CSE 535 Project 4

Analyzing the impact of political rhetoric in traditional and social media

Submitted by

Sai Suresh Dalli (sausures/50316378)

Sampreeth Reddy B(sboddire/50298591)

Paritosh Kumar Velalam(pvelalam/50295537)

Data Collection:

- Data for this project is collected from twitter API from three different languages and for 5 POIs per country (USA, Brazil and India).
- We collected this data as part of Project1 and used the same for this project.
- We have collected over 3 lakhs tweets over 3 different countries which consists both timeline tweets and replies

Converting the Multilingual data to English:

- As part of our project we need to handle multilingual data it will be difficult task to do Topic Modelling and sentiment analysis on the Portuguese and Hindi text so as a work around we have translated all the Portuguese and Hindi text to English using Google Translate cloud API and saved the translated text in the same json for each text.
- We used this translated text for the sentiment analysis and the Topic Modelling on the whole corpus collected
- This translated text is helpful for fetching results for queries in different languages if the English translated word is given as query.

Topic Modeling on collected data using LDA:

Topic modeling is a branch of unsupervised natural language processing which is used to represent a text document with the help of several topics, that can best explain the underlying information in a particular document. This can be thought in terms of clustering, but with a difference. Now, instead of numerical features, we have a collection of words that we want to group together in such a way that each group represents a topic in a document. We have huge amount of data around us in unstructured way so we need to understand it and make decisions on it. For this we use topic modeling.

We apply the same on our corpus to obtain main topics the POIs are talking about in their tweets.

Data Preprocessing for LDA Topic modelling:

- **Data Cleaning:** Remove all the special characters, emoticons, etc.
- **Tokenize**: we tokenize each sentence into list of words removing punctuations and unnecessary characters altogether. We used Gensim library for this
- **Stemming**: In this we reduced the word to word stem that is removing the prefixes and the suffixes. We used Spacy library for this

- **Create a word matrix**: We need to create a word matrix to feed as input to LDA model. CountVectorizer is used to create a word matrix on our data

Build LDA model:

- We have initialized the LDA model with number of topics equal to 15.
- Now we transformed the preprocessed data and fed it as input to LDA model.
- Which outputs the 15 clusters with the words saying that words belonging to the same cluster are more related.
- By manually seeing each cluster words we assigned a Topic to each cluster based on the prior knowledge we have on the data.
- Now we have created a LDA model which is used to predict the topic for any given text based on the fact that the text is more nearer to which cluster.
- Predict the topic for each tweet:
- Now we parse through the each tweet in our corpus and we feed the tweet text to LDA model which we created above to predict its topic and save the topic in same json for further implementation on data.

Sentiment Analysis:

Sentiment Analysis is the process of **computationally** determining whether a piece of writing is positive, negative or neutral. It's also known as **opinion mining**, deriving the opinion or attitude of a speaker.

- We have used TextBlob a python library for the sentiment analysis.
- When we feed a piece of text to this it gives the popularity of the text as greater than zero or equal to zero or less than zero.
- Based on above analysis we can sentiment any given text as follows:
    - analysis.sentiment.polarity > 0: positive
    - analysis.sentiment.polarity < 0: negative
    - analysis.sentiment.polarity == 0: neutral
- The Sentiment field is added to the same tweet json.

Related news article Collection:

We need to search for related news articles for the corpus collected as part of requirement for this we followed the below procedure

- **For Indian Tweets:** We collected the news links from the The NDTV news archive over the period we have collected our tweet corpus.
- **For USA Tweets:** We collected the news links from the CNN archive over the period we have collected our tweet corpus.
- **For BRAZIL Tweets:** We collected the news links from the Reuters archive over the period we have collected our tweet corpus.
- **Procedure for assigning news articles to the related Tweet:**
  - We have done the data cleaning, tokenization and stemming on the translated data for each tweet.
  - Then using the BeautifulSoup library we extracted the headlines and their repective urls
  - We apply all the preprocessing steps for each of the headlines text
  - We matched the Nouns in the tweet text and the Head line Text. If we get a count greater than or equal to 2 we have mapped the news article as the related news to the tweet of the particular POI.
- Related News articles and their urls are added to the same tweet json.

Web development:

We used Python Flask for backend and html and java script for front end. We used Plotly JS for plotting graphs.

Faceted Search:

Faceting is one of the important factor when we deal with the advance search systems. It helps us refine the results by breaking up the search results into multiple categories, typically showing counts for each, and allows the user to "drill down" or further restrict their search results based on those facets. We provided faceting on 4 different categories:

a. POI name
b. Language
c. Country
d. Verified profile

This helps user fetch information based on need, rather than getting all tweets matching the query. Note: When we apply POI name filter even the replies to the tweets of POI name selected are displayed. The replies can be further filtered by verified true filter as most of the replies are from unverified profiles.

