Turkish Online Journal of Qualitative Inquiry (TOJQI)

Volume 12, Issue 3, July 2021:2596- 2610

Research Article

Web-based Instant Messaging Application with Functionality for Filtering Distracting Content

Sarthak Maheshwari [1], Chinmay Pareek [2], Saad Yunus Sait [3*]

Abstract

The internet has provided us various modes for transmission of data from one device to another such as emails, SMS/MMS, instant messaging applications etc. Instant messaging applications are very widely used for their cost effectiveness and facility for quick communication. Within the meantime, several undesirable messages are received by users which can be classified as spam, distracting, redundant. Despite the fact that people have the flexibility to promptly acknowledge a message as undesirable, performing such a task could be a waste of time. A platform independent instant messaging web-based application is developed which can filter distracting messages in real time. The application uses socket.io library for bidirectional communication necessary for instant messaging and uses NodeJS, MongoDB for routing and storing data respectively. The real time filtering of messages is done by Support Vector Machine classifier and Stochastic Gradient Descent. The machine learning algorithms are integrated with the web part with the help of APIs.

Keywords: *Instant messaging, bidirectional communication, routing.*

---

[1] Computer Science and Engineering, College of Engineering and Technology, SRM Institute of Science and Technology, SRM Nagar, Kattankulathur, Kanchipuram, Chennai, TN, India, sy2787@srmist.edu.in

[2] Computer Science and Engineering, College of Engineering and Technology, SRM Institute of Science and Technology, SRM Nagar, Kattankulathur, Kanchipuram, Chennai, TN, India, cp9457@srmist.edu.in

[3*] Dr., Computer Science and Engineering, College of Engineering and Technology, SRM Institute of Science and Technology, SRM Nagar, Kattankulathur, Kanchipuram, Chennai, TN, India, saady@srmist.edu.in

Introduction

In this decade we encountered several ways of communicating online such as emails, content posting platforms such as twitter, Facebook etc., instant messaging platforms etc. communication. Instant messaging is an essential means of communication and data exchanging across the globe. Instant messaging is recognized to be meaningful to people, the main good factor is the exchanging of ideas among different people, groups and communities. Out of all ways instant messaging is widely used for its cost-effectiveness and facility for quick communication. Instant data transmission applications are majorly famous and user-friendly means to share and exchange information on different topics. Regular communications mean the sharing of the many styles of content, as message, picture, audio and video information. In step with Facebook survey, a user performs ninety things of content every month, whereas over thirty billion things of content are shared by users worldwide in a month.

Within the meanwhile, several undesirable messages are received by users which may be classified as spam or distracting. Currently, in OSNs the information filtering mechanisms take a different path due to various reasons. In on-line social networks, information filtering can help users manage the texts received on their own system, by filtering out unwanted messages from their walls. Also, social media platforms are already being used to recommend content to users. In this work, we basically provide an instant messaging service which can filter distracting messages.

In a previous work (Sait et. al), we showed that models with good performance can be learnt for the task of filtering distracting content. In this work, we actually implement a chat application which filters distracting messages, thereby saving user time, and enhancing productivity. The machine learning algorithms can perform information filtering and these algorithms can be embedded into the chat applications.

The reason behind developing a web-based application is to eliminate the platform dependency of the application. So, the application can run on any OS like Windows, iOS, Linux, Temple OS etc. The ML method used is called Support Vector Machine (SVM) which is implemented as a classifier and Stochastic Gradient Descent is implemented for optimization, which was also proposed by us (Akilan et al). The machine learning part is embedded with a web application. The flask API is generated and included in the web application code.

The web application part is built on the 'MEN' stack i.e., MongoDB, express and NodeJS. NodeJS allows us to run JavaScript on the server and write server-side scripts. Mongod B is used for storing user data and messages. Socket.io is a JavaScript library which helps us in implementing bidirectional communication between clients and servers. Socket helps us in enabling real time messaging features in applications. The machine learning algorithm is built on flask, a micro web framework written in Python. For filtering messages, Support vector machines (SVMs) are useful tools for information classification. They filter two-category points by allocating them to one of two disjoint half spaces in either the original input space of the problem for linear classifiers, or in a higher dimensional feature space for nonlinear classifiers.

Literature Survey

Andrew, et al, 2017 used a Bayesian project model trained by Genetic a (hybrid) for filtering spam messages. In this model various techniques are used such as classification, training the model and lastly the most important feature selection. For feature selection Genetic Algorithm is used and for classifier Bayesian algorithm is used. The difficulty faced was SMS size limitation 160 characters and the inclusion of emoticons, slangs makes the model ineffective to train. The output of the model can be spam or ham. The accuracy of the model can be closely predicted by the presence of FP and FN where FP is false positive and FN is false negatives.

A. Karim, S. Azam, 2017 used rule-based systems for detecting spam. The rule-based systems are used because they are capable of fighting spam by combining different pattern detection technologies and rules can be updated remotely. The model has been able to achieve constant time of O(1) for detecting spam. It is possible by using a data structure called hash forest. The constant time complexity is achieved as it only goes through very small pieces of text. The challenge faced by the author was anti-spam filtering as filtering speed slows and throughput decreases. The algorithm's speed is not dependent on text and vocabulary size.

