International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 10 | Oct 2021 www.irjet.net p-ISSN: 2395-0072

A Study of Content & Conversation Suggestion in Messaging Applications

Yash Kandalkar[1], Raj Jadhav[2], Dhruv Bhanushali[3], Manas Buchade[4], Deepali Kadam[5]

[1,2,3,4]Student, Information Technology, Datta Meghe College of Engineering, Airoli, India
[5]Asst. Professor, Information Technology, Datta Meghe College of Engineering, Airoli, India

---

Abstract - *Communication is an integral part of a person's day-to-day life. A major part of that is done digitally via messaging applications. With rapid developments in technology, messaging applications are getting more secure and easy to use. Simplicity, security, and high functionality are some of the key features to keep in mind while developing a messaging application. In this paper, a brief literature review is conducted to study the structure of messaging applications and Machine Learning Models to implement and create a web-based application that suggests real-time content based on the user's conversations and history.*

Keywords: **Messaging Application, Content Suggestion, Topic Extraction, Conversation Suggestion.**

1. Introduction:

Messaging applications play a very important role in a person's day-to-day life. From simple chatting to professional video conferencing, everything is now possible with the help of rapidly developing communication applications. A variety of messaging applications are available in the market such as WhatsApp, Facebook, Instagram, Telegram, etc. which serve different purposes for different users. Some of the most important features of a messaging application include a tamper-proof security system, reliable cloud storage to back up the data, an easy-to-use interface, smooth and fast connectivity between the users, etc. Modern messaging applications also use Machine Learning, Artificial Intelligence, and Deep Learning techniques to implement features like suggesting specific people, personalized advertisements, predictive texting, etc.

These advancements have inspired us to create a multi-service messaging application that takes inspiration from all the modern features. We propose to develop a web-based messaging application that suggests real-time content to a user while in a conversation, based on the user's interests and chat history. The application will also support additional features like topic extraction from conversation history, automatic client-side user clustering based on common interests and community recommendations.

2. Literature Review:

2.1 Encryption Techniques for Different Messenger Applications [1]:

In this paper, we study the major encryption methods used by popular messaging applications such as WhatsApp, HikeMessenger, KakaoTalk, and Telegram. According to the latest released Facebook figures, WhatsApp alone has more than 2 billion users worldwide, making it one of the most popular messaging applications. WhatsApp uses end-to-end encryption to protect user data. This protocol uses Diffie-Hellman Key Exchange over ECDH and Curve25519 in the native cryptographic library. It also uses AES-256 and HMAC-SHA256 for additional security and message integrity. Only the sender and receiver can read and access the messages. No third party, not even WhatsApp can read the messages exchanged.

Hike messenger uses a 128-bit Secure Socket Layer which is encrypted with the Firewall server. This encryption algorithm is considered to be logically unbreakable. The latest update of Hike includes encryption using 128-bit AES and 2048 RSA public-key cryptosystems. Hike web uses AES-256 to protect the data over the web.

KakaoTalk is a leading South Korean messaging application. It uses AES-128 encryption in CBC mode for database files. The hashing algorithm MD5 was used as the encryption standard to generate the key for the application.

Telegram provides end-to-end encryption as well as a feature of self-destruction for messages using a timer. The end-to-end protocol uses the SHA-256 algorithm. Telegram also uses Diffie-Hellman protocol for generating key and 256-bit AES to encrypt data and an API call is made.

2.2 Smarter Response with Proactive Suggestion: A New Generative Neural Conversation Paradigm [2]:

The paper focuses on enhancing the traditional generative conversational systems which, provided a human issued *query (q)*, will generate a *response (r)* using sequence-to-sequence models. To enhance this, the paper also generates another sequence which is termed as a *suggestion (s)* which is a proactive measure for introducing another utterance that will keep the conversing users engaged.

The *suggestion (s)* generated is still in line with the original *query (q).* The authors are able to achieve this using a novel sequence generation model: *Deep Dual Fusion Model*.

The two main tasks for generating the response r and suggestion s are:

1. Response Generation: A recurrent neural network structure with gated recurrent units (GRU) is applied to generate a fitting reply to the given query, as an unrelated response can generate an inappropriate suggestion as well.

