Topic Detection Using a Critical Term Graph on News-Related Tweets

Paraskevas Tsantarliotis
Department of Computer Science & Engineering
University of Ioannina
Ioannina, Greece
ptsantar@cs.uoi.gr

Evaggelia Pitoura
Department of Computer Science & Engineering
University of Ioannina
Ioannina, Greece
pitoura@cs.uoi.gr

ABSTRACT

Social media and online social networks are playing an increasingly important role in our lives, as they attract millions of users around the world. Twitter, one of the most popular micro-blogging services, holds a special position among them, since information shared through this service spreads faster than it would have been possible with traditional sources. There are many interesting works analyzing the information that flows through Twitter. Most of such research focuses on trending topic detection, i.e. what are the people talking about right now. We propose a new method to detect topics using a graph, where nodes correspond to terms and edges correspond to co-occurrence of the two terms in the tweet stream. Dense subgraphs, of this graph, pose special interest, as the nodes that are highly connected share a special relation. Thus, the corresponding terms potentially share a relation too. To explore this fact, we apply a community detection algorithm on the graph. The resulting communities correspond to topics related to various real world events. Experimental evaluation of the results of this technique is also provided on both synthetic and real data.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Clustering; H.4.m [**Information Systems Applications**]: Miscellaneous

General Terms

Algorithms

Keywords

topic detection, community detection, term mining

1. INTRODUCTION

In recent years, usage of social media has overcome any expectation. Millions of users from all over the world post content on online social networks, forums or their blogs or subscribe to micro-blogging services. As a result, social media, especially online social networks, have been transformed to a powerful mean of disseminating news (e.g. describing real-world events).

In particular, Twitter [15] is both an online social network and a micro-blogging service, which enables users to send and read short 140-character text messages, called "tweets". The context of tweets varies from chit-chat to political sentiment, creating a very interesting stream of information. Thus, Twitter can be described as an information/data network. Such a network contains important data and poses exploration opportunities and challenges, such as discovering and browsing valuable information.

Most of the efforts focus on trending topic detection. There are many reasons why researchers focus on this particular problem. First of all, it describes what the people are talking about right now. Furthermore, this can be a powerful tool for marketing specialists and opinion tracking companies, since trending topics can describe the opinion and intentions of a large group of people. There are a lot of services and sites dedicated to finding trending topics, such as Trends24[1], TrendsMap[2] and WhatTheTrend[3]. Usually, these services use - one or two - frequent terms or frequent hashtags ("#") to describe a topic. However, using more than a couple of terms to describe a topic would be more informative, e.g. instead of "Edward, Snowden", we would prefer something like: "Edward, Snowden, NSA, surveillance, illegal", which is much more expressive.

In this paper, we propose a new intuitive way to detect topics on data streams where a topic is described by a number of terms. We construct a graph whose nodes correspond to terms appearing in tweets. Two nodes are connected if only they co-occur in the same tweet and the weight of the edge corresponds to the co-occurrence frequency. We call this graph *critical term graph*. Constructing such a graph can be very expensive for large documents, but is suitable for short document, such as tweets. We will discuss how this graph is constructed and its properties in detail later in this paper. Based on this graph, we extract topics from its dense subgraphs. We use a community detection algorithm, which partitions the graph into sets of nodes that are tightly connected with each other and sparsely connected with nodes that belong to different communities. Thus, in this case the communities represent the topics. The output of the algo-

[1] http://trends24.in
[2] http://trendsmap.com
[3] http://whatthetrend.com

rithm is displayed using the visualization tool Gephi[1]. We also provide an evaluation of our system and its variants both on real and synthetic datasets.

In this paper we focus on detecting topics in general and not trending topics. Nevertheless, the technique we propose could possibly work for real time trending topic detection. The main contribution of the paper is the study of the critical term graph and the feasibility of applying community detection algorithms on this graph to identify topics. To this end, we design a new model for generating synthetic tweets by controlling the number of topics and the overlap between them. We also test our algorithms on real news-related tweets, empirically evaluating our model.

In Section 2 we describe works related to ours and in Section 3 we describe in detail the problem and the proposed solution. Evaluation of our results are presented in Section 4. Finally, in Sections 5, we discuss future work and summarize our conclusions.

