Topic Clustering on Twitter Data

561 Project Report Supervisor : Satya Siva Katragadda

Harika Karnati **hxk5815**

Sai Sree Bhagi **sxb8547**

ABSTRACT

To identify trends in social media(Twitter) over a long period of time by "Topic clustering" . The main issue here is that the size of tweet does not give a lot of leverage to communicate the complete story, so we try to identify all relevant tweets using topic clustering. All the tweets are clustered into related stories using cosine similarity model with tf-idf weights to identify similar tweets. We will also be looking at the size of the cluster and comparing it to the news reports during the same time period to identify possible correlation between data on twitter and traditional news sources.

Introduction:

Twitter is a short texted and fast growing social media. Currently, lot of research going on to identify and analysis trends using Twitter data. Tweet (single post on twitter) has limit of 140 characters in every post, it made easy to find social media trends rather than the conventional news articles, web documents and blogs. Our data contains twitter data related to medical domain.

When dealing with long-term trend analysis, the most prevalent way is to show frequency of words or terms that are given by the user. Another way is to identify related words to a given set of keywords. Although both the methods gives a basic idea of trending, they will not give us a complete picture in terms of 'why a topic is trending? or what other words are trending with it? So we need to cluster all of these words into groups of related tweets.

Our project uses the clustering approach to find relevant topics. Clustering is an easy and efficient way of processing streaming data and grouping based on topic. Clustering of tweets requires similarity and distance measure to cluster related topics. In our case, we use Cosine Similarity with tf-idf weight to compare the similarity between tweets and cluster, cluster and cluster. Every incoming tweet is compared with existing clusters to place in one of cluster or if it didn't satisfy it forms a new cluster.

In our project we reorganizes clusters time to time (in our case every week). Reorganization is process to find similarity between different clusters. If the clusters are similar, we merge two similar clusters to form into one cluster. This regrouping of similar clusters will helps to increase intra cluster similarity just by identifying and placing tweets that are similar to one topic into one cluster rather than simply treating themselves individually into different clusters and hence reduces inter cluster similarity.

Working model:

Implementation of project is diagrammatically shown in figure 1 and pseudo code below helps to understand implementation . Data is collected from twitter for medical domain between Feb - April 2013. Each tweet is taken from database. Tweets usually have lot of noise, unrelated data like stop words, links, name tags etc., First step is to eliminate the noise, uninformative words from tweet so that we can generate the informative words from tweets.

For each processed tweet, tf-idf is calculated, tf (term frequency) is ratio of word frequency to maximum frequency of all words in same tweet and idf (Inverse document frequency) is calculated by applying logarithmic function to ratio of total number of tweets word appear to total number of documents.

Clustering of each tweet is done based on the cosine similarity between tweet and existing clusters. Cosine similarity is measure based on tf-idf weights of the two tweets or clusters . Tweet is placed in cluster which similarity is greater than threshold and greater than other clusters. If none of clusters exists or satisfies threshold conditions then tweet is placed in new cluster.

Every cluster is reorganized periodically (in our implementation every week). In reorganization process, each cluster is compared with other clusters using cosine similarity as explain above. All similar groups are merge into one. The threshold used in reorganization process is different from the threshold used in formation of clusters. Least predominate words from the cluster is removed from the cluster by using mean of tf-idf of words. If tf-idf of the word is less than mean of tf-idf of cluster, we are assuming as least dominate word. In this way we can ensure cluster contains the words that emphasis topic of cluster.

Working Model talks about the detail implementation of clustering technique.