XUESONG WANG, 2019 used K-L divergence for detecting illegal spam links in twitter. The links may evolve over time and go undetected. Therefore, this multi scale detection test technique is used to level the probable upcoming and evolving messages often known as drifts. The divergence method is used to project the junk distributional change. MDDT is highly useful as it improves classification results and performs better than other methods such as CUMSUM. The accuracy achieved by using this method is 98.6%.

In this paper SAMI AZAM, et al, 2018 used spectral and K-means clustering algorithms to identify spam. However the clustering method performed slightly better than K means, spectral algorithm, 3.5% to be precise. It emphasizes on cyber-security and privacy of an individual and the goal is to detect the spam emails having the intention of inflicting damage such as identity, data theft or trying to do damage. The research is done on a dataset having 100000 fields of spam and ham. Using an unsupervised framework, the model achieved an accuracy of 94.57%.

AL-Rawashdeh, et al, 2019, There is no dispute to the fact that SVM and KNN have best performance in areas of spam detection but the author suggested using the WCA hybridization algorithm. The research paper aims to increase the detection precision and improve the performance of junk/unwanted messages detection algorithms using WCA. The optimization approach can be used to choose the best solution possible. The accuracy achieved by implementing WCA is 96.3%.

Devottam Gaurav, et al, 2020 used different classifiers which are popular for having high accuracy for classification in spam detection techniques. The author Gaurav used the technique on different sets to find how the proposed method worked. The spam detection part is categorized into three parts, detection of spam images, Detection of Bagged words and Detections of collaborative spam. The experiments were performed using all three classification algorithms and it was found that Random Forest has the highest accuracy than the rest. The challenges faced were to classify spam emails and to optimize NP-hard problems and find the relevant features. The time complexity in spam detection tasks is O[n].

In this paper the Author Emmanuel Gbenga Dada, et al, 2019 reviewed the spam filtering strategies of popular social media websites such as Gmail, Outlook and Yahoo to analyze them. According to the review, deep learning and deep adversarial learning are effective for controlling spam. The filtering techniques are divided into different categories, Content Based Filtering Technique, Case Base Spam Filtering Method, Heuristic or Rule Based Spam Filtering Technique, Previous Likeness Based Spam Filtering Technique and

Adaptive Spam Filtering Technique. It has been found that around one-fifth of spam remain undetected by techniques implemented by popular social media sites. The review journal discusses all possibilities, datasets and techniques for spam detection.

ZHIJIE ZHANG, et al, 2020 in “Detection of Social Network Spam Based on Improved Extreme Learning Machine " proposed a method which can stop the menace of twitter spam. The architecture of algorithms for detection of spam or junk messages has the foundation of the Fuzzy-kernel based machine learning method (I2FELM). This method is popular for detecting spam and uses fuzzy weights to solve prejudiced data complications and amplifying of accuracy. This algorithm can be used no matter whether there are balanced or unbalanced datasets. The results showed accuracy more than random forest algorithm.

In “Ham and Spam E Mails Classification Using Machine Learning Techniques' ' the author M. Bassiounia, et al, 2018 used a 10-fold technique to improve the accuracy for spam detection techniques. Among all the classifiers Random Forest is preferred for getting better and effective results. The dataset used for the testing is spam base UCI dataset with 57 attributes. The experiment is done on the datasets using different classifiers. The accuracy is calculated by the number of FP and FN. The accuracies achieved for these classifiers are 95.4, 92.4, 92.4, 91.8, 91.5, 90.7, 90.3, 89.8, 89.8, and 82.6, respectively. The maximum precision classifier among all is the random forest algorithm.
In the paper “Spam Review Detection Using the Linguistic and Spammer Behavioural Methods” the author Naveed Hussain, et al, 2020 aims to identify fake and spam reviews using Behavioral Method techniques. This technique identifies thirteen different features of spam review to measure spam amount. If the score is above a threshold the review is marked as spam. The previous studies have conducted tests on Amazon spam review dataset of around 26 million reviews. The use of SRD-LM and SRD-BM techniques resulted in improved accuracy. However, SRD-BM is more effective as it actively looks out for features which are present in every spam and it results in efficient detection of junk or spam emails. The study might help many companies in future as the people with motives to degrade rating of a product cannot spam reviews.

Architecture

Frontend

The technologies that are used for the front-end part of our application are Html, CSS, Bootstrap and JavaScript. Basic pages for the front-end are Login, Signup, Chat window.

- **Signup**

The first and foremost for any user is to sign up in the application with a suitable username, email id and with a password. After signing up their account will be created and their credentials will be stored in the database. Sign up page UI is depicted below in Fig 3.1.

