Report on Twitter topic detection and analysis

Alex Tsilingiris
Christina Pardalidou
Sotiris Karapostolakis

June 30, 2017

Contents

1 Introduction 2

2 Model description 3

3 Method implementation details 4

4 Sample results 5

5 Framework description 7

1 Introduction

To keimeno sth sinexeia einai mia texniki anafora tou tade project to opoio ekane auta k auta me vash auta. stoxos itan auto k auto kai gia tin diekpaireosh tou egine auto k auto.

2 Model description

Section

No idea. isws : List me TF sta hashtags? Preprocessing steps?

3 Method implementation details

Event detection

Our approach

A combination of approaches was used for event detection: Since the dataset was static and filtered based on a query (in our case the word ”trump”), terms were considered as living organisms in the given timespan, allowing us to sample a set of tweets as when each term was ”most alive”, while combining user reputation metrics to select a tweet from a reputable source.

Hashtags as terms

We solely operated on hashtags on this step since they represent an idea or a topic, which in other cases would be difficult, or not as accurate, to define using Natural Language Processing and Machine Learning.

The living organism implementation

We considered a term as most-alive at the point in which it was mostly detected in our dataset. The time window was set to 2 hours. Therefore, we had to query for the mostly found terms (using the Term Frequency statistic) and sort the results in descending order. We then generated the final that contained the original tweets with the most active terms in the dataset that were posted in the given timespan.

Selecting a tweet from the pool

We now have a pool of tweets that might contain information about an event. The approach we followed was to assume that a user with a high amount of followers represents an influential event source into a social community[1]. We sorted (descending) the tweet pool by the number of followers the original poster has and ended up with 1 tweet that was our result for each term.

Sentiment Analysis

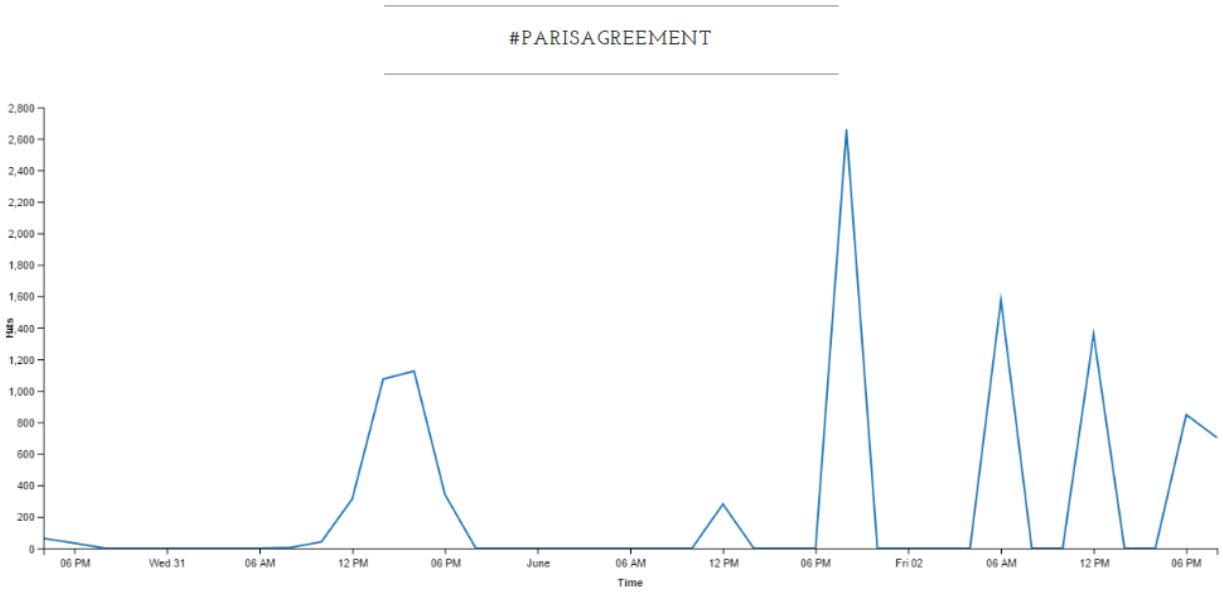
Event detection

Some top ranked terms (hashtags), their occurrences and the tweet that qualified as an event by our system will be presented in table 4.1

Term	Frequency
#parisagreement	10442
#covfefe	7085
#trumprussia	4229
#marchfortruth	973

Table 4.1: Terms and occurrences

A graph showing the top ranked term's occurrence over time can be seen in the following figure:



The tweet that our system determined as an event can be seen in the following figure:

Additional results can be found in the project's website [8].

5 Framework description

The software created was split into modules to allow for independent, on-demand functionality. A user can easily assemble a system from these independent modules based on their needs. The complete repository can be found in the project's Github page [4].

Getting the tweets

To begin with, Python and Tweepy [7] were used to access the Twitter API as efficiently as possible. The incoming stream was filter using the keyword "trump" and was directly saved into a MongoDb instance.

Preprocessing the dataset

A custom preprocessing module was created to combine all the required steps needed when analysing tweets. The nltk [5] library along with some custom regular expressions were used in order to complete the following actions: group hashtag symbols with the hashtag word, group mention symbols with the mention target, keep URLs functional and finally remove stop-words, punctuation and twitter specifics such as "via" and "rt". Finally, a script was created to convert datetime strings into MongoDb specific datetime strings so that range queries were possible.

Plotting

The pandas [6] library was used to resample the time and allow tweet bucketing based on a given time window. The window size used for term (hashtag) visualisation was 120minutes. The results were exported directly into .json format and are presented in our website [8] using the d3js [2] library.

