Prioritizing Urgent Messages on Instant Messaging Platforms

Harshaa Vardhan R, Mulugu Srinivas Shashank, Saad Yunus Sait

Abstract *Instant Messaging (IM) has become integral to everyday life, with a plethora of information exchange between users. Whatsapp, an IM service has about 65 billion messages being exchanged between users, in a single day (2017). Whatsapp is just one of several such applications which share countless amount of messages where all messages received by a single user are not in equal priority. With an ever-increasing volume of messages that a person exchanges, there is a need for prioritizing important messages. The messages that are more importance or urgency is predicted with a classification algorithm. Moreover, the trained classification model is wrapped as an API service, allowing messages to be classified and predicted in real time by calling the REST API service. The paper provides a mechanism to predict important messages that require more attention or urgency and thereby, offering a way to maximize productivity of a person to check important messages.*

Keywords —instant messaging, prioritization, urgent mes- sages

I. INTRODUCTION

Instant Messages contribute a significant portion of any person's time and not all messages are equal in priority for a user. An average person receives about 43 messages per day from different recipients, with each message having varied importance. Some messages require more attention like a message about a meeting schedule or emergency messages. In such cases, the urgent messages should be prioritized over the other non-urgent messages. The paper firdt reviews related papers and analyzes the varying mechanisms and frameworks by other papers. The literature survey is split based on the sub processes like collection of data, generating features it analyzes how these works performed each step in the process shown in Fig. 1. The proposed system, explained in this work hopes to achieve the distinguishing attribute of assigning priority for such instant messages. This allows the user to cater to urgent or important messages and thereby improve his/her productivity. The trained model has been wrapped as an API to allow predicting messages by just sending a particular message to the trained model and receiving the prediction - whether the message is urgent or not in real time. An implementation of a chat application, is used as a means to collect messages from the users of the application. Further, the chat application is built with features of a typical instant messaging applications and offers a way to label the messages as urgent messages. This is important because data for training machine learning models are not easily available. The classi- fication implementation module defines and illustrates the set of processes, the architecture, and the process flow used and implemented. It includes the complete background processes followed in developing the classification model - from the labelling mechanism followed, generating features, training the classification model and comparison between different models, to predicting urgent messages.

Revised Manuscript Received on September 22, 2019.

Harshaa Vardhan R, Computer Science Department, SRM Institute of Science and Technology Chennai, India harshaavardhan23@gmail.com

Mulugu Srinivas Shashank, Computer Science Department, SRM Institute of Science and Technology Chennai, India, mulugushashank3@gmail.com

Saad Yunus Sait, Computer Science Department, SRM Institute of Science and Technology Chennai, India, saad.y@ktr.srmuniv.ac.in

II. OBJECTIVE

- To gather messages and message meta-data from a chat application
- Classify the instant messages of the user based on vari- ous parameters/extracted features to determine priorities taking into account the message exchanges and activity of a user.
- Predict urgent or high priority messages
- To construct an API service for predicting the message importance

III. PROPOSED SYSTEM

The system gathers user text/instant messages in real time through a designed chat application with registered users along with public repositories. The messages contain vital informa- tion such as sender name, message received time, number of messages and the underlying message contents, which are further processed to fit into a classification model. The model determines a urgent response variable, which indicates the priority level of the received message. The classification model is wrapped into a REST API service which predicts if a particular message is a urgent message.

The figure 1 shows the processes followed by related work. The process flow starts from collection of data and how related literature have implemented the collection process. The process flow starts from the collection to the evaluation and statistical analysis done by the papers. Further, each sub processes are discussed in detail with comparison and insights.

