Discovering Global and Local Bursts in a Stream of News

Max Zimmermann
School of Computer Science
Otto-von-Guericke-University
of Magdeburg,Germany
max.zimmermann@iti.cs.uni-magdeburg.de

Irene Ntoutsi
Institute for Informatics
Ludwig-Maximilians-University
of Munich,Germany
ntoutsi@dbs.ifi.lmu.de

Zaigham Faraz Siddiqui
School of Computer Science
Otto-von-Guericke-University
of Magdeburg,Germany
siddiqui@iti.cs.uni-magdeburg.de

Myra Spiliopoulou
School of Computer Science
Otto-von-Guericke-University
of Magdeburg,Germany
myra@iti.cs.uni-magdeburg.de

Hans-Peter Kriegel
Institute for Informatics
Ludwig-Maximilians-University
of Munich,Germany
kriegel@dbs.ifi.lmu.de

ABSTRACT

Reports on major events like hurricanes and earthquakes, and major topics like the financial crisis or the Egyptian revolution appear in Internet news and become (ir)regularly updated, as new insights are acquired. Tracking emerging subtopics in a major or even local event is important for the news readers but challenging for the operator: subtopics may emerge gradually or in a bursty way; they may be of some importance inside the event, but too rare to be visible inside the whole stream of news. In this study, we propose a text stream clustering method that detects, tracks and updates large and small bursts of news in a two-level topic hierarchy. We report on our first results on a stream of news from February to April 2011.

1. INTRODUCTION

People resort increasingly on Internet sources to acquire up to date information about events of global or local importance. News providers care to perform frequent updates of the arriving information, while users of social platforms experience bursts of postings in association with major and rapidly evolving events. Delivering insights on the semantics of bursty events is a major challenge. In this study, we provide a two-level approach for the detection of small and major bursts in a fast stream of news.

There are first promising results on learning bursts in streams of news [8], and there is substantial research on the discovery of emerging and evolving topics, see e.g.[10, 4]. However, these approaches fall short of capturing novelty as a *local change* in non-major events: bursts are not only major events that draw everybody's attention; there are also events of local importance, occurring inside a part of a stream, which refers to a bursty or conventional topic. This leads to the challenge of simultaneously learning both *global* and *local bursts* in a stream.

Our approach deals with this challenge as follows: We consider a two-level hierarchy of topics over the text stream. Topics of the 1st level are global ones; global bursts are captured at this level. Topics of the 2nd level (we also call them "subtopics") are local ones; they capture events of local importance inside a 1st level topic, including local bursts. Subtopics are modeled inside a topic, using a feature space that is particular to the topic. For example, the keyword "Japan" may be used to distinguish the global event of the Japan quake from other global events, but may be ignored inside this event, because it does not contribute to distinguishing among its subtopics (like the tsunami, the Fukushima disaster, the international aid etc).

To detect emerging topics of the 1st or 2nd level, we pick arriving documents that fit to no topic or fit to some topic but to no subtopic inside it: we do not force these documents into a cluster, but rather retain them in a "container". We maintain a single global container at the 1st level, and one local container for each 1st level topic. When a local container is overfilled, we adjust the feature space and perform re-clustering - but we only consider the documents in the container and in the other subtopics of this 1st level topic. This allows us to identify bursts local to a topic, and adjust the model to them without re-computing other topics. Only documents that do not fit to any 1st level topic and thus flood the global container may lead to global re-clustering.

The rest of the paper is organized as follows. Related work is found in Section 2. Our method is presented in Section 3. Section 4 discusses our experimental results. Conclusions and outlook are presented in Section 5.

2. RELATED WORK

The discovery of emerging topics in a stream of news is studied in the context of *text stream clustering* and of *dynamic topic modeling*. Also, there are methods focussing on the detection of bursts in news posted in social platforms.

The *text stream clustering* algorithm of Aggarwal and Yu maintains a fixed number of K clusters/topics over time [1]. If a new document is too far from all existing clusters, it

SAC'12 March 25-29, 2012, Riva del Garda, Italy.

can become the seed of a new cluster, but *only if* some old cluster receives no new members and can thus be deleted. Otherwise, the document is assigned to the closest existing cluster, even if it is far from it. Liu et al [10] follow a similar approach for the actualization of K clusters as the text stream progresses. However, instead of using single words as document features, they use multiword phrases as topic signatures, and a more elaborate proximity computation. This family of methods has a number of shortcomings. First, they assume an a priori fixed feature space. When a burst occurs, it is likely to be associated with words or terms that are not part of the feature space. This caveat is addressed in the framework MONIC [14], which however focusses on the a posteriori interpretation of change, and not on the discovery of bursts. Second, the aforementioned text stream clustering methods assign each arriving document to some cluster. If the feature space is fixed, then this assignment may take place on the basis of keywords that are not characteristic of the burst. Even if the feature space is adjustable, as in [13] which builds upon [14], there is danger of overseeing keywords that are not yet frequent enough to become part of the feature space. Third, if the burst occurs inside a topic, i.e. it is a local burst, the keywords characteristic to it may never be frequent enough to become a topic of its own.