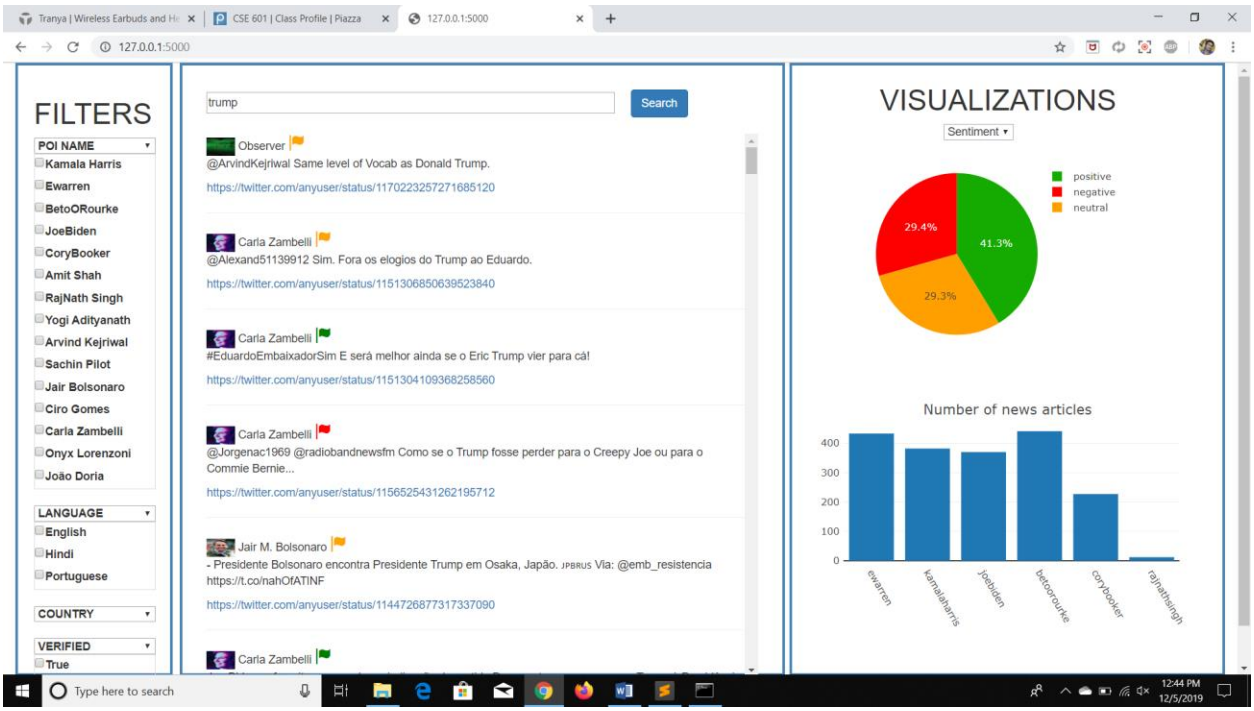

Fig 1. User Interface showing the filters by POI name, Language, Country and verified profile.

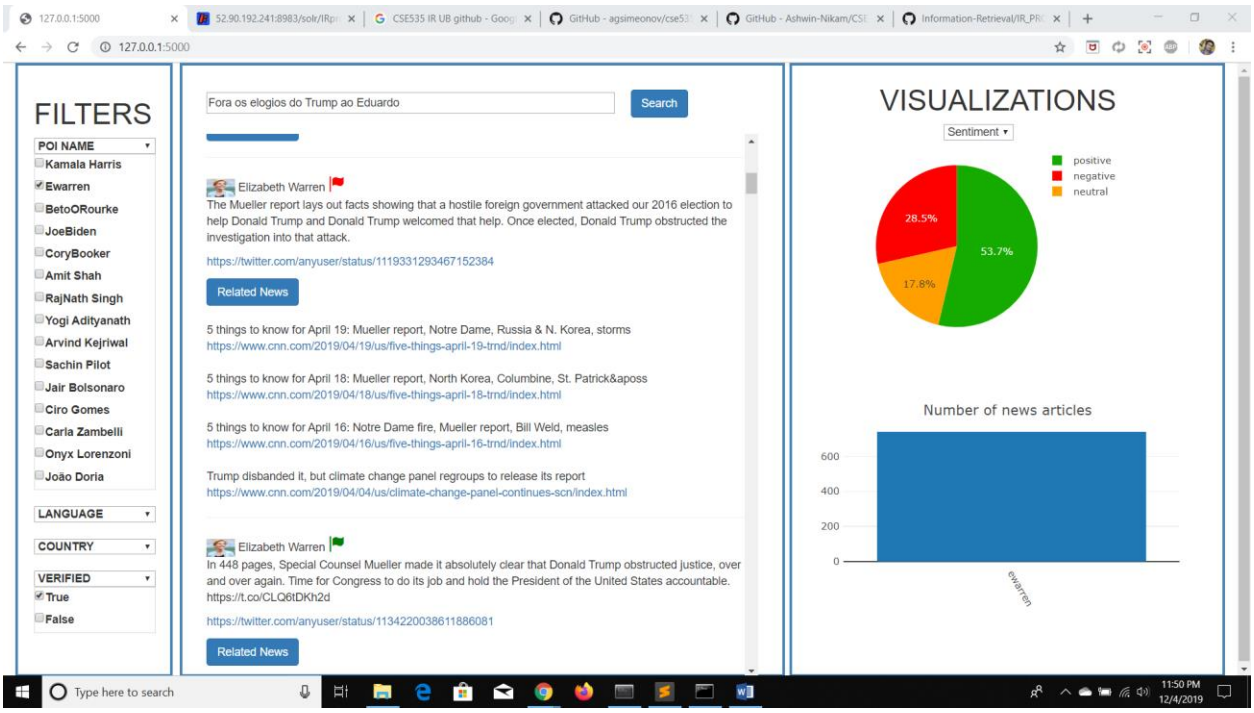

Fig 2. Results retrieved for a portuguese text

Multilingual Search:

We have implemented the search in such a way that it can accept Portuguese, Hindi and English languages. Also if a word of Portuguese translated to English is given as search

query, our search returns the results if the equivalent word is contained in the brazil corpus. The same applies to hindi text search. This has been achieved by translating the tweets and indexing them along with the original tweets. In the figure 1, it can be seen that if a portuguese

Analysis:

The following analysis is performed on the tweets retrieved. The results are obtained using the facet count of fields such as Sentiment, Tweet language, Topics, Country wise tweets, poi names, hashtags, mentions and tweet date.

