A Major Project On

**Multi-lingual SMS spam detection**

**Using**

**Recurrent Neural Networks**

Submitted By

Ch. Manohar (1407029)

I. Sai Shashank (1407007)

Pooja Bharti (1407011)



A report submitted in fulfilment for the

The degree of Bachelor of Technology

In Computer Science and Engineering

Under the guidance of

Dr Akshay Deepak

Department of Computer Science and Engineering

Jan-May 2018



**NATIONAL INSTITUTE OF TECHNOLOGY PATNA**

(An Institute under Ministry Of HRD, Govt. of India)

**ASHOK RAJPATH, PATNA-800005 (BIHAR)**

**CERTIFICATE**

This is to certify that **Ch. Manohar (1407029), I. Sai Shashank (1407007), Pooja Bharti (1407011)** has carried out the major project entitled as **Multi-lingual SMS spam detection using Recurrent Neural Networks** during 8th Semester under the supervision of **Dr. Akshay Deepak,** **Department of Computer Science and Engineering** in complete fulfilment of the requirements of the degree of Bachelor of Technology in Computer Science and Engineering.

Dr Akshay Deepak Dr Prabhat Kumar

Project Supervisor & Assistant Professor Head of Department

CSE Department CSE Department

May 2018 May 2018



**DECLARATION**

I hereby declare that this project entitled **“Multi-lingual SMS spam detection using Recurrent Neural Networks”** submitted as the major project has been carried out by me under the supervision of **Dr Akshay Deepak,** **Department of Computer Science and Engineering**. No part of this project has been submitted for the award degree or diploma to any other Institute.

|  |  |  |  |
| --- | --- | --- | --- |
| S. No. | Name | RollNo**.** | Signature |
|  | Ch. Manohar | 1407029 |  |
|  | I. Sai Shashank | 1407007 |  |
|  | Pooja Bharti | 1407011 |  |

Place: Patna

Date: May 2018

**ACKNOWLEDGEMENT**

We hereby take the privilege to express our gratitude to all the people who were directly or indirectly involved in the execution of this work, without whom this project would not have been a success. We extend our deep gratitude, respect and obligation to our project mentor, **Dr Akshay Deepak,** for being a constant source of inspiration. Our heartiest thanks to our classmates who have supported us in all ways. Words are inadequate to express our gratitude to our parents and friends who have been supportive all the time. We would also like to thank our institution and the faculty members without whom this project would have been a distant reality. Above all, we bow before the Almighty for his immense blessings throughout the life.

**Ch. Manohar I. Sai Shashank Pooja Bharti**

**(1407029) (1407007) (1407011)**

**Table of Contents**

**Table of Contents v**

**List of Figures vi**

**Abstract vii**

1. **Introduction 1**
   1. **What is Spam? 1**
   2. **Purpose of Spam 1**
   3. **Spam as a Problem 1**
2. **Dataset 3**
   1. **Data Collection 3**
   2. **Data Labelling 4**
   3. **Data Pre-processing 4**
3. **Numerical Representation 6**
   1. **Bag of Words 6**
   2. **Word2Vec 6**
   3. **Lang2Vec 7**
4. **Machine Learning Classification Models 8**
   1. **Support Vector Machines 8**
   2. **Multinomial Naive Bayes 8**
   3. **Logistic Regression 9**
   4. **K-Nearest Neighbours 10**
   5. **Decision Tree 11**
   6. **Random Forest 12**
5. **Recurrent Neural Networks 13**
   1. **Long Short Term Memory (LSTM) 14**
6. **Implementation 16**
   1. **Bag of Words 16**
   2. **Word2Vec 20**
7. **Result**
8. **Conclusion**
9. **Bibliography**

**List of Figures**

**2.1 Dataset Description 3**

**2.2 Data Labelled Screenshot 4**

**3.2.1 Word2Vec neural network 7**

**5.1 Unfolding RNN for each time step 13**

**5.1.1 LSTM CELL 14**

**ABSTRACT**

Over recent years, the growth of SMS Spam messages has been increased as the cost to exchange messages has decreased. SMS messages for spam detection are often classified using the frequently repeated words but it doesn't consider the context of the text message. In our report, we are building a robust recurrent neural model using LSTM which actually deals with the context of the text message by remembering the sequence of the words in a message. For our Experimental purposes, we have also build different machine learning classifiers on the same (multilingual) SMS dataset that does classification based on frequently repeated words, to compare ourselves with the proposed recurrent neural model results.