Dynamic topic modeling constitutes a second family of approaches on learning topics over a stream adaptively [12, 4, 15]. These methods probabilistically associate documents to topics (which are latent variables describing the data distribution), and are considered more flexible and robust than text stream clustering methods. For the discovery of bursts, though, they are subject to the same caveats mentioned above. First, the feature space is mostly assumed to be known a priori with the exceptions of [2, 7]. Second, even if the feature space can be modified, keywords that are important for the burst may never become frequent enough to be associated with any of the latent variables, especially if the burst is inside a given topic.

Very recently, scholars have proposed methods that model bursts of news in social platforms. Much research in this context has been devoted to classify opinions and to categorize sentiments, see e.g. [3]. In the context of topic monitoring, the "TwitterMonitor" of Mathioudakis and Koudas detects sharp increases ("bursts") in the frequency of keywords found in tweets, and marks sets of bursty, frequently co-occurring keywords as trends [11]. This approach is likely to detect keywords associated with major events, but cannot identify subtopics local to the same event. The incremental method of Gu et al. on topic monitoring in Twitter is hierarchical and can thus distinguish between global and local topics [8], albeit topics are subordinate to an a priori known event; disentangling the individual events of the stream into 1st level topics is not addressed. Gu et al. first identify the core information blocks of the single event, by finding key phrases that are adopted by many users for the description of this event. These blocks are then organized into a theme hierarchy based on their similarity and according to a list of properties that the hierarchy should have; for example, the parent of a node must have less keywords than the node itself. When a new tweet arrives, it may be assigned to an existing theme/node or become a new theme, whereupon the hierarchy is re-constructed. This decision is taken on the basis of snapshot quality versus temporal smoothness [6]. However, performing such a test (or re-constructing the hierarchy) in response to a single tweet does not seem appropriate, because a tweet that is too different from all others may be noise; a burst should be supported by several documents. Moreover, the approach is customized to the peculiarities of a stream of tweets (short documents, distinct users, repeated phrases and near-duplicates that must be eliminated) rather than an arbitrary stream of news. He and Parker study bursts in scientific publications, considering as basis for their method models of "burstiness" designed for social media [9]. Their approach is confined to platforms where information is propagated, rather than arbitrary news providers. Moreover, it is designed to find bursts rather than smoothly emerging (and declining) topics. In our approach, we do distinguish between bursty and gradually emerging (sub)topics, since both may occur in a stream of news.

3. EXTRACTING EVOLVING AND BURSTY TOPICS FROM THE STREAM

We observe a growing collection D of documents. The documents constitute a stream, which we monitor at discrete timepoints $t_0, t_1, \ldots, t_i, \ldots$ Our method aims to *organize* the stream of documents into a 2-level hierarchy of broad *topics* and more specific *subtopics*. For example, a topic might refer to the entire situation of the Japan earthquake while two of its subtopics might describe the particular situations of the nuclear meltdown and the plummet at the stock market. Since the document stream evolves over time, our method *updates* this hierarchy online so as to reflect the evolution of the underlying population. In particular, we propose a two-level approach that encompasses one pass per level of the topic/subtopic hierarchy and updates only the affected parts of the hierarchy.

3.1 Overview

A graphical representation of our method is depicted in Figure 1. It consists of three main components:
(i) a 2-level hierarchy of *topics* and *subtopics*,
(ii) the concept of (sub)*containers* for storing documents that are novel with respect to the existing hierarchy, and
(iii) indicators to decide on document *novelty* and hierarchy update or re-construction.

To extract the 2-level hierarchy of topics and subtopics from a collection of documents we employ text clustering (Section 3.3). Since the stream of documents evolves over time, the hierarchy might deteriorate and its (sub)topics might not be suitable anymore for the description of the new documents. We use the notion of *document novelty* to evaluate whether a document brings some new information with respect to the existing hierarchy of topics and subtopics. To decide on the novelty of a new document, we introduce document novelty (cf. Definition 2) which computes the cosine similarity to its most similar (sub)topic in the hierarchy. If the similarity score is below a given threshold, this entails that the document does not belong to the boundary of any topic and thus, it exhibits novelty. We store these documents into a 1st level *container*, since they represent something new that cannot be covered by the current topics. Such new documents may be part of a new topic thread, e.g., the first reports on the Japan tsunami would fit in no topic and thus go into the 1st level container at first.