a. Sentiment of Tweet

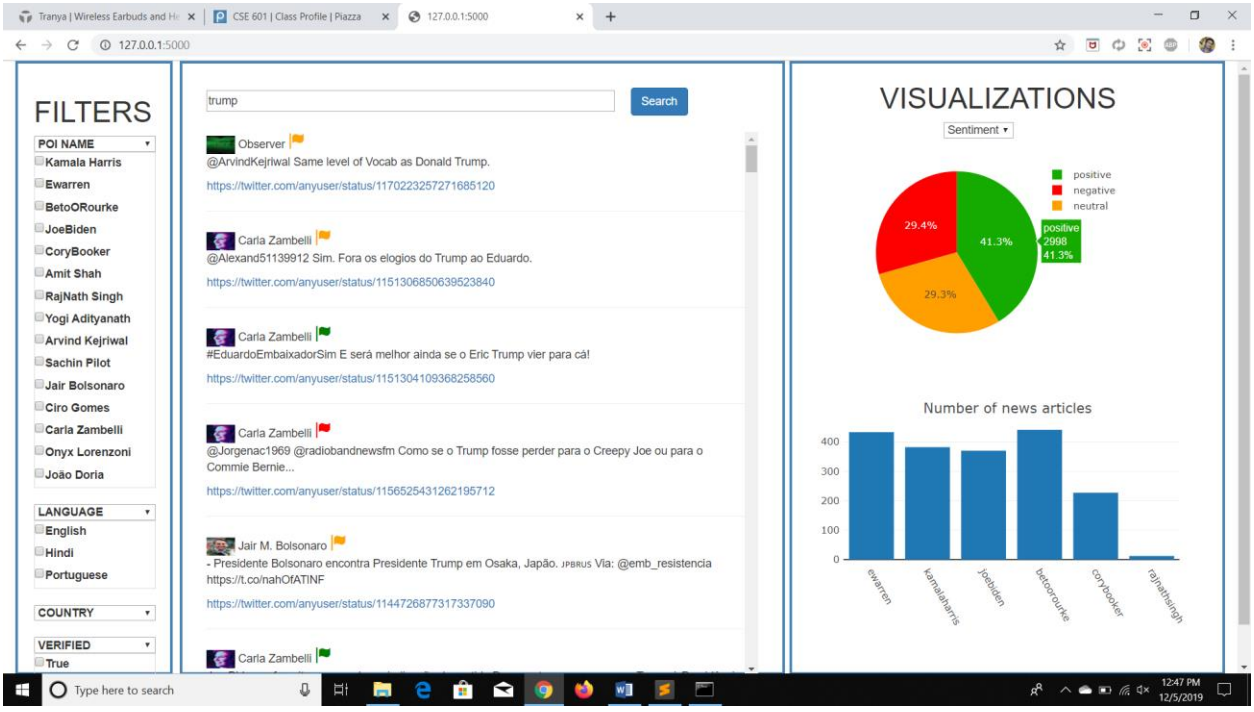

Fig 3. UI showing sentiment analysis of the tweets contained in the retrieved results

The Sentiment analysis is represented as a pie chart. If we hover around on the pie chart, the number of tweets related to positive, negative and neutral are displayed as well.

b. Language of tweet

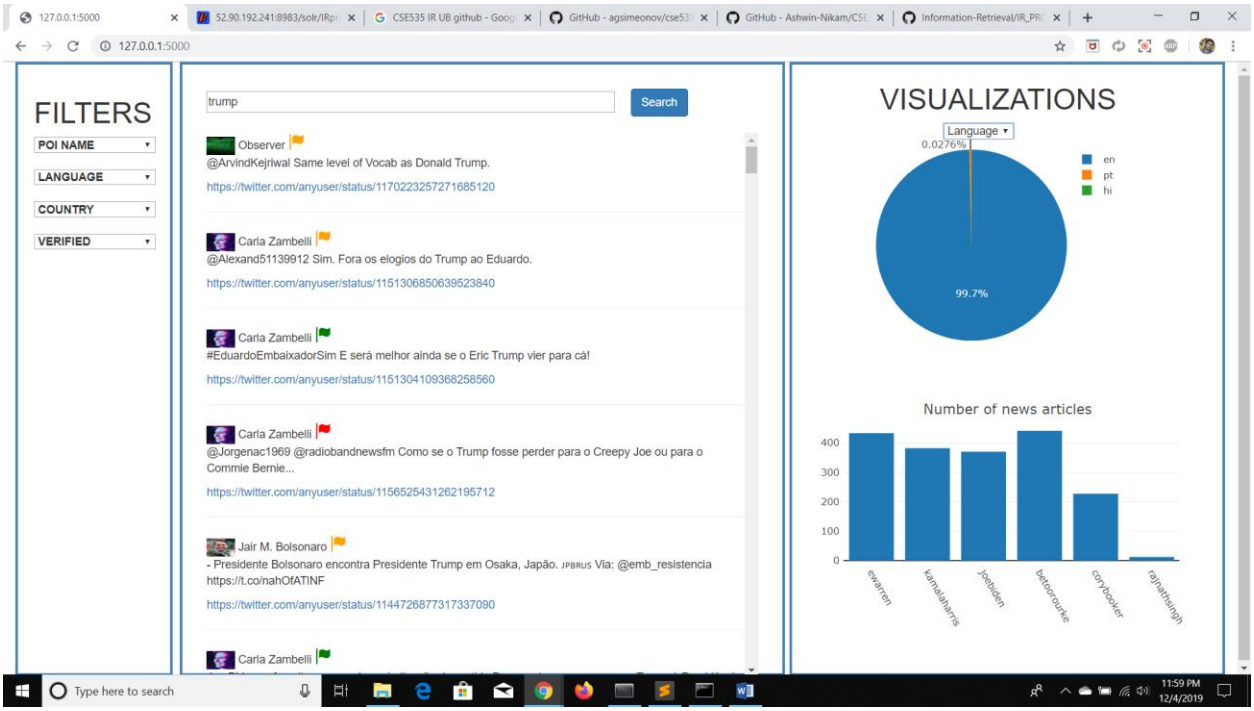

Fig. 4 Pie chart showing the number of tweets per particular language

The user can then apply language filters to see the tweet from a particular language.