**Chapter 1**

**Introduction**

The mobile phone market has grown exponentially in the recent years, an estimated 62.9 percent of the population worldwide already owned a mobile phone and the number of mobile phone users in the world is expected to pass the five billion mark by 2019. As the usage of mobile phones increased and cost to exchange SMS has decreased, SMS's exchanged between mobile phones has increased rapidly. Short Messaging Service (SMS) is a text messaging service that allows mobile phone users to exchange short text messages usually of length less than 160 characters. SMS is used as an alternate for voice calls in positions where voice communication is either not possible or not desired between the end phone users. As Popularity of SMS has increased, SMS's are becoming the main target for spammers.

* 1. **What is Spam?**

Spam is the use of electronic messaging systems to send an unsolicited or unwanted message especially advertising, as well as sending messages repeatedly. Sending messages to numerous recipients at the same time. Spamming remains economically viable because advertisers have no operating costs beyond the management of their spamming lists. Mobile phone spam is directed at the text messaging service of a mobile phone. This can be especially irritating to customers not only for the inconvenience but also because of the fee they may be charged per text message received in some markets

* 1. **Purpose of Spam**

Spams are mainly used for advertisement, multi-level marketing etc. These messages are sent by spammers for different ill wills of taking a hold over user’s personal data or tricking them into the subscription of their premium tariff facilities.

* 1. **Spam as Problem**

Spam is a big problem now-a-days due to following reasons:

* It is very annoying
* It consumes resources and takes times
* Fraud
* Identity Theft

**Chapter 2**

**2.1 Data Collection**

We make use of multi-language dataset .We have total 10,167 messages or dataset .In these dataset ,we have collected 5572 standard English dataset from kaggle.com and real SMS data from our own ,friends mobile phone .Our own dataset contains data from 3 different languages .These are English ,Hindi ,Telugu and mix of these 3 languages .Mix data is the combination of (Hindi ,English) , (English ,Telugu) ,(Hindi ,Telugu) SMS messages .After collecting the dataset we manually filter the data .So that same data doesn’t get repeated.

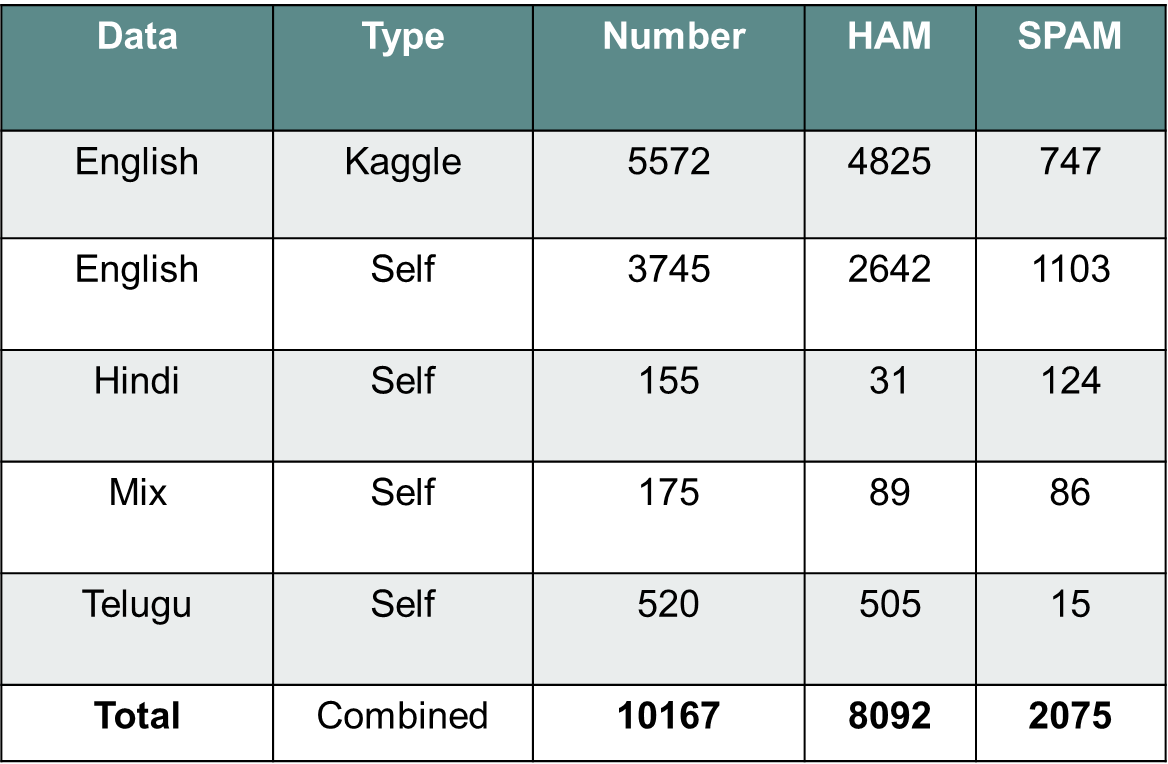


Figure 2.1 Dataset Description

**2.2 Data Labelling**

Manually labelled 4595 our own collected data as SPAM or HAM based on individual perspective.