2. Proactive Suggestion: A dual sequence-to-sequence generation process is needed to encode the information from two sequences, the *query (q)* and *response (r)* to generate a relevant *suggestion (s).*

The Deep Dual Fusion Model consists of *deep dual fusion units* where each unit is made up of 3 cells: 1) Sequential GRU Cell, 2) Alignment GRU Cell and 3) Fusion Cell.

The data used for training the model contains a large number of human conversations, crawled from the open Web, where publicly available user messages with replies were collected.

Crowdsourcing was used to test the model's output. In which, the appropriateness of a response for a given query, and the proactive suggestion for a certain query and response were judged.

Several prominent models for generation-based conversational systems were compared against the proposed Deep Dual Fusion model, such as Plain Seq2Seq, Multi-Seq2Seq, Attentive Multi-Seq2Seq, and Copying Seq2Seq.

The accuracy of generating suggestions from these models was evaluated in terms of BLEU, ROUGE, and human judgments. In all these metrics, Deep Fusion scored the highest.

2.3 Professional chat application based on natural language processing [3]:

Respectively. One round consists of several processing steps that include substitution, transposition, and mixing of the input plaintext to transform it into the final output of ciphertext.

The data is put in the array in the first step, and then cipher transformations are performed on it in multiple rounds. While decrypting, the input consists of ciphertext + encryption key + counter. The obtained output is the original plaintext. The implementation makes AES one of the most suitable algorithms for security implementation.

More than 5.3 billion of the world population use online platforms to converse with others. A large number of users are victims of cyberbullying using inappropriate lewd messages.

The algorithm proposed in this paper is very easy to understand. The database is stored at the personalized client-side. The algorithm starts with Data Processing of the typed message in which special characters are removed and then the entire text is converted to lowercase. After that, Natural Language Processing (NLP) techniques are applied. Various operations like Stemming, Word Tokenization, Sentence Tokenization, Stop-words removal, Parts of Speech Tagging are carried out. Following this, keywords are derived from the processed text and later those words are compared with the keywords present in the database. If the condition becomes true i.e., if the user typed message contains any inappropriate words, then a warning will be displayed to the user. If the condition becomes false, that is, the text doesn't contain any negative sentiment words then the message is encrypted and stored in the database.

2.4 Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt Data [4]:

This paper emphasizes the importance of security and the use of the AES algorithm for encryption. Security is a major concerning topic in every application, especially messaging applications that share a considerable amount of private data. A message before encryption is called plaintext. It is human-readable. Text obtained after performing encryption is called ciphertext. Advanced Encryption Standard (AES) algorithm is a very commonly used encryption cipher algorithm. It is so secure that it is used by the U.S. Government to protect classified information. It is extremely difficult to crack and safe against all sorts of brute-force attacks. AES includes three block ciphers namely AES-128, AES-192 & AES-256.

These ciphers encrypt and decrypt data in blocks of 128 bits using cryptographic keys of 128, 192, and 256 bits, respectively. There are 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys

2.5 Enhanced Chat Application [5]:

The authors of this paper have focused on creating an Android Chat Application in which the application will feature predictive texting.

Predictive texting means predicting what the user is typing from a set of letters already typed by the user. The predictive texting system proposed in this paper is a self-learning system, that is, a predefined database is not stored, rather the system learns and adds the words in the newly created database while the user is still having a conversation. This database is temporarily created which means that the words stored in it will be deleted

once the user ends their session. This helps in avoiding the mixing up of words of user A with words of user B.

The algorithm proposed for predictive texting is summarized below:

Step 1: A function checks if the total length of a word is 3, if yes then a hash-chain is called.

Step 2: A hash table is created for the size of 10 words.

Step 3: The 3-letter substring is passed to find the hash function of the hash table as parameters.

Step 4: A hashing value for the given substring is calculated by this function which is used in searching the words beginning with the same substring or else it is used to store the word at the specified location.

Step 5: The hash function is made by calculating the hash value by summing the ASCII value of the letters in the substring, and after that modular division is performed on the sum. In this way, a set of 10 words for a particular substring can be stored.

Along with this, a frequency count of words is also maintained so that frequently occurring words will be assigned a higher priority. And with this, the lower priority words will get deleted from the table.

2.6 Latent Dirichlet Allocation [6]:

The system that we propose contains a lot of features *(further explained in the framework overview)*, that are based on topic extraction. So, topic extraction is a very important part of our system. In the past, there has been a lot of contribution on the extraction of topics from a corpus like the Multi-Grain hierarchical model, Latent Semantic Analysis, LDA (Latent Dirichlet allocation) [6] and much more. We referred to these models and found out that LDA modeling was more relevant to our system.

