MIT | Academy of Engineering
(An Autonomous Institute affiliated to Savitribai Phule Pune University)

School of Computer Engineering & Technology

A Mini Project Report

on

"An LSTM Approach for SMS Classification using Recurrent Neural Network"

by

Arjun Yachwad (B194032)
Prashant Walunj (B194158)
Pranav Dighe (B194152)
Tushar Potale (B194140)

of

B.Tech

Academic Year: 2019-2020

CONTENT

INTRODUCTION:

In recent years, the use of mobile phones and smart systems has increasingly developed, and Short Message Service (SMS) has become one of the most important means of communication; so that 97% of cell phone users use this service. Unwanted short messages (known as spams) are transferred on the communication channel such as SMS, perform great advertising, but as a disturbing factor for users. In 2012, more than 6 billion messages were transferred on mobile phones in USA. According to a study in 2011, 3.5 billion people or 80% of active users in the world use SMS of mobile phone as a means of communication.

According to a study in 2011, 3.5 billion people or 80% of active users in the world use SMS of mobile phone as a means of communication. Of this large number of short messages, many are unwanted SMS messages produced for the following reasons.

- Sending SMS is low cost and many mobile operators offer SMS packages with very low price.
- Since users interact more with mobile phones compared to computers, they have more confidence in SMS, and it is very convenient to send confidential information.

AIM:

1) Reduce IT Administration Costs
2) Increase Security and Control
3) Provide sensitivity to the client and adapts well to the future spam techniques.
4) Reduce Network Resource Costs.

OBJECTIVE:

The objective of identification of Spam e-mails are:

1) To give knowledge to the user about the fke e-mails and relevant e-mails
2) To classify that mail spam or not

PROBLEM STATEMENT:

Short Message Service (SMS) is one of the well-known communication services in which a message sends electronically unwanted e-mails irritating internet connection, Critical e-mail message are missed or delayed, billions of dollars lost worldwideas well as spam can crash mail servers and fill up hard drives.

To minimize this problem we used An LSTM Approach for SMS Classification using Recurrent Neural Network.

LITERATURE SURVEY:

Spam mail, also called unsolicited bulk e-mail or junk mail that is sent to a group of recipients who have not requested it. The task of spam filtering is to rule out unsolicited e-mails automatically from a user's mail stream. These unsolicited mails have already caused many problems such as filling mailboxes, engulfing important personal mail, wasting network bandwidth, consuming users time and energy to sort through it, not to mention all the other problems associated with spam (crashed mail-servers, pornography adverts sent to children, and so on)[3].

According to a series of surveys conducted by CAUBE.AU 1, the number of total spasm received by 41 email addresses has increased by a factor of six in two years (from 1753 spams in 2000 to 10,847 spams in 2001)[4]. Therefore it is challenging to develop spam filters that can effectively eliminate the increasing volumes of unwanted mails automatically before they enter a user's mailbox. D. Puniskis [5] in his research applied the neural network approach to the classification of spam. His method employs attributes composed of descriptive characteristics of the evasive patterns that spammers employ rather than using the context or frequency of keywords in the message. The data used is corpus of 2788 legitimate and 1812 spam emails received during a period of several months.

PROPOSED METHOD:

The ability of the brain to process huge volumes of information in a short time, the use of parallel structure in data analysis, and the remarkable ability of the human brain in learning various issues are special features. Therefore, simulation is always been tempting, and deep neural networks have been created for this purpose. Among all machine learning algorithms, Deep Recurrent Neural Networks (DRNNs) work on data sequence [15]. Sequential data are data whose current values depend on previous values [16]. Among such data, the following can be noted.

- Frames (samples) of speech signal x Continuous Frames (images) of Video
- Climatic condition
- The stock price of a company / industry
- The sequences generated by the grammar
- Words within a text RNNs are very powerful because of combining the following features:

  Distributed hidden layers, which allow them to save a lot of information about previous layers.
  - Non-linear dynamics, which allows such networks to update hidden layers in a complex way.
  - RNNs have the potential to provide implementation and enforcement for small and parallel programs, and thus have a great interaction for producing more complicated results.
  - With the number of neurons at hand and enough time, RNNs have the ability to do any calculations performed by a computer.

Recurrent Neural Network (RNN):

A RNN contains an input layer, hidden layer, and output layer, as well as feedback connection weights, activation functions, and interconnection weights. In this study, the proposed RNN is designed by the combination of the locally recurrent and globally feed forward structure. The dynamic properties are achieved by utilizing the internal feedbacks as shown in Figure.