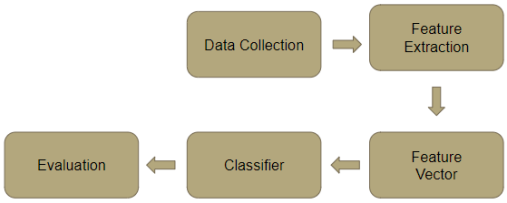

Fig. 1. Sub-process Flowchart

A. *Data Collection*

Collecting data from private messages are quite difficult and involves the user's consent and proper security mechanisms to be followed. There are various ways to collect data from such IM applications. The works of [1] defines a data collection methodology involving creating a curated list of public What- sapp groups that take in users. The researchers created public Whatsapp groups, which could be joined by any user. This method allows analyzing group conversations of the members and collection of such data, while one–one conversations aren't collected. Also, whatsapp has a limit on the number of users in a group and therefore, requires creation of many groups to facilitate the conversations of distinct users. The group's messages are then manually exported to an analytical tool to offer insights. [2] uses a data collection methodology where they developed a software tool that backs up entire data included messages and groups of an Android application (Whatsapp). They further used an encryption algorithm to mask the user's details to preserve the privacy of the individual from which the data is obtained. They obtain more data about participant's age, gender, demographic information and asked them rate their sociability level based on a five point rating system. They mention the drawbacks in getting participants to share their information, the main reason being a privacy concern, even though their identities are masked or encrypted. [3] uses the approach of a surveying the people/users of the IM service to offer insights and analyze the trends. The above data collection method allows to generate categories and patterns that evolve, which can be further used for feature extraction or information generation. [4] gathers data from publicly available datasets like DSTC 4 [5], MRDA [6], SwDA and [7] that aid in training the model for short text classification. [8] makes use of messages obtained from Facebook as a dataset and filter the unwanted messages, defined by the user.

B. *Feature Extraction*

The feature extraction process involves parsing and process- ing the collected data, to obtain relevant and useful features. These features are intended to be informative and distinct to facilitate the classification process. In text classification and processing, the usual way is to extract features from the messages or texts involved. The usual way is to extract features by reading and processing the texts and messages involved in the conversation. [2] deems this as a privacy concern and gets around it by extracting features from the conversation characteristics and general message information. Their objective being, not to violate the privacy of the users and also to reduce storage cost. The features used by them include message length, temporal properties (sent/received time, time elapsed between messages), size of the group, gender, etc. The 2 features extracted from [3] include type of messages sent/received, time period of interactions, the people involved in the conversation i.e. who they converse with, intents of the conversations – chatting, planning/coordination, work related. [3] uses these intentions to obtain a clearer interpretation of why and what users converse about in the messages. Chatting involves regular exchange of personal in- teractions. Planning/coordination is one of the most significant differentiating factor for using such services. By using the groups feature, (exchange of messages from many users at a time) coordinating and planning activities like going out for a movie or attending conferences becomes easier. The features used were extracted through surveying the users and the researchers categorizing based on the data. [2] does a similar implementation by categorizing the intention of user's usage. The categories included by [3] are: work, family and friends. [4] uses transcribed words as well as previously predicted dialogue acts as features for all the CNN, RNN as well as Naive Bayes model. Transcribed words are nothing but the conversion of the audio converted into textual format. [8] uses endogenous (interior) features of the short texts as well as the exogenous (external) knowledge from short texts (transcribed words) which are related to the context of the message. Three types of text features are used: Bag of Words (BoW) (Bag of Words: creates a vocabulary set of all words in a document), Contextual Features (CF) and Document Properties (DP) where BoW and DP are obtained from the information available from the text message. Some of their features include a collection of words like bad words, good words, capital words (the number of letters capitalized in a message), exclamation marks (the number of occurrences of exclamation marks in a message), punctuation characters, etc. The collected data after cleaning and pre-processing, is further used to extract significant features, to facilitate subsequent processing and for implementation of an accurate classifier model.

C. *Privacy Measures*

Instant messages contain sensitive information and conver- sations which pinpoint certain individuals along with their phone numbers during collection of data. It is therefore necessary to implement certain privacy and security measures while collecting such data. Encoding or encrypting the phone numbers and usernames are the common practice and most efficient. Almost all the researchers collecting IM data make use of this practice. They abstract the user phone numbers to avoid misuse. [2] goes a step further by completely skipping the content of the messages exchanged in conversations and makes use of general message and group information to build features, though, losing some important

features, makes sure that the message integrity and trust is maintained. Encryption or encoding the username and phone numbers allows the researchers to not know about the details of the users and exclude human bias from the process.

D. *Text Classification*

[4] uses a model consisting of two main parts: short-text representation using various RNN(Recurrent Neural Network) and CNN(Convolution Neural Network) models and classifying the vector representation of the current short text using the information from the current as well as the previous short texts. The RNN short-text representation is done by using a variant of RNN called LSTM (Long Short Term Memory) by giving an input of the words in the short text to different layers and short text representations are computed by using various pooling strategies (pooling is used to lower the dimensions of all the extracted features and lower overfitting) like last, mean, and max pooling (last pooling picks the last vector, mean pooling averages all the vectors and max pooling picks out the vector with maximum value). The CNN short-text representation done by using the ReLU activation function (Rectified Linear Unit is an activation function which is zero for all the values less than 0 therefore it is sometimes called half-rectified) to obtain a scalar feature (ct) from the word vectors (Xt). Then the short-text representations are finally calculated by in the max pooling layer by using the previously obtained scalar features. The classification is done by passing the short-text representation (st) to a feed-forward Neural Net with twolayers which gives the class representation of the short-text. The first layer of the Artificial Neural Network (ANN) takes the input of the short text representation, hyperparameters and gives out the class representations of the short-text. The second layer takes the input of the class representations from the previous layer and gives an output - probability of the text belonging to a specific label. In [8] the text representation is mainly done by using the Vector Space Model (converts textual information into vector form for eg., binary format or indexed words). Radial Base Function Network (RBFN) [9] is also used for the short text classification because it is proven to be very effective with noisy data and intrinsically ambivalent class data. A RBFN is a non-linear classifier which measures the distance (eg., Euclidean) between the input examples training set to the find the similarities and finally outputs a classification.