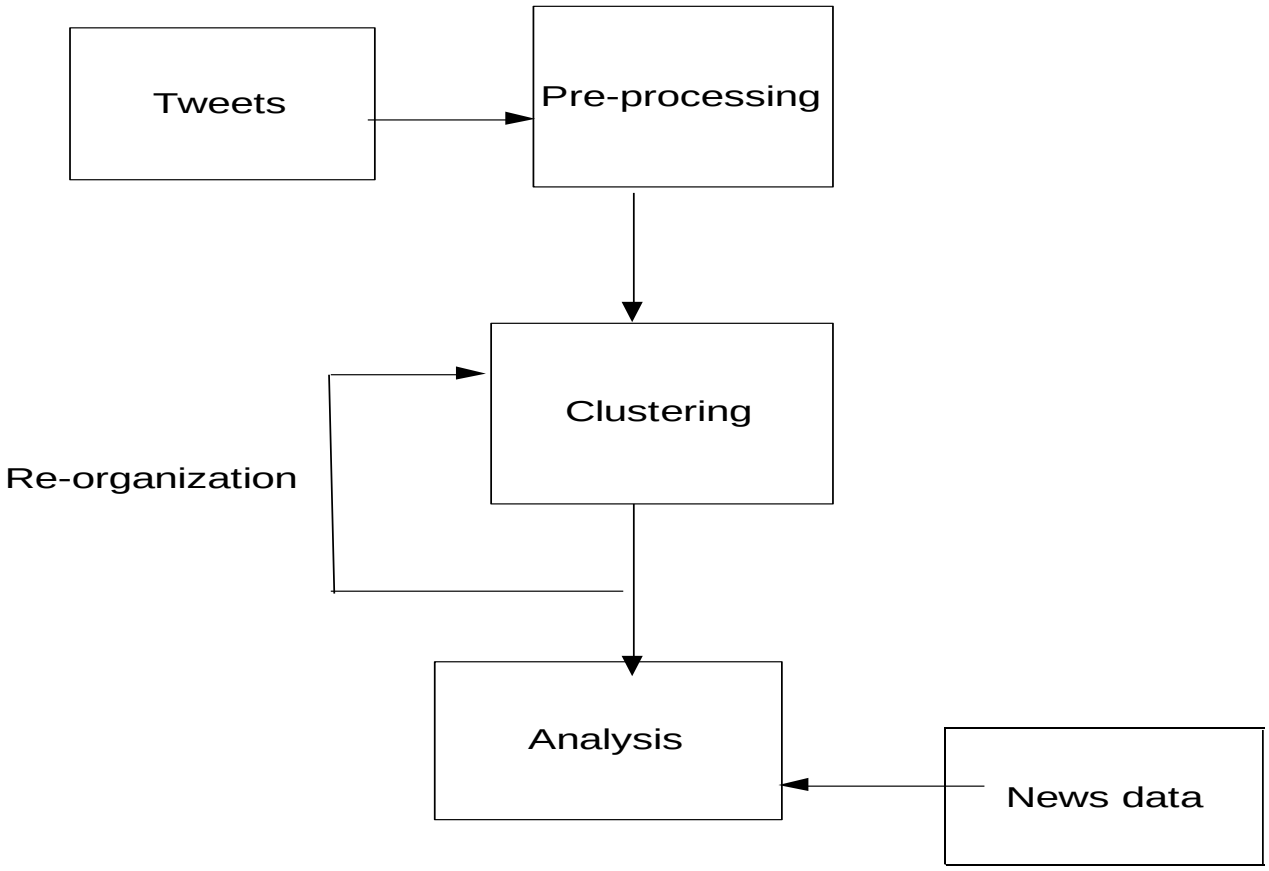


Figure 1

Input: Tweets in twitter data and news data

Output: Generating trend over analysis graph of most representative term versus normalized frequency for each cluster that form a topic in twitter data and news articles for users over a period of time.

pseudo code of Implementation

```
for each incoming tweet
 removeTags(tweet)
removeStopwords(tweet)
words = convertTweetstoWords(tweet)
if(time > week)
{
//reorganize
for (all_clusters)
```

{

calculate tfidf;

calculate cosine similarity with other clusters;

if(similarity > clusterThresold)

{

merge two clusters;

}

}

}

calculate tfidf of tweet;

for all_cluster:

calculate cosine similarity between tweet and cluster;

if similarity > threshold and similarity > othercluster:

place tweet in cluster

if no_clusters exist OR tweet_not_similar:

create a new cluster;

end tweets

Methodology:

We used agglomerative clustering approach to implement "Topic clustering on twitter data"

Steps for implementation:

1. **Reading data:** Reading all the tweets in twitter data and news data

2. **Pre-processing the data:** Removing uninformative words like hashes,URL's and special characters,length of words having size less than or equal to 2 in all the tweets in news data and drug data. In this step, we finally left with only informative words in all tweets.

3. **Training the data:** we train all pre-processed tweets in twitter data and news data to TF-IDF model. By using formula of tf-idf, we calculate term frequency and inverse-document frequency for each term in the tweet. Multiplying these factors together gives us tf-idf for each tweet.

