Two-level message clustering for topic detection in Twitter

Georgios Petkos
CERTH-ITI
Thessaloniki, Greece
gpetkos@iti.gr

Symeon Papadopoulos
CERTH-ITI
Thessaloniki, Greece
papadop@iti.gr

Yiannis Kompatsiaris
CERTH-ITI
Thessaloniki, Greece
ikom@iti.gr

Abstract

This paper presents our approach to the topic detection challenge organized by the 2014 SNOW workshop. The applied approach utilizes a document-pivot algorithm for topic detection, i.e. it clusters documents and treats each cluster as a topic. We modify a previous version of a common document-pivot algorithm by considering specific features of tweets that are strong indicators that particular sets of tweets belong to the same cluster. Additionally, we recognize that the granularity of topics is an important factor to consider when performing topic detection and we also take advantage of this when ranking topics.

1 Introduction

This paper presents our approach to the topic detection challenge organized by the 2014 SNOW workshop. Details about the challenge and the motivation behind it can be found in [Pap14]. The task did not only involve topic detection per se, but it also required the development of approaches related to the presentation of topics: topic ranking, relevant image retrieval, title and keyword extraction. We present the solutions we applied to each of these problems. Open source implementations of most of the methods used are already available in a public repository[1] and the rest will be made available soon.

In: S. Papadopoulos, D. Corney, L. Aiello (eds.): Proceedings of the SNOW 2014 Data Challenge, Seoul, Korea, 08-04-2014, published at http://ceur-ws.org

[1]`https://github.com/socialsensor/topic-detection`

The rest of the paper is structured as follows. In Section 2 we provide a brief overview of existing topic detection methods. Subsequently, Section 3 presents our approach for treating the different aspects of the challenge. Then, in Section 4 we present a preliminary evaluation of the overall approach and present some topics produced by the overall approach and finally Section 5 concludes the paper.

2 Related work

At a very high level there are three different classes of topic detection methods:

1. *Document-pivot methods*: these approaches cluster together documents using some measure of document similarity, e.g. cosine similarity using a bag of words representation and a *tf-idf* weighting scheme. For instance, the approach in [Pet10] is an approach that falls in this class and uses a incremental, threshold-based cluster assignment procedure. That is, it examines each document in turn, it finds its best match from the already examined documents and either assigns it to the same cluster as its best match or initializes a new cluster, depending on if the similarity to the best match is above some threshold or not. Documents are compared using cosine similarity on *tf-idf* representations, while a Locality Sensitive Hashing (LSH) scheme is utilized in order to rapidly retrieve the best match. A variant of this approach is utilized in this work.

2. *Feature-pivot methods*: these approaches cluster together terms according to their cooccurrence patterns. For instance, the algorithm presented in [?] performs a sequence of signal processing operations on a tf-idf-like representation of term occurrence through time in order to select the most "bursty" terms. Subsequently, the distribution of

appearance of the selected terms through time is modelled using a mixture of Gaussians. Eventually, a cooccurrence measure between terms is computed using the KL-Divergence of the corresponding distributions and terms are clustered using a greedy procedure based on this measure.

3. *Probabilistic topic models*: these represent the joint distribution of topics and terms using a generative probabilistic model which has a set of latent variables that represent topics, terms, hyperparameters, etc. Probably, the most commonly used probabilistic topic model and one that has been extended in many ways is LDA [Ble03]. LDA uses hidden variables that represent the per-topic term distribution and the per-document topic distribution. A concise review of probabilistic topic models can be found in [Ble12].