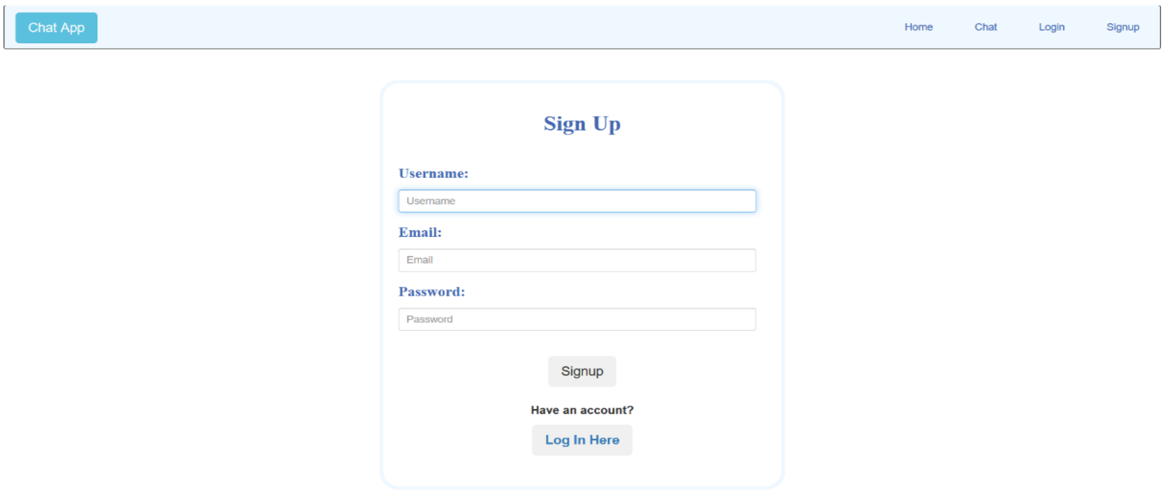


Fig. 3.1. Sign Up Page Interface

- **Login**

After the process of creating an account or signing up users can login into the application from the login page by entering their correct credentials which they used while creating their account. After logging in, users can use the features of the application. Login page UI is depicted below in Fig 3.2.

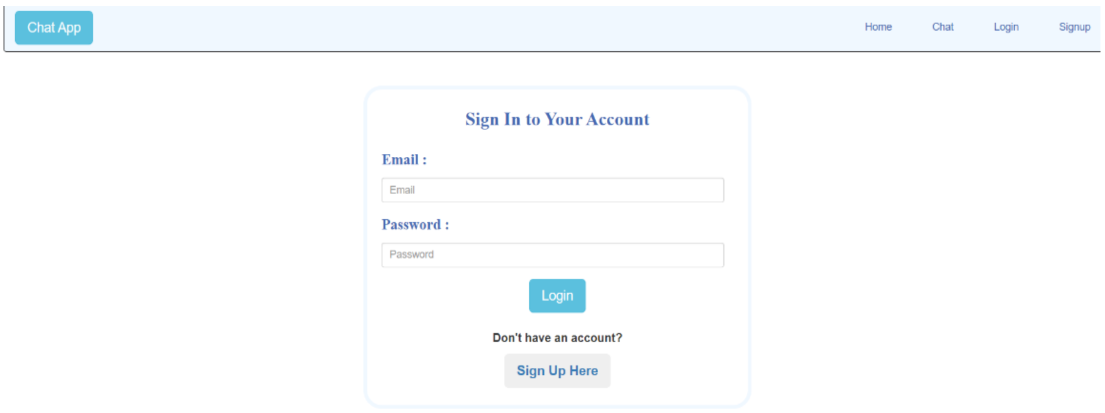
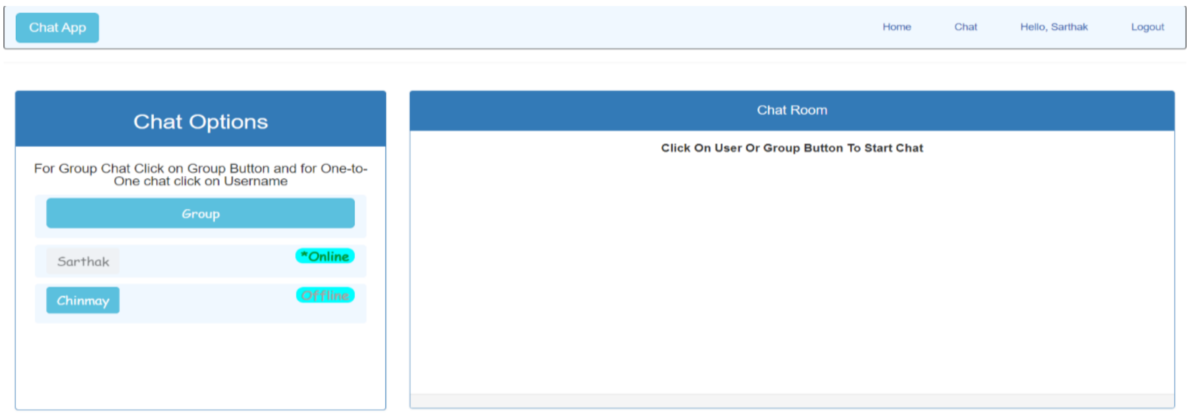


Fig. 3.2. Login Page Interface

- **User's Window**

The interface user gets after logging into the application, can be seen in Fig 3.3.



