A
Project Report
On

SPAM SMS DETECTION

B.Tech Sem-VII

Prepared by

AKASH A. PATEL (IT-075)
JIGAR A. PARMAR (IT-071)

DEPARTMENT OF INFORMATION TECHNOLOGY
FACULTY OF TECHNOLOGY,
DHARMSINH DESAI UNIVERSITY
COLLEGE ROAD, NADIAD- 387001
December-2017

A
Project Report
On

SPAM SMS DETECTION

B.Tech Sem-VII

Prepared at

Department of Information Technology
Faculty of technology, Dharmsinh Desai University
College Road, Nadiad-387001

Prepared by

AKASH A. PATEL (IT-075)
JIGAR A. PARMAR (IT-071)

Guided by

Dr. MUKESH M. GOSWAMI
(Associate Professor)
Department of Information Technology
Faculty of Technology
Dharmsinh Desai University

DEPARTMENT OF INFORMATION TECHNOLOGY
FACULTY OF TECHNOLOGY,
DHARMSINH DESAI UNIVERSITY
COLLEGE ROAD, NADIAD- 387001

CANDIDATE'S DECLARATION

We declare that pre-final semester report entitled **"Spam SMS Detection"** is our own work conducted under the supervision of the guide **Dr. Mukesh M. Goswami**.

We further declare that to the best of our knowledge the report for B.Tech. VII semester does not contain part of the work which has been submitted either in this or any other university without proper citation.

_____
AKASH A. PATEL
15ITUOD019

_____
JIGAR A. PARMAR
14ITUBS002

DHARMSINH DESAI UNIVERSITY
NADIAD-387001, GUJARAT

CERTIFICATE

This is to certify that the project carried out in the subject of Software Design Project, entitled "**Spam SMS Detector**" and recorded in this report is a bonafide report of work of

1) AKASH A. PATEL Roll No. IT-075 ID No. 15ITUOD019

2) JIGAR A. PARMAR Roll No. IT-071 ID No. 14ITUBS002

Of Department of Information Technology, semester VII. They were involved in Project work during academic year 2017 – 2018.

Dr. MUKESH M. GOSWAMI,
Department of Information Technology,
Faculty of Technology,
Dharmsinh Desai University, Nadiad
Date:

_____

Prof. R.S.Chhajed,
Head, Department of Information Technology,
Faculty of Technology,
Dharmsinh Desai University, Nadiad
Date:

_____

ACKNOWLEDGEMENT

It is our pleasure to express deep sense of gratitude to all who helped us with their co-operation and guidance to develop this project.

We would like to show our gratitude to our guide Dr.Mukesh M. Goswami, Associate Professor, Dharmsinh Desai University for his dedicated involvement in every step throughout the process of our project development. We appreciate all of their expertise, guidance and careful critique for this project. His valuable guidance and support encouraged us and demonstrated that learning never ends.

In addition, we would like to express our sincere thanks to our head of department Prof.R.S.Chhajed who gave us an opportunity to explore our self to accomplish this project successfully at an undergraduate level.

I am obliged to staff members of Dharmsinh Desai University for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of our project.

Lastly, we would like to thanks to almighty, our parents, brothers and friends for their constant encouragement without which this project would not be possible.

AKASH A. PATEL (IT-075)

JIGAR A. PARMAR (IT-071)

Table of Contents

ABSTRACT

Due to increase in benefits of SMS the supply of spam messages has increased in the recent years. In parts of Asia up to 30% of messages were spam in 2012.So, here we propose to incorporate different new content based features to improve the performance of SMS spam detection. The effectiveness of the proposed feature is empirically validated using SVM classification method. The result demonstrate that the proposed features can improve the performance of SMS spam detection.

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 DEFINITION

This project aims to lessen the hassle and enhance the usability of mobile devices by segregating the incoming messages into spam or not spam.

1.2 SCOPE

Scope of the project application is so wide because everyone who face the problem related to spam messages in inbox, can use this application. As the application is made in android many people worldwide can use it on their smartphone.