For a more thorough review of existing topic detection methods please see [Aie13].
Two of the most important problems for topic detection are fragmentation and merging of topics. Fragmentation occurs when the same actual story / topic is represented by many different produced topics. This is quite common in document-pivot methods, such as the one that we build upon (e.g. if the threshold is set too high). Merging is in some sense the opposite of fragmentation, i.e. it occurs when many distinct topics, not related to each other, are represented by a single topic. In the case of document-pivot methods, merging may occur when the threshold is set too low. In that case, it is possible that the occurrence of terms that are not important for a topic may result in two documents related to different topics being matched. These merged topics may either be higher level topics of related lower level topics or may be mixed topics of lower level topics that are not related to each other, depending on the features on which the assignment of tweets to clusters has occurred. The first case may be acceptable depending on the required granularity of topics, but the second case is undesirable as it will produce topics that are inconsistent and of limited use to the end user. Thus, it is crucial for document-pivot methods to both do the matching based on the important textual features and to select the threshold appropriately. From an end user's perspective, fragmentation is bad because it results in redundant and overly specific topics, whereas merging has a much more negative effect as it is quite likely to provide incomprehensible topics.

3 Approach

The challenge did not only involve topic detection per se; it also involved various aspects of topic presentation and enrichment: topic ranking, title and keyword extraction, as well as retrieval of relevant tweets and multimedia. In the following we present the pursued approaches for each of these problems.

3.1 Pre-processing

The pre-processing phase of the employed solution involves duplicate item aggregation and language-based filtering. Duplicate item aggregation is carried out because tweets posted on Twitter are often either retweets or copies of previous messages. Thus, it makes sense, for computational efficiency reasons, to process in subsequent steps only a single copy of each duplicate item, while also keeping the number (and ids) of occurrences for each of them. We implemented this by hashing the text of each tweet and only keeping the text of one tweet per bucket. In practice, we observed that we obtained a significant computational gain by doing this (the computational cost of the hashing procedure is very small). Indicatively, for the first test timeslot, the instance of our crawler collected 15,090 tweets and after duplicate removal we ended up with roughly half of them: 7,546 tweets in particular. It should be also noted that the hashing scheme we utilized did put in the same bucket all exact duplicates but not near-duplicates. For instance, cases when a user copies a message but adds or removes some characters are not typically captured as duplicates. It is possible though to modify the preprocessing so that most such cases are also captured, e.g. one could filter out the "RT" string and the user mentions and repeat the same hashing procedure or one could detect near duplicates using Jaccard similarity (using also an inverse index for speed). These options were briefly tested but thorough testing and deployment has been deferred.
The second step involves language detection. We use a public Java implementation[2], which provided almost perfect detection accuracy. As dictated by the challenge guidelines, we only keep content in the English language. This further reduces the number of tweets that needs to be processed in futher steps. For instance, in the first timeslot, after the removal of non-English tweets we end up with 6,359 tweets (from 7,546 non-duplicate tweets that were tested).

3.2 Topic detection

Having a collection of tweets (with duplicates and non-English tweets removed), we now proceed to detect topics in it. In previous work [Aie13], we experimented with all three classes of methods. All present many challenges when applied to a dataset retrieved from

