Auto-Categorization of Tweets and Efficient Information Retrieval

Vibhor Kumar
MT20126
vibhor20126@iiitd.ac.in

Anshu Kumar
MT20050
anshu20050@iiitd.ac.in

Divyang Patel
MT20114
divyang20105@iiitd.ac.in

Anupam Kumar
MT20084
anupam20084@iiitd.ac.in

Shivam Joshi
MT20130
shivam20130@iiitd.ac.in

1. INTRODUCTION

In today's digital era, data plays an important role. But knowing the relevant information from the data and being able to retrieve them in a most efficient way, is what everyone is looking for. Retrieving information and categorizing them into clusters for better readability has become a center of attraction among companies or researchers.

With the increasing popularity of social media sites, we are in the era of information explosion. As of June 2020, about 400 million tweets are being generated every day. Twitter provides a list of most popular topics people tweet about known as Trending Topics in real-time, but most of the time, it has been seen that these trending tweets contain the name of individuals, or words of different origin, etc. which is often hard to understand. Therefore, it is important and necessary to classify these tweets into general categories like sports, politics, technology, business, education with high accuracy for better information retrieval. In this project, our aim is to make an information retrieval system which reduces the search space by classifying user query and then retrieves most similar tweets.

2. LITERATURE REVIEW

In recent years, researchers have focused on summarizing and detecting topics for tweets. Due to the relatively new topic, some research problems are poorly defined and new problems are being added every day. Rosa et al.[4] used supervised as well as unsupervised clustering approach to cluster the tweets. Using the unsupervised clustering approach, the study shows that k-means perform better than LDA(Latent Dirichlet Allocation) but both of these methods produce low-quality clusters. The study shows that the supervised approach performs better. It also shows that the texts corresponding to the URL present inside tweets are mostly irrelevant to the topic and could degrade the performance if included.

Kathy Lee et al.[2] worked on the database of tweets on 768 trending topics. They try to classify these tweets into 18 classes like art and design, books, business, fashion, etc. In the first approach, researchers preprocess the text of tweets and then transform the tokens into tf-idf weights. They applied naive bays machine learning algorithm on these weights and got an accuracy of 65%. In the second approach, they use an algorithm from the user similarity model to find the five most similar topics for trending topic x. User similarity is represented by a metric that has a value that represents the similarity between the users commenting on topics a_i and a_j. Topic-specific influential users are computed using a variant of the Weighted Page Rank Algorithm and Twitter social network information such as tweet time, number of tweets made on a topic, and friend-follower relationship. This model assumes that if there is significant overlap among users generating tweets on two topics, then it implies a close relationship between the topics. Using this approach they got an accuracy of 70%. We can improve the accuracy by using neural network models.

Ibtihel e[1] focused on applying semantic approach based on supervised categorization. This approach would rely on ontological knowledge where the training of classifier was not required, unlike the traditional approaches such as supervised learning machine learning. Instead of a bag of words, the researchers used Knowledge Bases(KB) approach. The tweets are preprocessed, after that, Named Entity (NE) Recognition, Linking, and Expansion is done using DBPedia, which is used to extract structured info from Wikipedia and uses SPARQL to query DBPedia. NE Recognition is used to identify entity classes such as a person, location, or organization. Word Sense Disambiguity (WSD) is also used here as DBPedia alone is not sufficient. WordNet Synsets and DMPedia Concepts are used together for representing the Tweets. Then eXtended WordNet Domains(XWND) is used in the classification of Tweets which uses a more specific domain than WordNet. The evaluation of the model is done on the Tweets extracted from Twitter for six different classes, namely Sports, Business, Technology, Entertainment, Politics, and Education, and the accuracy reported was around 88% which is much better than other models.

Tare et al.[6] focused their research on applying Naive Bayes classifier and Hadoop's Map-Reduce technique to classify tweets belonging to various categories. The dataset was collected using two different Twitter APIs. The researchers used three categories to classify the tweets, i.e., politics, sports, and technology. After doing all the preprocessing

on the dataset, two classification techniques were used. In the first technique, Naïve based classifier was used, which makes an assumption that features are independent of each other. Despite these assumptions, naïve based performed well, giving 75% accuracy on test data. In the second technique, researchers used Naïve based classifier using the Map-reduce technique. The map-reduce framework uses Mappers and Reducers internally. Mappers take the labeled tweets from the training data and output category and word count as key-value. Reducer does the classification by calculating conditional probabilities of each feature. Using the research, the researchers explained how the Map-Reduce technique could be used to scale up the existing Naïve Bayes model.

