Evaluation of Tweet Clustering Metrics

Trishnendu Ghorai
Soham Poddar

Abstract: Various data was extracted from tweets and tested with different similarity metrics, clustering algorithms and evaluation metrics, to find which metrics are good for segregating of tweets.

1 Data Provided:

The tweets were collected by the following procedure:
9 broad topics were selected(e.g., music, technology, politics, etc) and some popular hashtags were chosen from each. 2000 recent tweets containing each hashtag were collected. Exact duplicates (retweets of the same tweet) were removed.

Each tweet was provided as a JSON string with the following attributes:
1. *The unique Identifier of the tweet (id)*
2. *The actual text of the tweet (text)*

2 Data Processing:

200 tweets were taken at random from each subtopic for testing and URL's , punctuations and stop words were removed from all tweets. Words with non ASCII characters were also removed.

Another dataset was formed by removing the '#' symbols from the above data. A third dataset was formed by stemming this. A fourth dataset was formed by keeping only the hashtag words.

3 Similarity Metrics

Similarity Metric or Measure is a function that defines the similarity between two documents, or in this case, two tweets.

3.1 Co-Occurrence based Metric

Each tweet was broken down into a set of words. For example "I'm Feeling good" becomes {"im", "feeling", "good"}.

The Intersection and Union of the sets of two tweets were calculated. Finally, Jaccard Similarity of two tweets was calculated by $\frac{|I|}{|U|}$.

3.2 Topic Modeling based Metric (LDA)

Topic Modelling is a technique of finding 'K' latent topics from a set of documents. Each topic can be represented by a set of words.

We used the LatentDirichletAllocation class from Scikit-Learn package to get the topics and we used them to classify each tweet as a 'K'-vector representing the probabilities of each topic being in the given tweet. Finally the cosine similarity between two tweets were calculated from these vectors using:

$$\frac{(A \,.\, B)}{(|A|\,|B|)}$$

3.3 Word2Vec Model

Word2Vec Module takes the tweet corpus and n-features are selected. For each word we get an n-vector containing the relation probability of the word with the n features.

We used the Word2Vec class from Gensim package. For each tweet we took the average score of the words in the tweets. We used them to classify each tweet as a 'K'-vector as before. Finally the cosine similarity between two tweets were calculated from these vectors as before.

4 Clustering

We calculated the similarity between all pairs of tweets using the above metrics. Now we label them into 'K' clusters using the couple of clustering algorithms below.

4.1 Kernighan-Lin Algorithm

The K-Lin algorithm is a general graph partitioning algorithm. The nodes are partitioned into two sets randomly at first. The total cost of inter-partition edges is calculated. It then repeatedly exchanges nodes to minimize the cost of inter-partition edges. After breaking it into 2 partition, it repeats the algorithm for each of the two clusters and keeps going for required depth.

We implemented an optimized version of the algorithm by Fiduccia and Mattheyses. It works good on cliques and almost equal partitions.

4.2 K-Means (Floyd's Algorithm)

K-Means Algorithm is a clustering algorithm that clusters various points in N-Dimensional space. K centres are chosen at random first, and all the points are assigned to the nearest centre. The centres are then re-calculated within each cluster and points are reassigned to each new centre. This process is repeated until we find a convergence.

We used KMeans class from Scikit-Learn to convert the list of tweet vectors into a list of labels for each tweet, denoting its predicted cluster.

5 Evaluation Metrics

To Measure the 'goodness' of a clustering, the following evaluation measures were used to compare the predicted cluster labels for the tweets(P), with the Gold-Standard labels of the tweets(S).

5.1 V-Measure

V-Measure is the an entropy based measure. It is the harmonic mean of the homogeneity h and completeness c. Its value ranges between [0,1] with 1 being the best cluster, and 0 being the worst.

$$V(S,P) = \frac{2.h.c}{h+c} \qquad h = 1 - \frac{H(S|P)}{H(S)} \qquad c = 1 - \frac{H(P|S)}{H(P)}$$

Where $H(A)$ *and* $H(A|B)$ represent the Entropy of A and conditional entropy of A w.r.t. B respectively.