2. RELATED WORK

Work related to ours includes research both in the broad areas of mining data streams and graph mining, specifically in the areas of topic detection in micro-blogging services and community detection.

There are many papers that focus on identifying trending topics on Twitter like [5, 7, 17, 19]. The authors of [5] describe some interesting methodologies of detecting and identifying trending topics, but they limit their results to unigrams and bigrams. The authors of [17] present TwitterMonitor, which detects bursty keywords and groups them to form clusters. In that short paper, it is made clear that a single pass over the data stream is not enough to detect bursty keywords, but no algorithmic nor experimental details are given. Another interesting approach is described in [7], they focus on detecting trending topics based on metrics that involve both the frequency of the terms and the authority of the users, who wrote the tweets. However, their method requires high computational load. Furthermore, information about the users may not be available in large datasets, because of the restrictions of the Twitter API.

In [19], the authors propose a complete system for detecting trending topics from Twitter posts in near-real time. Their algorithm is similar to the Apriori approach [4] and relies on finding frequent multi-word clusters, that represent a topic, and then calculating the burstiness of each topic. Their approach is based on a hypothesis, similar to ours, that if AB, BC and AC are frequent terms pairs then ABC is frequent. This is not necessarily true, but as we see in our results, it is very likely. Consequently, this hypothesis introduces false positive results. Furthermore, the algorithm used in [19] for topic detection is also interesting, because their work could possibly be extended to be used on the graph-based model we propose.

Graph mining is also related to our work, since we use such techniques to extract useful information. Currently, we focus on community detection algorithms, but other algorithms may be considered in the future. Community detection in graphs is thoroughly investigated in [10]. In general, community detection is used to uncover relation between nodes in complex networks in biology, computer science, sociology, etc. For example, in [9], they use similar technique in the Web to discover communities of Web pages dealing with the same topic. The authors of [8] propose an algorithm for dense subgraph extraction and they test it on a graph that represents the relationships between terms, obtained by a news agency. Their technique manages to group in the same subgraphs, terms that tend to be used together. However, no further analysis is provided. In [13] authors use a similar approach to ours, in order to detect topics related to a particular event, but their main focus is on the evolution of these topics through time. Instead, we use the model to detect topics under real conditions and provide more information about the model.

3. MODEL

In this section, we describe the basic concepts of our approach and the algorithm which we use to detect topics. For the description of the model, we assume one segment of the tweet stream. We discuss later how the method is extended to be applied to to a sequence of segments.

Topic. Similar to other works [17, 19] we define a topic to be a set of terms. Usually, a topic consists of three or more terms, in order to capture its essence. Terms in each set must be frequent, i.e. surpass some threshold, and co-occur with some other words in the topic. Thus, we are interested in looking for frequent term pairs and our goal is to merge these pair to form lager sets, which represent the topics.

Critical Pairs. Similar to [19], in order to detect frequent pairs of terms, we use a support measure:

$$sup(\tau_i, \tau_j) = \frac{|D_{ij}|}{|D|},$$

where τ_i, τ_j are terms, D_{ij} is the set of tweets that the two terms co-occur and D is the total set of tweets. A term pair is frequent if its support is equal or larger than a support threshold. In [19], the authors define a static value for the support. There are some drawbacks when a fixed support is used. As the input size varies, the output of the algorithms may elide some topics, or include too many topics. Instead of a static support threshold, we use a method defining a dynamic support threshold, i.e. the threshold adapts to the size and the content of the input. To achieve this, we sort the co-occurrence frequency of all possible two-word pairs in descending order. The N first pairs are called critical pairs and we use the support of the N-th pair as threshold.

Critical Term Graph. Here, we describe the basic concept of our work. Using the critical pairs, we create a graph, where nodes represent terms that are used in the tweets and edges indicate co-occurrence of the two terms in them. We assign a value to each node, which equals to the term frequency in the tweet corpus. Similarly, we assign to each edge

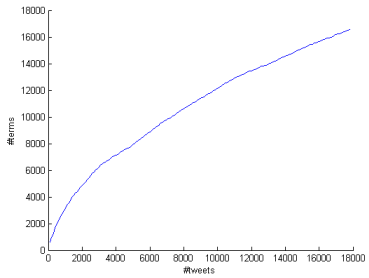

Figure 1: Number of unique terms in tweets.