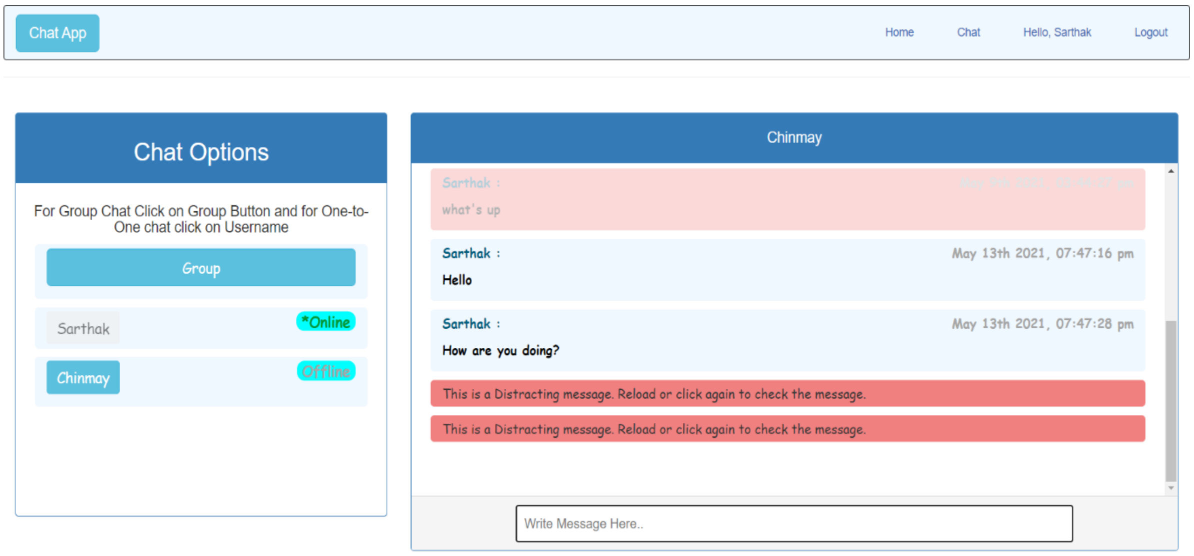
For Group Chat Click on Group Button and for One-to-One chat click on Username

Click On User Or Group Button To Start Chat

Fig. 3.3. User Window Interface

- **Chat Window**

After logging in, users can view the list of the people who are in their contact list. If the user wants to send a message to someone, he/she simply has to click on their username and the chat window will pop on the right side of the user's list where they can send the message. Fig 3.4 shows the chat window of the application.



For Group Chat Click on Group Button and for One-to-One chat click on Username

Sarthak : May 9th 2021, 03:44:27 pm
what's up

Sarthak : May 13th 2021, 07:47:16 pm
Hello

Sarthak : May 13th 2021, 07:47:28 pm
How are you doing?

This is a Distracting message. Reload or click again to check the message.

This is a Distracting message. Reload or click again to check the message.

Fig. 3.4. Chat Window Interface

Backend

The technologies that are used for the back-end part of our application are Node.js, MongoDB and Python for machine learning. Here we used Node.js to implement functions of our application. MongoDB is used as a database for storing user's data and messages. The dataset contains text messages labelled as distracting (1) or not-distracting (0). Whenever a user sends a message it is first processed to remove symbols, then converted to lowercase and stemming/lemmatization is done to get root words. Then, they are converted into vectors using TF-IDF. The classifier used is SGD Classifier that uses SVM (Support Vector Machine) as the classifier model and Stochastic Gradient Descent as the optimization technique. The SGD Classifier uses the aforementioned vectors for training. After the classification of messages through machine learning techniques it'll be stored in the database.

- **Database**

a. Schema

The chat application database schema is divided into three collections namely chats, rooms and users. The attributes of the collections are listed in Fig 3.5.