This is given by the formula:

$$\operatorname{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

$$\operatorname{tfidf}(t, d, D) = \operatorname{tf}(t, d) \times \operatorname{idf}(t, D)$$

where t be term,d be the document which contains 't' and 'D' be the total no. of documents in the collection.

4.**Clustering the tweets:** After calculating TF-IDF for each processed tweet, in the initial step of clustering, we made each tweet as one cluster.

5. **Cosine similarity calculation between two tweets(clusters):** In the present model we considered cosine similarity threshold of 0.01 and cluster cosine threshold of 0.75. Generally it is calculated between incoming tweet and clusters holding one tweet in early iterations, clusters holding multiple tweets in later iterations.Multiplying and summing up of TF-IDF factor together for all terms in incoming tweet vs tweet in clusters and dividing it by square-root of sum of squares of TF-IDF factor for all terms separately for each tweet gives us cosine similarity value between each two processed tweets in the corpus.

We calculate cosine similarity between two clusters. If the cosine similarity between any two clusters is greater than cluster cosine threshold, we are merging those clusters. Again we calculate similarity between incoming tweet and cluster holding multiple tweets. If the similarity value is greater than above mentioned threshold values, we place this incoming tweet into new cluster. Likewise, we did this process for all tweets & clusters untill we find no clusters and no tweet similar to tweets which are there in old clusters. Likewise,we calculate cosine similarity between incoming tweet and cluster holding one tweet,incoming tweet and cluster holding multiple tweets for all the processed tweets in twitter data and news data.

This is given by the formula:

$$\cos \operatorname{sim}_{tf}(t, s) = \frac{\Sigma_{d \in \mathcal{C}} Q_t(d) Q_s(d)}{\sqrt{\left(\sum_{d \in \mathcal{C}} Q_t^2(d)\right)\left(\sum_{d \in \mathcal{C}} Q_s^2(d)\right)}}.$$

where t,s are tweets which can be taken in vector notation.

6. **Re-organization of tweets in clusters:** If the cosine similarity between any two tweets is greater than any one of the threshold value in step 5, re-organization of tweets from one cluster to another cluster takes place dynamically for all tweets in corpus. Tweet is placed in cluster in which similarity is greater than threshold and greater than other clusters. If none of clusters exists or satisfies threshold conditions then tweet is placed in new cluster.

7. **Topic clustering:**

Based on the cosine similarity value between incoming tweet and tweets in clusters, we place similar tweets into one cluster everytime. Cosine similarity value of tweets in clusters can be compared with cosine threshold & cluster cosine threshold values everytime. Likewise, reorganizing and grouping of similar tweets back into the cluster related to same topic should be done for all the processed tweets in drug data and news data. Above step yields in "topic" identification for all clusters just by grouping all similar tweets in the corpus into one cluster.

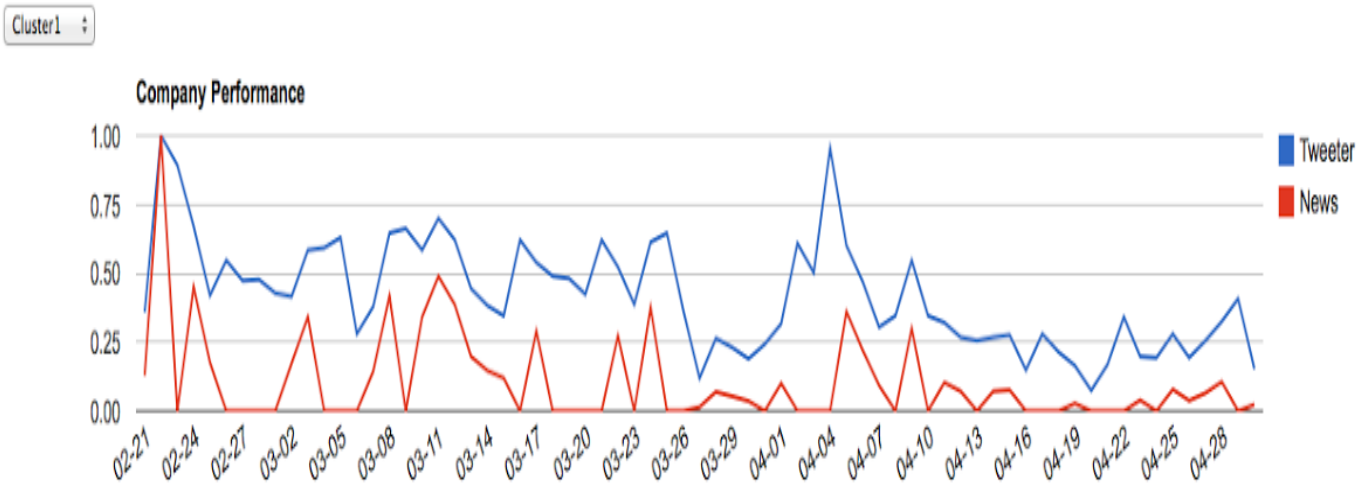
7. **Analysis of clusters:** It is done by trending the clusters we got in step 7 in twitter data & news data during Feb-April 2013. By plotting a graph between terms versus normalized frequencies in each cluster, we inferred how well the terms in cluster form a "topic" for users over a period of time. Through these graphs we identify possible correlations for all clusters that form various topics in twitter data versus news articles in corpus.

