NIPES

Journal of Science and Technology Research

Journal homepage: www.nipesjournals.org.ng

Modelling of a Deep Learning Based SMS Spam Detection Application

Osa E.[a] *, Elaigwu. V.O*[b]

[a]*Department of Electrical/Electronic Engineering, Faculty of Engineering, University of Benin, P.M.B. 1154, Benin City, Nigeria.*
[b]*Department of Electrical/Electronic Engineering, Faculty of Engineering, Benson Idahosa University, P.M.B. 1100, Okha, Benin City, Nigeria.*

Article Info

***Keywords*:**
Spam, SMS, Python, Machine Learning, Deep Learning, Model.

Received 04 September 2021
Revised 14 November 2021
Accepted 16 November 2021
Available online 12 December 2021

https://doi.org/10.37933/nipes/3.4.2021.17

https://nipesjournals.org.ng

Abstract

Various techniques for mitigating spam messages have been developed overtime. Methods that involve spam emails have also been applied to Short Message Service domain. Some of these methods are based on Artificial Intelligence. This work is aimed at the modelling and deployment of a spam detection application using Deep Learning algorithms. Spam data repositories were investigated for appropriate data required to create the model. Packages such as Google Colab, Pandas, Seaborn, Matplotlib and Wordcloud were implemented for expository data analysis to gain insight into the data. Packages such as Tensorflow, Python-dotenv, Scikit learn were used to create and evaluate the Deep Learning model. The model was thereafter deployed using Flask, a python library for web development. The web application was used to read SMS messages sent to a Twilio service phone number and messages as ***spam or ham****. The results show an accuracy score of 98.3% hence the developed model is highly reliable.*

1. Introduction

Short Message Service, known as SMS for short is one of the most ubiquitous communication services presently. SMS traffic volumes rose from around 1.46 billion in the year 2000 to about 7.9 trillion in 2012 [1]. Mobile phone users with SMS enabled devices had reached 6.1 billion users by the year 2015 [2]. Short Message Service (SMS) is actually a service for transmitting short length messages of around 160 characters between devices such as mobile phones by utilizing certain standard protocols. It is an alternative means of communication to voice calls in situations where voice communication is not possible or is not desired for communication. However, besides the legitimate use of such a communication service, criminals also take advantage of it for their nefarious activities. SMS is a major conduit for spam messages [3]. Spam refers to any kind of unwanted, unsolicited digital communication that is sent out in bulk. Unsolicited means that the recipient did not grant verifiable permission for the message to be sent. Bulk means that the message sent is part of a larger collection of messages which have virtually the same content [4]. Spam is an issue about consent of the recipient and not necessarily message content. A message is therefore said to be spam only if it is both unsolicited and bulk. Spam is commonly used as a medium for advertising products and services. According to [5], ninety-eight percent of all spam messages sent are a form of products or services advertisement which would apparently seem harmless while the remaining two percent pose the threat to the internet community and results in great losses. Spam includes, phishing mails that steal users' login information and identities, messages from internet fraudsters that mislead internet users to pay money in exchange for great riches or reward which are nonexistent, malspam that intend to make users download malicious web content attached to the messages. However, spam messages do not only appear in emails, they are also present in internet forums, text messages, blog comments and other social media. In 2009, China's three main mobile phone operators (China Telecom, China Mobile Ltd and China Unicom) signed an agreement to combat mobile spam by setting limitations on the number of text messages that could be sent each hour [6]. Spam first spread explosively as emails in the first decade of the 21th century as reflected by the statistical results in [7]. However, due to the fact that SMS is low-cost, bulk-sending, and reliably reaches intended recipients, spam began to gravitate towards this most popular and globally used communication service. In 2017 alone, the world sent 8.3 trillion SMS messages while the number of SMS messages sent monthly is 690 billion proving the importance of the service [8]. The companies hosting the bulk SMS sending service therefore have to continuously improve their spam filtering technology so as to combat spam SMS. SMS spammers tend to use legitimate words to increase the rank of spam messages in spam filters and also use obfuscated words to confuse these spam filters. Machine learning algorithms with its advantages of Natural Language Processing could be implemented to enhance SMS spam detection and filtering.