Xia el. al [7] concentrated on the application of deep learning in the information retrieval field. The authors have proposed a more specific way of dealing with law documents, combining word2Vec with legal document corpus. In order to expand the accuracy of the decision, Word2Vec is combined with specific law documents corpus. This has the advantage of discovering embedding meaning owing to the Word2Vec model. In addition, bag of words (BOW), which is carried out by creating a corpus full of words in law documents, can also be applied for that purpose. The disadvantages of these methods are that they cannot handle the embedded meaning in similar sentences.So, the usage of word2vec can significantly improve the accuracy by 0.20 compared with the BOW and TF-IDF models. Thus, the combination of Word2Vec and the law documents corpus is more compatible with the simple and efficient application of similarity analysis of law documents.

Selvaperumal and Suruliandi[5] manually collected tweets from various trending topics. In order to make the model unbiased, they included tweets containing URL, single or multiple trending topics, along with the tweets which do not contain URL and trending topics. Tweet URL, retweeted tweets, and tweets from a most influential user of a trending topic act as features of the model. If a URL is attached inside a tweet, then the webpage corresponding to that URL is categorized along with the tweet in the same category. If the tweets contain trending topics, then words from the top 5 tweets being retweeted and the top 5 tweets of influential users are collected and using the conventional text classifier, the collected words are classified. If the tweets do not contain URLs or trending topics, then a conventional text classifier is used to classify the tweets. The researchers used WEKA to classify the tweets. After building the model, the model performed better on the tweets belonging to the "Sports" category with an accuracy of 98%. Similarly, for the News, Politics, Entertainment and Education, the accuracies reported were 62%, 83%, 77%, and 66%, respectively. These accuracies came after applying SVM, which performs better compared to other classification algorithms.

Maceda et al.[3] focused on classifying the tweets which are related to earthquake using supervised machine learning algorithms. The tweets are converted into a tf-idf vector which acts as input to the model. Tweets were labeled into four categories, namely "drill/training", "earthquake feels", "extent of damages", and "government measures and rehabilitation". Out of 816 tweets, 204 tweets were labeled for each category. The researchers used SVM, Naïve Bayes, and Linear Logistic Regression using 10 and 15 folds validation. The model built using SVM correctly classifies 673 tweets out of 816 tweets. The accuracy obtained in this case was 82.48% with a .826 precision metric, .825 recall value, and .825 F-measure score. The model showed that the "drill/training" category is considered good while classification on "extent of damages," "government measures and rehabilitation" mandated for further review. The researchers finally concluded that SVM solves such problems due to its better generalization capacity on small datasets.

3. DATASET USED

In our project we are crawling tweets of the day using tweepy. We are collecting the trending tweets of 5 different categories that are Sports,Politics,Entertainment, Technology and Education using hashtags. We have collected around 21,000 tweets. Figure 1 shows the distribution of tweets per category and distribution of classes are balanced.

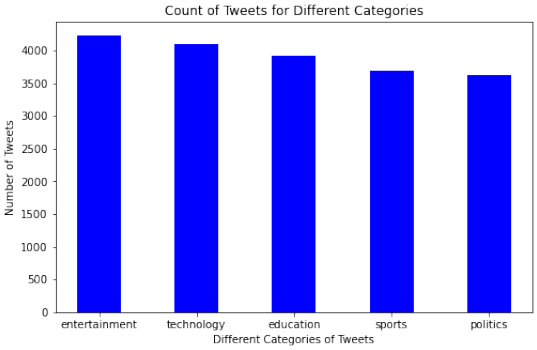

Figure 1: tweet class

Figure 2 shows TSNE plot of the data. The data was brought to the 2-D plane using TSNE so that linear data separation of data can be visualized.It can be observed that tweets are overlapping in nature as hashtags of a single category can also be used across other categories.