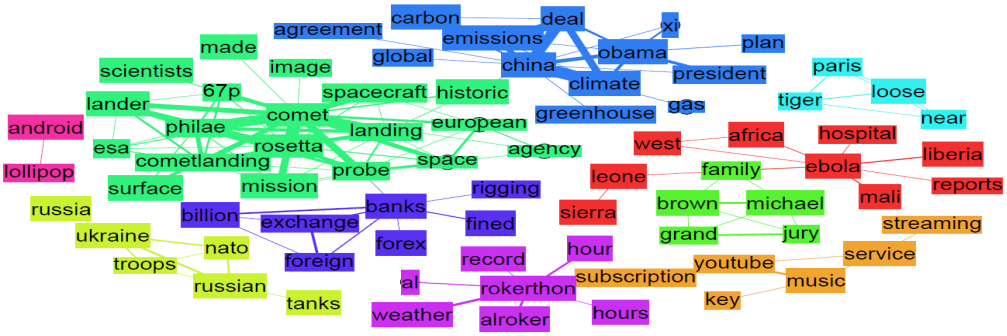

Figure 2: Results from November 12 2014.

a weight that corresponds to the co-occurrence frequency of its incident terms in the tweets. A normalization of these values, may also be considered.

An important property of this graph is that as the number of tweets increases, the number of newly seen terms decreases and at some point we expect to have more tweets than unique terms, i.e. the tweet corpus obeys to the Heap's Law. In Figure 1, we compare the number of unique terms to the number of tweets in a random stream of all public tweets, acquired by Twitter API[4]. At first, the number of unique terms is larger than the number of tweets, but in the end there are more tweets than unique terms. We expect this diagram to flat out much faster when tweets focus on specific topics, because the vocabulary should be limited. Thus, as the number of tweets increases, it is more likely new edges to be created or the existing edges to become stronger.

Generally, we expect the critical graph to be sparse, not well connected and have community structure (Figure 2). The communities are formed from terms that co-occur in the tweets, thus creating topics. The graph contains a lot of non-connected components. Usually, the small non-connected components (of size < 10) correspond to standalone topics that have no common terms with the other topics. The number of larger non-connected components, on the other hand, is much smaller and these components contain more topics.

3.1 Community Detection Algorithm

Preprocessing. When a new tweet arrives at the system, it gets preprocessed and then stored. Preprocessing involves splitting the tweet in a term list, removing stopwords, stemming the terms and generating all possible term pairs from the term list. We keep a record for the co-occurrence frequency of each term pair, in every time segment.

To detect communities we use a modularity based algorithm [6]. Several algorithms have been proposed to detect communities efficiently. Newman and Girvan [18] introduced modularity as a measure for the quality of the resulting communities. Modularity measures the density of edges inside a community compared to the expected density if the edges were distributed at random. Modularity ranges between -1 and 1. If the modularity of a subgraph tends to 1, it means that the sub-graph is tightly connected and may be a community and vice versa. In weighted networks modularity is defined in [6] as:

$$\frac{1}{2m}\sum_{i,j}[A_{ij} - \frac{k_i k_j}{2m}]\delta(c_i, c_j),$$

where A_{ij} represents the weight of the edge between i and j, $k_i = \Sigma_j A_{ij}$ is the sum of the weights of the edges attached to vertex i, c_i is the community to which node i is assigned, the δ-function $\delta(u,v)$ is 1 if $u = v$ and 0 otherwise and $m = \frac{1}{2}\ \Sigma_{ij} A_{ij}$.

It is easy to see that the community detection problem is equivalent to modularity optimization in a graph. In [6], the authors propose an algorithm consisting of two phases that are repeated iteratively.

Initially, every node is a community. Let i be a node of the graph and j its neighbors. The algorithm evaluates the gain of modularity that would be achieved if node i was removed from its community and placed in the community of j. If the maximum gain is positive, then node i is placed in the community for which the gain is maximum, otherwise the node i stays in its community. This step is repeated for every node in the graph until there is no improvement of modularity. Note that each node can be considered several times, during this phase.