E. *Evaluation*

In [4] the most frequent class is predicted by the models. Various RNN and CNN models are implemented by using CNN with an accuracy of 65.5% with the DSTC 4 [5], 84.6% with the MRDA [6] and 73.1% with the SwDA [7]. [4] was able to get an accuracy of 66.2% with the DSTC 4, 84.3% with the MRDA [6] and 69.3 with the SwDA [7] by using LSTM.
[8] was successfully able to predict the violent messages with a precision (P) of 92%, Vulgar (69%), Offensive (86%), Hate (58%) and Sex (75%). The average precision being achieved is 76% with a macro-average Recall of 38% and F-measure of 51%.

F. *Insights and Findings*

[1] offers insights like, The average image size is around 101 KB, while the average video size is very small-4.6MB. This gives information that most of the attached media sizes are on average, of small size. Whatsapp also uses image compression algorithms to compress the images, which could lead to smaller sized images. The average length of the text messages were 582 characters (median 136 characters). The conversations in the group carry messages from individual users where each message length would differ in number of characters present and length of the message. An average of 582 indicates that most 3 messages are quite long, unlike one– one conversations in the IM services. Each group has different levels of interactions and quantity of messages exchanged. [1]found that 30% of the groups have under 1000 messages during the 6 month measurement period. This indicates that not all group activity is the same and will differ based on a number of variables. [2] shows that the time duration of message exchanges were : 93 percent of messages were sent during day time and rest was, around 12 - 8 am. The time of day in which messages exchanges happen allow us to understand the way the users send or receive messages during a specific time of the day. Most messages i.e. around 93% of the messages are sent during the day. A high percentage of the messages in [2] were exclusively text messages while only a meagre percent included file attachments or links. The findings are in line with what [1] had concluded that, most of the conversations include textual communication and the media attachments were minimal in comparison. Media attachments include – images, video, audio, location or con- tact information.It is interesting to note that Men received and sent almost 15% lesser than women. This elucidates the participation and gender demographic in terms of their usage. Influence from peers and communities appeared to increase a non user to adopt such services. Women seemed to interact more in comparison to their male peers. When surveyed among the people, it was found that, the intents for using such services were found to be - having conversations, coordination of activities within groups, exchange of personal news, being active in groups, business related communications, and receiving ads.
The methods used by participants in [3] study denoted that there are very few ways to handle a barrage of notifications from Whatsapp and Short Messaging Service (SMS). The ways include switching off their phones or turning off audio. Results of [3] indicates that many people use WhatsApp and SMS for business,but do not use the methods above in fear of losing out clients or business prospects. [4] notes that the information in the short text representations is substantial and generalization than in class representations i.e., the information that can be elicited from short text representation is far greater than class representations.

Usage of sequential information on both the

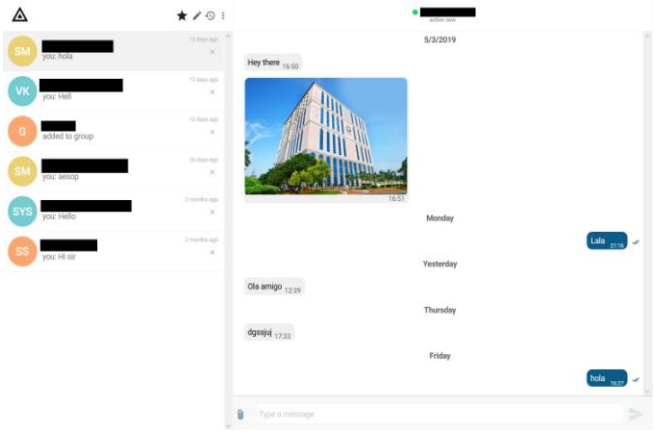

Fig. 2. Direct Chat in Chat Application

short text representation level and class representation level does not help and may even lower performance, especially in the case of class representations which do not contain rich information and may propagate inaccuracies from previous mis- classification. In [8], messages with ”hate” and ”offen- sive” content are not easily identified by term based approach because of the content being complex i.e., for distinguishing a message as hateful or offensive on the basis of the words used.