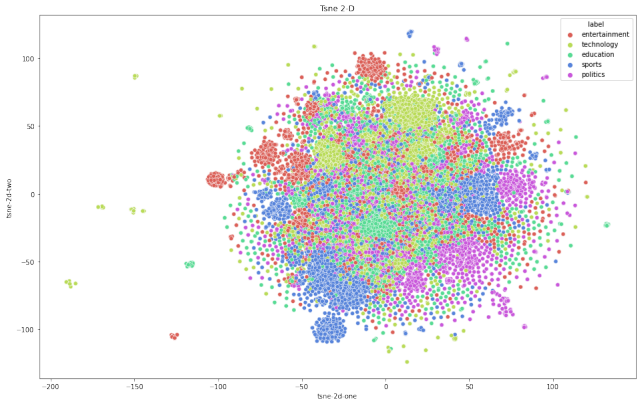

Figure 2: TSNE PLOT

The preprocessing of tweets is done to remove the noise from it. It includes following steps.

- Removal of punctuation.

- Removal of white spaces.
- Conversion of non-ascii characters.
- Tokenization of data.
- Handling of digits or numeric data.
- Stop words removal.
- Spell correction.
- Segmentation of words..
- lemmatization..

4. PLAN OF WORK

Our plan of work for this project is depicted in Figure 3.

- Downloading the trending tweets data based on different hashtags for different categories.
- Label each tweet into our predefined categories i.e., sports, politics, technology, business, and education.
- Train our model on those tweets.
- Taking query from the user thereby predicting its class using our model.
- Retrieve most similar tweets from a predicted class that match the user query
- We will also make an UI for the user to submit their Query.

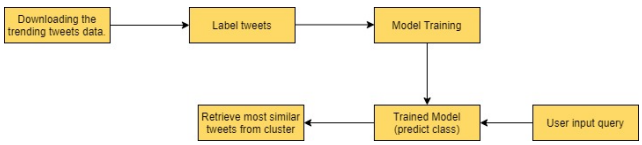

Figure 3: Model Overview

5. BASELINE RESULTS

We have implemented Kevin et. al [3] as a baseline system. Following this paper, we have implemented all the Machine Learning techniques given in the paper on the 19, 569 tweets of different and frequent social media categories like (Sports, Entertainment, Education, Technology, Politics).

5.1 Multinomial Naive-bias

Being a supervised algorithm, it assumes that each feature is independent of the other in the sample space. It is based on the Bayes theorem as formulated in Figure 4 .Given some prior information, Naive Bias computes the probability of occurrence of data in a specific category. In our project, we trained the Naive-bias model on 80% of the total corpus. The accuracy we got is Training Accuracy: 94.59%, Validation Accuracy: 85.25%.

$$P(y|x_1,...,x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1)P(x_2)...P(x_n)}$$

Figure 4: Multinomial Naive-Bias

5.2 Rocchio

In this classifier, we perform classification on the basis of tf-idf vectors. Rocchio's classifier is based on the idea that each class can be represented by its centroid. In our project, we trained the Rocchio model on 80% of the total corpus. The training accuracy and validation accuracy we got is 68.47% and 66.55% respectively.

5.3 SVM

A support vector machine (SVM) is a classification model that discriminates classes using a hyperplane defined using the formula w.x + b =0 where w represents the vectors, x represents the set of points in 'n' dimensions and b is a constant the given Multi-class problem. In our project, we trained the SVM model on 80% of the total corpus. The training accuracy and validation accuracy we got is 99.97% and 87.199% respectively.

5.4 Random Forest

Random forests is a supervised learning algorithm. It is utilized for classification. Firstly we have arbitrary data samples which are picked out. After that Random Forest generates a decision tree on these arbitrary data samples. The conjecture of each tree is obtained and the best and optimized solution is chosen. This is done by means of voting.In our project, we trained the Random Forest model on 80% of the total corpus. The training accuracy and validation accuracy we got is 99.98% and 83.82% respectively.

5.5 XG Boost

XGBoost is a highly efficient boosting algorithm. It is an execution of gradient boosting machines. It provides a collateral tree boosting that solves most data science and machine learning quickly.In our project, we trained the XG-Boost model on 80% of the total corpus. The training accuracy and validation accuracy we got is 88.27% and 81.34% respectively.

5.6 Extra Trees

In Extra Tree it implements various meta estimators that fit a number of randomized decision trees on various of the sub-samples of the dataset. Uses averaging for the improvement of predictive accuracy and control over-fitting.In our project, we trained the Extra Tree model on 80% of the total corpus. The training accuracy and validation accuracy we got is 99.98% and 85.20% respectively.

Apart from this, we have also applied Deep Learning Technique.

5.7 LSTM

Long short-term memory is the ARNN model. It has a feedback connection and it can process a complete sequence of data. In our project, we trained the LSTM neural network model on 80% of the data. We got the training and validation accuracy as 93.79% and 82.01% respectively.