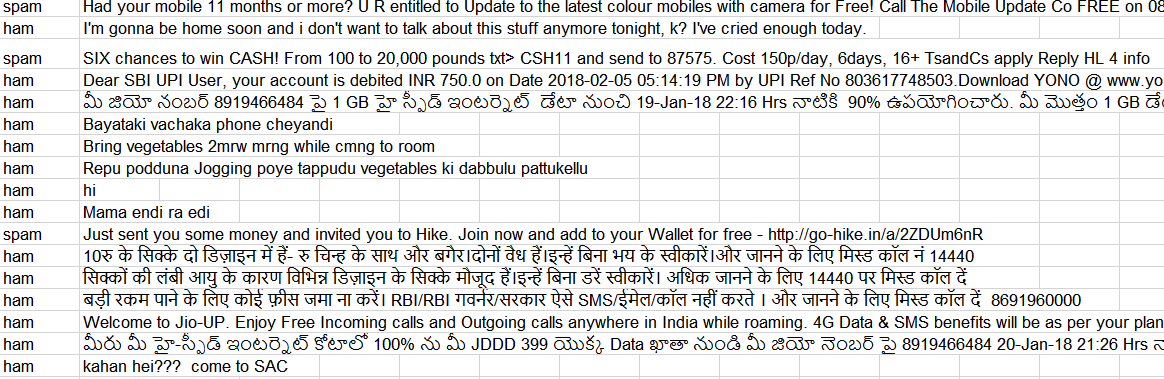
****

Figure 2.2 Data Labelled Screenshot

**2.3 Data Pre-processing**

Data pre-processing simply means the cleaning the data .Since every real world data is redundant .Hence Data pre-processing is very important for any machine learning model to give better results .In our project data pre-processing includes removal of unnecessary words i.e. stop words and stemming .It also includes converting all the letters into lowercase letters .The data pre-processing has done by using **nltk** library.

**Stop words:**

These are the most commonly occurring word in any language .It should be removed because these are unnecessary and frequent words that carry no information .Hence removal of these stop words has proven as very important.

List of nltk stop words:

[ the ,is ,of ,are ,am ,what ,when ,whom ,this ,that……., etc.]

**Stemming:**

Stemming is the process of reducing words to its root word by reducing affixes of a word to get its stem or root even if root has no dictionary meaning .The hypothesis behind stemming is that words with the same root mostly describe relatively close or similar meaning.

For example- beautiful, beautifully, beauty will be converted to **beauti** even if it has no meaning in dictionary. It has done by using **Porter Stemmer** algorithm.

**Chapter 3**

**Numerical Representation**

Many algorithms in machinelearning require a numericalrepresentation of objects, since such representations facilitate processing and statistical analysis.

**3.1 Bag of Words**

In this technique, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and its each unique word frequency. The bag-of-words model is commonly used in methods of document classification where the (frequency of) occurrence of each word is used as a feature for training a classifier. In practice, the Bag-of-words model is mainly used as a tool of feature generation. After transforming the text into a "bag of words", we can calculate various measures to characterize the text. The most common type of characteristics, or features calculated from the Bag-of-words model is term frequency, the number of times a term appears in the text.

Uses: Spam filtering, Sentiment analysis ….

Drawbacks: context of message is ignored.

**3.2 Word2Vec**

These models are two-layer neural networks that are trained to reconstruct contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector. Word2vec can give near numerical representation of similar words.

The purpose and usefulness of Word2vec is to group the vectors of similar words together in vector space. That is, it detects similarities mathematically. Word2vec creates vectors that are distributed numerical representations of word features, features such as the context of individual words.