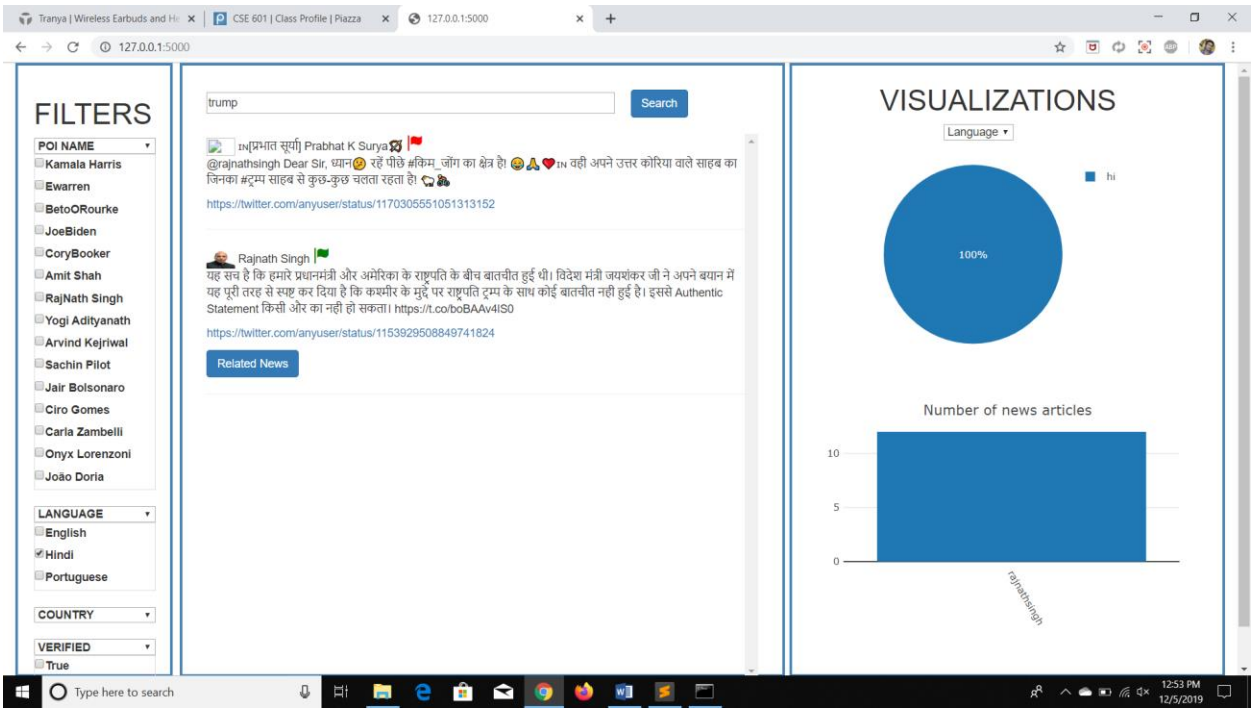

Fig 5. Results after applying language filter. Our UI handles hindi text as well.

c. Topics related to tweets retrieved based on search query

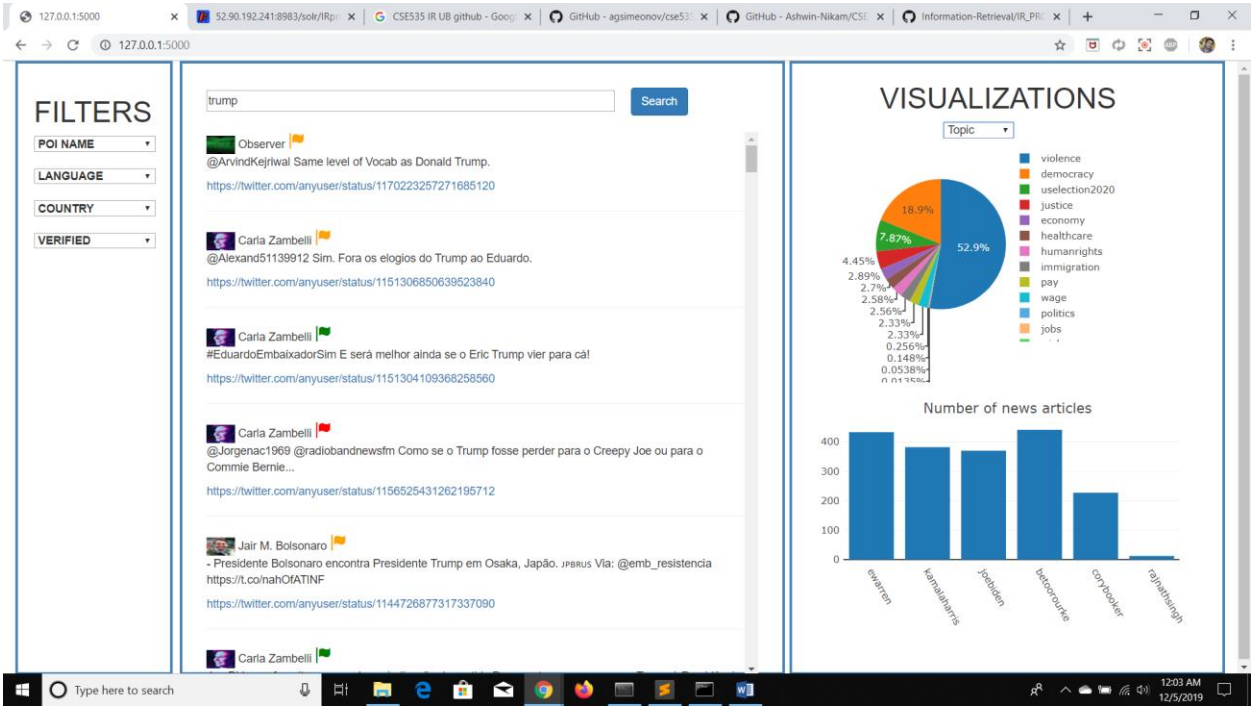

Fig 6. UI showing pie chart of the topics contained in the retrieved tweets.