For a clear understanding of the computational model for the proposed RNN through the proposed structure, the mathematical function of each node is described as follows: Figure.

The structure of recurrent neural network Input layer: There are M nodes, which represent the input variables in this layer. The output values of each node can be described as Equation (1): where ui(t) is the ith output value at time t, i = 1, 2, . . ., M, and the input vector is x(t) = [x1(t), x2(t), . . ., xM(t)]. Hidden layer: each node in hidden layer connects with the input nodes and output node of RNN. The output of each hidden node.

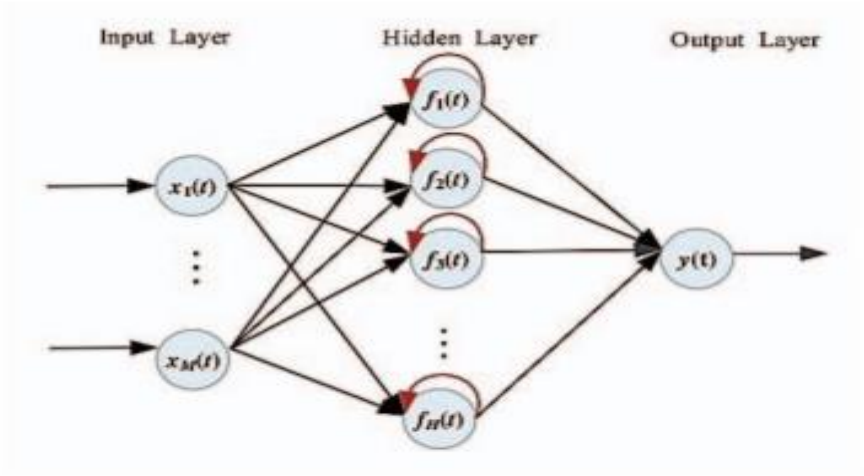


Figure 2: The structure of recurrent neural network

PREPROCESSING AND FEATURE EXTRACTION

First, short texts are collected with custom-made crawlers for different social media. Then, pre-processing tasks are needed to clean-up post contents. For example, URL links, hashtags, and emoticons are filtered. Also, stopword removal is performed to focus on content words. For Chinese posts, we used Jieba for word segmentation. Then, we extract metadata such as poster ID, posting time, and the number of retweets and likes. These will be used as additional features for classification.

SENTIMENT CLASSIFICATION:

After sentiment classification model is trained using LSTM, all posts in test set are preprocessed with the same procedure as the training set, and represented using the same word embedding model. Then, for testing step, the same processes of LSTM in Fig. 2 are followed, except for the weights update. The output of LSTM model will then be evaluated with the labels of each post in test data in the experiments.

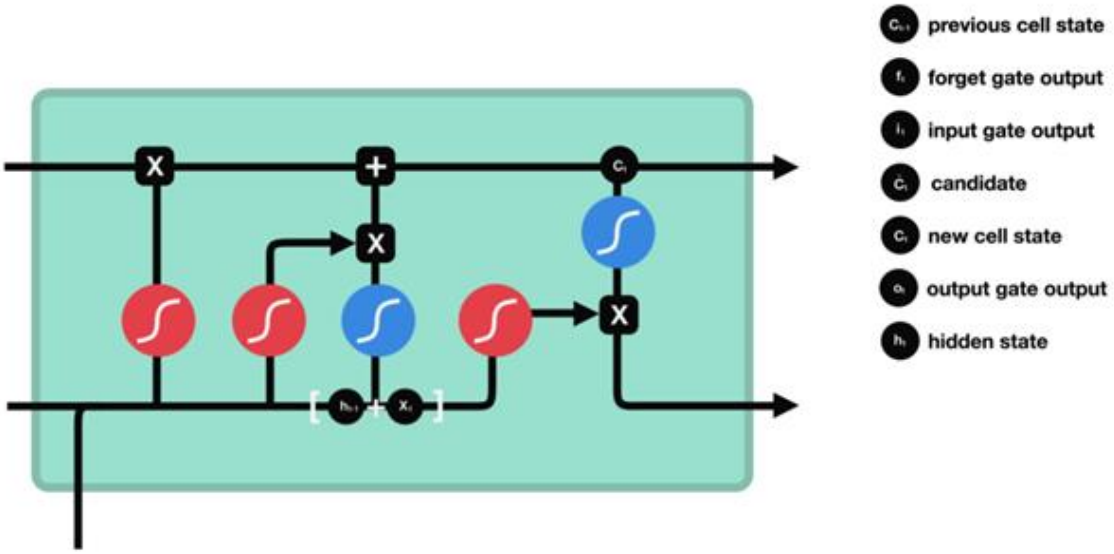
LSTM

Long short-term memory network was first introduced in 1997 by Sepp Hochreiter and his supervisor for a Ph.D. thesis.

