Detection of Potential Societal Threat In a Forwarded Message

[1]Pranali Nunes, [2]Parag Nar, [3]Sakshi Nakashe, [4]Prashant Katarnavare

[1]Assistant Professor, [2,3,4]UG Student, [1,2,3,4]Department of Information Technology, Fr. Conceicao Rodrigues College of Engineering, Mumbai, Maharashtra, India.

[2]27pnar@gmail.com, [3]sakshinakashe@gmail.com, [4]prashantkatarnaware@gmail.com

Abstract - Rapidly emerging new technologies generate exponential data on a daily basis. In this scenario, it becomes increasingly difficult to distinguish between the true and the fake data. Also, in this age of data sharing, there is no way of testing the genuinity of the data under consideration. All you can rely on is the source from which you obtained the data. However, that source may not be the creator of that concerned data. We can extend this analogy to social media message forwards. There have been lots of serious issues revolving around "WhatsApp" forwards. If a 'general information' message is forwarded, there is no way of knowing if the message is genuine or fake. Whatsapp also tried to make things easier by marking messages as 'forwards' but this does not complete the need. We need to know if the message which is blindly being circulated is harmless or harmful as long as the general societal health is concerned. One workaround to the problem is that we can classify the WhatsApp forwards as harmful and harmless. If the message is harmless then it is fit to be a forward.

Key Words: *messages, Data Mining, Support Vector Machines, Data Analysis, forwarded messages, Classification*

I. Introduction

Social media has become very powerful in today's generation. It is used by people of almost all ages and all disciplines. It has brought the world closer and given an opportunity to connect with friends, family, and others.

As we all know, it has become handy to communicate with message conversations. We all receive and forward messages. It is a common phenomenon which we don't even realize taking place. This is one of the quickest media for sending messages than ever before. We are living in an era where digitization is on its peak

We also know that if we are blessed with something so good, it naturally comes with some side-effects. One of the side-effects of this golden age of information and message exchange is the quick spread of rumors. We know how fast a wrong message gets circulated and what consequences it can have in a country where we live.

We would just like to throw light on the recent incidents happening because of social media forwards. These incidents of human assaults because of mere forward messages prove that we need to develop tools to increase the maturity of the people using this tool which can be used both, constructively as well as destructively.

II. Literature Survey

Identification of Relations from IndoWordNet for Indian Languages using Support Vector Machine[1]

This paper describes the inspection of the SVM classifier for identification of relations in Indian Languages apart from English.

SVM was successfully implemented for identification of relations in Indian Languages from the IndoWordNet[1].

This aims at implementing SVM in the field of text classification. This paper was used in the context of SVM to study the implementation of SVM to address the problem of text classification.

Sentiment Classification based on Ontology and SVM Classifier[2]

SVM has been used for sentiment classification by making SVM act as a binary classifier to classify sentiments

In the document level and sentence level sentiment classifications assume that each document or sentence focus on a single object and contains only one opinion or opinion from a single opinion holder

SVM is better at text classification and use of ontology makes SVM suitable for the application[2]. This reference was used for studying the construction of a binary classifier by studying the values and implication of intricacies of various parameters concerned with the algorithm like regularization parameters and kernel types.

User Intention Understanding From Scratch[3]

The main evaluation criterion of the performance is the prediction accuracy of all users in the test set. We can observe that on overall, the method Threshold-V achieves the best accuracy of 67%.

The drawback of the system is that the use of a numerical classifier led to unwanted overheads.[3] This reference gives a thorough procedure for the implementation of SVM algorithm along with the procedures to be followed for data pre processing which will prepare the data to be given as input to the SVM classifier.

Tkinter 8.5 reference: a GUI for Python[4]

This paper describes the Tkinter widget set for constructing graphical user interfaces (GUIs) in the Python programming language. It Includes coverage of the ttk themed widgets[4]. This paper was referenced for creating the user interface with python's tkinter module.

ViPEr, a Visual Programming Environment for Python[5]

In this paper we describe a Python- and Tkinter-based visual-programming environment called ViPEr. This tool enables non-programmers to build computational and visualization networks

interactively. Computational nodes can be placed onto a canvas and their input and output ports can

be connected using the mouse. The connections between the nodes define a directed graph that will be used to propagate data and trigger the execution of nodes that have new input data[5]. This tool is under consideration for the user interface enhancements by enabling reusability of the available code.

III. Proposed System

The proposed system will be a portal. The user will use this portal to type the message which has to be sent. Then the user will click on the "send" button. As the user clicks on the send button, the system will build a classifier in the backend using the dataset provided. The system will take the user's message as a test tuple and will predict the class of that message.