[2]https://code.google.com/p/language-detection/

Twitter. The main reason is that Twitter messages are very short. For document-pivot methods this exacerbates the problem of fragmentation, as it is more likely, at least compared to longer documents, that although a pair of messages discusses the same topic, there may not be enough terms present in both of them to link them. For feature-pivot methods, the problem with short documents is very similar: i.e. in short documents it is more likely that all terms that represent a topic will not cooccur frequently enough in order to be clustered together. In this work, we opt for a document-pivot approach, similar to that of [Pet10], but we modify it in order to take advantage of some features that can significantly improve the document clustering procedure. In particular, we recognize two facts: a) tweets that contain the same URL refer to the same topic and b) a tweet and a reply to it refer to the same topic. Therefore, we can immediately cluster together tweets that contain the same URL and we can also cluster together tweets with their replies. Considering that there will be cases that these initial clusters will contain tweets that do not contain the same textual features, we can expect that taking into account such information should improve the results of a pure document-pivot approach by reducing fragmentation.
Thus, the idea is to perform some first-level grouping of items based on the above features, which will subsequently be taken into account as part of a second-level document-pivot procedure. In order to obtain the first-level grouping, we utilize a Union-Find structure [Cor09]. Essentially, we create a graph that contains one node for each tweet, connect pairs of tweets that contain the same URL or that are related by a reply and obtain the set of connected components. Components that have more than one tweets are the first-level groups that we will subsequently use in our second-level clustering procedure. Clearly, a large number of tweets, those that are the only members of a component with a single element, are not put into any first-level cluster. Those tweets are not discarded and are also considered in the second-level clustering algorithm.
The algorithm employed for the second-level clustering is similar to that of [Pet10] (i.e. we use an incremental threshold based clustering procedure and LSH for fast retrieval), but has some modifications. We take into account the first-level clustering by examining if each new tweet to be clustered (it is reminded that all tweets are examined, either they belong to some first-level cluster or not) has been assigned to a first-level cluster and if it has, the other tweets from the first-level cluster are immediately assigned to the same second-level cluster (and are not further examined in subsequent clustering steps). Thus, all the first-level clusters become members of the second-level clusters produced by the document-pivot procedure. Additionally, a second-level cluster may also contain tweets that were not members of a first-level cluster and also second-level clusters may be created from tweets that did not belong in first-level clusters.
In practice, by inspection of the results of early experiments, it turns out that there still is some fragmentation: some topics are represented by multiple second-level clusters. Therefore we seeked ways to reduce this fragmentation.
We first experimented with a semantic representation, utilizing WordNet. In particular, instead of representing the documents with a plain bag of words representation that uses the raw textual features, we tried to use the synsets of the verbs and nouns in each document. Such a representation could improve the results, since it would introduce some semantics in the document matching procedure and could match documents that do not contain the same raw terms. In practice, preliminary results showed that this is indeed true; however it is also very likely to have the opposite effect: i.e. topic merging. Eventually, we dropped the idea of using WordNet features to represent documents and pursued a more moderate approach in order to deal with fragmentation.
This consisted of two things. First, we utilized lemmatized terms instead of raw terms in order to be able to better match terms. We also considered the use of stemming, but stemming is a much less reliable process and may introduce false matches. Additionally, we recognize that some features are more important than others for text matching. These features include named entities and hashtags. We use a *tf-idf* representation of documents and we boost the terms that correspond to named entities and hashtags by some constant factor (1.5 in our experiments, later we will also examine the effect of using non-constant boost factors). More formally, for the stemmed term j in the i^{th} document we compute the $tf\text{-}idf_i^j$ weight as follows:

$$tf\text{-}idf_i^j = \begin{cases} 1.5 \times tf_i^j \times idf^j, & \text{if } j \text{ is entity or hashtag} \\ tf_i^j \times idf^j, & \text{otherwise} \end{cases} \tag{1}$$

where tf_i^j is the frequency of the term in the document and idf^j is the inverse document frequency of the term in an independent randomly collected corpus (more details on this corpus will be provided later). For lemmatization and named entity recognition we utilize the Stanford Core NLP library [3].
Finally, as we mentioned before, the threshold value is an important parameter of the process. We opt for a high threshold (0.9) so that there is no merging, at the

[3]http://nlp.stanford.edu/software/corenlp.shtml

cost of some fragmentation (despite the modifications that we did to avoid it). As will be shown in a later section, where we present some empirical results, the produced topics are quite clear, meaning that there is no merging, and come at the appropriate level of granularity.

3.3 Ranking

The challenge required that only 10 topics per timeslot are returned. The preliminary tweet grouping step resulted in a few hundred first-level topics (483 for the first timeslot). When we apply the document-pivot clustering procedure we end up with considerably more second-level topics (2,669 for the first timeslot using a threshold of 0.9). Although, as verified by inspection, there still is some fragmentation , the number of actual topics is quite large. Thus, we need to rank the produced second-level topics in order to select the most important.