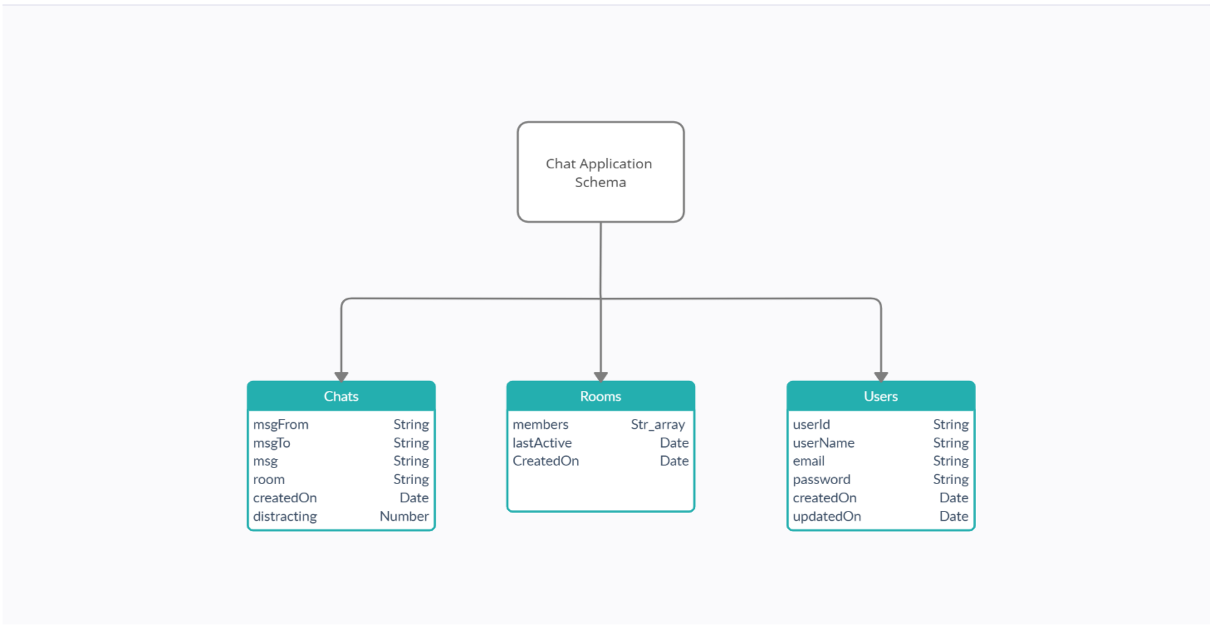
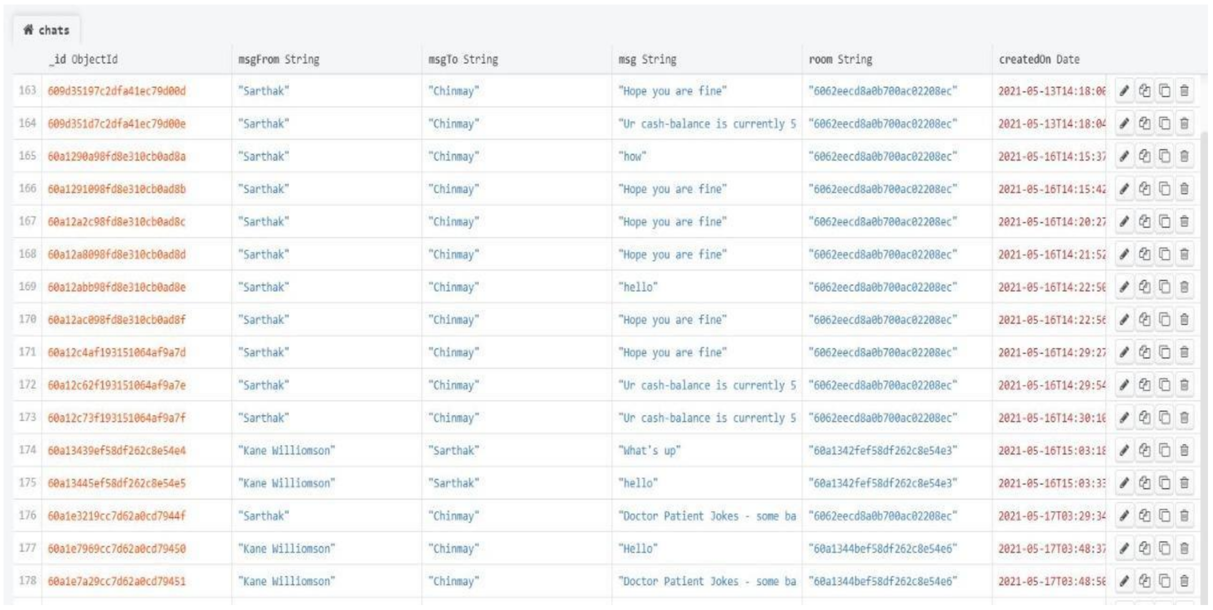


Fig. 3.5. Database Schema

b. Chat Database

The chat database contains columns like msgFrom, msgTo, msg, room, createdOn and distracting number. The chat database of the application is presented in the following Fig 3.6.