1.2. Overview of Spam

The history of spam dates back to 1864, before the advent of the internet around 1969, with a telegram sent out en-masse to a group of British politicians. By the 1980s, internet users came together on regional online communities, called bulletin boards (BBSes), that were administrated by hobbyists on their home servers. On a typical BBS, users were able to share files, post notices, and exchange messages online. During heated online exchanges, users would type the word "spam" over and over again to drown each other out. This was done in reference to a Monty Python sketch from the year 1970 in which *a husband and wife eating at a working-class café find that almost everything on the menu contains spam. As the wife argues with the waitress over the preponderance of spam on the menu, a chorus of Vikings drowns out the conversation with a song about spam.* The use of the word "spam" in this context, i.e., loud annoying messaging, caught on much to the chagrin of ***Hormel Foods***, the maker of spam recipe. In 1999, ***Melissa***, the first virus that spread via macro-enabled Word documents attached to emails was let loose upon the digital world. It spread by ransacking victims' contact lists and spamming itself to everyone the contact list of the victim. *Melissa* resulted in $80 million in damages, according to the FBI (Federal Bureau of Investigation). Due to the absence of any anti-spam legislation, professional spammers rose to prominence, including the self-proclaimed "Spam King" Sanford Wallace. Wallace was at one time reckoned the biggest sender of spam emails and social media spam on websites like Myspace and Facebook. It was in the early 2000s that governments the world over began to get serious about regulating spam. Notable among them are all member countries of the European Union and the United Kingdom who have strict legislations in place to restrict spam. Likewise, in 2003 the United States enacted laws cheekily called the CAN-SPAM Act. Despite all the mitigation measures put in place, spam propagation is still an issue.

1.3 Types of Spam

Types of Spam include: Advance-fee scams, Phishing emails, Malspam and Spam on mobile/Android devices [9].

1.4 Spam Text Messages and Phishing

Professional scammers send fake text messages to trick victims into giving personal information such as password, account number, or Social Security number. Once they lay hold on such information, they could gain access to email, bank, or other accounts of victims. Scammers also send fake messages that may state that they have some information about victim's account or transaction procedures. They may impersonate bank authorities stating that they have noticed some suspicious activity on the account of the victim. They may go further to claim there is a problem with payment details and send a fake invoice requesting victims to contact them if they did not authorize the purchase. Other spam messages may install harmful malware on mobile phone of victims thereby stealing personal information without the knowledge of the victims [10].

1.5 Machine Learning

Machine learning is the field of study that provides machines the ability to learn without being explicitly programmed. Machine learning systems can be classified according to the amount and type of supervision they get during training into supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning.

1.5.1 Supervised Learning

In supervised learning, the training data contains both features and label. For example, predicting the price of a car given a set of features. To train the model, we give the systems many examples of cars which include the features and the correct label. Some examples of supervised learning algorithms are k-Nearest Neighbors, Linear Regression, Logistic Regression, Decision Trees and Random Forest, Neural Networks.

1.5.2 Unsupervised Learning

In this type of machine learning, the training data is unlabeled. The system tries to learn without a "supervisor". Here the model tries to group similar data together: a form of clustering. Examples of unsupervised learning algorithms are K-means, DBSCAN, Hierarchical Cluster Analysis (HCA), Principal component analysis (PCA).

1.5.3 Semi-supervised learning

Labeling data is usually time-consuming and costly because sometimes the data could contain plenty of unlabeled data as well as labeled instances. Some algorithms can deal with data that is partially labeled. Most semi-supervised learning algorithms are a combination of unsupervised and supervised algorithms. For example, Deep Belief Networks (DBNs) are based on unsupervised components called Restricted Boltzmann Machines (RBMs) stacked on top of one another. RBMs are trained sequentially in an unsupervised manner and then the whole system is fine-tuned using unsupervised learning techniques.

1.5.4 Reinforcement Learning

In this kind of learning, the learning system called the Agent can observe the environment, select and perform actions, and get rewards in return. It must then learn by itself what the best strategy is (called a policy) to get the most reward over time. A policy defines what action the agent should choose when it is in a given situation.

In addition to these categories, machine learning can also be classified as either a regression problem or a classification problem. In regression, continuous values are predicted, for example, the price of a house, quantity of goods to be sold in the coming month etc. For classification, discrete values are predicted, the model tries to predict the class of a particular feature for example, predicting breast cancer as either malignant or benign, classifying emails as either spam or non-spam. The Spam detection task falls under the natural language processing aspect of machine learning as the messages are text data. It is a binary classification task as we are predicting whether the message is either spam or ham.

1.6 Deep Learning

Deep learning is a class of the broad machine learning family based on artificial neural networks. It is inspired by the structure and function of the brain. Artificial neural network is a mathematical model that changes its structure during learning. In deep learning, a computer model learns from images, text, or sound. These models transform its input data into meaningful outputs, a process that is learned from exposure to known examples of inputs and outputs. Deep learning has experienced huge advancements in the artificial intelligence space and has been instrumental in many applications. Deep neural networks consist of multiple layers of interconnected nodes, each of which uses a progressively more complex deep learning algorithm to extract and identify features and patterns in the data. They then calculate the likelihood or confidence that the object or information can be classified or identified in one or more ways. The input and output layers of a deep neural network are called *visible* layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final identification, classification, or description is calculated. In between the input and output layers are *hidden* layers where the calculations of each previous layer are weighted and refined by progressively more complex algorithms to zero in on the final outcome. This movement of calculations through the network is called *forward propagation.* Another process called *back propagation* identifies errors in calculated predictions, assigns those weights and biases, and pushes them back to previous layers to *train* or refine the model. Together, forward propagation and back propagation allow the network to make predictions about the identity or class of the object while learning from inconsistencies in the outcomes. The result is a system that learns as it works and gets more efficient and accurate over time when processing large amounts of data.