G. *Current Challenges*

The studies show that the acquisition of instant messages from users is often viewed with skepticism from the users as they have privacy concerns. Also, some IM services (What- sapp) use end-end encryption, where complete encryption is done on the messages. As [3] highlights that even in the contemporary world, there exists very little mechanisms to handle the vast amounts of messages on a mobile device. There is also a growing concern on handling forwarded mis- information and fake news through instant messages.

IV. CHAT APPLICATION IMPLEMENTATION

To streamline the process of data collection we have moved to using a chat application Chat21, created and supported by Frontiere21. Chat21 is an open source, easy to implement and multi-platform (Android, iOS and Web) chat application which was built on Firebase. ([10]) Firebase is a mobile and web application development platform.

A. *User Registration*

Any new user is required to sign up using his/her email address, password, first and last name. The user's details are stored in the firebase realtime database. The user details stored include the above mentioned details, along with newly created User ID (UID), imageurl and timestamp as shown in Figure 3

UID - It denotes the unique user ID generated for each regis- tered user. ImageURL - It contains the link to a user uploaded profile picture. The url points to an image uploaded in Cloud FireStore, stored in a bucket. Timestamp - Time of registration represented in Unix Epoch Format Once registered, the user can login using the registered email address and access the application.

B. *Direct Chat*

Direct chat enables one-to-one messaging between two users. The supported formats are: text, image and binary. Lets the user start a new conversation by viewing a list of contacts which also comes with a search feature to find the recipient of the message with ease. The application sends outbound messages even when a user's device is disconnected.

The application saves the messages locally and sends it when the device reconnects. The left side of the direct chat (refer fig. 2) contains the contact names of the people with whom the user has conversed with. the selected conversation is highlighted and the messages exchanged between the user and the selected contact are shown in the conversation page - right hand side of direct chat (refer fig. 2).

C. *Group Chat*

Using group chat makes it possible to communicate with all the members of a group. All these messages are stored on the real-time database. The messages of this type are uniquely identified in the backend by their channel type ”group”. Group Chat 4 is similar in design of direct chat(2 but the selected conversation is a group called ”General” as shown in figure

4. The messages exchanged between the group members are present in the conversations pane with information on the group members - name, joining timestamp, user id, info about sender and receiver, group names in the firebase backend database - figure 5.

D. *Cloud Storage and Database*

Details regarding the users and their corresponding activ- ities and interactions are stored in Firebase Database. The database is a NoSQL database with read/write operations can be performed with using the REST APIs of the Realtime Database. The Cloud Storage acts like a folder system where the attachments are stored. The images being sent in messages,

profile pictures of the users are stored using the Firebase Cloud. The path of the image i.e the url pointing to this cloud folder is saved in the database for referencing.

The parent node, Ionic Chat has child nodes - contacts, groups, messages, presence and users. Each user identity is hashed to ensure privacy. In representation of a messages contains (6) :

Contacts - It contains the details of all the users and their corresponding details such as email, last name, first name, user id and timestamp.

Groups - Contains information on the groups created and their corresponding details such as creation date, name of group, members and member info.

messages - The messages exchanged in pertaining to each groups are stored under this node.

Presence - The last accessed date and time of each user Users - All user information along with their conversations, messages exchanged and groups that their part of.

E. *Other Functionalities*

- Notifications: Users will get push notifications for all the messages that are received (on all the platforms) when the application is running in the background.
- Profile Photo: There are options to set a custom profile pictures.
- Archive: Important Chats can be archived or stored for viewing later.
- Delete Conversations: Chats which are no longer required can be deleted.

V. BASIC CONCEPTS IN TEXT CLASSIFICATION

A. Lemmatization and Stemming

Stemming and Lemmatization are ways to implement text processing, highly common in natural language processing for mining the text. The messages exchanged contain texts which have features that required to be extracted. The messages are first broken into words or tokens as stemming and lemmatiza- tion requires the words to be split into tokens. Stemming aids in pre processing before extracting the features. The paper makes use of Stemmer function to strip the suffixes of the words to obtain a simple root of the word. As shown in figure 8, ”playing”, ”plays”, and ”play” all denote the same underlying word of ”play”. Stemming achieves the root a word by removing the suffixes after play - like ”ing”,”s” . Stemming is relatively faster than lemmatization as it only involves removing or striping the suffixes, offering fast mining.

