Using Social Media to Enhance Emergency Situation Awareness

Danilo Marzilli - Andrea Lombardo - Daniele Davoli

Project for the Web Information Retrieval course-La Sapienza, Università di Roma.

July 7, 2019

Using Social Media to Enhance Emergency Situation Awareness:

The project wants to detect an emergency situation in real time through tweets flow scanning.

1 ABSTRACT

Microblogging platforms, such as Twitter, provide active communication channels during crisis or calamity events like earthquakes, floods, typhoons or conflicts.
During these events, affected people often post useful information on Twitter which can be used for increasing situational awareness and other humanitarian disaster responses, if they are processed timely and effectively.
The examined paper [1] investigates real-time detection of earthquakes and floods through tweet's analysis.
In order to detect a target event, the authors produced a classifier of tweets based on some specific features: the presence of keywords in a tweet, the number of words and their context.
It is made up of three steps: the first one aimed to detect bursts for unexpected incidents, the second one for the classification of impact assessment and the last one to make online clustering for topic discovery.

At the first step, they detected bursts comparing the frequency of a word in the last window time respect to its historical average.
In order to assess the impact of the earthquake on the infrastructures, they extracted different features from tweets to instance word length, word bigram, number of hashtag and then they used two classifier, SVM and Naïve Bayes, to identify if that tweet was about damage to infrastructures or not.
For online clustering, they represented each tweet as a vector in a multi-dimensional space in which each component was the TF-IDF weight related to that word; then they used cosine similarity to aggregate similar tweets to the same centroid and if the centroid became too old it was deleted.

These three steps were linked together in order to help the Australian Crisis Coordination Centre to improve their hazard monitoring and situation awareness and to be more precise when delivering assistance to population and local forces.

1.1 Authors Keywords

Event detection; Twitter; stemming; semantic analysis; Cresci-SWDM15 dataset, feature extraction, SVM, Naïve Bayes, clustering based upon centroid, NumPy library, cosine similarity.

2 INTRODUCTION

Inspired by the previously described paper, we designed a Twitter-based earthquake and flood detection system applying some changes and simplifications.
In this section, we will briefly describe the guidelines we used for the development of our project; we will explain them more in detail in the following sections.
The authors developed a real-time earthquake reporting system using Australian tweets, in this way they could filter tweets and focus on only the ones which were related to earthquake.
For the dataset they recorded tweets coming from a one-year period, and cause the dataset was not publicly available we could not follow the same approach. To overcome these issues, we used a dataset which will be described in detail in Section 2.
In developing this project, we avoided to do the experiment the authors have done about burst detection since it used to detect a burst regarding a word of the tweet or a hashtag knowing its historical average frequency and comparing it with the current frequency.
Not having a real-time stream from Twitter is a difficult problem to overcome, in order to carry on this experiment.
After having prepared the training data through some preprocessing techniques and having selected the types of features to extract from tweets (Section 3), we devised a classifier based on centroids in order to look for topics (Section 6), a Support Vector Machine (SVM) to classify if a tweet was relevant or not and to distinguish if the tweet was about a flood or an earthquake (Section 7). These experiments has been done also with the Naïve Bayes classifier in order to compare their performances upon this dataset (Section 8).
Then we produced the multi-dimensional representation of the tweets (Section 4) to train the centroid classifier and re-mapped not relevant/damage/no damage in the dataset to train the SVM and the Naïve Bayes classifier.
At this point, we made some simplifications with respect to the assigned paper in particular due to the different language of the dataset we used for the project which is in Italian rather than English.
Lastly, we were not interested in the development of a classifier that get in real time tweets from Twitter and aggregate them to the cluster more similar since this was unpractical and far beyond the project's scope.

3 DATASET

Availability of data - in particular human-annotated one - is one of the major problems that may occur in the development of a learning system.
Since the authors of the paper did not share any dataset because they created it from scratch retrieving tweets in real time for one year, we decided to use a dataset that contain tweets regarding earthquake and floods happened in Italy.
The dataset is composed of 5,642 manually annotated tweets in the Italian language.
The tweets are related to 4 different natural disasters occurred in Italy between 2009 and 2014.

For each tweet is reported:

1. tweet ID;
2. text;
3. source;
4. author's screen name;
5. author's ID;
6. latitude and longitude (if available);
7. time;
8. disaster ID (see below);
9. class.

Tweets have been manually annotated by humans and divided among 3 classes according to the information they convey:

damage class: tweets related to the disaster and carrying information about damages to the infrastructures or on the population;

no damage class: tweets related to the disaster but not carrying relevant information for the assessment of damages;

not relevant class: tweets collected while building the dataset, but not related to any disaster (noise).