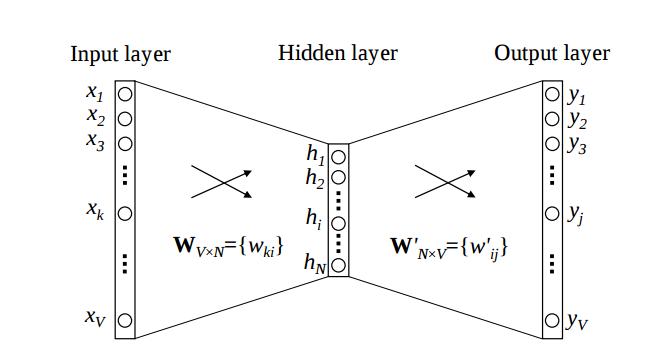


Figure 3.2.1 Word2Vec neural network

**3.3 Lang2Vec**

The purpose and usefulness of Word2vec is to group the vectors of similar words together in vector space and also using their language information.

**Chapter 4**

**4.1 Support Vector Machines**

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyper plane. In other words, given labelled training data (supervised learning), the algorithm outputs an optimal hyper plane which categorizes new examples. In two dimensional space this hyper plane is a line dividing a plane in two parts where in each class lay in either side. The learning of the hyper plane in linear SVM is done by transforming the problem using some linear algebra. This is where the kernel plays role. For linear kernel the equation for prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

**F(x) = B(0) + sum(ai \* (x,xi))**

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B0 and ai (for each input) must be estimated from the training data by the learning algorithm. And finally last but very important characteristic of SVM classifier. SVM to core tries to achieve a good margin. A margin is a separation of line to the closest class points. A good margin is one where this separation is larger for both the classes.

**4.2 Multinomial Naive Bayes**

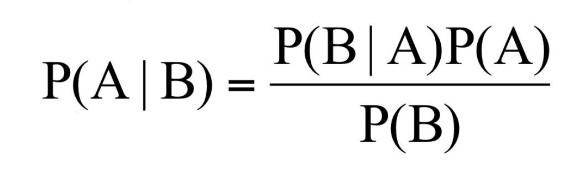
It tells us how often A happens given that B happens, written P(A|B), when we know how often B happens given that A happens, written P(B|A) , and how likely A and B are on their own.

P (A|B) is “Probability of A given B”, the probability of A given that B happens

P (A) is Probability of A

P (B|A) is “Probability of B given A”, the probability of B given that A happens

P (B) is Probability of B



Naive Bayes classifier calculates the probabilities for every factor. Then it selects the outcome with highest probability. This classifier assumes the features are independent. Hence the word naive. Even with this it is powerful algorithm used for

1. Real time Prediction

2. Text classification/ Spam Filtering

3. Recommendation System

**4.3 Logistic Regression**

Logistic Regression is a classification algorithm. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables. To represent binary / categorical outcome, we use dummy variables. You can also think of logistic regression as a special case of linear regression when the outcome variable is categorical, where we are using log of odds as dependent variable. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function.

Logistic Regression is part of a larger class of algorithms known as Generalized Linear Model

The fundamental equation of generalized linear model is:



In logistic regression, we are only concerned about the probability of outcome dependent variable. This function is established using two things: Probability of Success (p) and Probability of Failure (1-p). P should meet following criteria:

It must always be positive (since p >= 0)

It must always be less than equals to 1 (since p <= 1)

**4.4 K-Nearest Neighbours**

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

1. Ease to interpret output

2. Calculation time

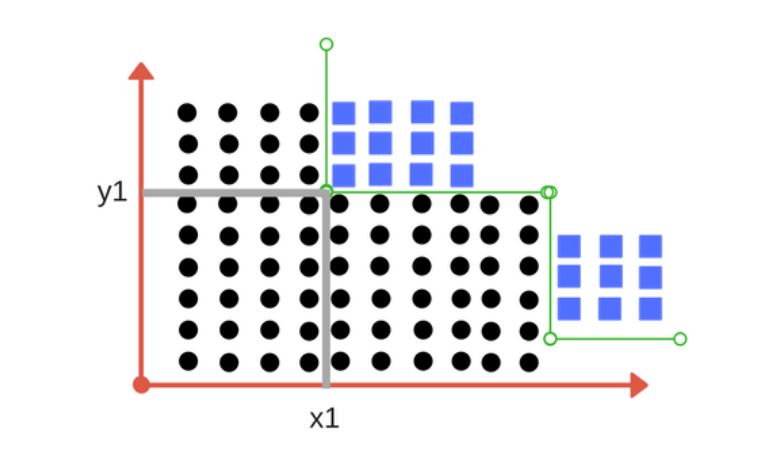
3. Predictive Power

KNN algorithm fairs across all parameters of considerations. It is commonly used for its easy of interpretation and low calculation time. The “K” is KNN algorithm is the nearest neighbours we wish to take vote from. The choice of the parameter K is very crucial in this algorithm. The boundary becomes smoother with increasing value of K.