2. Methodology

2.1 Importing Libraries & Data

The first step is to import libraries that we require to execute various codes. Thereafter the data is imported. We treated the ham and spam message classification as a supervised machine learning problem. In a supervised machine learning problem, the inputs and the corresponding outputs are available during the algorithm training phase. During the training phase, the machine learning algorithm statistically learns to find the relationship between input texts and output labels. While testing, inputs are fed to the trained machine learning algorithm which then predicts the expected outputs without knowing the actual outputs. For supervised ham and spam message classification, we needed a dataset that contains both ham and spam messages along with the labels that specify whether a message is a ham or spam. In this paper, we used that open-source spambase dataset from the UCI machine learning repository [11]. To import the above dataset into the work, we used the *read_csv()* method of the Pandas library as shown in appendix.

2.2 Data Exploration and Visualization

We explored the data to identify the target variables and do a bit of preprocessing on the dataset. Before application of machine learning algorithms to a dataset, it is always a good practice to visualize data to identify important data trends. We first plot the distribution of ham and spam messages in our dataset using a Bar chart. The dataset consists of 5569 messages of two classes: spam and ham messages. There are 745 spam messages (shown in Figure 1). The dataset comprises two columns, the message and the label. The label or target variable is what we are trying to predict (i.e., spam or ham).

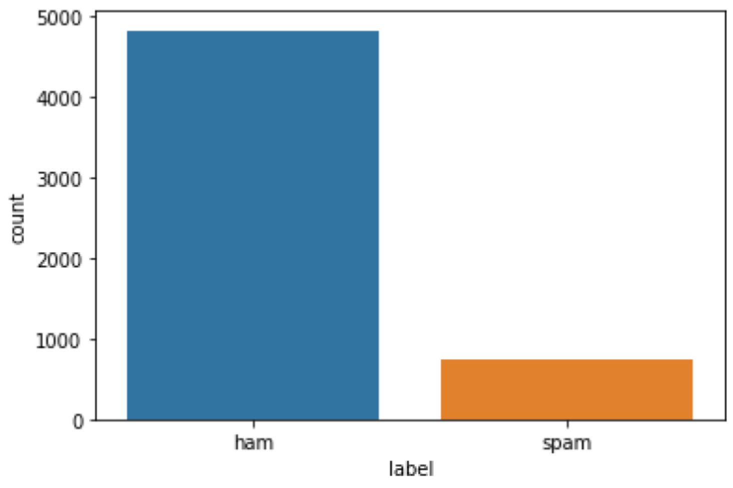


Figure 1: Bar Chart Visualizing Data Distribution.

2.3 Exploratory Data Analysis.

Exploratory Data Analysis is one of the most crucial steps in data science processes. It helps the scientist to understand and gain insight from the data. In this paper, we used Word Cloud, an open source tool for visualizing and analyzing the data. Wordclouds are a simple yet effective method of text visualization. The spam/ham messages were visualized using some wordclouds as shown in Figures 2a and b respectively. The frequency or importance of the words in the data is represented by the size of the text in the image.

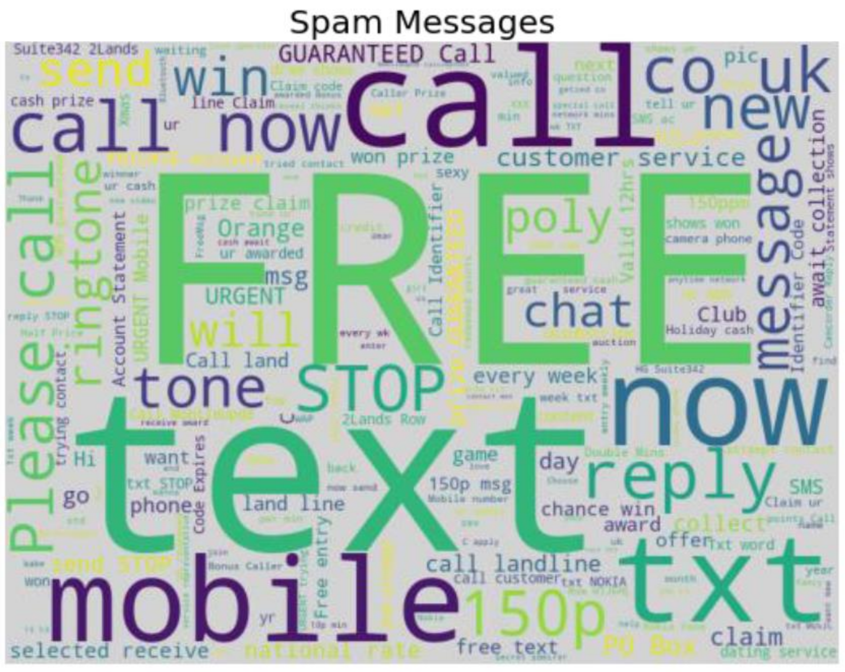


Figure 2a: WordCloud Spam Messages.