Latent Dirichlet Allocation (LDA)

is a topic modeling algorithm that is used to classify words in a document under a particular topic. It creates a multinomial distribution of topics over documents and multinomial distribution of words over topics, modeled as Dirichlet distributions.

LDA model gives two matrices:

1) $\Theta td = P(t|d)$ which is the probability

distribution of topics in documents.

2) $\Phi wt = P(w|t)$ which is the probability

distribution of words in topics.

So, the probability of a word given in document $P(w|d)$ is equal to the summation of the dot product of $P(w|t)$ and $P(w|d)$ (assuming conditional independence), which implies that $P(w|d)$ is equal to the summation of the dot product of Φwt and θtd.

To extract topics from a document, the model has some defined steps:

Step 1: Assign random weights to both the matrices.

Step 2: Choose a topic randomly from topics over document distribution based on their weights.

Step 3: Select a word at random based on the distribution of words for the chosen topic and put it in the document.

Step 4: Repeat this step for the entire document.

This process will validate the assigned weight assumptions to the matrices.

The algorithm for implementing LDA starts with Preprocessing of raw text which includes Tokenization, Stop-words removal, Lemmatization, and Stemming. Followed by the conversion of text to DTM (Document Term Matrix). The final step is running the LDA model.

TITLE OF PAPER	AUTHOR	YEAR	SELECTED FEATURES
Encryption Techniques for Different Messenger Applications	Maganti Manasa, Dasari Veera Reddy, AmanapuYaswanth, G.V.S Raj Kumar	2019	Getting acquainted with widely used encryption algorithms in modern messaging applications.
Smarter Response with Proactive Suggestion: A New Generative Neural Conversation Paradigm	Rui Yan, Dongyan Zhao	2018	Implementation of Deep Dual Fusion Model for Conversation Suggestion using Response Generation and Proactive Suggestion.

Professional Chat Application based on Natural Language Processing	S Karthick, R John Victor, S Manikandan, Bhargavi Goswami	2018	Censoring of Cuss Words using NLP techniques like Stop-words removal, tokenizations, and stemming.
Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt Data	Ako Muhamad Abdullah	2017	Studying AES-256 Encryption and Decryption in detail and implementing it.
Enhanced Chat Application	Avinash Bamane, Parikshit Bhoyar, Ashish Dugar & Lineesh Antony	2012	Predictive Texting using hash values, hash function, and hash table.
Latent Dirichlet Allocation	David M. Blei, Andrew Y. Ng, Michael I. Jordan, John Lafferty	2003	Topic Extraction Technique using Latent Dirichlet Allocation modeling.

Table 1: Literature survey in Tabular Format

3. Framework Overview of Proposed System

Our whole framework can be broken down into the client-side website, server, a real-time database, and machine learning models. The diagrammatic representation of the framework is given in Figure [1].

The website is a ReactJS Progressive Web Application (PWA), which will allow the site to be installable to any device (Android, iOS, Windows, etc.) so that the site is easily accessible to users. PWA also allows the site to provide push notifications to the users, which is crucial in messaging applications. The frontend is connected with the backend server which is a RESTful API.

The front-end makes an HTTP request to the server at an appropriate endpoint for performing various operations like sending, receiving, starring, deleting messages, etc.

3.1 Authorization:

The user authorization is carried out with the traditional JWT based authorization. The client sends an HTTP request to the server with the user-provided email and password in the JSON notation. On the server, the email and password are provided to the Firebase Admin SDK's Auth module which checks these credentials in the user's database and returns a success or error message depending on the correctness of the credentials. If the credentials are valid, the server generates a JWT token with an expiry date of 7 days. The token is signed with HMAC using the SHA-256 hash algorithm. Upon expiration of the token, the user is asked to log in again.