Initially, we considered to simply rank the topics according to the number of documents they include and the number of retweets these documents receive. However, we realized that the granularity and hierarchy of topics is also important for topic ranking. As already discussed, some topics may be considered as subtopics of larger topics and it is reasonable that the attention that a larger topic attracts should affect the ranking of related finer topics. For instance, the most popular high-level topic in our corpus are the events in Ukraine (this was determined in an early exploratory stage of our study by examining the ratio of likelihood of appearance – for more details on this likelihood please see the section on title extraction – of terms in the test corpus and in an independent randomly collected corpus, the term “Ukraine” had the highest ratio). It makes sense then that although a topic about some event in Ukraine may be linked to as many documents as another topic about a concert, however considering the overall attention that the events in Ukraine received, the Ukraine related topic should be ranked higher.

In order to take advantage of this, we apply the following procedure. We perform a new clustering of the documents, but this time we boost further the weight of hashtags and entities. The boost factor is not the same for each entity and hashtag, instead, it is linear to its frequency of appearance in the corpus. More formally, $tf\text{-}idf_i^j$ weights are computed as follows (cf. Eq. 1):

$$tf\text{-}idf_i^j = \begin{cases} cf^j \times tf_i^j \times idf^j, & \text{if } j \text{ is entity or hashtag} \\ tf_i^j \times idf^j, & \text{otherwise} \end{cases} \tag{2}$$

where cf_j is the frequency of appearance of the entity or hashtag j in the test corpus. This significantly reduces the number of produced topics (1,345 for the first timeslot whereas 2,669 topics were produced from the second-level clustering for the same timeslot) and by inspection it appears that it reduces fragmentation a lot. Importantly, merging takes place, but only related topics are merged into clean higher level topics. For example, the algorithm manages to put all documents related to Ukraine to the same cluster. Subsequently, we rank these high-level topics by the number of documents in the corresponding clusters and link each second-level topic produced by the initial document-pivot procedure to the corresponding high-level topic. The linking is carried out by finding which high-level topic contains the largest number of tweets of each second-level topic. Finally, we rank all second-level topics belonging to the same high-level topic according to the number of tweets they contain. Eventually, we have a two-level clustering, one for high-level topics and one for the low/second-level topics within each of them. In order to select the 10 topics to return, we apply a simple heuristic procedure with the number of low level topics selected from each high-level topic dropping linearly as its rank drops. More specifically, we apply the following procedure. First we examine only the top-ranked high-level topic and select a single low-level topic from it. Then, we examine the top two high-level topics and select one low-level cluster from each of them and so on until we obtain 10 topics. Of course, selected second-level topics are not reconsidered for selection when a high-level topic is revisited during the described procedure. Also, in case that there are not enough low-level topics in some high-level topic we just skip it.

It should also be noted that we attempted to produce high-level topics without additional boosting of entities or hashtags, either by lowering the similarity threshold or by clustering second-level super-documents, but both these approaches resulted in mixed topics. It appears that these mixed topics were formed based on less important textual features which are more common across different topics. On the other hand, the applied approach of boosting entities and hashtags in a more aggressive manner did not produce any mixed topics and did indeed manage to surface the higher level topics.

3.4 Title extraction

We first split the text of each tweet in the cluster into sentences to obtain a set of candidate titles. Clearly, splitting the text into sentences makes sense, as the title has to be a coherent piece of language. To obtain sentence separation we again use the Stanford

NLP library. Having an initial set of candidate titles, we subsequently compute the Levehnstein distance between each pair of candidate titles in order to reduce the number of actual candidates. In the final step we rank the candidate titles using both their frequency and their textual features. The score of the title is the product of its frequency and the average likelihood of appearance of the terms that it contains in an independent corpus. The likelihood of appearance of a term t was obtained using a smoothed estimate in order to account for terms not appearing in the independent corpus:

$$p(t) = \frac{c_t + 1}{N + V} \tag{3}$$

where c_t is the count of appearances of t in the independent corpus, N is the total number of (non-unique) terms in the corpus and V is the vocabulary size (larger than the number of unique terms in the corpus). The corpus that was utilized to obtain these estimates was collected by randomly sampling from the Twitter streaming API and consisted of 1,954,095 tweets. It should also be noted that removed candidates increase the frequency count of their most similar candidate and also that, despite the fact that we do not process duplicate items, the count of duplicates removed for each processed item contributes to the frequency of the sentences extracted from it.