d. Volume of tweets from countries retrieved based on search query

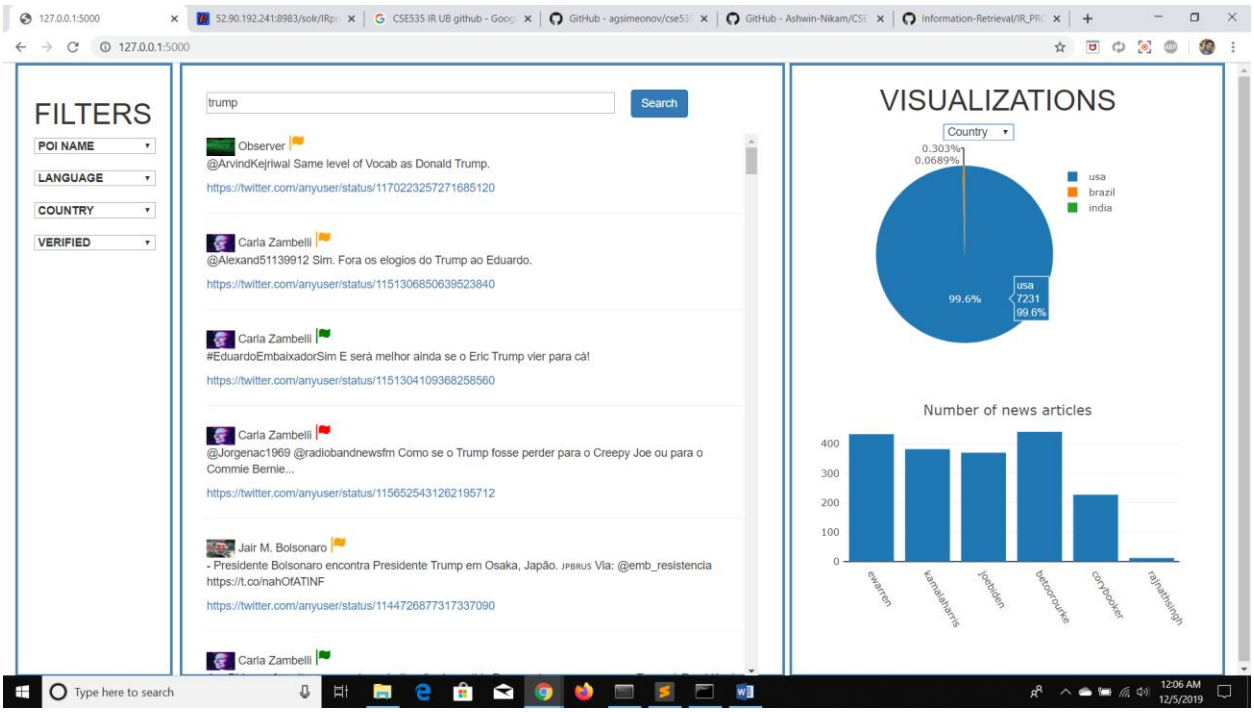

Fig 7. UI showing pie chart for number of tweets per country

e. Volume of tweets per POI retrieved based on search query

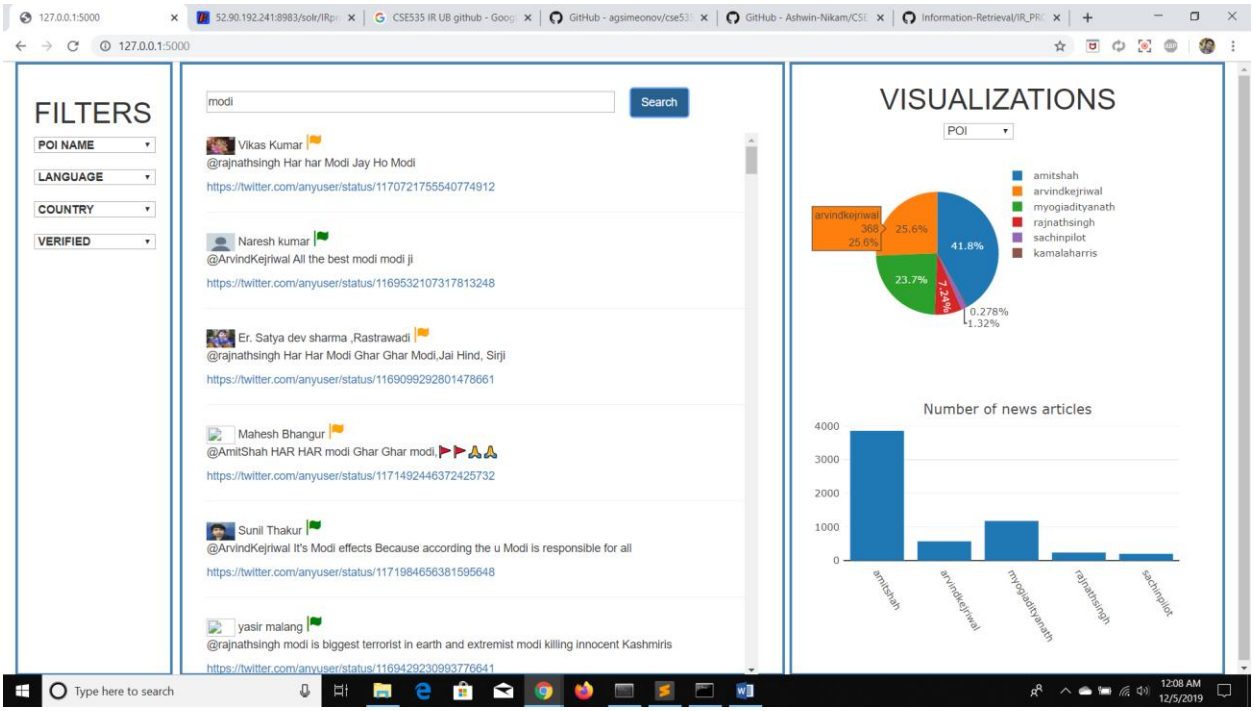

Fig 8. UI showing pie chart for the number of tweets per POI.