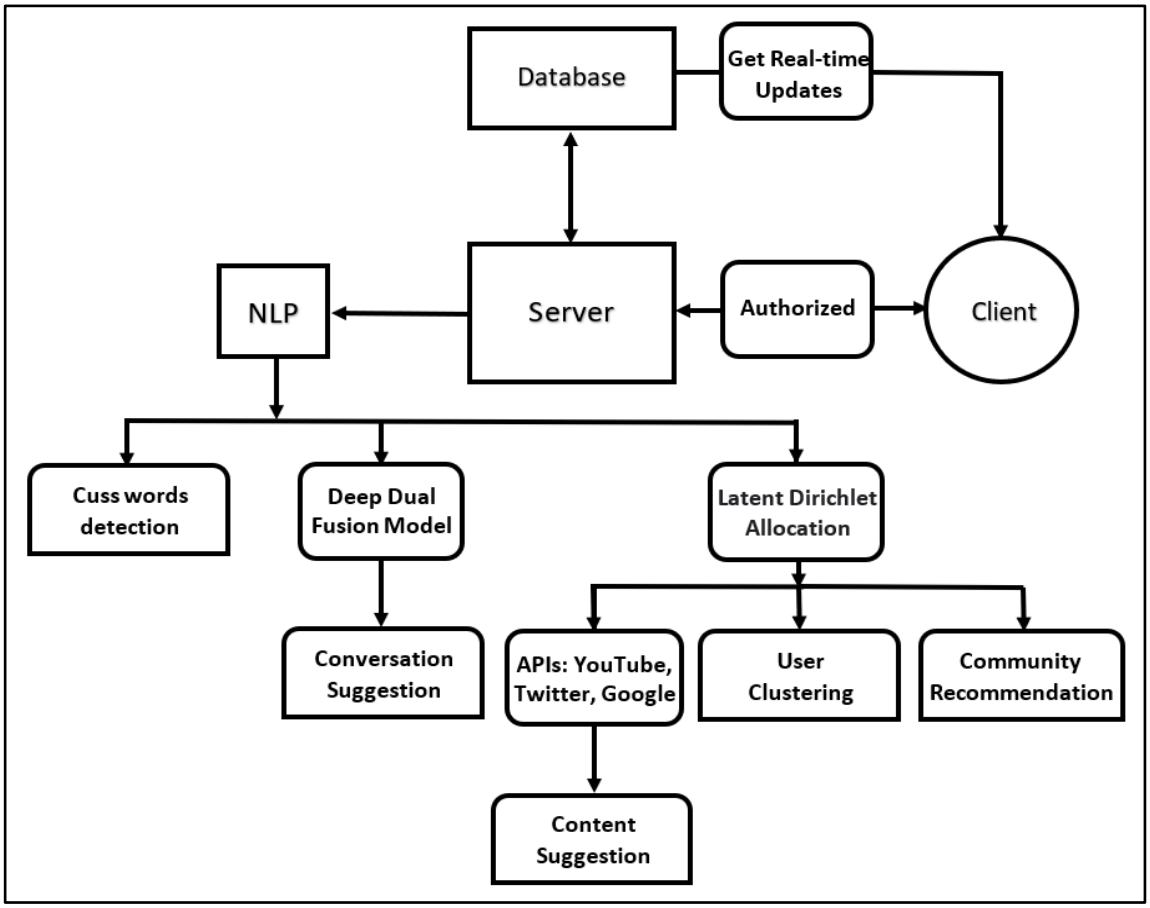

Figure 1: Framework Overview

3.2 Frontend:

The front end is made with ReactJS and Ant Design components.

The client-side is divided into 3 tabs:

1) Messaging, 2) Explore, and 3) Communities.

3.2.1 Messaging:

This section will contain private chats and groups the user is a part of. The interface is very similar to traditional messaging platforms.

From the user's chat history, using the Latent Dirichlet Allocation (LDA) [6] model, the topics are extracted and ranked based on their frequency in the user's chat history and its timestamp.

These topics (further referred to in the paper as *interests)* are used for a) Local User Clustering, b) Communities Recommendation, c) Mutual Users Recommendation, d) Topic Suggestion, and e) Content Suggestion.

a) Local User Clustering:

On the Messaging tab, the people user converses with will be clustered under different topics. The clustering will be based completely on both the user's common *interests*.