Lemmatization of the words is used along with or as an alternative to Stemming. Lemmatization reduces the words into their root words or ”lemmas” but with greater precision. The unprocessed words are called inflected words, where such inflected words i.e. the words present in messages are converted to their root words. The paper uses parts of speech tagging, where the words are tagged along with their parts of speech - noun, verb, pronoun. This enables the lemmatizer to understand the words better for further reduction to their root words. Lemmatizer looks through a corpus of words that contain root words to obtain best results. It is important to note that lemmatization requires a sufficiently high time to lemmatize, when compared to Stemmers but offers better reduction to root words.

B. TF-IDF N-gram

The Term Frequency (TF) of a word w in a document d just counts the number of times the w is present in d. It can be computed by using the equation below:

$$\mathrm{TF}(w, d) = \mathrm{frequency}_{d}(w)$$

The Inverse Document Frequency (IDF) of a word is calculated for a word w over the entire size of the corpus D. The IDF is computed by using the following equation.

$$\mathrm{IDF}(w, d, D) = log \frac{1 + |D|}{1 + df(d, w)}$$

Fig. 4. Group Chat in Chat Application

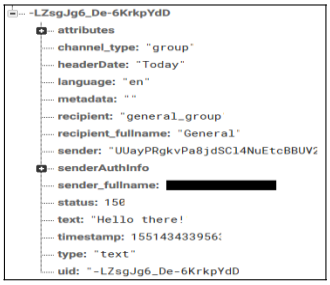

Fig. 5. Group Chat representation in Firebase

Numerator of above equation signifies - size the corpus (i.e., number of documents) and the denominator is the frequency of the word ”w” in the number of documents in the corpus containing the word. The computed logarithm value of the aforementioned fraction for each of the words in the corpus will be lesser if the words occurs more frequently in the corpus. For example, words like ”a”, ”an”, ”is” and ”the” will occur almost throughout the corpus but will not contribute to the nuances of a sentence. So such words will have a high document frequency which will reduce the IDF value.

Fig. 6. Single message representation in Firebase

So to calculate the TF-IDF of a word w in a document d with a corpus size of D, the values of TF and IDF are multiplied as shown in the equation below:

$$\text{TF-IDF}(w, d, D) = \text{TF}(w, d) * \text{IDF}(w, D)$$

Based on this equation it is clear that a word that appears frequently in the corpus will have a lower TF-IDF value than

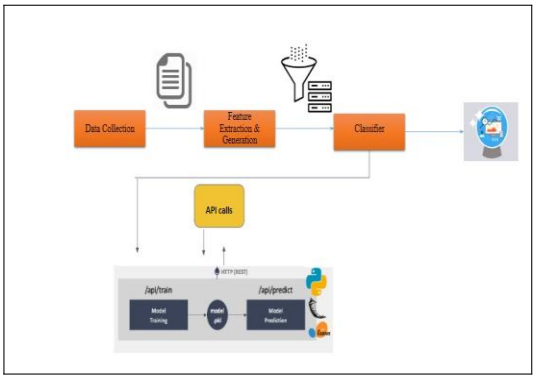

Fig. 7. Architecture of the classification Process

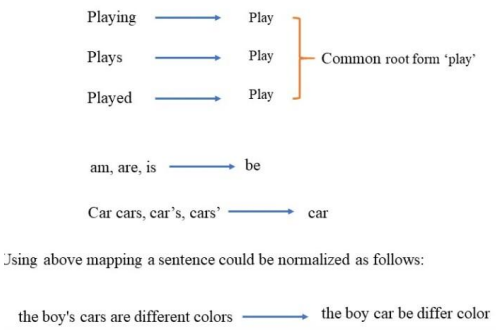

Fig. 8. Example of stemming

others because of the IDF values than more specific words which don't occur as frequently.

calculating the tf-idf values of all the words in the document requires the generation of vocabulary sets. these can be generated by using the n-gram model. a n-gram is called as a unigram when one word is considered at a time in the vocabulary set.

For example consider the sentences, S1="The house is maintained by John". S2="John maintained the house".

The vocabulary set v would be:

{"The", "house", "is", "maintained", "by", "John"}

The bi-grams vocabulary set for S1 and S2 would be as follows:

{"The house", "house is", "is maintained", "by John"}

Similarly, tri-gram, four-gram, etc. can be calculated to improve the accuracy of the model as all the sequences of words are considered.

C. Extreme Gradient Boost Algorithm

Extreme Gradient Boost (XGBoost) is an optimized wrapper that performs gradient boosting using several Machine Learning (ML) algorithms under gradient boosting. Gradient boosting is a supervised ML classification technique which generates predictions by using the ensemble of weak models to create a strong model. These weak models are typically decision trees. The algorithm is built sequentially where each decision tree adjusts the bias and variance by correcting the errors observed in the previous samples by using an arbitrary loss function. A loss function is used to get better and more optimized future predictions by penalizing the model for wrongly predicted samples.