3.5 Keyword extraction

The keyword extraction process is similar to the title extraction process. However, instead of complete sentences, we now examine either noun phrases or verb phrases. We decided to work with noun phrases and verb phrases instead of unigram terms because they generally provide a less ambiguous summary of topics. In particular, short phrases can be more meaningful, regardless of the order that they appear in, as compared to single terms. For instance, let us consider one of the topics in the first timeslot in the test set. That topic is about Ukrainian journalists publishing a number of documents found in president Yanukovich's house. The set of keywords we produced was: "secret documents", "Yanukovich 's estate", "Ukraine euromaidan", "was trying", "president 's estate". One can see that regardless of the sequence of these phrases, one can grasp a fairly good idea about the topic. If however we used single terms, it could be possible, depending on the order of terms, that some of them may be incorrectly associated, e.g. "secret" could be associated to the term "estate" instead of the term "documents".
Eventually, in order to select the keywords we rank them according to their frequency in the clustered documents and their likelihood of appearance in an independent vocabulary as we did for the titles. However, for keyword extraction we are not limited to selecting a single candidate, as is the case for title extraction. Thus, we need a mechanism for selecting the number of top ranked candidate titles. We utilize a "largest gap" heuristic to do this. That is, after ranking the candidate keywords we compute the score difference between subsequent candidates, we find the position in the ranked list with the largest difference and select all terms until that position.
At the final step of the process, we add to the set of keywords the set of most important entities. These are determined using a similar "largest gap" heuristic and we only add them if they do not already appear as part of a phrase in the set of keywords. Finally, it should be noted that we use the Stanford NLP library to obtain the noun and verb phrases. However, instead of doing a full parsing of the texts, which would be computationally costly, we perform part of speech tagging and apply some heuristic rules to obtain noun and verb phrases from part of speech tags. More particularly, we identify sequences of terms consisting only of nouns, adjectives and possessive endings (e.g. "'s") as noun phrases and we identify sequences of terms consisting only of verbs as verb phrases.

3.6 Representative tweets selection

The challenge also requires that a number of representative and as much as possible diverse tweets is provided for each topic. The set of related tweets can be easily obtained in our approach, since we utilize a document-pivot method. Regarding diversity, the duplicate removal step that we apply at the first stage of our processing partly takes care of this requirement. However, there are still some near duplicates that were not captured by the duplicate removal step. Additionally, to introduce as much diversity as possible, we make sure that all replies from the topic's cluster are included in the set of representative tweets and additionally we include the most frequent tweets (making sure that the total number of selected tweets is at most 10).

3.7 Relevant image extraction

We retrieve relevant images by applying a very simple procedure. In particular, if the tweets associated with a topic contain the URL of some images, then we find the most frequent image and return it. Otherwise, we issue a query to the Google search API, searching by the title of the topic and associate to the topic the first image returned. In a few cases, this did not return any results; then we issue a further query, this time using the most popular keyword. It should be noted though that this approach has a limitation: the Google search

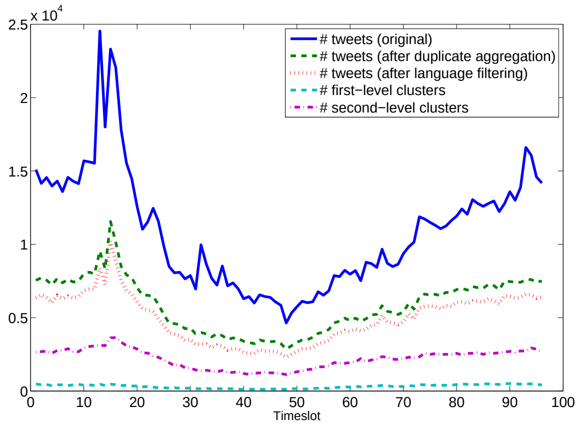


Figure 1: Number of tweets before and after duplicate aggregation and language filtering, as well as the number of first-level and second-level clusters produced

API allows only a specific number of queries per day and thus we had to issue repeated queries for a long period of time in order to obtain results for each image. A potentially better option in that respect would be to use a different search API, such as Twitter's.