In the second phase, a new graph is built with nodes the communities found in the first phase and edges that have weights equal to the sum of weight of the edges between nodes in the corresponding two communities (edges between nodes of the same community are marked as self-loops). Phase one is applied to the new graph. The two phases are repeated until there are no changes and modularity is maximized.We chose this algorithm because it is fast; its complexity is linear on typical and sparse data [6]. In addition, the algorithm follows an unsupervised approach, so we need no previous knowledge about the number of topics we are looking for.

It is known that modularity optimization algorithms suffer a resolution limit [11], i.e. they fail to resolve communities smaller than a certain scale, even if they are well defined. However, because this algorithm, in intermediate steps, merges communities in order to create new communities, there is a sense of hierarchy. Thus, we are able to use this hierarchical structure to detect smaller communities.

3.2 Snapshots

So far, we have considered topics in a single sequence of tweets. To handle streams of tweets, we divide the stream in small segments of fixed time length. Each segment is marked

[4] `https://dev.twitter.com/streaming/`

with a time id, based on its start and end timestamps. For each segment we store all the possible term pairs and their frequencies, that we have extracted from tweets.Given a time range, to detect topics within this time range, we first identify the appropriate segments that correspond to this specific segment. Then, we group all the term pairs from the appropriate segments, get the critical pairs and finally create the critical term graph. This graph is a snapshot of the graph for the specific time range. Snapshots are useful for tracking topics over time, which is considered as future work (see Section 5).

4. EXPERIMENTAL RESULTS

In this section, we demonstrate the initial results of our approach both on synthetic and real data.

4.1 Synthetic Data

One of the difficulties we faced is the lack of datasets with known ground truth. In order to test our approach under various conditions, we created a model to generate synthetic datasets similar to the one described in [14]. Since real tweets obey to the Heaps' law, the synthetic tweets should be generated by randomly sampling terms based on a Zipf distribution. We present a short description and preliminary results of the model we used to generate synthetic tweets.

We assume that we have k topics. Our goal is to generate tweets for each of the k topics. Initially, we generate the terms, which will be used in the tweets. Similar to [14], we consider *k+1* topic vocabulary bags, i.e. bag B_i=1,...,k contains the terms for the i-th topic and bag B_{k+1} contains general terms. General terms can be used in all topics, they are like stopwords but contain more valuable information for the topic.

Then, we create a $k \times$ *(k+1)* matrix P. Each cell P_{ij} of the matrix corresponds to the probability to pick a term from vocabulary bag B_j while generating tweets for the i-th topic. We call overlap between topics, the case in which when we generate a tweet for a topic i, we include a term from a bag B_j that is different from the bag corresponding to the topic, i.e., $B_i \neq B_j$ Note that vocabulary bag B_{k+1} should have similar probability in every topic. We describe the process that generates tweets for each topic below.

Let t and M be the topic that the tweets refer to and the number of tweets we want to generate for this topic, respectively. At first, we assign occurrence frequency to all terms based on Pareto distribution. We must point out that general terms should have the same frequency in all topics, in order to avoid special relation with any topic. Then, we generate M tweets for the topic. Note that we do not want duplicate tweets. Tweets are considered as a set of terms that make up the tweet. We noticed in our dataset that the number of terms used in the tweets (not including stopwords) follows the Poisson distribution. Thus, we first decide the size of the tweet S and then pick S terms. Term picking can be summarized in two steps:

1. Select a vocabulary bag B_i based on the probability of matrix P_k.

2. Select term from B_i based on occurrence frequency, i.e. terms with high occurrence frequency are more likely to be picked. Note that we don't want duplicate terms to avoid spam.

We also need to check if the tweet is unique. If so, we decrease the frequency of each term used in the tweet by 1. Otherwise, we create a new tweet. This process is repeated until we create tweets for all k topics. Therefore, the output the of the model is $k{\cdot}M$ synthetic tweets.

In order to evaluate the results of our model, we consider community detection as a problem of assigning all similar nodes to the same communities [3]. Thus, based on pair counting, we can predict the following cases:

- True Positive (TP): Term pairs that belong to the same topic are assigned to the same communities.
- True Negative (TN): Term pairs that belong to different topics are assigned to different communities.
- False Negative (FN): Term pairs that belong to the same topic are assigned to different communities.
- False Positive (FP): Term pairs that belong to different topics are assigned to the same community.