1.3 TECHNOLOGY REVIEW

What is ANDROID?

Android is a mobile operating system developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets. Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. In addition to touchscreen devices, Google has further developed Android TV for televisions, Android Auto for cars, and Android Wear for wristwatches,each with a specialized user.

Features of ANDROID

- ✓ Application Framework
- ✓ Integrated Browser
- ✓ Optimized Graphics
- ✓ Java Support
- ✓ Storage and Media Support
- ✓ Various Connectivity Support
- ✓ Voice Based and Multitasking Feature
- ✓ Provide Multiple language support and Accessibility

What is ANDROID Studio?

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, MacOS, Linux based operating system. It is replacement of Eclipse Android Development tools as primary IDE for native Android Application Development.

Features of ANDROID Studio

- ✓ Gradle-based build support
- ✓ Android-specific refactoring and quick fixes
- ✓ Lint tools to catch performance, usability, version compatibility and other problem
- ✓ ProGuard integration and app-signing capabilities
- ✓ Template-based wizards to create common Android designs and components
- ✓ A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
- ✓ Support for building Android Wear apps

What is Gradle?

Gradle is an open source based automation system that builds upon the concepts of Apache Ant and Apache Maven and introduces a Groovy-based domain-specific language (DSL) instead of the XML form used by Apache Maven for declaring the project configuration. Gradle uses a directed acyclic graph ("DAG") to determine the order in which tasks can be run. Gradle was designed for multi-project builds which can grow to be quite large, and supports incremental builds by intelligently determining which parts of the build tree are up-to-date, so that any task dependent upon those parts will not need to be re-executed. The initial plugins are primarily focused on Java, Groovy and Scala development and deployment, but more languages and project workflows are on the roadmap.

FIREBASE

- **FIREBASE AUTH**

Firebase Auth is a service that can authenticate users using only client-side code. It supports social login providers Facebook, GitHub, Twitter and Google. Additionally, it includes a user management system whereby developers can enable user authentication with email and password login stored with Firebase.

2. BACKGROUND

2.1 MACHINE LEARNING

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning focuses on the development of computer programs** that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. **The primary aim is to allow the computers learn automatically** without human intervention or assistance and adjust actions accordingly.

Machine Learning methods

Machine learning algorithms are often categorized as supervised or unsupervised.

Supervised machine learning algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

In contrast, **unsupervised machine learning algorithms** are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn't figure out the right output,

but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.

2.2 FLASK

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website. Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins.

2.3 COUNT VECTORIZER

It converts a collection of text documents to a matrix of token counts. This implementation produces a sparse representation of the counts using scipy.sparse.csr_matrix.

In this scheme, features and samples are defined as follows:

1.) Each individual token occurrence frequency (normalized or not) is treated as a feature.

2.) The vector of all the token frequencies for a given document is considered a multivariate sample.

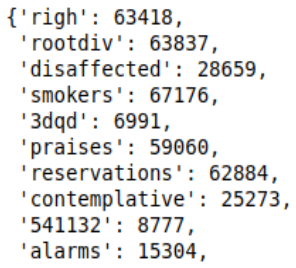

Figure 1 Token Counts generated by CountVectorizer

2.4 TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY

Term Frequency : The number of times each term *(T)* occurs in a document *(D)*is called its Term Frequency.

$$\text{Term Frequency} = F_{T,D} \; / \sum F_{T',D} \qquad (2.4.1)$$

Inverse Document Frequency : Its is a factor which diminishes the weight of term*(t)* that occur very frequently in the document*(d)* set and increases the weight of term that occur rarely.

$$\text{Inverse Document Frequency } (t,D) = \log(N \; / \; |\{d \in D : t \in d\}|) \qquad (2.4.2)$$

N: Total number of Documents. $N = |D|$

$|\{d \in D : t \in d\}|$: Number of documents where the term *t* appears.