LSTM is a special kind of RNN, capable of learning long term dependencies.

Remembering information for long period of time is it's default behaviour.

Long short-term memory (LSTM) network is the most popular solution to the vanishing gradient problem.



PROJECT CODE:

In[1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Model
from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding
from keras.optimizers import RMSprop
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
get_ipython().run_line_magic('matplotlib', 'inline')

df = pd.read_csv('spam.csv',delimiter=',',encoding='latin-1')
df.head()
```

In[2]:

```
sns.countplot(df.v1)
plt.xlabel('Label')
plt.title('Number of ham and spam messages')
```

In[3]:

```
X = df.v2
Y = df.v1
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
```

In[7]:

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)
```

In[8]:

```
# A good first step when working with text is to split it into words. Words are called tokens and the process of splitting text into tokens is called tokenization.
# Keras provides the text_to_word_sequence() function that you can use to split text into a list of words...

max_words = 1000
max_len = 150
tok = Tokenizer(num_words=max_words)
tok.fit_on_texts(X_train)
sequences = tok.texts_to_sequences(X_train)
sequences_matrix = sequence.pad_sequences(sequences,maxlen=max_len)
```

In[9]:

```
# Keras Embedding Layer. Keras offers an Embedding layer that can be used for neural networks on text data. It requires that the input data be integer
# encoded, so that each word is represented by a unique integer. ... It can be used to load a pre-trained word embedding model, a type of transfer learning

def RNN():
    inputs = Input(name='inputs',shape=[max_len])
    layer = Embedding(max_words,50,input_length=max_len)(inputs)
    layer = LSTM(64)(layer)
    layer = Dense(256,name='FC1')(layer)
    layer = Activation('relu')(layer)
    layer = Dropout(0.5)(layer)
    layer = Dense(1,name='out_layer')(layer)
    layer = Activation('sigmoid')(layer)
    model = Model(inputs=inputs,outputs=layer)
    return model
```

In[10]:

```
model = RNN()
model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
```

In[11]:

```
# A callback is a set of functions to be applied at given stages of the training procedure. You can use callbacks to get a view on internal states
```

```
# and statistics of the model during training. You can pass a list of callbacks (as the keyword argument callbacks) to the .fit() method of the Sequential
#  or Model classes. The relevant methods of the callbacks will then be called at each stage of the training.

model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
        validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.0001)])


# In[12]:


test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = sequence.pad_sequences(test_sequences,maxlen=max_len)


# In[13]:


accr = model.evaluate(test_sequences_matrix,Y_test)


# In[14]:


print('Test set\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(accr[0],accr[1]))


# In[15]:


Testing_context = ["Oh k...i'm watching here:)"]

txts = tok.texts_to_sequences(Testing_context)
txts = sequence.pad_sequences(txts, maxlen=max_len)


# In[16]:


preds = model.predict(txts)
print(preds)
```