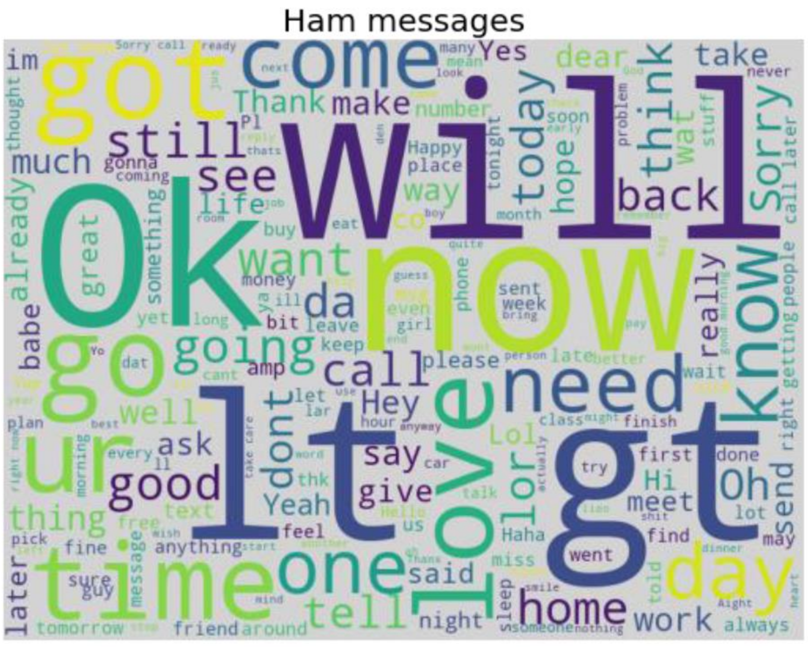


Figure 2b: WordCloud Ham Messages.

The visualizations in Figure 2 shows clearly how the two categories of messages are differentiated.

a.**Spam messages**
It can be noticed that in the spam messages, there is a higher frequency of "spammy" words such as: free, text, stop, call, message, etc. This gave our model better insight into what makes up a spam message.

b. **Ham messages**
It is also noticed that Ham messages contain words such as love, time, home, work, etc.

2.4 Data Preprocessing
The datasets were collected from different sources without processed inputs for the machine learning algorithm. The first preprocessing step employed was cleaning the raw message data. This involved the following steps:

• **Conversion to lower case**: This was carried out to capture the unique tokens and not differentiate between alphabet cases. Lowering the case of the text is essential because the word "SPAM", "Spam", "spam" all means the same thing and add the same value to the sentence but the model may see them different. Words that occur for a very few times or in a large number of documents are not very good for classification hence they were removed.

• **Removing special characters**: Special characters such as "", (),! , were also removed.

• **Removing stop words:** English stop words such as *a, to, i, am, is* were removed as they do not help much in classification procedure. The important words were the tokens other than the stop words. So, we removed the stop words from the messages as they do not represent any differentiating factor.

• **Removing hyperlinks.**

• **Removing numbers.**

• **Removing whitespaces**.

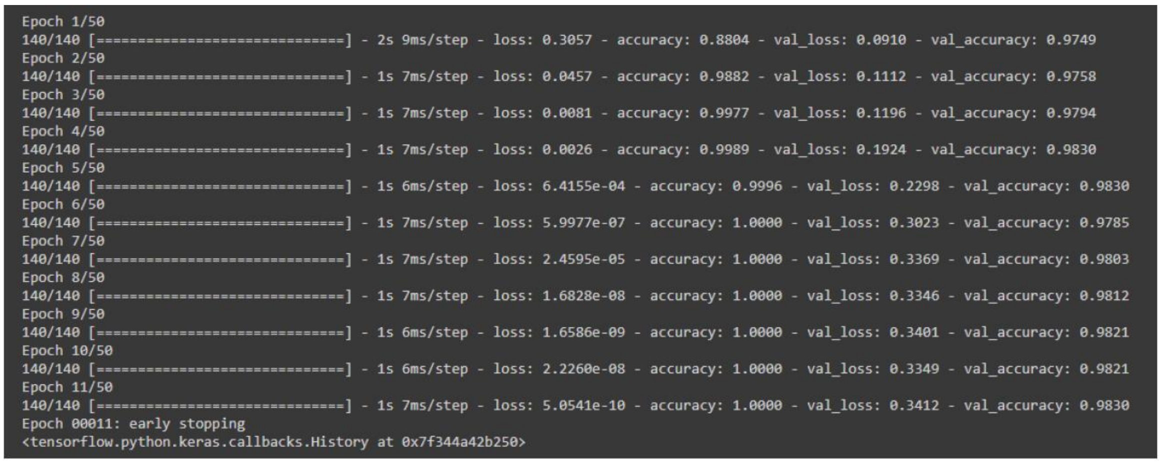
• **Word Stemming:** Stemming algorithms work by removing the end or the beginning of the words, using a list of common prefixes and suffixes that can be found in that language. We implemented stemming on words to bring them to their root form.