VI. CLASSIFICATION IMPLEMENTATION

Figure 7 shows the overall architecture and process flow of the classification process. The flow starts from collecting relevant data, parsing them to generate features in the Feature Extraction and generation process. Once the data is structured with sufficient features, it is fit into a classifier to make predictions. The predictions can be called using API calls, made from REST API service, which has the trained classifier wrapped for making predictions.

A. Data Collection

The data collection process involves obtaining data relevant to message exchanges, conversations, text message exchanges. The data set needs to include different categories of con - versations or message texts like- personal, business, spam, advertisements, etc. One source of data is from the chat application mentioned in section V section. The messages can only be accessed by using the registered firebase authenticated account. The app, being deployed in firebase enables the the real time database to be exported to a json format, which is further parsed. Along with the messages from the chat application, to gather more records to create a better trained classifier, public datasets mentioned above - NUS Corpus, Maluba dataset is used. The parsing process involves cleaning the individual datasets. The cleaning process involves processing the data sets to remove anomalies, missing data or tuples. Once the the individual datasets are cleansed, they are further combined together in a coherent structure containing the required columns. The parsed data set is labelled as urgent/attention required text in a two factor approach - It is generated by combining the labelled messages from the chat application, maluba and nus

corpus dataset. The labelled dataset is shown in the figure, with each message having a urgent label. urgent messages are labelled ”1” while non- urgent messages are labelled as ”0”.

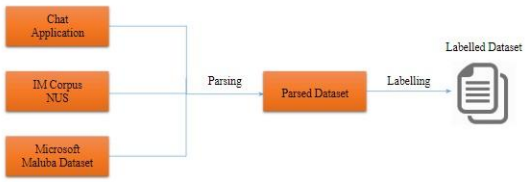

Fig. 9. Data Collection Process

B. Feature Extraction

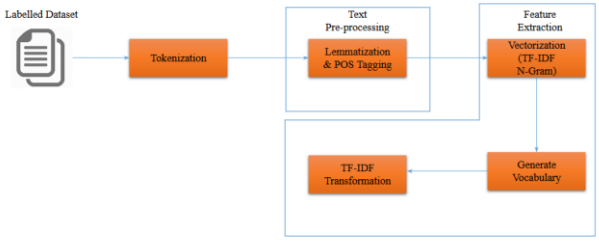

Fig. 10. Feature Extraction

The feature extraction process involves generation of var- ious textual features like splitting the textual data into to- kens (i.e., Tokenization) in the textual features, then tagging the words with their respective parts of speech (i.e., noun, pronoun, verb, etc.). Further, document properties features like word count, average word length, number of punctuation marks used, number of capitalized letters through the labelled textual dataset. In this step, the textual data is also converted into numerical features (a.k.a Feature Vectorization) which can be given as an input to the ML model to generate the predictions. The method used for Vectorization in the current implementation is Term Frequency Inverse Document Frequency (TF-IDF) with N-grams. It is an important process that is required here to understand the significance of a particular collection of words(2 or 3 words) to the document among the entire dataset,by taking N-words at a time. Then the vocabulary of all the significant words is generated and the counts of word tokens are vectorized which can then be given as an input to a ML model to get the predictions.

C. Classification

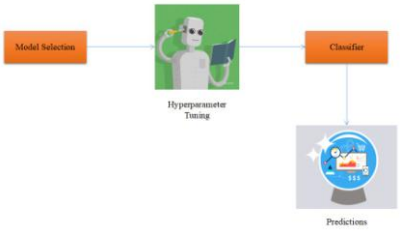

Fig. 11. Classification process

The classification process in any ML work-flow starts with model selection. Among the myriad of the available ML models available, it is vital to select the best model for the problem at hand. XGBoost ([11]) is known to be very effective for text classification problems. The effectiveness of any ML model depends on the parameter selection. For the purpose of selection of these parameters there are several methods like grid search and random search which use cross-validation ([12]). After the best possible parameters are picked, the input file is split into training and testing data. The model is trained over training and tested using test data.

D. Performance metrics

This section include the metrics used to truly comprehend how well the model works in the classification process. Some of them are listed as follows: accuracy, precision, f1-score, recall, confusion matrix, Area Under Curve - Receiver Operating Characteristic (AUC-ROC).