TFIDF stands for TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY, is a numerical static that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval, text mining and user modelling.The tf-idf increases proportionally to a number of times a words appear in the document.

Let’s take an example with the following counts. The first term is present 100% of the time hence not very interesting. The two other features only in less than 50% of the time hence probably more representative of the content of the documents:

```
>>> counts = [[3, 0, 1],
...           [2, 0, 0],
...           [3, 0, 0],
...           [4, 0, 0],
...           [3, 2, 0],
...           [3, 0, 2]]
...
>>> tfidf = transformer.fit_transform(counts)
>>> tfidf
<6x3 sparse matrix of type '<... 'numpy.float64'>'
    with 9 stored elements in Compressed Sparse ... format>

>>> tfidf.toarray()
array([[ 0.81940995,  0.        ,  0.57320793],
       [ 1.        ,  0.        ,  0.        ],
       [ 1.        ,  0.        ,  0.        ],
       [ 1.        ,  0.        ,  0.        ],
       [ 0.47330339,  0.88089948,  0.        ],
       [ 0.58149261,  0.        ,  0.81355169]])
```

Figure 2 Output of tf-idf

2.5 SUPPORT VECTOR MACHINES

A Support Vector Machine(SVM) is a very powerful and versatile Machine Learning model, capable of performing linear and nonlinear classification, regression, and even outlier detection. It is one of the most popular models in Machine Learning.

Support Vectors are the data points nearest to the hyperplane, the points of the dataset that removed, would alter the position of the dividing hyperplane. Because of this, they can be considered critical points of a data set.

Pros and Cons of Support Vector Machines.

Pros:

- Accuracy
- Works well on smaller cleaner datasets.
- It can be more efficient because it uses a subset of training points

Cons:

- Isn’t suited to larger datasets as the training time with SVMs can be high.
- Less effective on noiser datasets with overlapping classes.

SVMs are based on idea of finding a hyperplane that best divides a dataset into two classes.SVMs are particularly well suited for classification of complex but small- or Medium-sized datasets.

Why use Support Vector machine?

According to a literature paper published by **Houshmand Shirani-Mehr of Stanford University,** SVM seems to have better accuracy compare to all the other algorithm like RainForest, Adaboost with decision trees, k-nearest neighbour.

3. LITERATURE REVIEW

3.1 SPAM DETECTION USING TEXT CLUSTERING - M. SASAKI AND H. SHINNOU OF IBARAKI UNIVERSITY, JAPAN.

Abstract

They propose a new spam detection technique using the text clustering based on vector space model. Their method computes disjoint clusters automatically using a spherical k-means algorithm for all spam/non-spam mails and obtains centroid vectors of the clusters for extracting the cluster description. For each centroid vectors, the label (`spam' or `non-spam') is assigned by calculating the number of spam email in the cluster. When new mail arrives, the cosine similarity between the new mail vector and centroid vector is calculated. Finally, the label of the most relevant cluster is assigned to the new mail. By using their method, we can extract many kinds of topics in spam/non-spam email and detect the spam email efficiently. In this paper, they describe the their spam detection system and show the result of their experiments using the Ling-Spam test collection.

3.2 THE CURSE OF 140 CHARACTERS: EVALUATING EFFICIENCY OF SMS SPAM DETECTION ON ANDROID BY AKSHAY NARAYAN AND PRATEEK SAKSENA

Abstract

Here, they have compared behaviour of applications present in Google play store to detect spam. They created a Naive Bayes classifier at first to detect spam with different tokenization method like Word split, EmailLike, N - Grams, Ignore case, Retain case.

Their second approach was to create a two layer stacked classifier where they tried two strategies one where the second layer was of naive bayes and second where second layer was of SVM and the first layer is always of naive Bayes in both cases. If the first layer i.e. Naive Bayesian records probability higher than 65% then it is set to further input to second higher layer for classification.