DETAILS OF DATASET

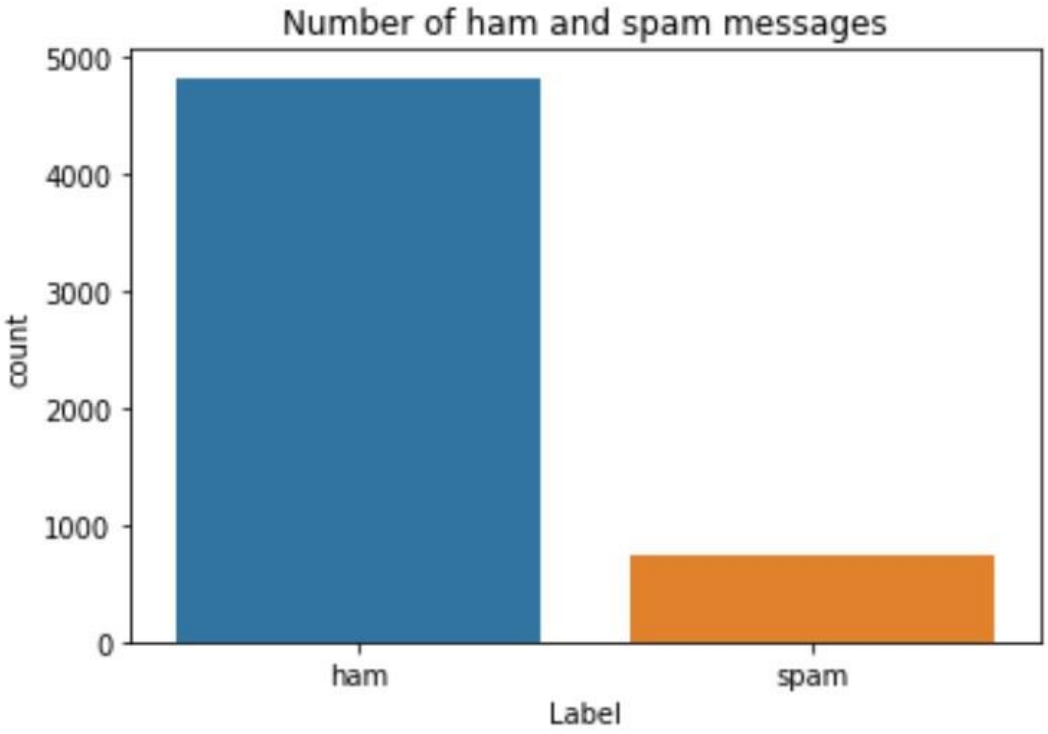
The SMS Spam Collection is a set of SMS tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged according being ham (legitimate) or spam.

The files contain one message per line. Each line is composed by two columns: v1 contains the label (ham or spam) and v2 contains the raw text.

spam.csv (491.86 KB) — 4 of 4 columns — Vie

	v1 class	v2 sms		
	ham 87% spam 13%	5169 unique values	[null] 99% bt not his girlfrnd..... 0% Other (42) 1%	[null] 100% MK17 92H. 450Pp... 0% Other (9) 0%
1	ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...		
2	ham	Ok lar... Joking wif u oni...		
3	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's		
4	ham	U dun say so early hor... U c already then say...		
5	ham	Nah I don't think he goes to usf, he		

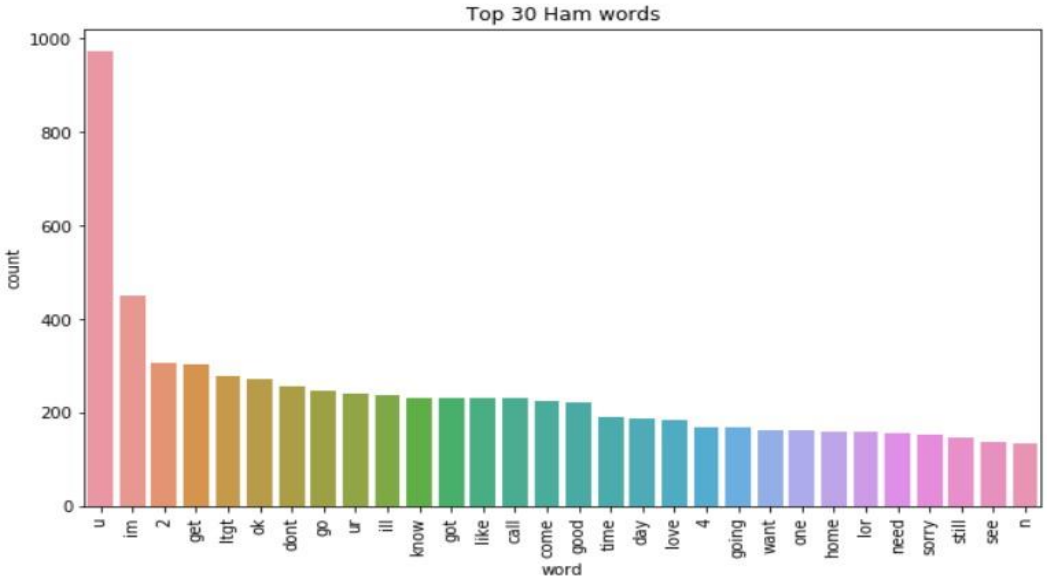
Following is The graph showing Number of Spam vs Ham SMS.