KNN algorithm is one of the simplest classification algorithm. Even with such simplicity, it can give highly competitive results. KNN algorithm can also be used for regression problems. The only difference from the discussed methodology will be using averages of nearest neighbours rather than voting from nearest neighbours.

**4.5 Decision Tree**

Decision Tree Classifier, repetitively divides the working area (plot) into sub part by identifying lines. (Repetitively because there may be two distant regions of same class divided by other as shown in image below).



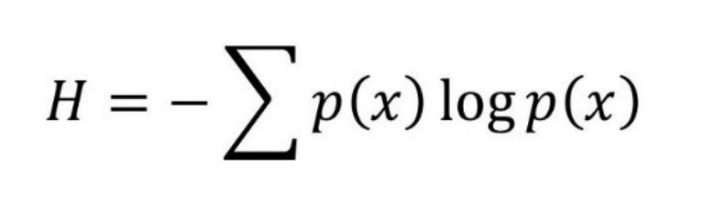
In above division, we had clear separation of classes. But what if we don’t have?

Impurity is when we have a traces of one class division into other. This can arise due to following reason

1. We run out of available features to divide the class upon.

2. We tolerate some percentage of impurity (we stop further division) for faster performance. (There is always trade-off between accuracy and performance).

Entropy is degree of randomness of elements or in other words it is measure of impurity. Mathematically, it can be calculated with the help of probability of the items as:



Suppose we have multiple features to divide the current working set. What feature should we select for division? Perhaps one that gives us less Impurity.

Decision tree at every stage selects the one that gives best information gain. When information gain is 0 means the feature does not divide the working set at all. Dividing efficiently based on maximum information gain is key to decision tree classifier.

**4.6 Random Forest**

Random Forest Classifier is ensemble algorithm. Ensembled algorithms are those which combines more than one algorithms of same or different kind for classifying objects.

Random forest classifier creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object. So finally, it predicts based on the majority of votes from each of the decision trees made. This works well because a single decision tree may be prone to a noise, but aggregate of many decision trees reduce the effect of noise giving more accurate results.

n\_estimators: Number of trees in forest. Default is 10.

Criterion: “gini” or “entropy” same as decision tree classifier.

Random Forest Classifier being ensemble algorithm tends to give more accurate result. This is because it works on principle,

Number of weak estimators when combined forms strong estimator.

**Chapter 5**

**Recurrent Neural Networks**

Language as we saw earlier- the sequence of words define their meaning, a time series data – where time defines the occurrence of events, the data of a genome sequence- where every sequence has a different meaning.

**Uses:**

* **Sentiment Classification** – This can be a task of simply classifying tweets into positive and negative sentiment. So here the input would be a tweet of varying lengths, while output is of a fixed type and size.
* **Image Captioning** – Let’s say we have an image for which we need a textual description. So we have a single input – the image, and a series or sequence of words as output. Here the image might be of a fixed size, but the output is a description of varying lengths.
* **Language Translation** – This basically means that we have some text in a particular language let’s say English, and we wish to translate it in French. Each language has its own semantics and would have varying lengths for the same sentence. So here the inputs as well as outputs are of varying lengths.