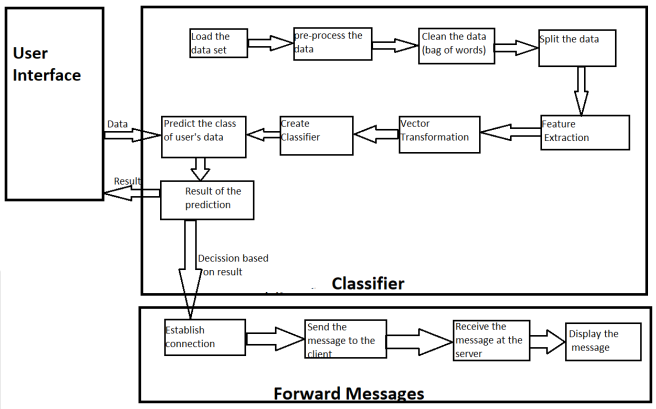
If the message is detected to be "safe", then it will be forwarded to another user. In that case, a second terminal will open up (as we are simulating in the localhost) and the message will be forwarded for the new user to view.

Now, in case if the message is classified as "not safe" by the classifier at the backend, then the user will not be allowed to forward the message. The system will display an alert saying that the message which the user intends to forward it not safe for the societal health.

A. OBJECTIVES

1) We aim to achieve the following through this project:
2) Classify a forward message based on its potential societal threat
3) The first step is the classification
4) A useful tool for the cyber cell to detect potentially harmful forwards and try to stop them
5) Can be used on an individual level
6) Design a system with a highly usable UI

B. FLOW CHART



Modules explanation

User Interface: Provides facility to the user to access the system using a user friendly interface

Load data: Create the data provided by the users is now loaded in the dataset.

Split: The dataset is split into training and testing dataset.

Create classifier: create a classifier module will create a classifier which follows a specific algorithm to classify objects. In our case, it will create SVM classifier with a linear kernel and with the specified value of c. C is a regularization parameter that controls the trade-off between achieving a low training error and a low testing error.

Train classifier: Train classifier will take the classifier created in the previous step and the training data set which you have provided to "teach" the classifier how to classify the data.

IV. Methodology

The system makes use of SVM classifier for accurate diagnosis of diseases. SVM is a supervised binary classification algorithm. It trains a classifier using the data set provided to the algorithm. This classifier aims at classifying all the tuples given to it, which follow the norms of the way in which the classifier has been built.

A. WORKING

The classifier considers "n-dimensional" space while constructing a classifier and subsequently while classifying "foreign tuples" which are given to the classifier as a "test".Here, n is the number of classes or labels into which the input from the user has to be classified by the system/classifier.

SVM is a numerical classifier i.e. it can only classify numerical data. The data set which we have considered is a textual data set, in the form of text files. Hence, here we are dealing with the problem of document classification. Now, SVM being a numerical classifier, needs data input in numeric format only. Hence, we have to convert the documents into vectors. This process is called Vectorization. The process used for vectorizing the documents or, for converting textual documents into mathematical vectors is called TFIDF (Term Frequency-Inverse Document Frequency) vectorization. This process gets rid of "local stop words" in the document and then converts each document into a vector.

SVM takes each vector and processes it in such a way that it can be represented as a point in the n-dimensional space. After representing each document as a point in the n-dimensional space, SVM tries to separate the data which is being represented. This data is separated by drawing a hyperplane.

A hyperplane is an n-dimensional plane which separates the data plotted on the space according to their labels.

SVM constructs a hyperplane by considering appropriate distances from farthest points in each label. It finds "gutter values" to find the hyperplane.

These gutter values consider points which are farthest from the rest of the points grouped together in that particular label. The hyperplane is plotted by taking these gutter values into consideration.

The hyperplane accomplishes the actual task of classification.

Once the hyperplane has been constructed, the classifier is ready to classify "foreign tuples". When a tuple(which follows the way in which the classifier was built, but is serendipitous enough pose a new challenge to the already built classifier) is given to the classifier, it plots the tuple on the already existing n-dimensional plane (which also consists of all the training data plotted over it and separated by a hyperplane).

Then the algorithm checks on which area the new tuple has been plotted. Because the n-dimensional space gets divided into different areas when it is separated by a hyperplane.

Each area defines its own label or class, one of which the classifier is supposed to predict. Depending upon the current area on the n-dimensional plane where the new tuple has been plotted, SVM finds the class of all the other points in that area (because all the points in that area of the space have the same label).

This is the class of the "foreign tuple" which was given to the classifier for prediction.

V. IMPLEMENTATION DETAIL

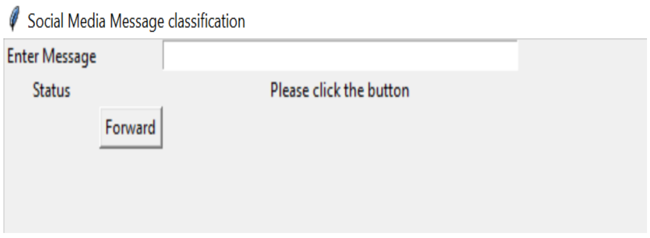


Fig 2. Sender UI

This is the user interface of the sender. The user interface of the sender consists of the following elements:

- Labels

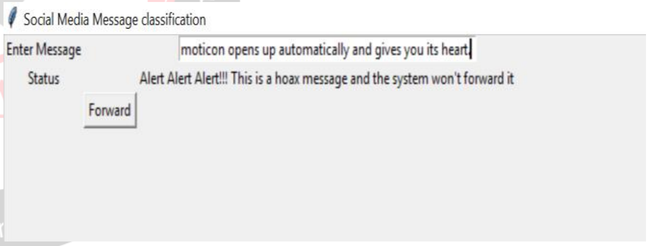
  These elements of the user interface have been used for enhancing the usability of the user interface by embedding instructions on it.