Using the above, we can calculate the following evaluation measures, which are defined in [3, 12]:

- Precision: $P = \dfrac{TP}{TP + FP}$
- Recall: $R = \dfrac{TP}{TP + FN}$
- Jaccard Coefficient: $J = \dfrac{TP}{TP + FP + FN}$
- Rand Statistic: $RS = \dfrac{TP + TN}{TP + TN + FP + FN}$

Figures 3 and 4 show the results of our approach. In Figure 3, we present three different plots comparing the number of topics that our model estimated to the real number of topics. We test our model using different amounts of critical pairs, in three different percentages of overlap between topics, 5%, 15% and 25%. The overlap between topics is distributed randomly. The synthetic datasets are generated using 3% general terms in each topic and they contain 3 to 10 frequent topics and 2 to 10 less frequent topics, for each experiment. Note that the experiments have been repeated multiple times and we present average values.

We notice that our model in most cases predicts the correct number of topics. The maximum discord, between the estimated and the real number of topics occurs when using 1000 critical pairs to detect topics in larger datasets. In these cases, the model predicts less topics than the actual. But as we can see in Figure 4 the detected topics are the frequent ones, because all the evaluation metrics are close to 1. We must also point out that using more than 1000 critical pairs in the datasets with 5 to 10 topics does not perform well. Even though the model predicts almost the correct number of topics, the critical pairs contain noise and this leads to estimating false topics. As we can see in Figure 4 as the noise increases the corresponding metrics are decreasing.

The number N of critical pairs affects the number of topics detected. A default value of 2000 seems to work in most cases. By reducing N, we may end up loosing some topics, however, the ones detected are the most important (i.e., the most frequent) ones. By increasing N, we may get false results when the number of actual topics is small. This is one

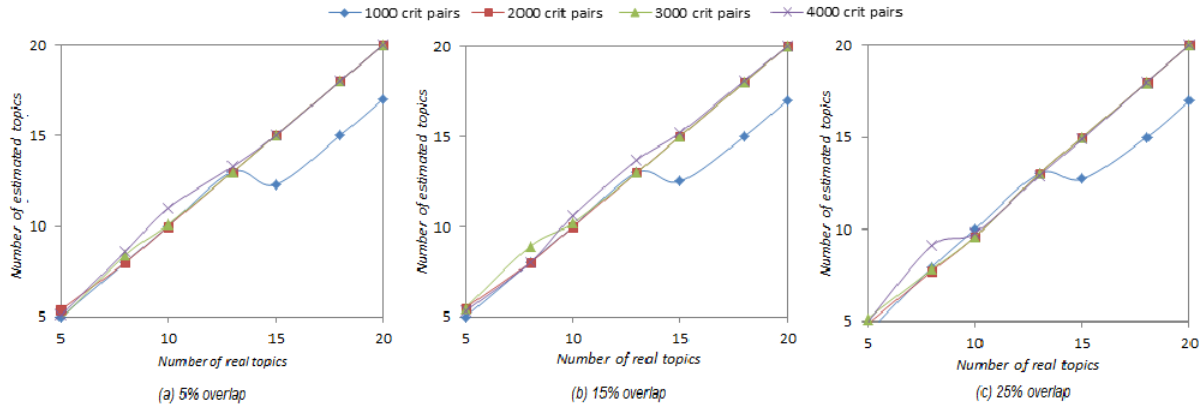

Figure 3: Comparison of the number of estimated topics to the number of real topics, using different amounts of critical pairs and increasingly overlapping topics ((a) 5%, (b) 15% and (c) 25%).