• **Word Lemmatization:** Lemmatization involves utilizing the dictionary of a particular language and trying to convert the words back to their base form. In the paper, the NLTK library in python was used to implement the Word Stemming and Word Lemmatization algorithms.

The next step after cleaning the data was to tokenize the cleaned data. Most deep learning algorithms do not work with texts and thus the data needs to be converted to numerical values. Tokenization is the process of splitting text into smaller chunks called tokens that can then be fed into the deep learning model as features.

2.5 Dividing Data into Training and Test Sets
Machine learning algorithms learn from the training set, and to evaluate how well the trained machine learning algorithms performs, predictions are made on the test set. It was therefore necessary to divide our data into the training and test sets. To achieve this, the train_test_split() method from the sklearn.model_selection module was implemented as shown in appendix.

2.6 Training the Deep Learning Model
Very simple neural network architecture was used in this work because of the small data size and to avoid over fitting. The input consists of an embedding layer that takes integers as input, then looks up these integers in an internal dictionary, and returns the associated vectors. Summary of the model is shown in the appendix. The model was thereafter fitted with the training set. The model trained for a number of epochs and stopped when there was no further improvement. This was made possible by the early stopping callback. The model training ran for about 11 or 12 epochs as shown in Figure 3. This variation was due to the stochastic nature of the model and even data splitting.



```
Epoch 1/50
140/140 [==============================] - 2s 9ms/step - loss: 0.3057 - accuracy: 0.8804 - val_loss: 0.0910 - val_accuracy: 0.9749
Epoch 2/50
140/140 [==============================] - 1s 7ms/step - loss: 0.0457 - accuracy: 0.9882 - val_loss: 0.1112 - val_accuracy: 0.9758
Epoch 3/50
140/140 [==============================] - 1s 7ms/step - loss: 0.0081 - accuracy: 0.9977 - val_loss: 0.1196 - val_accuracy: 0.9794
Epoch 4/50
140/140 [==============================] - 1s 7ms/step - loss: 0.0026 - accuracy: 0.9989 - val_loss: 0.1924 - val_accuracy: 0.9830
Epoch 5/50
140/140 [==============================] - 1s 6ms/step - loss: 6.4155e-04 - accuracy: 0.9996 - val_loss: 0.2298 - val_accuracy: 0.9830
Epoch 6/50
140/140 [==============================] - 1s 7ms/step - loss: 5.9977e-07 - accuracy: 1.0000 - val_loss: 0.3023 - val_accuracy: 0.9785
Epoch 7/50
140/140 [==============================] - 1s 7ms/step - loss: 2.4595e-05 - accuracy: 1.0000 - val_loss: 0.3369 - val_accuracy: 0.9803
Epoch 8/50
140/140 [==============================] - 1s 7ms/step - loss: 1.6828e-08 - accuracy: 1.0000 - val_loss: 0.3346 - val_accuracy: 0.9812
Epoch 9/50
140/140 [==============================] - 1s 6ms/step - loss: 1.6586e-09 - accuracy: 1.0000 - val_loss: 0.3401 - val_accuracy: 0.9821
Epoch 10/50
140/140 [==============================] - 1s 6ms/step - loss: 2.2260e-08 - accuracy: 1.0000 - val_loss: 0.3349 - val_accuracy: 0.9821
Epoch 11/50
140/140 [==============================] - 1s 7ms/step - loss: 5.0541e-10 - accuracy: 1.0000 - val_loss: 0.3412 - val_accuracy: 0.9830
Epoch 00011: early stopping
<tensorflow.python.keras.callbacks.History at 0x7f344a42b250>
```

Figure 3: Model Training with Epoch Display.