3.3 IMPROVING STATIC SMS SPAM DETECTING BY USING NEW CONTENT BASED FEATURES BY AMIR KARAMI AND LINA ZHOU

Abstract

They used Linguistic inquiry and word count to include 80 different feature of linguistic process and with that they used various algorithm. Among all of them boosting Random Forest and SVM showed better performance with accuracy of 92% and 98% respectively.

3.4 LITERATURE REVIEW CONCLUSION

Here we conclude that there are many other methods that have been applied by others like, Naive Bayes, two stacked classifier, using a LIWC with classifier, SVM, Random Forest. But out of all that SVM performed very well.

4. PROPOSAL

4.1 ARCHITECTURE

There are two architecture.

1) Android based Client Architecture
2) Flask based server Architecture

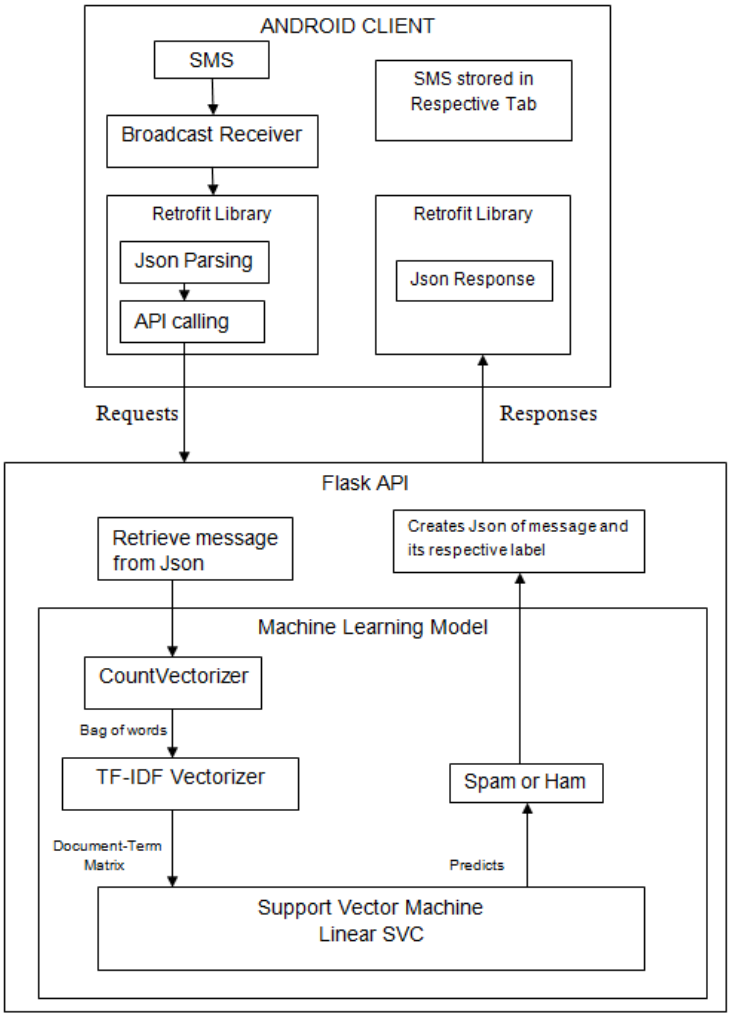

Figure 3 Project Architecture

4.1.1 Android Architecture

- First of all Application connect to the firebase for Authentication using Email.
- Once logged In user have to give following permission:
  - Read SMS
  - Receive SMS
  - Internet
  - Read Phone State
- When the user receives the SMS it will be detected by OnReceive() method of Broadcast Receiver and it will fetch the content of text message and contact No. and pass to Retrofit. Retrofit converts message into Json object and send it to Flask server using API.
- When server responses it will be handled by Retrofit and the message and predicted label will be fetched from Json.
- Message will be stored in respected Tab(Spam or Normal).