5.8 Word2Vec

We convert each word of tweet into the vector of 300*1 dimension and then combine them to represent whole tweet as a vector of dimension 300*1. we used en_core_web_lg model by spacy to convert word into vectors. This model is trained on web documents and its size is 800mb. after converting our tweets to vectors we apply various machine learning models. when we apply SVM on Word2Vec, the accuracy is reduced to 81% in comparison to 87% of tf-idf.

6. PROPOSED METHOD

In the project, we aim to develop a framework, which is going to auto categorize the tweets into relevant topics it belongs to. We have used Twitter as the main source of documents since twitter is the platform where everyone shares their opinion and search for others' opinion, so we have taken tweets data as the main source of documents. Our framework helps users to auto categorise the tweets and present the most relevant tweets according to the user's entered query.

We retrieved $17,000$ tweets from twitter for each relevant and frequent categories that most of the tweets come under, then because of the rawness of the tweets, it has lots of noise that might hamper the model poorly if we feed these tweets to the model directly, so we performed various preprocessing techniques as follows:

- Removal of Duplicate Tweets: We have only Taken the unique tweets from the mentioned five categories. After removal of the duplicate tweets we are left with the $23,147$ unique tweets.
- Removal of punctuation: Punctuation marks act as noise, so we performed some pre-processing techniques to remove these punctuation marks. We made all the words in small letters to prevent ambiguity.
- Tokenizing of data: Prior to applying the most NLP and Deep Learning techniques we have to convert the given corpus into a set of tokens.
- Stop word removal: There are many words that don't contribute to the performance of the model and they occur more frequently and at times they act as a noise in the model. So we removed all the punctuation signs from the corpus.
- Segmentation of words: Many words tend to overlap and join together, that might make our model to wrongly interpret the corpus. To prevent this, we performed segmentation on our corpus.
- Lemmatization: In the fetched tweet there might be many words which might have the same meaning, which makes it really ambiguous to interpret anything out of it. So we performed lemmatization on the corpus.
- Removal of Small Tweets: From the preprocessed tweets we have removed the tweets whole length is smaller than 4. After this, We have left with 19,569 tweets on which we have formed the tf-idf vector of the tweets.

After the preprocessing, we performed various Machine Learning techniques on the preprocessed data.. The query that we got from users will be fed into the trained Machine Learning model and our model will predict one of the most 5 frequent social media trending topics (Sports, Entertainment, Education, Technology, Politics) it belongs to, Hence it will reduce the search space to the particular category only and will perform Information retrieval techniques to output the most relevant(Similar) tweets in that category. To retrieve the most similar tweets we first perform TF-IDF to convert each category of tweets into its individual weight matrix, We will convert the user query into vector form and perform various proximity measures to retrieve most relevant tweets.

We have also applied the following two measure to find the tweet Similarity:

- Cosine Similarity
- Euclidean Distance

Along with the above TF-IDF, we have also implemented the probabilistic Information retrieval models like Okapi BM25.

7. RETRIEVAL

Tweets retrieval comes into light when one needs to find the Tweets which are the best match for a query vector. For efficient retrieval it is important that the query vector and the tweets have maximal sentence or tweet similarity. We have implemented and evaluated mechanisms that measures similarity.

For document(tweets) retrieval, we have calculated the similarity between the tweet present with us and the Query Vector. Here, we calculate the similarity score S(Q,R) where Q represents the query vector and R represents the tweet. It calculates the degree up to which the tweet and the query vector are similar to each other or have common information between them. The tweet with maximum score is the best match for the query vector.Here, we have calculated the similarity score query vector and tweet that belong to the same class that is being predicted by SVM classifier algorithms for the given particular query. For techniques like tf-idf, cosine and euclidean we have used the tf-idf matrix and for Jaccard we have used the length of the query vectors and the Tweets to find the best match whereas in BM25 we have used the average tweet length to find the idf value of the tweets. Similarity scores can be calculated using various approaches :

7.1 Jaccard Coefficient

It calculates the score of document using number of words common in query and document(intersection) divided by total words in query and document(union). The idea behind Jaccard similarity is document which has more common words with query will have more score.