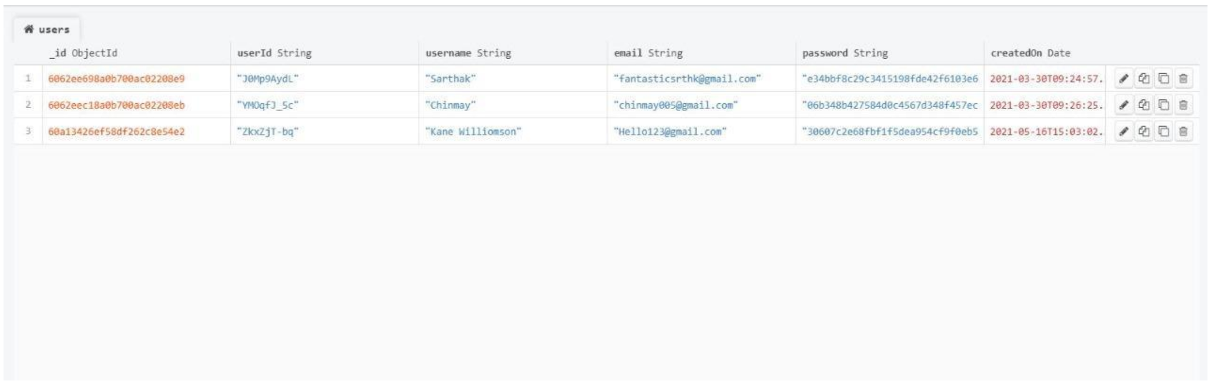
chats

	_id ObjectId	msgFrom String	msgTo String	msg String	room String	createdOn Date
163	609d35197c2dfa41ec79d00d	"Sarthak"	"Chinmay"	"Hope you are fine"	"6062eecd8a0b700ac02208ec"	2021-05-13T14:18:06
164	609d351d7c2dfa41ec79d00e	"Sarthak"	"Chinmay"	"Ur cash-balance is currently 5	"6062eecd8a0b700ac02208ec"	2021-05-13T14:18:04
165	60a1290a98fd8e310cb0ad8a	"Sarthak"	"Chinmay"	"how"	"6062eecd8a0b700ac02208ec"	2021-05-16T14:15:37
166	60a1291098fd8e310cb0ad8b	"Sarthak"	"Chinmay"	"Hope you are fine"	"6062eecd8a0b700ac02208ec"	2021-05-16T14:15:42
167	60a12a2c98fd8e310cb0ad8c	"Sarthak"	"Chinmay"	"Hope you are fine"	"6062eecd8a0b700ac02208ec"	2021-05-16T14:20:27
168	60a12a8098fd8e310cb0ad8d	"Sarthak"	"Chinmay"	"Hope you are fine"	"6062eecd8a0b700ac02208ec"	2021-05-16T14:21:52
169	60a12abb98fd8e310cb0ad8e	"Sarthak"	"Chinmay"	"hello"	"6062eecd8a0b700ac02208ec"	2021-05-16T14:22:56
170	60a12ac098fd8e310cb0ad8f	"Sarthak"	"Chinmay"	"Hope you are fine"	"6062eecd8a0b700ac02208ec"	2021-05-16T14:22:56
171	60a12c4af193151064af9a7d	"Sarthak"	"Chinmay"	"Hope you are fine"	"6062eecd8a0b700ac02208ec"	2021-05-16T14:29:27
172	60a12c62f193151064af9a7e	"Sarthak"	"Chinmay"	"Ur cash-balance is currently 5	"6062eecd8a0b700ac02208ec"	2021-05-16T14:29:54
173	60a12c73f193151064af9a7f	"Sarthak"	"Chinmay"	"Ur cash-balance is currently 5	"6062eecd8a0b700ac02208ec"	2021-05-16T14:30:16
174	60a13439ef58df262c8e54e4	"Kane Williomson"	"Sarthak"	"What's up"	"60a1342fef58df262c8e54e3"	2021-05-16T15:03:18
175	60a13445ef58df262c8e54e5	"Kane Williomson"	"Sarthak"	"hello"	"60a1342fef58df262c8e54e3"	2021-05-16T15:03:33
176	60a1e3219cc7d62a0cd7944f	"Sarthak"	"Chinmay"	"Doctor Patient Jokes - some ba	"6062eecd8a0b700ac02208ec"	2021-05-17T03:29:34
177	60a1e7969cc7d62a0cd79450	"Kane Williomson"	"Chinmay"	"Hello"	"60a1344bef58df262c8e54e6"	2021-05-17T03:48:37
178	60a1e7a29cc7d62a0cd79451	"Kane Williomson"	"Chinmay"	"Doctor Patient Jokes - some ba	"60a1344bef58df262c8e54e6"	2021-05-17T03:48:56

Fig. 3.6. Chat Database View in Mongo Compass

c. Users Database

The users database contains columns like userId, username, email, created On, updated On and password. The user details of the application are show in the Fig 3.7 below.



users

	_id ObjectId	userId String	username String	email String	password String	createdOn Date
1	6062ee698a0b700ac02208e9	"J0Mp9AydL"	"Sarthak"	"fantasticsrthk@gmail.com"	"e34bbf8c29c3415198fde42f6103e6	2021-03-30T09:24:57.
2	6062eec18a0b700ac02208eb	"YMOqfJ_5c"	"Chinmay"	"chinmay005@gmail.com"	"06b348b427584d0c4567d348f457ec	2021-03-30T09:26:25.
3	60a13426ef58df262c8e54e2	"ZkxZjT-bq"	"Kane Williomson"	"Hello123@gmail.com"	"30607c2e68fbf1f5dea954cf9f0eb5	2021-05-16T15:03:02.

Fig. 3.7. User Database View in Mongo Compass

- **Filtering Mechanisms**

The classifier used is SGD Classifier that uses SVM (Support Vector Machine) as the classifier model and Stochastic Gradient Descent as the optimization technique. The SGD Classifier uses the TF-IDF vectors for training. The SGD Classifier also supports online learning, that is, if the prediction is wrong, the user can label it correctly and the model gets incrementally updated/trained.

- **Integration of Web and ML**

Integration is done by enabling the ML model's prediction and updation through a REST API. An API call can be made with the necessary input packed as a JSON object (text message as well as user's feedback in the case of update. The output would be a JSON object which would contain the prediction as distracting (1) or non-distracting (0).

An event emitter function is used for classifying messages in real time. In the emitter function the data is feeded to the flask server through an API. In the emitter function a JSON object is created which has the URL of API path and the text data and all the data is added into a JSON object. An asynchronous JavaScript function is created and takes a JSON object as input and returns the global pred variable. Now using the prediction variable, we can identify whether the message is distracting or not distracting.