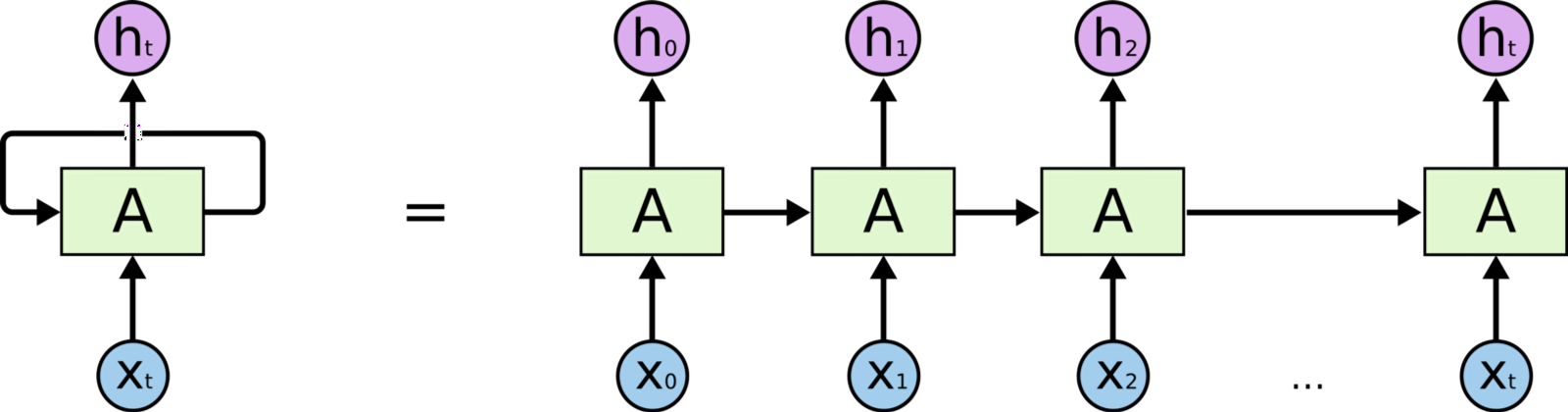


Figure 5.1 Unfolding RNN for each time step

**5.1 Long Short Term Memory (LSTM)**

Recurrent Neural Networks work just fine when we are dealing with short-term dependencies. RNNs fail to understand the context behind an input. Something that was said long before, cannot be recalled when making predictions in the present. The reason behind this is the problem of **Vanishing Gradient.** RNN remembers things for just small durations of time, i.e. if we need the information after a small time it may be reproducible, but once a lot of words are fed in, this information gets lost somewhere. This issue can be resolved by applying a slightly tweaked version of RNNs – the Long Short-Term Memory Networks.LSTMs have an edge over conventional feed-forward neural networks and RNN in many ways. This is because of their property of selectively remembering patterns for long durations of time.

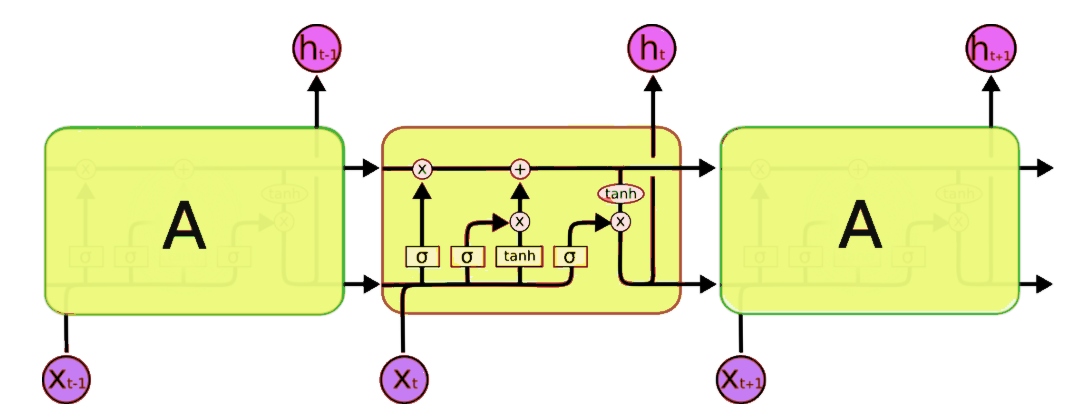
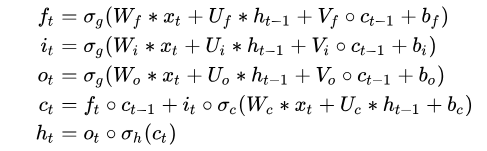


Figure 5.1.1 LSTM CELL

Now, this is nowhere close to the simplified version which we saw before, but let me walk you through it.

A typical LSTM network is comprised of different memory blocks called cells (the rectangles that we see in the image)**.** There are two states that are being transferred to the next cell; the cellstate and thehiddenstate. The memory blocks are responsible for remembering things and manipulations to this memory is done through three major mechanisms, called **gates.**

**Mathematics formulas behind LSTM:**

****

LSTMs are a very promising solution to sequence and time series related problems. However, the one disadvantage that I find about them, is the difficulty in training them. A lot of time and system resources go into training even a simple model.

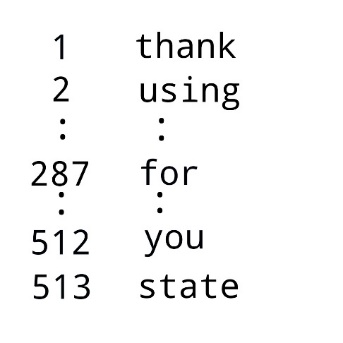
**Chapter 6**

**Implementation**

**6.1 Bag of Words**

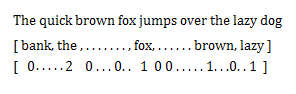
We made a list of 410 most frequent words into our model dictionary.

