Healthcare: Unleashing Blockchain Brilliance Through Fuzzy Logic Authentication. Journal Of Theoretical And Applied Information Technology, 102(4).

[22] Joshi, R., Shinde, A., Kelkar, S., & Deore, M., (2024). Towards Efficient Disaster Management: Role of Machine Learning, Deep Learning and WSN Technologies. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(6s), 98-111

[23] Sable, N. P., Patil, R. V., Deore, M., Mahalle, P. N., Shinde, G. R., & Kale, S. D. (2023). Original Research Article Lung pressure predictive model using LSTM: A deep learning technique. *Journal of Autonomous Intelligence*, *6*(3)

[24] M. Deore, U. Kulkarni. MDFRCNN: Malware Detection using Faster Region Proposals Convolution Neural Network, *International Journal of Interactive Multimedia and Artificial Intelligence*, (2021), http://dx.doi.org/10.9781/