Component Diagram

In figure 3.8 the component diagram of the chat application is presented. At first a new user has to sign Up and an existing user has to log in. After validation and authentication, users will have the option to Chat with the users logged in. There are two chat options available one is Group chat and other is personal chat i.e., one to one chat. When a user sends a message to another user or in a group the message is emitted by Socket IO. The message is sent to the flask server with the help of an API in the form of a JSON object. The machine learning SVM classifies the data. Socket IO then receives the data and displays the data to the receiver and stores it in the database as well.

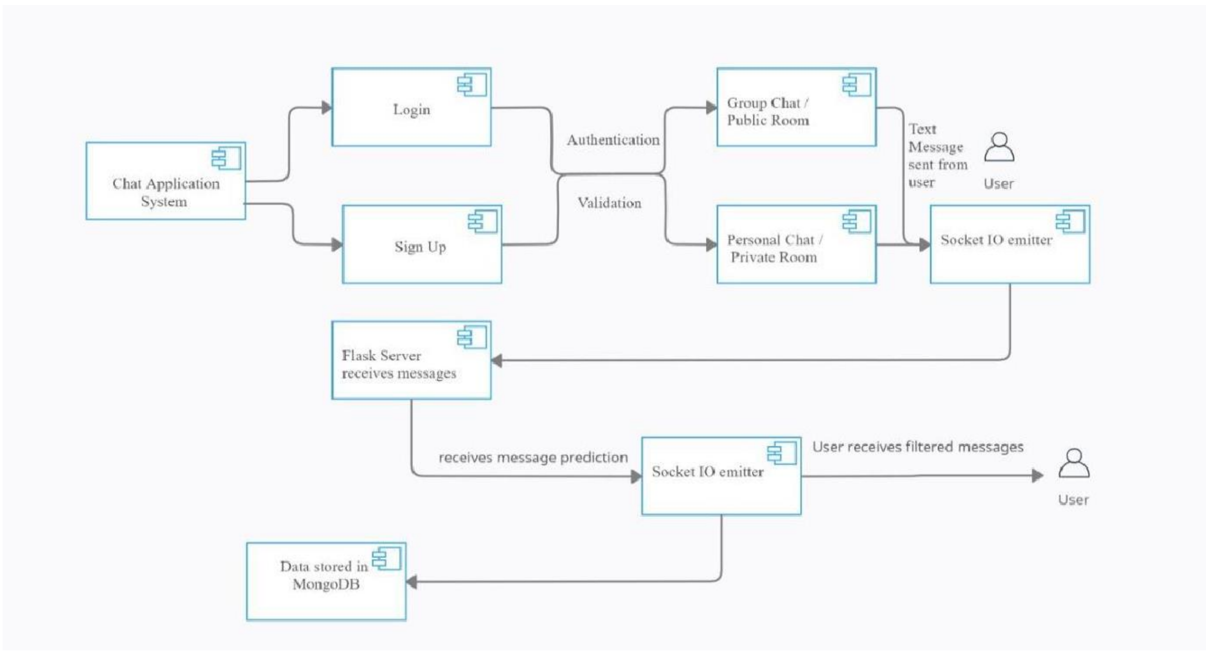


Fig. 3.8. Component Diagram

Working of Application

Frontend

So, in this part there are two important pages which are available in any website i.e Signup and Login Page. Therefore, the first and foremost step for a user to register themselves on the application and create a profile by adding username and password. By signing up on the application, users can login in our website through their correct credentials. After logging in, users should know their friend's username to whom they want to send a text. If the searched user is found in the database, the user can send a text message to him/her and If the text message is found out to be distractive it'll be displayed with a red warning colour.

Backend

The technologies that are used for the back-end part of our application are Node.js, MongoDB and Python for machine learning. Here we used Node.js to implement functions of our application. MongoDB is used as a database for storing user's data and messages. The dataset contains text messages labelled as distracting (1) or not-distracting (0). Whenever a user sends a message it'll first be processed to remove symbols, then converted to lowercase and stemming/lemmatization is done to get root words. Then, they are converted into vectors using

TF-IDF. The classifier used is SGD Classifier that uses SVM (Support Vector Machine) as the classifier model and Stochastic Gradient Descent as the optimization technique. The SGD Classifier uses the aforementioned vectors for training. After the classification of messages through machine learning techniques it'll be stored in the database.

Integration

Integration is done by enabling the ML model's prediction and updating through a REST API. An API call can be made with the necessary input packed as a JSON object (text message as well as user's feedback in the case of updating). The output would be a JSON object which would contain the prediction as distracting (1) or non-distracting (0).

Conclusion

Spamming is a major problem nowadays because of which people can be easily distracted. This paper shows how a chat application can be implemented for filtering distracting messages using a linear SVM classifier with SDG. This can help people in various fields to work with full efficiency. By using this project, we can decrease the number of distracting messages significantly.

The possibilities of future enhancements can lead to being a major factor in the messaging industry. Those enhancements being:

1. The filtering mechanism could be extended to other languages besides English.
2. The classification of images, videos, audio and other media files can also be done.