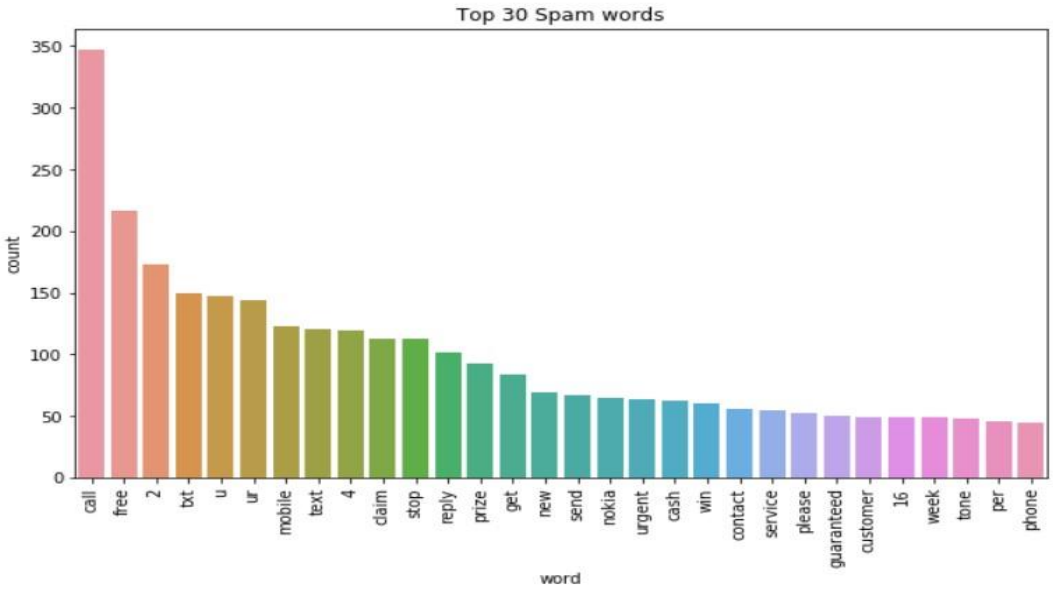
RESULTS & ANALYSIS:

Classification method is provided for detecting unwanted and normal messages. So far, in the previous studies, SVM or Bayesian methods is used. The proposed method RNNs is used. Test results on standard datasets UCI SMS spam showed the efficiency of the proposed method. In the proposed method, after 10 initial epochs, when a steady state is observed, a high accuracy of 98% is obtained .

In this following Graph the top 30 Ham words are given



In this following Graph the top 30 Spam words are given



The accuracy Which we Got from our model is 98.7%.which if we compare it with any other model which is very high.

```
print('Test set\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(accr[0],accr[1]))
```

```
Test set
  Loss: 0.040
  Accuracy: 0.987
```

As shown Down below In Testing_context we are providing input to our trained model . Then it will classify that SMS based on our trained model.
Then we will predict from our model . we will get output in percentage.
As in this result output is 0.00024814 which means it is ham SMS .

In [15]:
```
Testing_context = ["Oh k...i'm watching here:)"]

txts = tok.texts_to_sequences(Testing_context)
txts = sequence.pad_sequences(txts, maxlen=max_len)
```

In [16]:
```
preds = model.predict(txts)
print(preds)
```

```
[[0.00024814]]
```

CONCLUSION:

In this paper, we have proposed a sentiment classification approach based on LSTM for short texts in social media. Using word embeddings such as Word2Vec model, it's feasible to train the contextual semantics of words in short texts. Also, deep learning methods such as LSTM show better performance of sentiment classification when there are more amounts of training data. For special community behaviors, further experiments using "community"-specific sentiment lexicon and larger data sizes are needed in future.

REFERENCES:

[1] C. Pu and S. Webb, "Observed trends in spam construction techniques: A case study of spam evolution", Proceeding of 3 rd Conference on E-Mail and Anti-Spam, 2006.

[2] M. Embrechts, B. Szymanski, K. Sternickel, T. Naenna, and R. Bragaspathi,"Use of Machine Learning for Classification of Magnetocardiograms", Proceedings of IEEE Conference on System, Man and Cybernetics, Washington DC, pp. 1400-05, 2003.

[3] Duncan Cook, Jacky Hartnett, Kevin Manderson and Joel Scanlan, "Catching Spam before it arrives: Domain Specific Dynamic Blacklists", in ACSW Frontiers, Australian Computer Society, Vol. 54, pp. 193 – 202, 2006.

[4] Bekker S, "Spam to Cost U.S. Companies $10 Billion in 2003", ENT News, http://www.entmag.com/news/article.asp?EditorialsID=565 1>.

[5] D. Puniškis, R. Laurutis and R. Dirmeikis, "An Artificial Neural Nets for Spam e-mail Recognition", Electronics and electrical engineering, Vol. 69, No. 5, pp. 73 – 76, 2006.