We made some modifications to the dataset in order to adapt it to our needs, in particular for the second experiment we have done (SVM) we collapsed the 3 classes (not relevant/damage/no damage) into 2 classes (relevant or not), considering a binary classification problem.

This dataset carries with it the difficulty to deal with Italian tweet, for this reason we made some simplification during the preprocessing.

4 PREPROCESSING

In this dataset, tweets were roughly clean but they contained punctuation, numbers, symbols that are not necessary for classification purposes since they can lead to biased results.
For this reason, we applied some standard pre-processing methods like punctuation and stop words elimination, symbols and digits elimination.
As a consequence, in order to reduce the size of the vocabulary and the dimension of the related space, we applied some pre-processing techniques using the nltk library.
For the stemming, that is a heuristic which simply chops off the end of a word, we applied the Snowball stemmer which has been adapted to deal with Italian language.

Unfortunately, it was not possible to do the same for lemmatization, since it is not available an algorithm or implementation for doing lemmatization to Italian words.
The consequence to this problem is that the size of the dictionary can increase a lot because Italian language shows a wide variety of terms to express the same meaning, verbs are a great example of this problem.

5 TWEET TO VECTOR

After the preprocessing, we needed to represent the tweets in an appropriate way to work with the following algorithms.
The idea is to represent a single tweet as a vector in a multi-dimensional vector space, where each axis is a term of our vocabulary, furthermore we weighted each vector component with the TF-IDF weight function to reach the best results.
We reached our purpose using the CountVectorizer module from SKLearn library. The module builds the vocabulary, scanning the preprocessed tweets, and for each term it counts the relative frequency.
Then it transforms each tweet in a vector that lives in the multi-dimensional space built according to the vocabulary.
At that point we were ready to implement the following algorithms.

6 CLUSTERING AND CLASSIFICATION

In the paper, the authors used an algorithm based on prototype/centroids to carry on experiment on online clustering in order to find relevant topics.
More precisely, as soon as the first tweet (now a vector) arrived, the system created a new centroid with it and the related cluster, then whenever a new vector arrived from the data stream it was compared, in the sense of cosine similarity, with all centroid present in that space. If the similarity is above a certain threshold, that vector is attached to the related centroid and then the centroid/prototype is recalculated else it is created a new centroid with that tweet.
Another experiment they executed was about classification on whether or not a tweet was about damage to infrastructures.
As a consequence, we realized the same experiment using a very different dataset, and

in particular we re-made the online clustering experiment and the classification using two different supervised machine learning algorithms. Now we explain the differences between clustering and classification:

- **classification** -> you know in advance the classes in which to assign the elements and you have at your disposal a set of examples for each class.

- **clustering** -> it is assumed that there is a natural subdivision of the elements into categories but it is not known in advance how many and which are the categories. In this hypothesis there are of course no explicit examples for each category.

7 ONLINE CLUSTERING EXPERIMENT

In order to reproduce the experiment on online/hierarchical clustering, we cannot be able to use a real-time Twitter stream therefore we trained and tested our algorithm on the dataset we talked about.
To implement the algorithm, we could have used some "precompiled" library like SciPy and SKLearn that make available all the necessary function to calculate the centroid and the cosine similarity, instead we have decided to implement the entire algorithm from scratch using only the numerical manipulation library NumPy.
At the beginning we consider our first vector as the centroid of the first cluster, then we compared each vector with every centroid already present in the vector space, we calculated the cosine similarity (defined as), and if the value was less than the threshold, a new centroid was created, otherwise, since we keep track of the maximum cosine similarity value and with which centroid it obtained that value, we could append that tweet to the relative cluster, then we recalculated the centroid vector for the cluster with the new vector included.
The main data structure of this part was aimed to keep track of all the centroid and, for each one, of all the vector/tweet that belonged to it.

8 SVM EXPERIMENT

In order to reproduce another experiment that has been done from the authors of the paper, we focused on whether the tweet was relevant or not from the point of view of the damage to infrastructures, and being a binary class problem we thought to use an SVM (Support Vector Machines) classifier to separate with a hyperplane the two classes. Before to use the classifier, we built a new dataset made only of tweet-id, tweet-text and relevant/not relevant.
Since in the original dataset there were present classed about relevant/not relevant and damage/not damage we collapsed the class damage/not damage into relevant class which was represented with number 0, whereas the class not relevant was represented with 1. Then, we trained the SVM classifier using the 0.70 of the dataset and the other part used for testing. The same experiment has been done considering, instead of relevant or not, the two classes of earthquake and flood. The results of the experiment are described in detail in the next chapters.