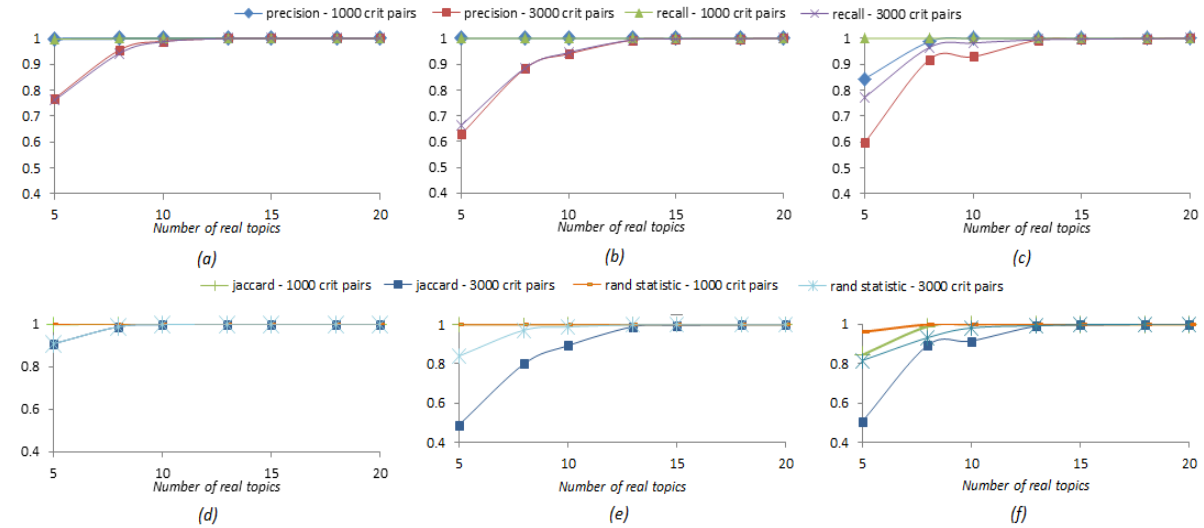

Figure 4: Evaluation metrics on synthetic datasets using 1000 and 3000 critical pairs and increasingly overlapping topics ((a) & (d) 5%, (b) & (e) 15% and (c) & (f) 25%).

of the issues we plan to address in future work. An initial idea is to dynamically adjust N, by looking at the actual frequency of the N-th pair in adjacent segments.

4.2 Real Data

We provide a set of experimental results based on real data and discuss their quality. The datasets used in these experiments are closely related to news, sports and lifestyle. This can help us to get an empirical evaluation of the results. We have implemented a crawler, which follows public accounts of popular news agencies, magazines, politicians, celebrities, etc and people that are related to them. For example, some of the accounts are Barak Obama, The Guardian, various journalists, N.A.S.A. and F.I.F.A.com. Most of these accounts are verified by Twitter, to avoid spam, and their tweets are written in English.

Thus, the context of the dataset varies from political to sport related events. We have to point out that the tweets are related to real world events, the popularity of these events directly affects our results. An example it is shown in Figure 2. We got 10 topics that happened at November 12 of 2014. The most obvious of the them refers to first ever successful land of a spacecraft (Philae) on a comet (Comet 67P). We can also see information about the U.S.-China emissions deal, the Forex scandal, the Ebola outbreak and a rather bizarre news about a loose tiger near Paris.

Similarly, Figure 5 displays topic from (Monday) December 1 2014. We see topics that refer to World AIDS Day, new Star Wars movie trailer, cyber Monday and black Friday sales. Another interesting example in the same figure is that the algorithm manages to detect different topics for two different protests in two different places, Hong Kong and Ferguson U.S. respectively, even though they have some common words. Still, there are some terms that could belong to both topics, but since the algorithm produces non-overlapping communities, this is not possible.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a model to detect topics in

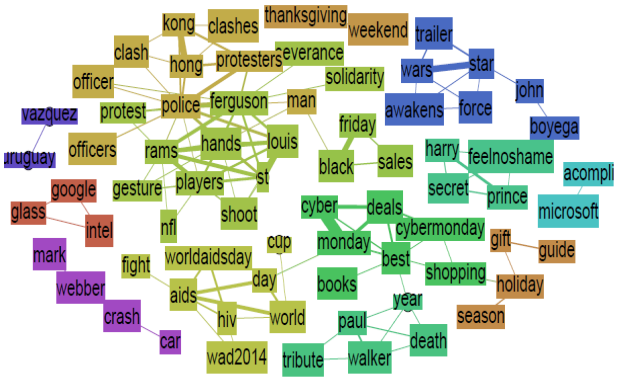

Figure 5: Results from December 1 2014.

short text documents, such as tweets, using a graph-based model, the critical term graph. This graph consists of nodes, which represent the terms found in tweets, and edges, which represent the co-occurrence frequency of the corresponding terms. Detecting communities in such graph can be an efficient tool for topic detection. We empirically proved the effectiveness of our model with experiments on real news-related tweets. In addition, we proposed a method to create synthetic tweets, and tested our model under various conditions.