Getting the tweets for sentiment analysis

After detecting the top event that occurred during the time the tweets were crawled, the dataset was filtered in order to get only the tweets containing hashtags related to the emerging event (e.g #parisagreement, #climatechange). These specific tweets were written to text files so they can be imported to the Opinion-Mining tool[3] for prediction.

Opinion-Mining tool

The tool uses a custom lexicon implementation that is shown in figure 5.1. The training set used includes a total of 25k labeled reviews taken from the published dataset of imdb. Half of them are positive and the other half are negative.

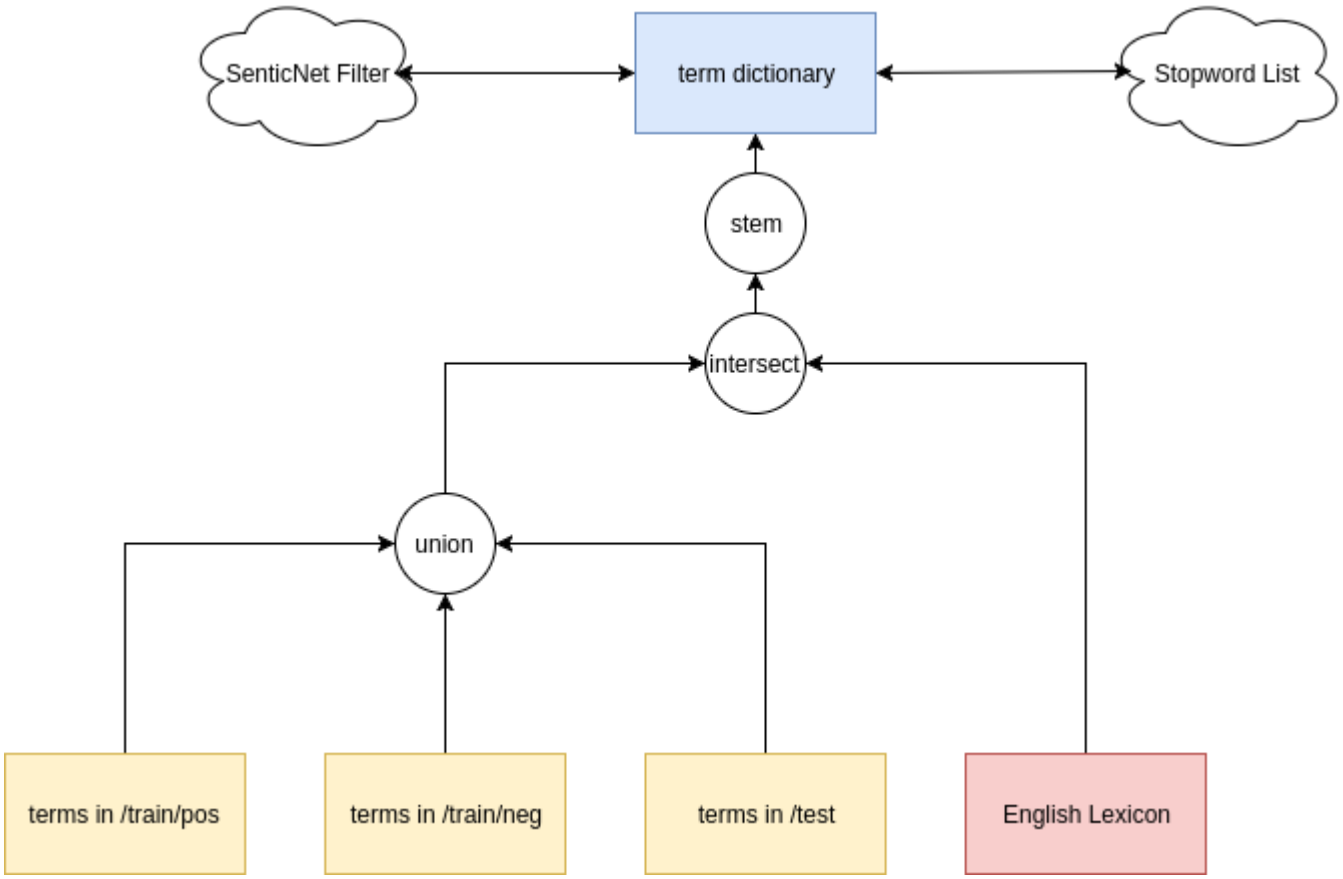


Figure 5.1: Depiction of the diagram of the lexicon.

Preprocessing process

- Punctuation removal, tokenization.
- Stemming (non-English terms are deleted).
- Terms frequencies.
- TF/IDF Vectors.

Prediction

The model uses a SVM classifier with SGD tuning for prediction. The vectors are defined with TF: log normalized and IDF: smooth normalized. The output is a text file that has in each row the name of the input text file and the output of the prediction for this file i.e 1.0 for positive and 0.0 for negative tweets.

Bibliography

[1] M. Cataldi, L. D. Caro, and C. Schifanella. Personalized emerging topic detection based on a term aging model. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(1):7, 2013.

[2] d3js. d3js. https://d3js.org/.

[3] Github. Opinion mining tool. https://github.com/lefterisKl/Opinion-Mining/.

[4] Github. Project repository. https://github.com/alextsil/twitter-topic-detection-and-analysis/.

[5] nltk. nltk. http://www.nltk.org/.

[6] pandas. pandas. http://pandas.pydata.org/.

[7] Tweepy. Tweepy. http://www.tweepy.org/.

[8] Website. Project website. http://snf-755277.vm.okeanos.grnet.gr/.