In Figure 5, A and B are the sets which contains the words

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Figure 5: Jaccard Coefficient

present in query and document respectively. Jaccard coefficient does not consider the term frequency as a measure to calculate score because set do not contain duplicate entries. rare words are more informative than frequent words.It also ignore the rarity of the word. first we create separate set for each document and then find its score by taking intersection and union with query set.

7.2 TF-IDF

In TF-IDF, TF denotes term frequency, where we first calculate the frequency of a term in a particular sentence. Then we calculate IDF, IDF denotes inverse of document frequency which is basically inverse of no of sentences which contain a particular word to the total no of sentences in a particular document. So, when we multiply TF and IDF together we get TF-IDF using Figure 6. The idea is that if two or more documents have similar words then those documents are more likely to be similar. It preserves the semantic differences between two documents unlike Bag of Words or other techniques.

Different variants of TF-IDF are used by different search

$$w_{t,d} = \log(1 + tf_{t,d}) \times \log_{10}(N / df_t)$$

Figure 6: TF-IDF

systems. So likewise in this project we have used TF-IDF as one of the techniques for scoring and ranking the retrieved documents on a given query. It ensures that among the various retrieved documents the document which has high IDF terms will have greater relevance or in other words it measures how important a term is in deciding the different class of tweets or document.

7.3 BM25

BM25 improves upon Tf-idf. BM25 has its roots in probabilistic information retrieval. It casts relevance as a probability problem. More the probability, more relevant document. To calculate the idf value in BM25, we add 1(smoothing) before taking log as shown in Figure 7 which makes it impossible to have to have a negative idf value.

Term frequency in BM25 dampens the impact of term fre-

$$\mathrm{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

Figure 7: IDF

quency even further than traditional Tf-Idf. The impact of term frequency is always increasing, but asymptotically approaches a value. The TF score above is further influenced by whether the document is above or below the average length of a document in the corpus. so the documents with huge length will not dominate the ranking. In Figure 8, f(qi, D) is qi's term frequency in the document D, $|D|$ is the length of the document D in words, and avgdl is the average document length in the text collection from which documents are drawn. k1 and b are free parameters, usually chosen, in absence of an advanced optimization, as k1 in [1.2,2.0] and b = 0.75. We made a matrix whose each column represent vector representation of a tweet.

$$\mathrm{score}(D,Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\mathrm{avgdl}}\right)}$$

Figure 8: BM25

7.4 Cosine

After we convert the tweets into vector using Tf-Idf matrix, cosine similarity is calculated using Figure 9.It is measure of angle between two vectors, thus it is judgement of orientation and not magnitude. Therefore two similar statements(Vector) will have cosine similarity as 1. Cosine similarity is calculated between the query and all the tweets of the query class and the tweets with highest cosine similarity score are retrieved.

$$\mathrm{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Figure 9: Cosine

7.5 Euclidean

In euclidean distance, we calculate the euclidean distance between two documents using Figure 10. In this project we calculate the distance between tweets and query vector while retrieving tweets. Euclidean distance indicates the similarity between two documents or tweets. When a user enters the query we retrieve the most similar tweets with the entered query in terms of euclidean distance between the query and the tweets.

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

Figure 10: Euclidean

8. RESULT

8.1 Auto-categorization of tweets

For auto-categorisation of tweets, Rocchio algorithm, Naive Bayes ,SVM, Random forest, XGBoost, Extratrees, and LSTM classifier are tested on tweets. Figure 11 represents results of these classsification algorithm.It is clearly seen from the

table that SVM Classifier performs best on taken test set with an accuracy of 0.8719.

Classifier	Multinomial Naïve Bayes	Rocchio	SVM	Random Forest	XG Boost	Extra Trees	LSTM
Accuracy	0.8525	0.6655	0.8719	0.8382	0.8134	0.8520	0.8201
Macro-Average Precision	0.86	0.80	0.88	0.85	0.85	0.87	
Weighted-Average Precision	0.86	0.80	0.88	0.85	0.85	0.86	
Macro-Average Recall	0.85	0.66	0.87	0.84	0.81	0.85	
Weighted-Average Recall	0.85	0.67	0.87	0.84	0.81	0.85	
Macro-Average F1-Score	0.85	0.68	0.88	0.84	0.82	0.86	
Weighted-Average F1-Score	0.85	0.68	0.87	0.84	0.82	0.85	

Figure 11: Results of Auto-Categorization

Query1	UPSC CSE Exams
Query2	Marvel new series falcon
Query3	narendra modi bengal election
Query4	Mithali Raj Indian Women cRickter
Query5	crypto currency bitcoin

Figure 12: Query Set

8.2 Retrieval

For comparing the result of various retrieval models, we have taken a list of queries given in Figure 12.