Sample Dictionary:



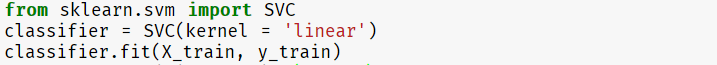
The next step is to score the words in each document. The objective is to turn each document of free text into a vector that we can use as input or output for a machine learning model. The simplest scoring method is to mark the number of words as a value, 0 for absent, or its corresponding frequency.

Example:



**Using Different Machine Learning Techniques:**

1. **Support Vector Machine**

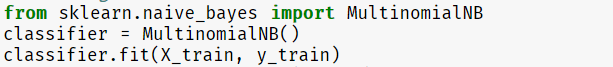


Using **svm and linear kernel** we got 97.98% accuracy for kaggle dataset, 96.21% accuracy for our multi-lingual dataset and kaggle combined.

Confusion matrix for multi-lingual:



**2. Multinomial Naive Bayes**

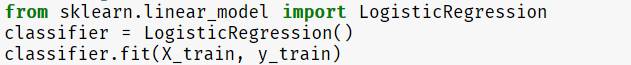


Using **Multinomial Naive Bayes** got 98.42% accuracy for kaggle dataset, 92.07% accuracy for our multi-lingual dataset and kaggle combined.

Confusion matrix for multi-lingual:



1. **Logistic Regression**

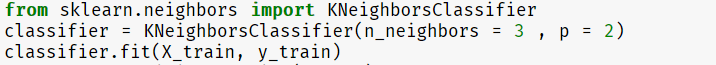


Using **Logistic regression** got 97.48% accuracy for kaggle dataset, 95.46% accuracy for our multi-lingual dataset and kaggle combined.

Confusion matrix for multi-lingual:



1. **K-Nearest Neighbours**

****

Using **K-Nearest Neighbours** got 93.10% accuracy for kaggle dataset, 94.24% accuracy for our multi-lingual dataset and kaggle combined.

Confusion matrix for multi-lingual:



1. **Decision Tree**

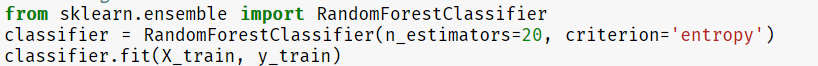
****

Using **Decision Tree** got 96.62% accuracy for kaggle dataset, 95.26% accuracy for our multi-lingual dataset and kaggle combined.

Confusion matrix for multi-lingual:



1. **Random Forest**



Using **Random Forest** got 97.98% accuracy for kaggle dataset, 96.50% accuracy for our multi-lingual dataset and kaggle combined.

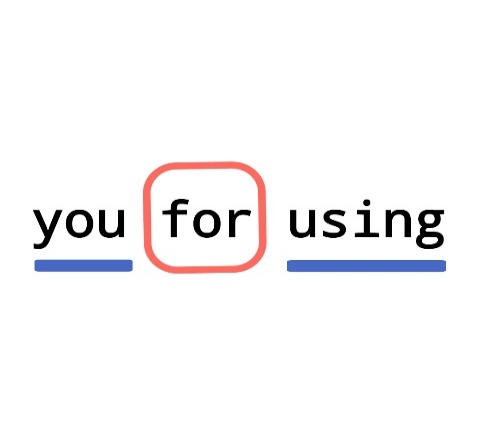
Confusion matrix for multi-lingual:



**6.2 Word2Vec**

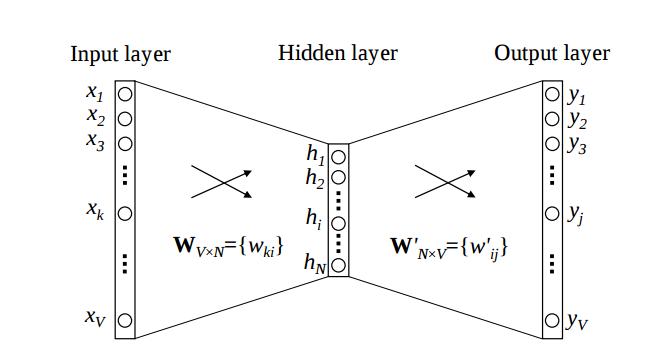
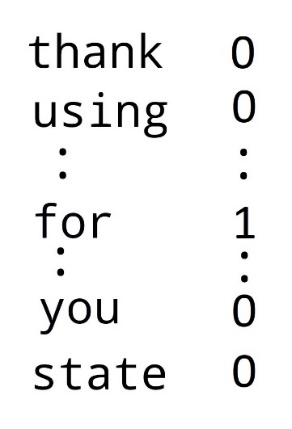
We need to convert message to an **input output pair**such that if we input a word, it should it predict that the neighbouring words: the n words before and after it.