9 NAÏVE BAYES EXPERIMENT

In this experiment, we wanted to use another supervised machine learning algorithm to check if it was able to reach performance significantly different from the ones showed from SVM. The main difference between the two methods is that SVM try to calculate the hyperplane that divide in the best way the vectors of the two classes and at the same time trying to keep the error the lowest possible and to be more general possible (since it has high bias).
The Naïve Bayes, instead, use probability, in this case, the number of times a word occurs in a tweet and in all of them and how this information is related to the meaning it carry on.
The experimental results and how it behaved against the SVM classifier are shown in the next chapters.

10 RESULTS

10.1 SVM

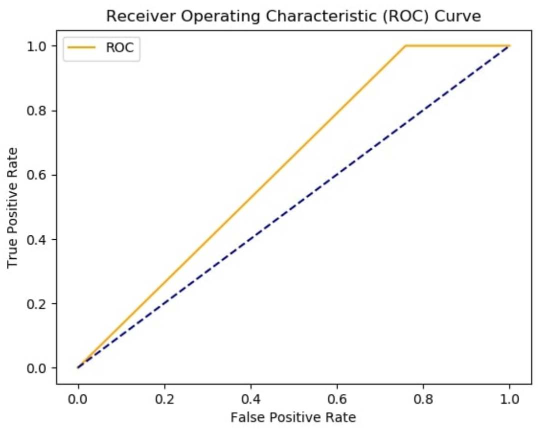
In order to do these experiments, we put into the SVM classifier the tweet represented as a vector and the class it belonged, which was in one case flood or earthquake and in the second relevant or not relevant.
Then we decided to use the 0.70 of the dataset as a training set and the remaining part as a test set. As a kernel, we used a linear one and for gamma we used different values in order to allow some misclassification if the two class were not perfectly linearly separable.
The gamma parameter can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.
Since we are not able to represent the vectors because they live in a high dimensional space (about 1600 axes), we decided to represent the results we obtained and the relative performance through ROC curve and AUC.

The ROC curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied, and the AUC is the Area Under Curve.
The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. An AUC of 1 means a perfect classifier, instead a value of 0 means that for every element it decides for the opposite class.

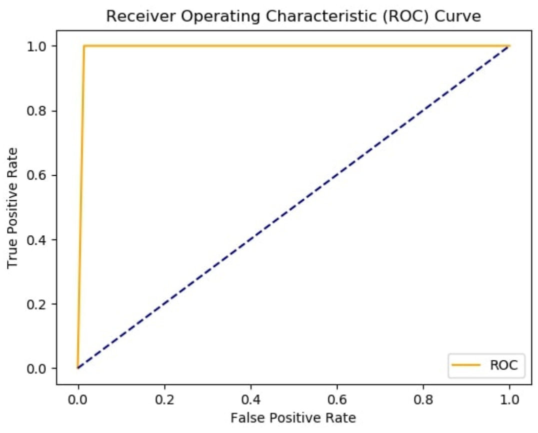
The confusion matrix is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one. Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class (or vice versa).
Next it is possible to view the graphical results.

These graphics represent the behavior of this classifier in the experiment flood/earthquake for different values of gamma, ranging from 0.05 to 1. We don't have investigated further because from 1 onwards the ROC curve remain the same.