XGBOOST: The evaluation metrics observed in the predictions of the XGBoost Model ([11]) to detect urgent messages (from various datasets as described in Section VII.A) are as follows: Precision: 84%

TABLE I

	precision(%)	*recall(%)*	*f1-score(%)*	*support*
non urgent	83	98	90	8603
urgent	84	31	45	2564
micro average	83	83	83	11167
macro average	84	65	68	11167
weighted average	83	83	80	11167

XGBOOST CLASSIFIER PERFORMANCE METRICS

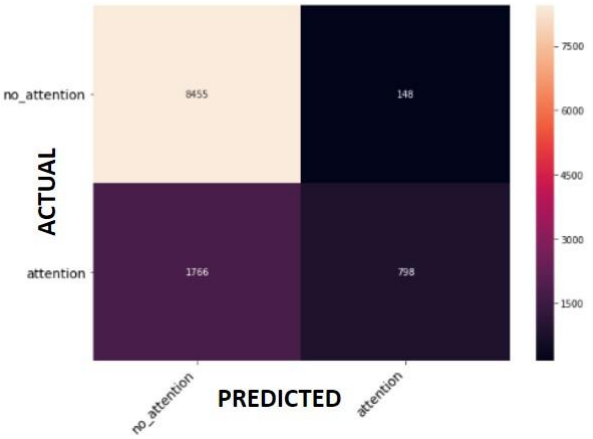

Fig. 12. XGBoost Classifier Confusion Matrix

The classification report for XGBOOST model I, in com- parison with SVM report offers better classification and per- formance metrics. A precision of 84 percent I indicates the correctly predicted messages from the overall observations. Even though the accuracy is high, it is imperative

to look into other key metrics to understand the predictive capability of the model. The recall value being 31% is an important metric for the model, indicating the percentage of correctly identified urgent messages.

AUC-ROC curve:

The Roc curve of the XGBOOST model shows the performance of the classifier at different classifying thresholds, plotted between the True Positive rate and True Negative rate. True positive being the actual and predicted label of urgent messages are same while true Negative is the actual and predicted label of non urgent messages are same. The Area under the curve ranges from 0 to 1, with 0 denoting a bad prediction model and 1 denoting the best. While ideally the AUC-ROC should be 100%, the area is around 65% in the figure 13. In classifying urgent messages, it is significant that the urgent messages are not misclassified and thus, such a metric offers greater insights. The observed metrics prove that XGBoost is an efficient solution to the problem of detecting messages that require the user's immediate attention.

E. Model API Service

The model api service allows wrapping up the trained classifier model as an api (Figure 14), which can take in requests from HTTP API services. The requests are the messages that can be sent through HTTP POST. HTTP Post allows greater security of information and message integrity

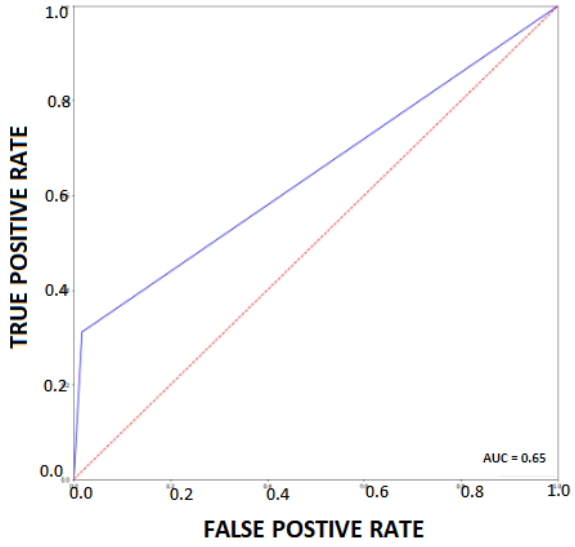

Fig. 13. XGBoost Classifier AUC-ROC Curve

than GET requests. The RESTful API is run through Flask and Python. The back-end support of the REST API is supplemented through a python web application running in the background. This is achieved by the flask web framework for Python, enabling the web application to listen to a particular port in the local server. This acts as the server side script that provides the back-end support for the REST Api. The trained model as shown in the figure of 14 is the trained classifier which classifies a message as urgent or not urgent. Client applications can send messages/texts through HTTP POST requests to the web app to classify the particular message as urgent or not. Once the model is deployed and saved, each REST API call invokes the function to extract and generate features and further, the ”predict” method of the Machine Learning algorithm. The predict method, is triggered after feature extraction to generate predictions. The features are structured in the format required (10 and 11), is triggered to generate predictions. The predictions can be sent as response in JSON format to the client requesting the predictions. The API is designed to receive the message/text that is to be classified.

1) BACKEND MODULE: The trained model described in the previous section is pickled or saved in a model.pkl file. Pickle files allow loading and dumping trained classifiers objects This allows reusing the trained classifier and removes the need to train the classifier for every single request.