3. Results and Discussion

3.1 Evaluation of Algorithms

Once predictions were made, the next step was to evaluate the algorithm. Algorithm evaluation involved comparing actual outputs in the test set with the outputs predicted by the algorithm. Performance evaluation of a classification algorithm can be done using accuracy, F1, recall, and confusion matrix as performance metrics. In this paper, confusion matrix and accuracy were used as the performance metrics.

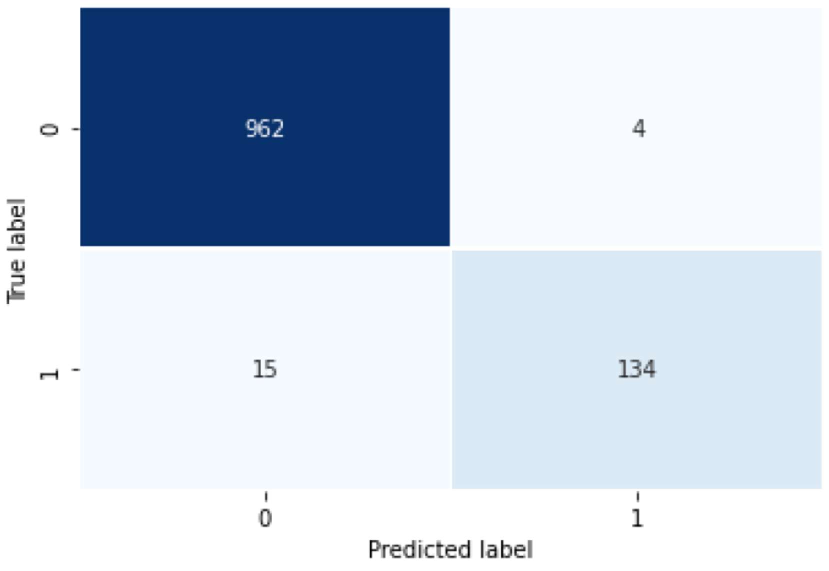
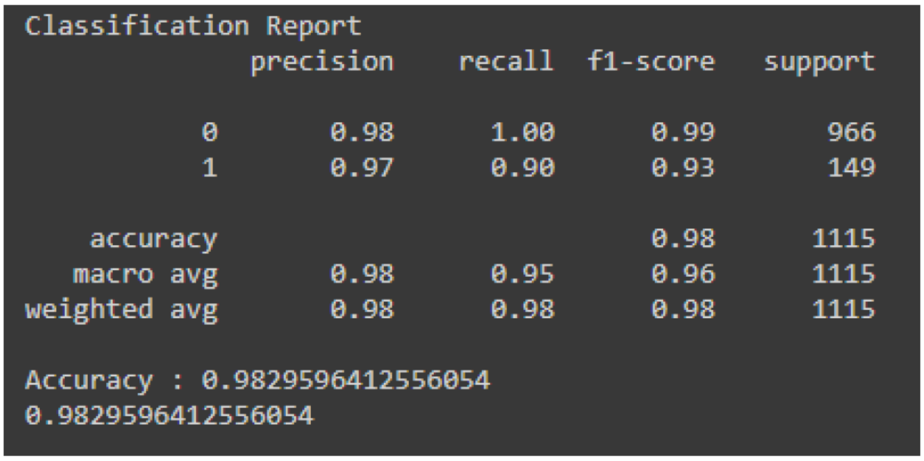


Figure 4: Confusion Matrix.



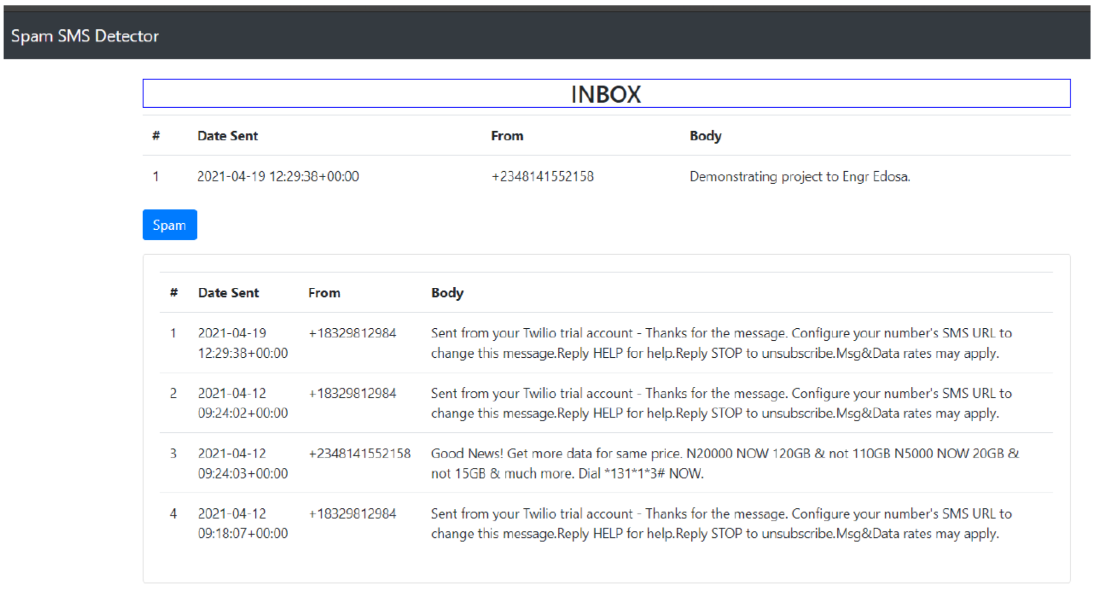
```
Classification Report
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       966
           1       0.97      0.90      0.93       149

    accuracy                           0.98      1115
   macro avg       0.98      0.95      0.96      1115
weighted avg       0.98      0.98      0.98      1115

Accuracy : 0.9829596412556054
0.9829596412556054
```