4 Evaluation

In the following we examine different aspects of the applied approach and then we comment on the quality of the produced topics. Figure 1 displays the number of tweets before and after duplicate aggregation and language filtering, as well as the number of first-level and second-level clusters produced. One thing to note is that for all timeslots there is a significant reduction on the number of tweets to be clustered after duplicate aggregation and language filtering. Additionally, for all timeslots there is a number of – typically a few hundred – first-level clusters, each of which contains at least two tweets, meaning that we immediately and without resorting to any complicated clustering operations we have obtained initial clusterings for a significant part of the tweets to be clustered. The number of second-level topics is typically larger though as tweets that did not form first-level clusters also participate in the second-level clustering procedure. It is also interesting to note that the computational cost of the complete procedure for each timeslot is not that high. In particular, the complete set of operations (first and second level clustering, ranking, title and keyword extraction as well as relevant image retrieval) took on average 65.33 seconds per timeslot on a machine with moderate computational resources (Intel Q9300 CPU running at 2.5 GHz and 4GBs of RAM).
Table 1 presents the ten topics produced by our approach for the first test timeslot. As a first remark, it appears that all topics are related to a distinct event, which is fairly well represented by both the title and the keywords. It should be noted though, that in some cases the set of keywords may not be enough by itself to provide a very clear picture of the essence of the story. For instance, in the story about the Ukrainian parliament voting to send Yanukovich to Hague, the keyword Hague is missing, although it should be included. Thus, the keyword extraction process may be improved, by appropriately changing the mechanism of automatically selecting the number of phrases and entities to return (across the complete test collection, the minimum number of keywords retrieved was 1, the maximum was 5 and the average was 2.625). Also, due to the heuristic that we applied to rapidly retrieve noun and verb phrases, we occasionaly have mixed noun and verb phrases, e.g. the phrase "Ukrainian parliament votes". The title on the other hand makes perfect sense and is in all displayed topics (and most other topics as well) very indicative of the topic. Finally, the multimedia retrieved are sometimes very relevant and some times not too much; e.g., for the topic about the cost of Yanukovich's house, the retrieved image is the frontpage of some newspaper.

5 Conclusions

In this paper we presented the approach pursued by our team for participating to the SNOW 2014 Data Challenge. In short, we have utilized a document-pivot approach, however we have taken advantage of features that allow us to improve the quality of the detected clusters. In particular, we have taken advantage of commonly appearing URLs and of reply relationships between tweets, formulating a two-level clustering procedure. We have tuned our clustering, so that it provides a set of topics at the required granularity, i.e. low level stories rather than high-level topics at the cost of some fragmentation. In practice, this provided very good topics. Subsequently, we apply a number of NLP techniques in order to enrich the representation of topics: we use sentence splitting for title extraction and we use noun and verb phrase extraction for identifying key phrases. Additionally, we identify that the ranking of some topic should be related to the importance of any larger topic that it may be linked to and we apply an appropriate procedure in order to achieve a two-level ranking of topics

Acknowledgments

This work is supported by the SocialSensor FP7 project, partially funded by the EC under contract number 287975.

References

[Pap14] S. Papadopoulos, D. Corney, L. Aiello. SNOW 2014 Data Challenge: Assessing the Performance of News Topic Detection Methods in Social Media. *Proceedings of SNOW 2014 Data Challenge*, 2014.

[Pet10] S. Petrović, M. Osborne, V. Lavrenko. Streaming First Story Detection with Application to Twitter. *HLT: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2010.

[Wen10] J. Weng, E. Lim, J. Jiang, Q. He. TwitterRank: finding topic-sensitive influential twitterers. *Proceedings of the third ACM international conference on Web search and data mining*, 2010.

[Ble03] D. Blei, A. Ng, M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[Ble12] D. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, 2012.