f. Hashtags present in the tweets retrieved based on search query

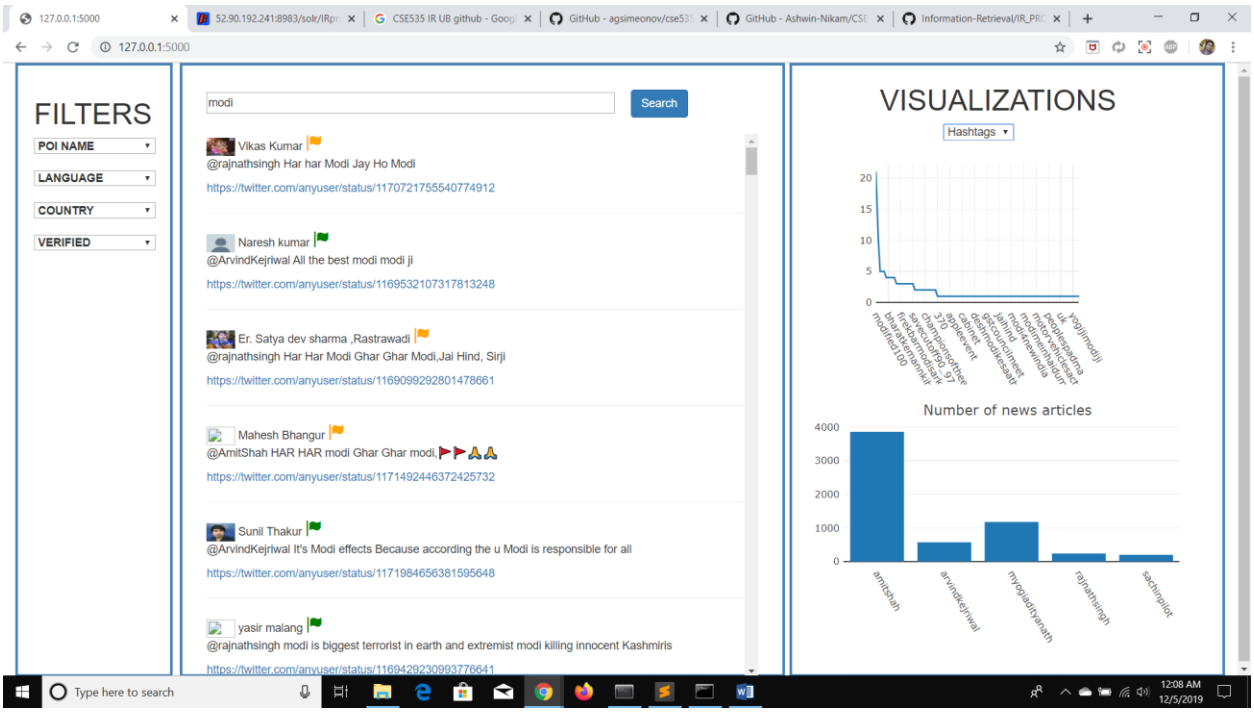

Fig 9. UI showing the top hashtags in the retrieved tweets.