5.2 NVI-Measure

NVI measure is another evaluation metric which is a normalization of the VI-measure. It is calculated as follows. Greater values represent worse clusters, 0 representing the best cluster.

$$NVI = \begin{cases} H(P) & if\ H(S) = 0 \\ \dfrac{H(S|P) + H(P|S)}{H(S)} & else \end{cases}$$

6 Results

Dataset : with no hash			
Clustering : FM			
	Similarity	Vmeasure	NVImeasure
32/42	Jaccard	0.773423332	0.437649036
	Word2vec	0.375059953	1.207648402
	Lda	0.474227198	1.016016105
8/9	Jaccard	0.379119975	1.210202747
	Word2vec	0.196027806	1.566706137
	Lda	0.260302404	1.441400195
Clustering : Kmeans			
	Similarity	Vmeasure	NVImeasure
42/42	Jaccard	0.740147468	0.488584126
	Word2vec	0.193280063	1.498893916
	Lda	0.615380678	0.764187584
32/42	Jaccard	0.673425082	0.579521196
	Word2vec	0.182407142	1.473019739
	Lda	0.59954583	0.755235236
8/9	Jaccard	0.21046175	1.098018514
	Word2vec	0.097839295	1.670153464
	Lda	0.359270043	1.204202989

Fig 1.a

Dataset : with hash			
Clustering : FM			
	Similrity	Vmeasure	NVImeasure
32/42	Jaccard	0.80343457	0.407527156
	Word2vec	0.380360655	1.197407011
	Lda	0.493759845	0.978270017
8/9	Jaccard	0.366618555	1.234557078
	Word2vec	0.21887125	1.522165688
	Lda	0.283872646	1.395492161
Clustering : Kmeans			
	Similrity	Vmeasure	NVImeasure
42/42	Jaccard	0.745536035	0.477210485
	Word2vec	0.197770306	1.451976144
	Lda	0.624863305	0.73924886
32/42	Jaccard	0.705135619	0.52300068
	Word2vec	0.195366876	1.414116138
	Lda	0.621553269	0.719550339
8/9	Jaccard	0.145873583	1.180181563
	Word2vec	0.12194282	1.52650726
	Lda	0.326418591	1.240187632

Fig 1.b

Dataset : with no hash stem			
Clustering : FM			
	Similarity	Vmeasure	NVImeasure
32/42	Jaccard	0.767051634	0.449952768
	Word2vec	0.369593201	1.218213682
	Lda	0.480959253	1.003006182
8/9	Jaccard	0.421046465	1.502149753
	Word2vec	0.184626303	1.588885605
	Lda	0.260633379	1.440757449
Clustering : Kmeans			
	Similarity	Vmeasure	NVImeasure
42/42	Jaccard	0.733270724	0.50205389
	Word2vec	0.188767414	1.522516595
	Lda	0.657148827	0.677746312
32/42	Jaccard	0.705183572	0.520746473
	Word2vec	0.172452346	1.485030326
	Lda	0.622198127	0.716741311
8/9	Jaccard	0.152650012	1.101985098
	Word2vec	0.093655977	1.674792024
	Lda	0.383447081	1.18290703

Fig 1.c

7 Inference

The Jaccard Similarity performed best in most of the cases, and LDA performed good in the other cases. Word2Vec lagged behind.

References

Programing Tools:

1. Scikit-Learn Modules
2. Gensim Module

Papers/Lectures:

1. Efficient Clustering of Short Messages into General Domains - Oren Tsur, Adi Littman, Ari Rappoport.
2. Graph Partitioning and Graph Clustering in theory and practice by Christian Schulz - Karlsruhe Institute of Technology