Results:

Program saves clusters into output file in JSON format. Validation of results is done by correlation of twitter tweets trends to news trends. News trends are generated manually by searching the google news website with dominant words of cluster over same time of tweets.

Effect of Threshold: We tried different thresholds to assign a tweet to a cluster. The parameter seems to affect the quality of clusters formed. We tested 8 different threshold values, 0.01 to 0.5. As the threshold value increases the number of clusters increases, due to the requirement of high relevance between incoming tweet and established cluster. In the table below we show the number of clusters formed along with average number of tweets in cluster and representative words in cluster. From the results we can see that as the threshold increases the number from clusters also increases. From the median value decrease, most of these clusters are noise, but based on representative words we can see that most word clusters retain their topic model through different values.

Threshold	# Cluster	Avg # Tweets per cluster	Median # of tweets per cluster	Avg # of representative words	Median # of representative words
0.01	29	296.8965517	10	18.10344828	9
0.05	93	92.58064516	12	22	16
0.1	226	38.09734513	8	16.13716814	14
0.2	694	12.40634006	4	11.52161383	10
0.3	1165	7.39055794	2	10.02832618	9
0.4	1456	5.98430154	1	7.7643178	6
0.5	2134	3.12539467	1	5.2445234	6



Words in Cluster

cymbalta depression help hurts

Words in Cluster

Or take cymbalta

Or take cymbalta

@HauteMessFiasco swollen hands, sleepy all day, not enuff sleep at night, achy bones, headaches. feels similar to coming OFF cymbalta to me.

@HauteMessFiasco swollen hands, sleepy all day, not enuff sleep at night, achy bones, headaches. feels similar to coming OFF cymbalta to me.

depression hurts, but cymbalta helps....I think that's what that commercial says

depression hurts, but cymbalta helps....I think that's what that commercial says

Both data from clusters and data from news feeds are plotted to line graph. Plotting is done using google Visualization Library. The figure shows correlation of one cluster over time for twitter data and news feeds. In figure, It shows correlation graph, words in the cluster and actual tweets.

Future Work:

After the most representative words are selected, there should be a way to identify co- occurrences in the tweets to identify the actual number of words occurring instead of just using max frequency to identify trends. Another possible direction is to incorporate a way to identify temporal distance along with cosine similarity.

References:

1. Wartena, C.; Brussee, R., "Topic Detection by Clustering Keywords," Database and Expert Systems Application, 2008. DEXA '08. 19th International Workshop on , vol., no., pp.54,58, 1-5 Sept. 2008!

2. http://en.wikipedia.org/wiki/Tf HYPERLINK "http://en.wikipedia.org/wiki/Tf–idf"–idf!

3. http://en.wikipedia.org/wiki/Cosine_similarity!

4. Sungchul Kim; Sungho Jeon; Jinha Kim; Young-Ho Park; Hwanjo Yu, "Finding Core Topics: Topic Extraction with Clustering on Tweet," Cloud and Green Computing (CGC), 2012 Second International Conference on , vol., no., pp.777,782, 1-3 Nov. 2012