g. Mentions present in the tweets retrieved based on search query

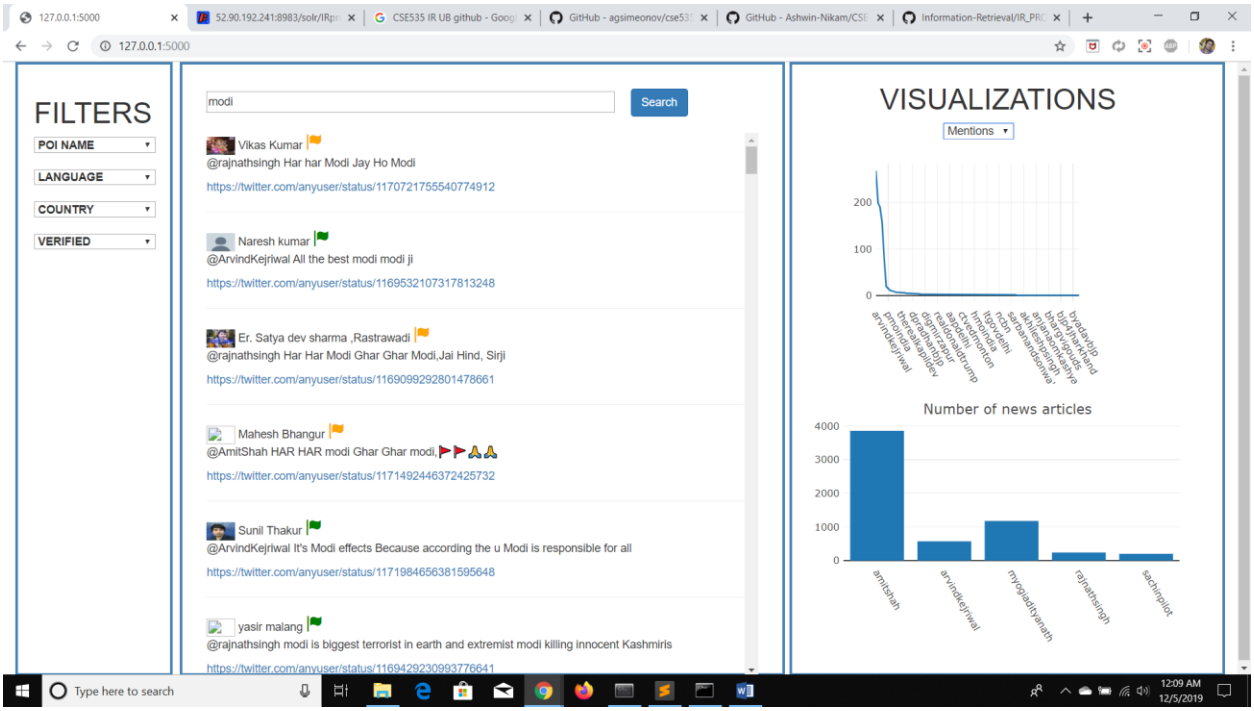

Fig 10. UI showing the top mentions in the retrieved tweets.

h. Timeline of tweets

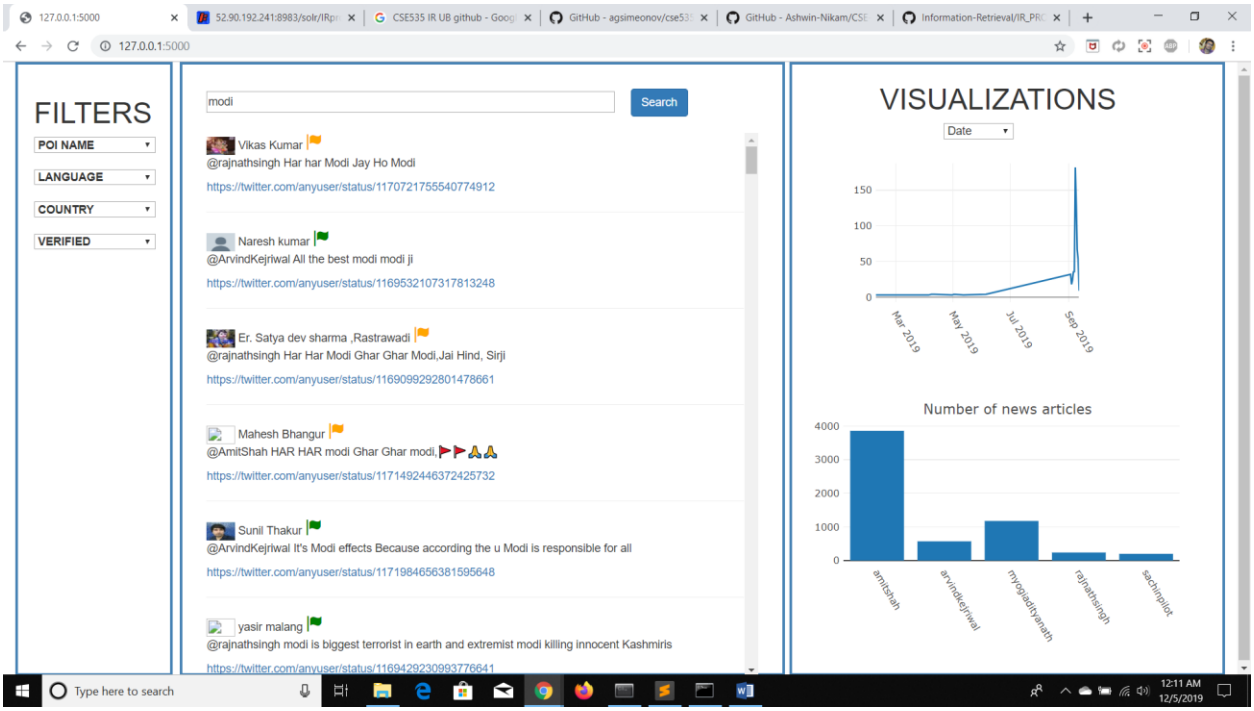

Fig 11. UI showing the time line of tweets in the retrieved results.

i. Impact of POI tweets:

To see the impact of political rhetoric of POI, the number of news articles related to the POI tweets retrieved in the search results is shown. Higher the number of news articles, greater the impact of POI tweets. Only the news articles related to POI timeline tweets are collected, as we are interested only in the impact of tweets by POI.

Fig 12. UI showing the time line of tweets in the retrieved results.

Based on the search results, the user can then observe the number of articles per POI plotted as a bar plot and then select each POI tweets by applying POI name filter and verified true filter to see the news articles.