Figure 5: Training Logs displaying Accuracy of Model.

The final accuracy for the validation set was around 98% (0.983) as seen in the training logs. The output shows that our algorithm achieves an accuracy of 98.3% for spam message detection which is impressive. The model was thereafter saved using the python pickle library.

3.2 Model Deployment

The model was finally deployed using **Flask**, a python library for web development. The web application reads SMS messages sent to a Twilio phone number and classifies them into spam or ham. The final result is displayed in an SMS dashboard as shown in the Figure 6.



#	Date Sent	From	Body
1	2021-04-19 12:29:38+00:00	+2348141552158	Demonstrating project to Engr Edosa.

#	Date Sent	From	Body
1	2021-04-19 12:29:38+00:00	+18329812984	Sent from your Twilio trial account - Thanks for the message. Configure your number's SMS URL to change this message.Reply HELP for help.Reply STOP to unsubscribe.Msg&Data rates may apply.
2	2021-04-12 09:24:02+00:00	+18329812984	Sent from your Twilio trial account - Thanks for the message. Configure your number's SMS URL to change this message.Reply HELP for help.Reply STOP to unsubscribe.Msg&Data rates may apply.
3	2021-04-12 09:24:03+00:00	+2348141552158	Good News! Get more data for same price. N20000 NOW 120GB & not 110GB N5000 NOW 20GB & not 15GB & much more. Dial *131*1*3# NOW.
4	2021-04-12 09:18:07+00:00	+18329812984	Sent from your Twilio trial account - Thanks for the message. Configure your number's SMS URL to change this message.Reply HELP for help.Reply STOP to unsubscribe.Msg&Data rates may apply.

Figure 6: SMS Classification by Web Application.

4. Conclusion

Deep learning as a prominent algorithm is employed in several cyber security areas. In this paper, a double layer Deep Learning Neural Network was applied to classify SMS messages as either Ham or Spam. Findings from result show that the developed model is highly accurate with a percentage of 98.3. For future improvements of this work, it is recommended that the model be integrated in form of APIs (Application Programming Interfaces) for mobile devices. The model can also be extended to accommodate online learning (i.e., continuous learning process while in operation).

References

[1] PortioResearch Worldwide A2P SMS Markets (2014–2017). Understanding and Analysis of Application to-Person Text Messaging Markets Worldwide; Portio Research Limited: Chippenham, UK, 2014.
[2] E. Ezpeleta (2017) Short Messages Spam Filtering Combining Personality Recognition and Sentiment Analysis. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 2017, 175–189.
[3] N. Choudhary and A.K Jain (2017). Towards filtering of SMS spam messages using machine learning based technique. Paper presented at the International Conference on Advanced Informatics for Computing Research.
[4] Spamhaus, (2021).The Spamhaus Project-The Definition of Spam. Available at: https://www.spamhaus.org.
[5] S. Forbes (2002). Web of deception: Misinformation on the Internet: Information Today, Inc. inc, c. s. (2018). Security report.
[6] TechTarget Contributor (2007). What is SMS spam (cell phone spam or short messaging service spam)? Available at: https://searchmobilecomputing.techtarget.com.
[7] V.V. Arutyunov (2013) Spam: Its past, present, and future. Sci. Tech. Inf. Process. 2013, 40, 205–211.
[8] M. Morreale (2017). Daily SMS Mobile Usage Statistics. 2017. Available online: https://www.smseagle.eu/2017/ 03/06/ daily-sms-mobile-statistics.

[9] Malwarebytes, (2021). SPAM: stupid pointless annoying…malware? https://www.malwarebytes.com.
[10] U.S. FEDERAL TRADE COMMISSION Consumer Information (2020). How to Recognize and Report Spam Text Messages. Available at: https://www.consumer.ftc.gov.
[11] Mark Hopkins, E. R., George, F. and Jaap, S. Classifying Email as Spam or Non-Spam. Retrieved from: http://archive.ics.uci.edu/ml/datasets/Spambase.