There are many possible improvements and extensions. At first, we can improve the critical term graph, using Natural Language Processing techniques on the tweets. Synonyms could be considered duplicate information in the critical term graph, thus these terms could be grouped. Furthermore, verbs could also be eliminated, because in most cases they do not add much valuable information. These improvements would reduce the graph size and potentially improve the performance of the model. Both synonyms and verbs can be detected using simple dictionary based methods, such as WordNet[2]. Alternatively, synonyms can be detected using more advanced methods like [16].

Furthermore, an evaluation process of the results based on real data can be considered by using information from external sources, such as Google Trends or news agencies.

We are currently focusing on different ways on assigning weights on the edges and how the algorithm responds to these changes. The quality of results when using other overlapping and non-overlapping community detection (or graph partitioning) algorithms is, also, another interesting topic.

Another extension could be the use of this model in trending topic detection. This can be achieved by using the concept of burstiness. In [19], the authors define the trending topics as topics that exhibit high frequency in a specific time segment compared to its previous segments. In our model, this could be achieved by using additional constraints on selecting the critical pairs, e.g. the occurrence frequency of the corresponding terms should exhibit bursty behavior.

Our ultimate goal is to use the critical term graph for tracking the evolution of the topics through time. We are considering two approaches on this problem. The first approach involves building different critical term graphs for each snapshot and then trying to map communities, and thus the corresponding topics, between these snapshots. Alternatively, we can maintain a single critical term graph and incrementally update the graph by adding, modifying or removing edges and nodes.

6. ACKNOWLEDGMENTS

This work was co-financed by the European Union (ERDF) and Greek national funds through the Operational Program Thessaly-Mainland Greece and Epirus 2007-2013 of the NSRF 2007-2013. We also thank A. Likas for the insightful discussions and suggestions.

7. REFERENCES

[1] Gephi, `https://gephi.org/`.
[2] Wordnet, `http://wordnet.princeton.edu/`.
[3] *Social Media Mining*, chapter 6. Cambridge University Press, 2014.
[4] R. Agrawal, R. Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th int. conf. VLDB*, volume 1215, pages 487–499, 1994.
[5] J. Benhardus and J. Kalita. Streaming trend detection in twitter. *International Journal of Web Based Communities*, 9(1):122–139, 2013.
[6] V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of community hierarchies in large networks. *CoRR*, abs/0803.0476, 2008.
[7] M. Cataldi, L. Di Caro, and C. Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, page 4. ACM, 2010.
[8] J. Chen and Y. Saad. Dense subgraph extraction with application to community detection. *Knowledge and Data Engineering, IEEE Transactions on*, 24(7):1216–1230, 2012.
[9] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee. Self-organization and identification of web communities. *Computer*, 35(3):66–70, 2002.
[10] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.
[11] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.
[12] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Cluster validity methods: Part I. *SIGMOD Record*, 31(2):40–45, 2002.
[13] S. Higuchi, T. Hashimoto, T. Kuboyama, and K. Hirata. Exploring social context from buzz marketing site-community mapping based on tree edit distance. In *PERCOM Workshops, 2013 IEEE International Conference*, pages 187–192. IEEE, 2013.
[14] A. Kalogeratos and A. Likas. Document clustering using synthetic cluster prototypes. *Data Knowl. Eng.*, 70(3):284–306, 2011.
[15] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on WWW*, pages 591–600. ACM, 2010.
[16] Y. Li, B.-J. P. Hsu, C. Zhai, and K. Wang. Mining entity attribute synonyms via compact clustering. In *Proceedings of the 22nd ACM CIKM*, pages 867–872. ACM, 2013.
[17] M. Mathioudakis and N. Koudas. Twittermonitor: Trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD*, pages 1155–1158. ACM, 2010.
[18] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.
[19] N. Pervin, F. Fang, A. Datta, K. Dutta, and D. Vandermeer. Fast, scalable, and context-sensitive detection of trending topics in microblog post streams. *ACM TMIS*, 3(4):19, 2013.