[Aie13] L. Aiello, G. Petkos, C. Martin, D. Corney, S. Papadopoulos, R. Skraba, A. Goker, I. Kompatsiaris, A. Jaimes. Sensing trending topics in Twitter. *IEEE Transactions on Multimedia*, 15(6):1268–1282, Oct 2013.

[Cor09] T. Cormen, C. Leiserson, R. Rivest, C. Stein. Introduction to Algorithms, Third Edition. *The MIT Press*, 2009.

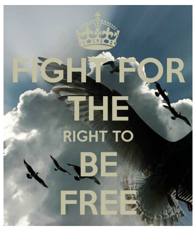
Title: Fight for the right to be free! *Keywords:* Ukraine, madonna, free !! fight fascism *Relevant tweet:* @Madonna: Fight for the right to be free!! Fight Fascism everywhere! Free Venezuela the Ukraine&Russia #artforfreedom	
Title: Ukraine's toppling craze reaches even legendary Russian commander, who fought Napoleon *Keywords:* Legendary russian commander, Ukraine *Relevant tweet:* @RT_com #Ukraine toppling craze reaches even legendary Russian commander,who fought Napoleon http://on.rt.com/izqunf	
Title: Ukraine parliament votes to send Yanukovych to The Hague *Keywords:* Ukraine parliament votes, Yanukovych *Relevant tweet:* #Ukraine parliament votes to send Yanukovych to The Hague	
Title: Ukraine's president spent $ 2.3 m on dining room decor, $ 17k tablecloths, $ 1m to water his lawn *Keywords:* Ukraine 's president, dining room decor *Relevant tweet:* #Ukraine's president spent $2.3M on dining room decor, $17K tablecloths, $1M to water his lawn	Criminal probe in Quebec oil train derailment Charlotte Sun EGYPT SPEEDS TRANSITION RESURRECTING EXTINCT BIRDS? Rough waters for YMCA Summer camp kayaks, canoe stolen over weekend
Title: Journalists in Ukraine are in the process of uploading 1000s of secret documents found at Yanukovich's estate *Keywords:* Secret documents, Yanukovich 's estate, Ukraine euromaidan, was trying, president 's estate *Relevant tweet:* The #YanukovychLeaks is up! Here are the documents recovered at the ousted presidents estate. #Ukraine #euromaidan	
Title: Mt. Gox takes Bitcoin exchange offline as currency woes mount, does not say when transactions/withdawals will resume *Keywords:* Gox, bitcoin exchange offline, currency woes *Relevant tweet:* Mt. Gox takes #Bitcoin exchange offline as currency woes mount, http://fxn.ws/1ppoMGk @joerogan	
Title: Can't decide if I want to write this week's Most Googled Song about Seth Myers Jimmy Fallon or Bitcoin. Thoughts?? *Keywords:* Bitcoin, Seth *Relevant tweet: Can't decide if I want to write this week's Most Googled Song about Seth Myers & Jimmy Fallon or Bitcoin... Thoughts??*	
Title: Syria aid still stalled after UN. *Keywords:* Melawanlupa syria aid, resolution, stalled *Relevant tweet: #MelawanLupa RT #Syria #aid still stalled after #UN. resolution http://reut.rs/1mwaqlh*	
Title: Remarks at today's UN General Assembly briefing on the Humanitarian Situation in Syria *Keywords:* Today 's un general assembly briefing *Relevant tweet:* Remarks by @AmbassadorPower at today's UN General Assembly briefing on the Humanitarian Situation in #Syria: http://go.usa.gov/Bt2d	
Title: Usmnt's friendly vs Ukraine on March 5 moved to Cyprus, according to Ukraine's Football Federation. *Keywords:* Usmnt 's friendly, Ukraine's football federation *Relevant tweet:* #USMNT's friendly vs Ukraine on March 5 moved to Cyprus, according to Ukraine's Football Federation. http://foxs.pt/NuDSvq	

Table 1: The 10 topics produced by our approach for the first timeslot.