Appendix

Entire Code Snippets

```
# -*- coding: utf-8 -*-
"""Copy of SPAM_DETECTOR.ipynb
Automatically generated by Colaboratory.
Original file is located at
https://colab.research.google.com/drive/1eCuG4ThRk2InjltJPKbMjfddw6gh_a_s
"""
import tensorflow as tf
print(tf.__version__)
# Commented out IPython magic to ensure Python compatibility.
# import libraries for reading data, exploring and plotting
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import wordcloud
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
# %matplotlib inline
# library for train test split
from sklearn.model_selection import train_test_split
#deep learning libraries for text pre-processing
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
#Modeling
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, GlobalAveragePooling1D, Dense, Dropout, LSTM, Bidirectional,Flatten
url =
'https://raw.githubusercontent.com/ShresthaSudip/SMS_Spam_Detection_DNN_LSTM_BiLSTM/master/SMSSpamCollection'
messages = pd.read_csv(url, sep ='\t',names=["label", "message"])
messages.to_csv("drive/MyDrive/SpamDetector/spam.csv")
messages.head()
messages.describe()
messages.label.value_counts()
data = messages.copy()
sns.countplot(data['label'])
plt.show()
data['label'] = data['label'].map( (Wu et al., 2017) )
data_ham = data[data['label'] == 0].copy()
data_spam = data[data['label'] == 1].copy()
def show_wordcloud(df, title):
text = ' '.join(df.astype(str).tolist())
stopwords = set(wordcloud.STOPWORDS)
fig_wordcloud = wordcloud.WordCloud(stopwords=stopwords,background_color='lightgrey',
colormap='viridis', width=800, height=600).generate(text)
plt.figure(figsize=(10,7), frameon=True)
plt.imshow(fig_wordcloud)
plt.axis('off')
plt.title(title, fontsize=20 )
plt.show()
show_wordcloud(data_spam["message"],"Spam Messages")
show_wordcloud(data_ham["message"], "Ham messages")
"""Preparing Data for Training"""
X = data['message'].values
y = data['label'].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
# prepare tokenizer
```

```
t = Tokenizer()
t.fit_on_texts(X_train)

# integer encode the documents
encoded_train = t.texts_to_sequences(X_train)
encoded_test = t.texts_to_sequences(X_test)
print(encoded_train[0:2])
# pad documents to a max length of 8words
max_length = 8
padded_train = pad_sequences(encoded_train, maxlen=max_length, padding='post')
padded_test = pad_sequences(encoded_test, maxlen=max_length, padding='post')
print(padded_train)

#model setup
vocab_size = len(t.word_index) +1

#define model
model = Sequential()
model.add(Embedding(vocab_size, 24, input_length=max_length))
model.add(Flatten())
model.add(Dense(500, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(100, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

#compile model
model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['accuracy'])
model.summary()
early_stop = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=10)
# fit the model
model.fit(x=padded_train,
y=y_train,
epochs=50,
validation_data=(padded_test, y_test), verbose=1,
callbacks=[early_stop]
(from sklearn metrics import classification report, confusion matrix, accuracy score
def c_report(y_true, y_pred):
print("Classification Report")
print(classification_report(y_true, y_pred))
acc_sc = accuracy_score(y_true, y_pred)
print("Accuracy : "+ str(acc_sc))
return acc_sc
def plot_confusion_matrix(y_true, y_pred):
mtx = confusion_matrix(y_true, y_pred)
sns.heatmap(mtx, annot=True, fmt='d', linewidths=.5,
cmap="Blues", cbar=False)
plt.ylabel('True label')
plt.xlabel('Predicted label')
preds = (model.predict(padded_test) > 0.5).astype("int32")
c_report(y_test, preds)
plot_confusion_matrix(y_test, preds)
model.save("drive/MyDrive/SpamDetector/spam_model")
import pickle
with open('drive/MyDrive/SpamDetector/spam_model/tokenizer.pkl', 'wb') as output:
pickle.dump(t, output, pickle.HIGHEST_PROTOCOL)
"""<h1>Inference</h1>"""
s_model = tf.keras.models.load_model("drive/MyDrive/SpamDetector/spam_model")
with open('drive/MyDrive/SpamDetector/spam_model/tokenizer.pkl', 'rb') as input:
tokenizer = pickle.load(input)
sms = ["Congratulations! you just won 2 million dollars. Send you account details for processing "]
sms_proc = tokenizer.texts_to_sequences(sms)
sms_proc = pad_sequences(sms_proc, maxlen=max_length, padding='post')
pred = (model.predict(sms_proc) > 0.5).astype("int32").item()
print(pred)
```