References

Wang, X., Kang, Q., An, J., & Zhou, M. (2019). Drifted Twitter spam classification using multiscale detection test on KL divergence. *IEEE Access*, *7*, 108384-108394. https://doi.org/10.1109/ACCESS.2019.2932018

Karim, A., Azam, S., Shanmugam, B., & Kannoorpatti, K. (2020). Efficient Clustering of Emails Into Spam and Ham: The Foundational Study of a Comprehensive Unsupervised Framework. *IEEE Access, 8*, 154759-154788.

https://doi.org/10.1109/ACCESS.2020.3017082.

Al-Rawashdeh, G., Mamat, R., & Abd Rahim, N.H.B. (2019). Hybrid water cycle optimization algorithm with simulated annealing for spam E-mail detection. *IEEE Access*, *7*, 143721-143734. https://doi.org/10.1109/ACCESS.2019.2944089.

Gaurav, D., Tiwari, S.M., Goyal, A., Gandhi, N., & Abraham, A. (2020). Machine intelligence-based algorithms for spam filtering on document labeling. *Soft Computing*, *24*(13), 9625-9638. https://doi.org/10.1007/s00500-019-04473-7

Dada, E.G., Bassi, J.S., Chiroma, H., Adetunmbi, A.O., & Ajibuwa, O.E. (2019). Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon, 5*(6), e01802.

Zhang, Z., Hou, R., & Yang, J. (2020). Detection of Social Network Spam Based on Improved Extreme Learning Machine. *IEEE Access*, *8*, 112003-112014.

https://doi.org/10.1109/ACCESS.2020.3002940

Bassiouni, M., Ali, M., & El-Dahshan, E.A. (2018). Ham and spam e-mails classification using machine learning techniques. *Journal of Applied Security Research*, *13*(3), 315-331. https://doi.org/10.1080/19361610.2018.1463136

Hussain, N., Mirza, H.T., Hussain, I., Iqbal, F., & Memon, I. (2020). Spam review detection using the linguistic and spammer Behavioral methods. *IEEE Access, 8*, 53801-53816. https://doi.org/10.1109/ACCESS.2020.2979226

Ahuja, L. (2018). Handling Web Spamming Using Logic Approach. *In International Conference on Advances in Computing and Data Sciences*, Springer, Singapore, 380-387

Attenberg, J., Weinberger, K., Dasgupta, A., Smola, A., & Zinkevich, M. (2009). *Collaborative email-spam filtering with the hashing trick. In: Proceedings of the sixth conference on email and anti-spam*

Bassiouni, M., Ali, M., & El-Dahshan, E.A. (2018). Ham and spam e-mails classification using machine learning techniques. *Journal of Applied Security Research*, *13*(3), 315-331.

Camastra, F., Ciaramella, A., & Staiano, A. (2013). Machine learning and soft computing for ICT security: an overview of current trends. *Journal of Ambient Intelligence and Humanized Computing*, *4*(2), 235-247.

Ojugo, A.A., & Eboka, A.O. (2018). Comparative evaluation for performance adaptive model for spam phishing detection. *Digital Technologies*, *3*(1), 9-15.

Ojugo, A., & Eboka, A.O. (2019). Signature-based malware detection using approximate Boyer Moore string matching algorithm. *International Journal of Mathematical Sciences and Computing*, *5*(3), 49-62.

Charninda, T., Dayaratne, T.T., Amarasinghe, H.K.N., & Jayakody, J.M.R.S. (2013). Content based hybrid sms spam filtering system. *Proceedings of ITRU Research Symposium, University of Moratuwa*, 31–35.

Hidalgo, J.M.G., De Buenaga Rodríguez, M., & Pérez, J.C.C. (2005). The role of word sense disambiguation in automated text categorization. *In International Conference on Application of Natural Language to Information Systems*. Springer, Berlin, Heidelberg, 298-309.

Arutyunov, V.V. (2013). Spam: Its past, present, and future. *Scientific and Technical Information Processing*, *40*(4), 205-211.

H. Shaban. (Sep. 19, 2019). *Nearly Half of Cellphone Calls Will be Scams by 2019*, Report Says. The Washington Post. *International Journal of Computer Applications* (0975 – 8887) Volume *– No.*, 2013.

Saad, O., Darwish, A., & Faraj, R. (2012). A survey of machine learning techniques for Spam filtering. *International Journal of Computer Science and Network Security (IJCSNS)*, *12*(2), 66.

E. Bauer. 15 *Outrageous Email Spam Statistics that Still Ring True in 2018*. https://www.propellercrm.com/blog/email-spamstatistics

Lyncova, D. (2019). *The surprising reality of how may emails are sent per day*. https://techjury.net/stats-about/how-many-emails-are-s e nt-per-day.

Sait, S.Y., Adak, R., Sharma, D., Prasad, A., Venkatesh, S.V., & Gaurav, A. Enhancing Mobile Instant Messaging Productivity by Filtering Distracting Messages. *Journal of Information Science, SAGE Journals.*

Akilan, N.M.K., Ghosal, A., & Sait, S.Y. *User-specific models for filtering distracting messages on mobile instant messaging. Turkish Online Journal of Qualitative Inquiry*.