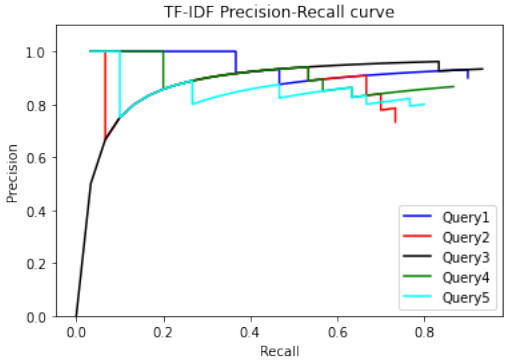

Figure 13: PR curve of Tf-Idf

With this test query set, PR curves for Tf-Idf and BM25 are drawn. Figure 13 shows PR curve of all query set for Tf-Idf Model where Figure 14 shows PR curve of all query set for BM25 model.

9. CONCLUSION AND INFERENCES

Machine learning methods are very useful in case of text based information retrieval as they help in categorizing the texts thereby reducing the entire search space. With the less search space, information retrieval techniques will perform better as time complexity in such cases will drastically improved.

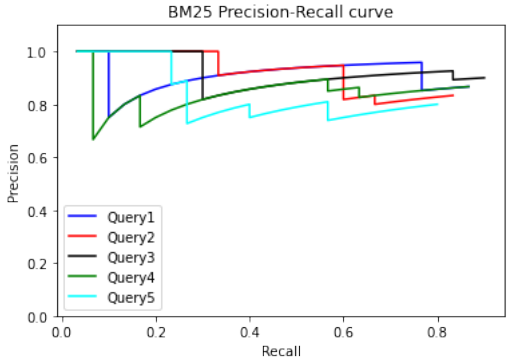

Figure 14: PR curve of BM25

We have observed that the models based on term frequency perform better than vector space model as when it comes to tweets user tends to use mixed vocabulary, combine multiple languages, and tend to use more of acronyms and slang languages which leads to poor vectorization of tweets and therefore poor similarity results.

We have also observed that Okapi BM25 and Relative Frequency techniques perform well for such problems. Cosine and Tf-Idf performs almost similar cosine measures similarity using angles between vectors. Jaccard performed average because it only considered union and intersection between set of terms and ignored the term frequency of rareness of the terms. Euclidean performed worst as it measures similarity by calculating distance between vectors.

10. USER INTERFACE

We have designed an interactive web application based on HTML, Python and Flask to retrieve top 10 tweets from collection of models chosen by users (see Figure 15). Users can search their queries using two modes.

- Text: Input from keyboard
- Audio: voice interaction via microphone and speakers.

Figure 15: Web Based User Interface

11. REFERENCES

[1] B. L. Ibtihel, H. Lobna, and B. J. Maher. A Semantic Approach for Tweet Categorization. *Procedia Computer Science*, 126:335–344, Jan. 2018.

[2] K. Lee, D. Palsetia, R. Narayanan, M. M. A. Patwary, A. Agrawal, and A. Choudhary. Twitter Trending Topic Classification. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 251–258, Dec. 2011.

[3] L. Maceda, J. Llovido, and A. Satuito. Categorization of Earthquake-Related Tweets Using Machine Learning Approaches. In *2018 International Symposium on Computer, Consumer and Control (IS3C)*, pages 229–232, Dec. 2018.

[4] K. D. Rosa, R. Shah, B. Lin, A. Gershman, and R. Frederking. Topical clustering of tweets. *Proceedings of the ACM SIGIR: SWSM*, 63, 2011.

[5] P. Selvaperumal and A. Suruliandi. A short message classification algorithm for tweet classification. In *2014 International Conference on Recent Trends in Information Technology*, pages 1–3, Apr. 2014.

[6] M. Tare, I. Gohokar, J. Sable, D. Paratwar, and R. Wajgi. Multi-Class Tweet Categorization Using Map Reduce Paradigm. *International Journal of Computer Trends and Technology*, 9:78–81, Mar. 2014.

[7] C. Xia, T. He, W. Li, Z. Qin, and Z. Zou. Similarity Analysis of Law Documents Based on Word2vec. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pages 354–357, July 2019.