Using Social Media to Enhance Emergency Situation Awareness

Web Information Retrieval

Group Name: Kernel_Panic()

Andrea Mastropietro 1652886

Umberto Mazziotta 1647818

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza
Università di Roma

Papers used

- Selected paper: Jie Yin, Andrew Lampert, Mark Cameron, Bella Robinson, and Robert Power. Using social media to enhance emergency situation awareness. IEEE Intelligent Systems, 27(6):52–59, 2012.
- Support papers:

1. Kevin Stowe, Michael J Paul, Martha Palmer, Leysia Palen, and Kenneth Anderson. Identifying and categorizing disaster-related tweets. In Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media, pages 1–6,2016.
2. Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Philip S Yu, and Hongjun Lu. Parameter free bursty events detection in text streams. In Proceedings of the 31st international conference on Very large data bases, pages 181–192. VLDB Endowment, 2005.

Datasets

We have two datasets consisting of collection of tweets gathered during the outbreak of Hurricane Sandy in NYC in 2012

1. Labeled dataset used for classification and clustering. Original dimension: 7’490 tweets (retrieved 5821).
2. Unlabeled dataset used for burst detection and clustering. Original dimension: 6’554’744 tweets (retrieved 1003864).

Due to Twitter privacy policy the text of the tweets had to be retrieved via Twitter APIs by passing the tweet ID.

Aim of the project

Our project consists into the classification and clustering of disaster-related tweets.

1. Firstly, we performed the classification of tweets among given classes, based on support paper 1. We determine if a tweet is relevant for the event or not. We then determine if a relevant tweets belongs to a specific class or not.

2. Secondly, we developed a Burst Detection module in order to find tweets regarding upcoming unordinary events, based on the selected paper and on support paper 2.

3. Thirdly, we performed online clustering for topic discovery on unlabeled tweets, based on the selected paper.

4. Finally, we run clustering also on labelled tweets in order to compare the patterns found by the clustering algorithm and the annotations made by human experts.

Classification – Baseline Features

- We performed model selection and parameter tuning using a grid search. We selected the model with the highest F1 score, which was a linear SVM using tf-idf with stemming, stop-word elimination and capitalization removal.
- The model employed in the paper is a linear SVM using unigram counts.
- The use of tf-idf as baseline features led us to higher scores for certain classes by capturing the importance of rare terms.
- On the contrary, due to the scarcity of data, some classes are terribly classified.

	Our model			Paper model			Samples
	F1	P	R	F1	P	R	
Relevance	.66	.83	.55	.66	.80	.56	1077
Actions	.23	.89	.13	.26	.44	.19	151
Information	.61	.67	.56	.33	.57	.24	346
Movement	.00	.00	.00	.04	.04	.04	30
Preparation	.24	.83	.14	.30	.44	.23	107
Reporting	.77	.72	.82	.52	.76	.40	701
Sentiment	.54	.68	.45	.37	.64	.26	415

Classification – All Features and Feature Selection

- Along with tf-idf, we also use the date of the tweet as a one-hot vector and Word Embedding vector by taking the mean embedding of vectors (200-entry vector).
- For WE we used a model by Stanford and a model trained by us on the unlabeled dataset, yielding very similar scores.
- We performed classification also using feature selection. (Truncated SVD for tf-idf and PCA for Word Embeddings).
- Tf-idf resulted to be the most characterizing feature.

	Our model			Paper model		
	F1	P	R	F1	P	R
Relevance	.72	.82	.65	.71	.81	.64
Actions	.40	.74	.27	.39	.46	.65
Information	.65	.73	.59	.48	.57	.41
Movement	.00	.00	.00	.07	.10	.07
Preparation	.20	.50	.12	.36	.41	.32
Reporting	.76	.70	.83	.73	.71	.75
Sentiment	.55	.56	.54	.53	.58	.49

All features

	Our model			Paper model		
	F1	P	R	F1	P	R
Relevance	.73	.81	.67	.72	.79	.66
Actions	.31	.69	.20	.41	.42	.40
Information	.64	.71	.59	.49	.50	.49
Movement	.00	.00	.00	.08	.10	.07
Preparation	.16	.75	.09	.36	.38	.35
Reporting	.77	.72	.83	.75	.71	.80
Sentiment	.60	.65	.55	.52	.52	.52

Best features

Burst Detection Module

- In order to reduce the amount of tweets processed by the clustering, we use this module to filter non relevant tweets
- The module considers relevant tweets that are about uncommon events, some of these events in the dataset are Hurricane Sandy hitting NYC, or the presidential elections.
- We reduce all the tweets to a set of features, stemming the tweets and removing stop-words.
- The module compares the probability of a feature to appear in a time window with the expected probability in order to determine whether a tweet is bursty or not.
- For each time window we build an in-memory inverted index on features, once we determine what feature are bursty, we merge the posting list getting the documents containing such features.

Burst Detection Evaluation

- We don’t have labels on the dataset, so we cannot evaluate how well the module performs, so what we do is to build one test set artificially.
- We used the main events happening in the days of the tweets forming the dataset, to identify some features that are for sure bursty.
- From the tweets on which we didn’t perform the training of the module, we collect all tweets containing such feature + some random tweets to add some noise.
- We considered as bursty only the features identified, while the others are considered non bursty.
- Running the module we computed the recall and the false alarm rate.
- In a first version we didn’t filter any tweet since we performed too high on false alarm, so we improved it, deepening the algorithm grasping the ideas of the second support paper.
- We get a recall of 47.16% against the 72.16% of the authors of the paper.
- We get a false alarm rate of 14.53% against 1.4%.

Online Clustering

- We use the clustering module in order to discover topics form the tweets.
- Problem: simulate online arrival of tweets; done using time windows.
- Since we access the tweets one time window at a time, we cannot use algorithms like k-means, where we need all the documents at once.
- We perform the clustering in a vector space.
- The feature vectors representing the tweets are the tf-idf weights of the terms contained in a tweet or a feature vector computed using word embeddings.
- We use a modified version of cosine similarity in order to clusterize the tweets, taking into account also the time differences.
- We add a tweet to the cluster with the closest centroid, with a distance above a certain threshold delta. If no suitable cluster is found, a new one is created.
- Problem: no ground truth, we used silhouette score. Evaluating on few time windows we get a score of 0.71 against 0.42 of the paper.

$$sim(T_i, C_j) = cos(T_i, C_j) e^{-\frac{(t_{T_i} - t_{C_j})^2}{2\sigma^2}}$$

Conclusions

- General problem: tweets are noisy! People tweet almost about anything.
- Classification problems given by the scarcity of data (main problem, even for the authors of the paper). Using tf-idf helped us to improve classification results even better than more sophisticated techniques.
- Burst Detection problem: bursty events are not only about disasters and emergencies.
- Running the clustering on the labeled dataset we discovered interesting things:
  - Either we obtain an high silhouette score (0.76) with a small number of clusters (2), thus an having a minimal overlapping.
  - Or we get a very low score (-0.12) but with a more realistic number of clusters (9). Such overlapping is normal, due to the fact that the dataset is multilabeled, so a cluster containing Information and Sentiment tweets will for example overlap with a cluster containing Sentiment and Reporting tweets.
- For further details see the report.
- Source is available at: https://github.com/AndMastro/EmergencyAwareness