H

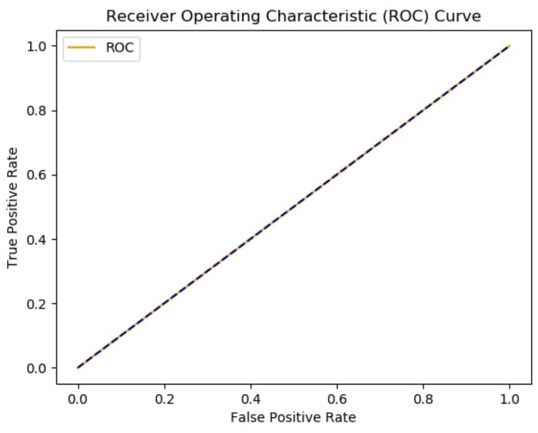


(a) *case of gamma ->0.09*



(b) *case of gamma ->1*

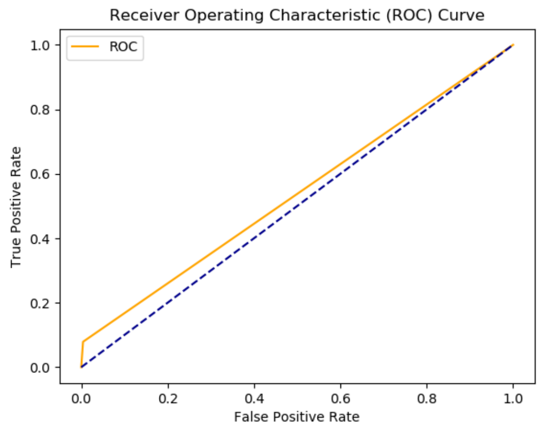
Figure 2



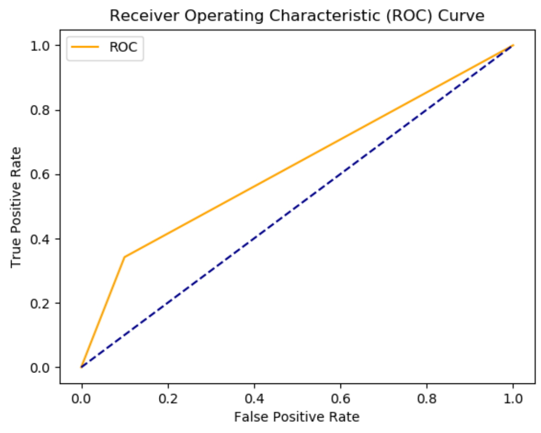
(a) *case of gamma ->0.05*

Accurancy	Precision	Recall	AUC	gamma
0.92	0.91	1	0.5	0.05
0.94	0.94	1	0.62	0.09
0.99	0.99	1	0.99	1

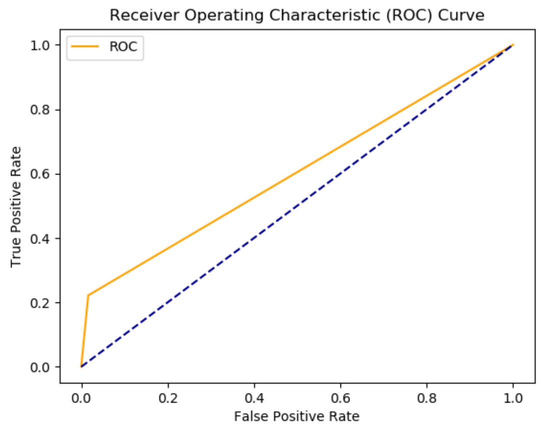
These three graphics represent the behavior of the SVM classifier in the experiment relevant/not relevant varying the value of gamma from 0.05, 0.09 and 1.



(a) *case of gamma ->0.05*



(b) *case of gamma ->500*



(c) *case of gamma ->1*

Figure 3

Accurancy	Precision	Recall	AUC	gamma
0.85	0.78	0.08	0.54	0.5
0.81	0.39	0.34	0.62	500
0.85	0.73	0.22	0.6	1

10.2 NAÏVE BAYES

In this experiment, we put into the Naïve Bayes classifier the tweet represented as a vector and the class it belonged, which was in one case flood or earthquake and in the second relevant or not relevant. We have used the same graphical approach used in SVM experiments to plot its performance.

Accurancy	Precision	Recall	F1-score	AUC
0.93	0.93	1	0.97	0.57
0.85	1	0.04	0.08	0.52

The first row is about the experiment of Kind of disaster instead the second row is about the experiment of Relevant(damage-not damage) and Not Relevant

10.3 CLUSTERING

In this experiment, after representing our tweets as vectors, we give them to the agglomerative clustering algorithm. We trained the algorithm more times, each time with different values of treshold. The treshold is the decision value thanks to the algorithm determinates if the vector is enough close to the centroid to put the vector in the cluster or less. The table below shows the number of clusters we obtained regarding the treshold value we set.

Try	Threshold	Clusters
0	0.005	5
1	0.01	10
2	0.1	198
3	0.5	3568
4	0.95	5171

11 IMPROVEMENTS

In this section, we will briefly describe some possible improvements to our project. Through a new approach to this project, it could be interesting to do the same experiment using other social networks such as Facebook and Instagram because they are continuously changing and improving the information they offer to population. Therefore, it could be more useful other social network depending on the situation we are dealing with, for instance in case of terrorist and criminal attacks.

12 CODE OF PROJECT

To see the project you can follow the link:

$\rightarrow CLICKME$

13 REFERENCES

1. Jie Yin, Andrew Lampert, Mark Cameron, Bella Robinson, and Robert Power, Using Social Media to Enhance Emergency Situation Awareness, 2012, IEEEE, 1541-1672
2. CRESCI-SWDM15
3. NumPy library
4. Sklearn library
5. Natural Language ToolKit library
6. Matplotlib Library