4.1.2 Server Architecture

- The server accepts Json object and abstracts the string of the message.
- Then the message is passed to CountVectorizer where it creates token counts and generates Document-Term Matrix.
- Document-Term Matrix is further passed to Tf-Idf Vectorizer where an array of samples and features is generated of the message.
- Generated matrix will be passed to SVM classifier, which will predict the label for that message.
- Jsonify will convert the message and its label to Json format.
- Converted Json will be sent back to Android client as a response to its request vai API.

4.2 FEATURE EXTRACTION

- For this project we use a dataset of UCI Machine Learning Repository which consists of 5574 messages including spam and not spam. Extracting features will result in 7726 features.

CLASS	TEXT
ham	Ok lar... Joking wif u oni...
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
ham	U dun say so early hor... U c already then say...
ham	Nah I don't think he goes to usf, he lives around here though

Table 1 Dataset from UCI Machine Learning Repository

Number of Spam messages in data set : 747

Number of Ham messages in data set : 4825

Label	Percentage in Dataset
SPAM	13.40%
HAM	86.60%

Table 2 Dataset Statistics

- Feature extraction is done using CountVectorizer and TfidfTransformer of scikit-learn library.
- CountVectorizer Converts a collection of text documents to a matrix of token counts. This implementation produces a sparse representation of the counts which is called Document-term matrix.
- TfidfTransformer converts a collection of raw documents to a matrix of TF-IDF features (TF = Term frequency, IDF = Inverse Document

Frequency). Apply Term Frequency Inverse Document Frequency normalization to the sparse matrix of occurrence counts by CountVectorizer.

4.3 CLASSIFIER

We apply LinearSVC classifier of SVM to for training the data. The advantage of using LinearSVC over SVM is that the training complexity in LinearSVC is O(m x n) where

'm' = number of training instances

'n' = number of features

while the time complexity of SVM is usually between $O(m^2 \times n)$ and $O(m^3 \times n)$ this means that it gets dreadfully slow when number of training instances gets large (e.g. hundreds of thousands of instances).

In LinearSVC we use 'C' hyperparameter to control the balancing. The higher the value of C leads to a narrow margin that leads to fewer margin violations and lower value of C leads to wider margin which leads to more margin violations.

4.4 ACCURACY

Accuracy is defined as the number of true positives (TP) plus the number of true negative (TN) over the total number of sample (N).

$$accuracy(total) = (TP + TN) / N$$

Here, we have used two SVM classifiers objects one with cross validation and other without it. With cross-validation we divide the dataset in 3 parts (default fold for cross-validation), and now we compare the average accuracy of cross-validation with accuracy without cross-validation.

Average accuracy with Cross-validation: 0.981521349121

Accuracy without Cross-validation: 0.987443946188

4.5 CONFUSION MATRIX

It is a specific table layout that allows visualization of the performance of an algorithm, Typically for supervised learning. It row in the matrix represents the instance in a predicted class while each column represents the instances in the actual class.

N=1115	Ham	Spam
Ham	951	11
Spam	3	150

Table 3 Confusion Matrix

C[0,0] contains True negatives i.e. True Not Spam detected as Not Spam

C[0,1] contains False Positives i.e. True Not Spam detected as Spam

C[1,0] contains False Negatives i.e. True Spam detected as Not Spam

C[1,1] Contains False Positives i.e. True Spam detected as Spam

5. LIMITATION AND FUTURE EXTENSION

5.1 LIMITATION

As it uses supervised machine learning so the amount of dataset applied to it is very less and thus sometimes a legit message from a known person can also be classified as spam due to content of the message.

5.2 FUTURE EXTENSION

We can use the application to gather sms from the user's device on user's permission and use the data to train our classifier very accurately and in future we can also separate Bank messages, Promotional messages, Spam messages, and various other types. And to make this more efficient we can use LSTM Neural network for text classification.

6. CONCLUSION

Here we conclude that by applying machine learning techniques we can filter out the spam messages and enhance the usability of mobile and make users free from unwanted messages