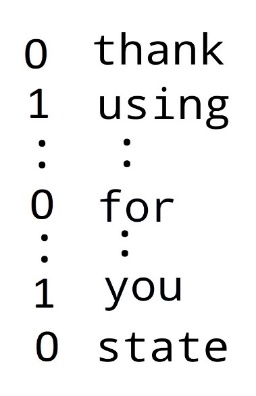
Sample SMS: **thank you for using state bank internet banking**.



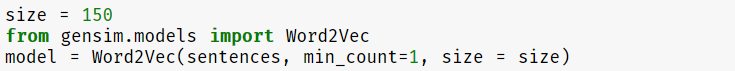
If we consider word ‘for’, you and using are target words. Then, we created these combinations for every word in dataset.

And then we feeded it into network.

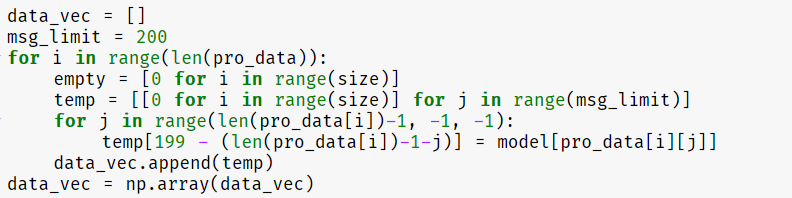




Gensim module for word2vec purpose.

****

Next, Pre-padding is done to make every input data of equal length. We made every message of length 200.

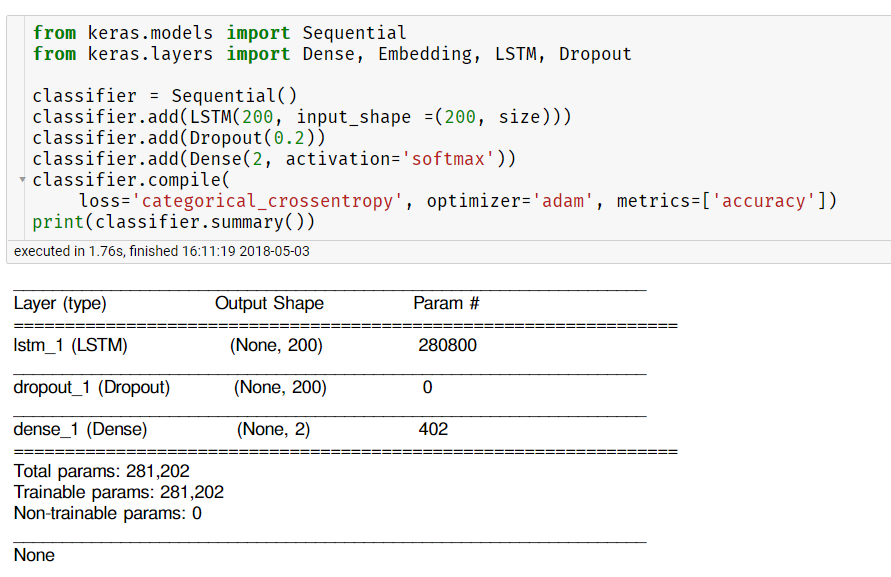
****

**LSTM:**

LSTM Networks are special kind of RNN, capable of learning long- term dependencies

In our model, output length of LSTM is 200 and it is connected 2 node output layer.

**Model:**

****

We used categorical cross entropy as loss function and Adam optimizer.

**Chapter 7**

**Results:**

**Bag of Words:**

|  |  |  |
| --- | --- | --- |
| **Model** | **Kaggle** | **Combined** |
| **SVM** | 97.98 | 96.21 |
| **Naive Bayes** | 98.42 | 92.07 |
| **Logistic Regression** | 97.48 | 95.46 |
| **K-Nearest Neighbors** | 93.10 | 94.24 |
| **Decision Tree** | 96.62 | 95.26 |
| **Random Forest** | 97.98 | 96.50 |

**LSTM:**

**English (Kaggle):**

|  |  |  |  |
| --- | --- | --- | --- |
| **Accuracy** | **Error Rate** | **Spam Caught** | **Blocked HAM** |
| 97.91 | 2.09 | 87 | 0.27 |

**Multi-lingual:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Accuracy** | **Error Rate** | **Spam Caught** | **Blocked HAM** |
| **96.8** | **3.2** | **89.4** | **1.31** |