The requests sent to the flask app contains text: ”string” or a message in string format. The requests and responses are sent in json format for efficient exchange of information through HTTP. To predict if the particular message is urgent or not, the string must be processed to extract features and be of the format required for the classifier input. The text is passed through the feature extraction phase to generate features and

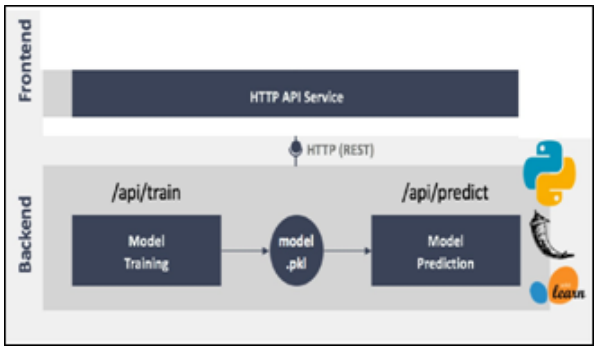

Fig. 14. Rest API architecture

made to fit with the tf-idf vector to generate a sparse matrix containing the tf-idf transformed vector based on the learned vocabulary and combine it with the generated features such as word density, message length, stop words and frequency of special characters

After processing the requests into the input structure required by the model, the binary prediction of urgent or not is made. The prediction is sent as a response to the client app in json format.

2) FRONT-END MODULE: The front end modules can make use of the REST Api using HTTP requests to send mes- sages that can be predicted through the classifier model. Front end modules running through Angular, Javascript, Python or HTML can make use of the MODEL REST API service.

VII. RESULT AND CONCLUSION

Text classification coupled with the handling and processing such messages enable a way to reduce the burden of unwanted messages and texts and give prominence to the more impor- tant messages for users, thereby increasing productivity and decreasing unwanted messages. The paper offers a way to classify messages into urgent or not by training the classifier to detect an important message. The trained classifier is wrapped into a API service, allowing messages to be classified in real time by calling the API service. The performance can be improved by trying other classifiers.

The classification of messages as important or unimportant has a wide scope, with the API service that can be used by any instant messaging service to classify a particular message. The API service is limited with respect to privacy. Thus, the future scope revolves around ensuring good security mechanisms to protect message data and also on achieving excellent prediction and performance metrics. The model can further be used as an integral part to increase the productivity of a user, thereby helping improve the life of a instant messaging user or any internet user. With increased precedence given to important messages, emergency situations can also be handled quicker.

REFERENCES

1) K. Garimella and G. Tyson, "Whatapp doc? a first look at whatsapp public group data," in Twelfth International AAAI Conference on Web and Social Media, 2018.
2) A. Rosenfeld, S. Sina, D. Sarne, O. Avidov, and S. Kraus, "A study of whatsapp usage patterns and prediction models without message content," arXiv preprint arXiv:1802.03393, 2018.
3) K. Church and R. De Oliveira, "What's up with whatsapp?: comparing mobile instant messaging behaviors with traditional sms," in Proceedings of the 15th international conference on Human-computer interaction with mobile devices and services. ACM, 2013, pp. 352–361.
4) J. Y. Lee and F. Dernoncourt, "Sequential short-text classification with recurrent and convolutional neural networks," arXiv preprint arXiv:1603.03827, 2016.
5) S. Kim, L. F. D'Haro, R. E. Banchs, J. D. Williams, M. Henderson, and K. Yoshino, "The fifth dialog state tracking challenge," in 2016 IEEE Spoken Language Technology Workshop (SLT). IEEE, 2016, pp. 511–517.
6) E. Shriberg, R. Dhillon, S. Bhagat, J. Ang, and H. Carvey, "The icsi meeting recorder dialog act (mrda) corpus," in Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue at HLT-NAACL 2004, 2004.
7) D. Jurafsky, E. Shriberg, and D. Biasca, "Switchboard dialog act corpus," International Computer Science Inst. Berkeley CA, Tech. Rep, 1997.
8) M. Vanetti, E. Binaghi, B. Carminati, M. Carullo, and E. Ferrari, "Content-based filtering in on-line social networks," in International Workshop on Privacy and Security Issues in Data Mining and Machine Learning. Springer, 2010, pp. 127–140.
9) D. S. Broomhead and D. Lowe, "Radial basis functions, multi-variable functional interpolation and adaptive networks," Royal Signals and Radar Establishment Malvern (United Kingdom), Tech. Rep., 1988.
10) "Firebase," Apr 2019. [Online]. Available: https://en.wikipedia.org/wiki/Firebase
11) T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016, pp. 785–794.
12) "Cross-validation (statistics)," Apr 2019. [Online].Available:https://en.wikipedia.org/wiki/Cross-validation(statistics)