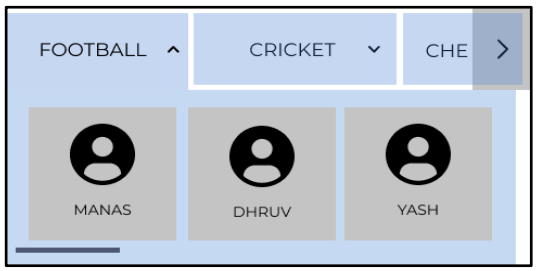

Figure 2: User Clustering based on user's *interests.*

b) Communities Recommendation:

Users are recommended communities to join based on the user's *interests.* The users are invited to join these public communities, where the participants will share common *interests.*

c) Mutual Users Recommendation:

Based on the user's *interests* and the people they converse with; we recommend the users who haven't interacted with each other but are mutually connected through another user and have common *interests*.

d) Topics Suggestion:

When strangers are conversing with each other, the conversation sometimes becomes boring (conversation stops after initial greetings). So, we recommend to each participant some topics which the person at the other end of the conversation is interested in.

e) Content Suggestion:

Content is provided to the user in the Explore tab based on the user's *interests.*

This will be explained in detail in the *Explore* section paper.

When conversing with other people, our proposed system will analyze the context of the ongoing conversation, and based on this context, we form a *query,* which is fed into the Deep Dual Fusion Model [2] for Generative Conversation Suggestion.

The model takes in a *query* and returns a *response* and a *suggestion* which is an utterance generated if the *query* was responded with the *response.*

Such conversation suggestions are excellent to keep the conversation going between the involved parties.

3.2.2 Communities:

This tab will contain the communities the user has already joined and will also recommend new public communities for the user based on their *interests.* Note that the user will not be added to a community directly, but only prompted to join.

These communities are *interest-oriented*, so only the users who have a certain *interest* will be prompted to join this community, which will help them connect with strangers sharing common *interests.*

3.2.3 Explore:

The Explore tab will aggregate content from different platforms like Google Search, Twitter, YouTube videos, etc. similar to *posts* from popular social media platforms. The content will be related to the user's *interests.*

Here, the user will be able to save these *posts* or *remove* them, which will restrict these *posts* from the user's Explore tab.

3.3 Server:

Our proposed framework will consist of 2 servers: a) Website Backend and b) Machine Learning Models's API

3.3.1 Website Backend:

This server is a RESTful Flask API, written in Python.

The server handles all the frontend requests and is connected with the Firebase Realtime Database. This server will connect with the Machine Learning Model's Server and will provide the client-side with the ML server's output.

3.3.2 Machine Learning Model's Server:

The two models: Latent Dirichlet Allocation [6] and Deep Dual Fusion Model [2] will be deployed on this server for topic extraction and conversation suggestion respectively.

3.4 Database:

The database used in the proposed system is a Firebase real-time non-relational database, which is used to handle messages in the application and other persistent data.

Since Firebase has a separate Authentication module, we do not need to store users' credentials, and they are not visible to developers as well. Firebase can manage the user's credentials.

4. Conclusions:

We studied various papers on topics to get a clear understanding of algorithms on Topic Extraction, Conversation Suggestion, and algorithms to secure our proposed system to protect users' privacy.

For Topic Extraction, we have decided to use Latent Dirichlet Allocation [6], which is easy to implement and a highly performant model. The extracted topics or user's *interests* will then be ranked based on their frequency in the user's chat history, and timestamp.

The Conversation Suggestions will be generated using the Deep Dual Fusion Model [2]. This novel model scores the highest in comparison to other generative conversation suggestion models.

The project is under development as of now.

References:

[1] M. Manasa, D. V. Reddy, AmanapuYaswanth, G.V.S R. Kumar, "Encryption Techniques for Different Messenger Applications". In proceedings of International Journal of Innovative Technology and Exploring Engineering (IJITEE), 2019.

[2] R. Yan and D. Zhao, "Smarter Response with Proactive Suggestion: A New Generative Neural Conversation Paradigm", In Proceedings of International Joint Conferences on Artificial Intelligence, 2018.

[3] Karthick S1, R. J. Victor2, Manikandan S3 and B. Goswami4, "Professional Chat Application based on Natural Language Processing". In Proceedings of IEEE International Conference on Current Trends in Advanced Computing (ICCTAC), 2018.

[4] A. M. Abdullah, "Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt Data". In proceedings of the Department of Applied Mathematics & Computer Science, 2017.

[5] A. Bamane, P. Bhoyar, A. Dugar and L. Antony, "Enhanced Chat Application". In Proceedings of Global Journal of Computer Science and Technology Network, Web & Security, 2012.

[6] D. M. Blei, A. Y. Ng, M. I. Jordan and J. Lafferty, "Latent Dirichlet Allocation". In proceedings of Eastern Mediterranean University - Cyprus, 2003.