- Entry

  This element of the user interface has been used for taking the message to be classified from the user of the system. It is a text box for taking data from the user.

- Button

  This element of the user interface enhances the interactivity of the interface. It triggers the backend working of the application when clicked.

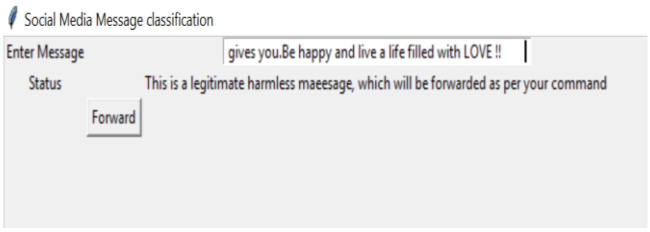


moticon opens up automatically and gives you its heart

Alert Alert Alert!!! This is a hoax message and the system won't forward it

Fig 3. Message detected as Dangerous

The user enters a message and hits a forward button. The message is classified as harmful and an alert is displayed on the screen, informing the same to the user. The user cannot share the message which has been classified as harmful by the system.

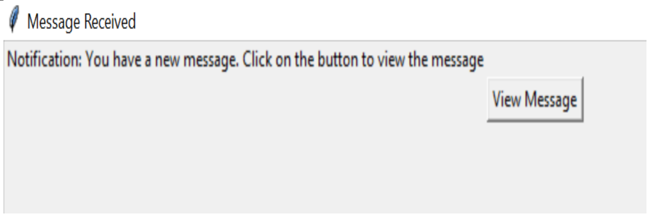


gives you.Be happy and live a life filled with LOVE !!

This is a legitimate harmless maeesage, which will be forwarded as per your command

Fig 4. Message detected as unharmful

The user enters a message and hits a forward button. The message is classified as harmless and the user is told that the message has been sent.


Message Received

Notification: You have a new message. Click on the button to view the message

Fig 5. Receiver UI

The user interface at the receiver side shows a notification which tells the user that a new message has been received. clicking on the view message button will display the message which the receiver has received from the sender.

Technology Used:

Front End: Python

Back End: Python

VI. CONCLUSION

The problem has condensed down to text classification. Text classification is an example of machine learning in Natural Language Processing (NLP). Document classification is a branch of Machine Learning under NLP which makes the text easier to sort. The system will use the SVM algorithm for classifying documents based on NLP. The system falls under the category of a binary classifier.

REFERENCES

[1]Ziheng Wang, Yonggang Qi, Jun Liu, Zhanyu Ma, "User Intention Understanding From Scratch", 2016

[2]Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin, "A Practical Guide to Support Vector Classification", 2016

[3] Khin Phyu Phyu Shein, Thi Thi Soe Nyunt, "Sentiment Classification based on Ontology and SVM Classifier", 2017

[4]John W. Shipman, "Tkinter 8.5 reference: a GUI for Python", 2012

[5]Michel F. Sanner, Daniel. Stoffler and Arthur J.Olson, "ViPEr, a Visual Programming Environment for Python", [Coon et al. 00] Sophie I. Coon, Michel F. Sanner and Arthur J. Olson. Re-usable components for structural bioinformatics. In Proceedings of the 9th International Python Conference. Xx-xx (2000)

[DX 93] Technical Report, ICM Corporation, p. various, February 1993

[Macke et al. 98] Thomas J. Macke, Bruce S. Duncan, David S. Goodsell and Arthur J. Olson.

Interactive modeling of supramolecular assemblies. J. Mol Graph and Model. (1998) 16, 115-120.

[Python at TSRI] http://www.scripps.edu/~sanner/python

[Upson et al. 89] C. Upson et al. IEEE Comput. Comput. Graphics Appl. 9(4), 30-42 (1989)

[Sanner et al. 96] M.F. Sanner, J.C. Spehner, and A.J. Olson. (1996) Reduced surface: an efficient

way to compute molecular surfaces. Biopolymers, Vol. 38, (3), 305-320.

[Sanner 99a] Michel F. Sanner. Python: A Programming Language for Software Integration and Development. J. Mol. Graphics Mod., 1999, Vol 17, February. pp57-61.

[Sanner et al. 99b] Michel F. Sanner, Bruce S. Duncan, Christian J. Carrillo and Arthur J. Olson.

Integrating Computation and Visualization for Biomolecular Analysis: An example using Python and AVS. Proc. Pacific Symposium on Biocomputing. (1999) pp 401-412.