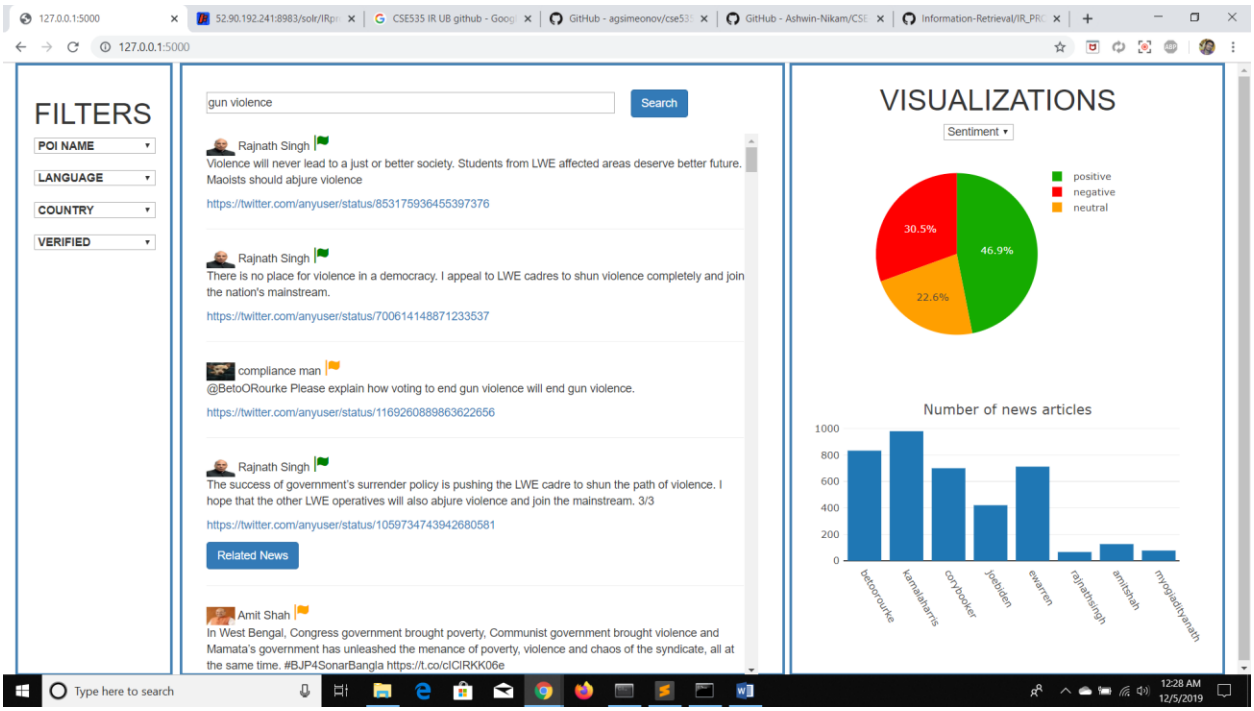

Fig 13. Results retrieved for search “gun violence” without filters

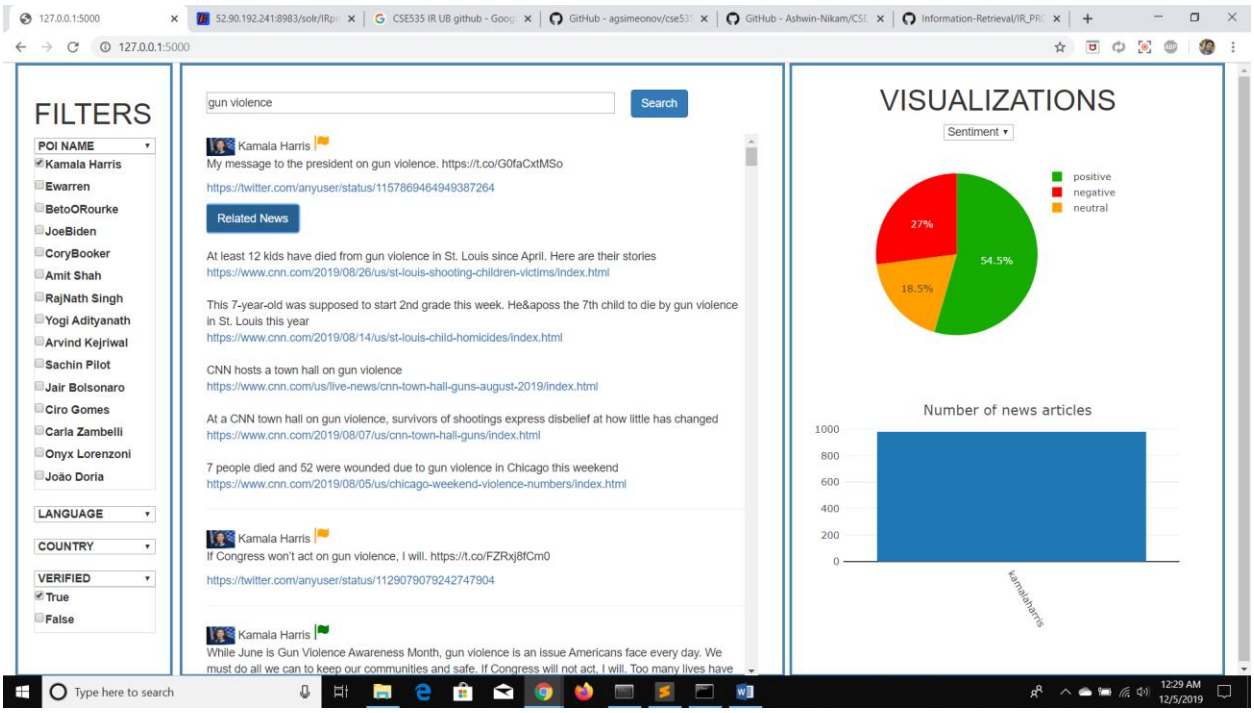

Fig 14. Results retrieved for search “gun violence” after applying
POI name: Kamala harris and Verified: True filter

Video Demonstration URL:

The functionality of our IR system is demonstrated in the video.

https://youtu.be/8Y1Ar7uyVuQ

Application URL:

ec2-54-158-114-17.compute-1.amazonaws.com

Display Resolution Settings:

Our UI has been developed on a laptop which has these display settings. It is highly suggested to use these settings when testing the application.

- Scale and Layout: 125%
- Resolution: 1920 x 1080

Team Contributions:

Team Member	UBIT ID	Person Number	Tasks
Sai Suresh Dalli	saisures	50316378	Front end
Sampreeth Reddy Boddireddy	sboddire	50298591	LDA Analysis, News articles scraping, Language Translation
Paritosh Kumar Velalam	pvelalam	50295537	Back end

References:

1. https://medium.com/@yanlinc/how-to-build-a-lda-topic-model-using-from-text-601cdcbfd3a6
2. https://cloud.google.com/translate/docs/
3. https://plot.ly/javascript/
4. https://www.crummy.com/software/BeautifulSoup/bs4/doc/
5. https://www.nltk.org/
6. https://textblob.readthedocs.io/en/dev/
7. http://flask.palletsprojects.com/en/1.1.x/