Novelty may also appear at the 2nd level: subtopics on tsunami and nuclear meltdown formed early within the Japan

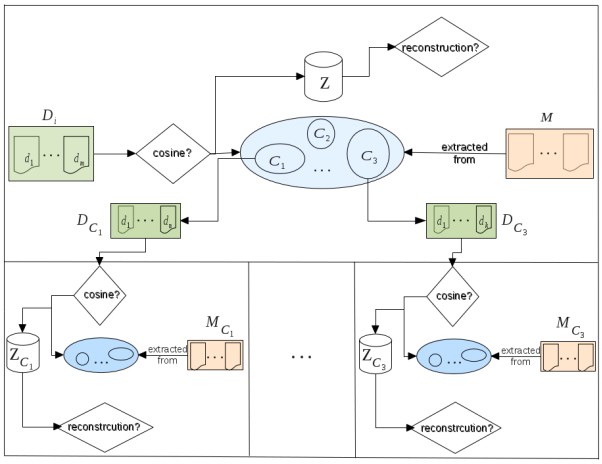

Figure 1: Learning the 2-level topic hierarchy over time based on the novelty of the arriving documents

topic, while reports on the extreme weather conditions, to which the homeless people were subjected, arrive later on. We store such documents in a 2nd level container, a *subcontainer*. We associate one subcontainer to each 1st level topic. This has the advantage that a subcontainer accommodates documents that are within a topic (otherwise they would have been assigned to a different topic), but they are not within any of the already known subtopics. The documents in a subcontainer may thus be the seed of a new subtopic in the specific topic.

The size of the (sub)container, i.e. the number of documents displaying novelty with respect to the existing (sub) topics, is our second novelty indicator, called *stream novelty* (cf. Definition 3). Whenever the number of documents that exhibit novelty exceeds a certain threshold, i.e. the corresponding 1st or 2nd level container is overflown, a local or global re-adjustment is needed: if a 2nd level container overflows, we re-learn the subtopics inside the affected topic only; we learn the 1st level topics only if the 1st level container is full. This way we avoid the re-structuring at each new arriving document and also we are more robust to noise/outliers.

Figure 1 shows the decision on *document novelty* by the "cosine" rhombus and on *stream novelty* by the "reconstruction" rhombus. In Figure 1 the (sub)containers are denoted by ζ. The update of the hierarchy is described in Section 3.4.

3.2 Definitions and Notations

We express topics and subtopics as labels of clusters.

Definition 1. [Topic Label] Let M be a dataset of documents and let $f(M)$ be the feature space learned upon M (through TF–IDF) . Let $\mathcal{C}$ be a 1st or 2nd level topic learned from M based on $f(M)$. The label of $\mathcal{C}$, denoted by $\hat{\mathcal{C}}$, is a $|f(M)|$-dimensional vector:

$$\hat{\mathcal{C}} = \prec w_1, w_2, \ldots, w_{|f(M)|} \succ$$

where $w_i, i = 1 \ldots |f(M)|$ is the average weight of keyword k_i in the documents belonging to $\mathcal{C}$.

When a new document arrives, we assess its novelty on the basis of its similarity to its most similar topic per level.

Definition 2. [Document Novelty] Let d be a new document. Let θ be a set of topics extracted from a dataset M. Let $f(M)$ be the feature space derived from M (through TF–IDF keyword weighting). Given a similarity threshold $\delta \in [0, 1]$, d is novel with respect to θ if:

$$\max_{\mathcal{C} \in \theta} cosine^{f(M)}(\widehat{\mathcal{C}}, d) < \delta$$

where $\hat{\mathcal{C}}$ is the label of $\mathcal{C}$ and $cosine()$ is the cosine similarity function.

For a new document, we first compute the *document novelty* towards the topics of the 1st level. If the document is similar to a 1st level topic with respect to δ, then we also compute the novelty of the document for the subtopics of this topic.

Note that the document novelty is computed with respect to a feature space $f(M)$, which is either the feature space of the 1st level or the feature space of the documents inside a 1st level topic. This means that before the comparison, the new document d is transformed into the feature space $f(M)$. In particular, the TF–IDF scores of d are computed based on the TF scores of d and the IDF scores from $f(M)$.

Document novelty assesses the novelty of a single document d. A single document, though, is not sufficient for modifying the hierarchy. Rather, we accumulate the novelty of the incoming documents by monitoring the size of the containers where they are stored. In particular:

Definition 3. [Stream Novelty] Let θ be the set of (1st or 2nd level) topics. Let $\mathcal{Z}$ be the container associated with θ; it contains all those documents that exhibit novelty with respect to θ. Given a size threshold parameter σ, $\mathcal{Z}$ exhibits novelty towards θ if:

$$|\mathcal{Z}| \geq \sigma$$

The size threshold σ might be a constant number or it may be set to a fraction of the number of documents in θ.

3.3 Extracting the Hierarchy of Topics

Let D_0 be the set of documents arriving at timepoint t_0. These documents comprise our initial collection M upon which the 2-level topic hierarchy is built. We first construct the 1st level clusters (general topics) from M and then, we cluster the documents in each 1st level cluster into 2nd level clusters/subclusters (subtopics).

For the topic extraction (clustering at the 1st level), we apply *fuzzy c-means* upon the feature space $f(M)$ derived from the collection M, and obtain a set of clusters $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$. We crisply assign each document $d \in M$ to its most similar cluster $\mathcal{C}_i, i = 1 : K$. This way a set of documents $M_{\mathcal{C}_i} \subset M$ is assigned to each cluster $\mathcal{C}_i$ such that $\cup_{i=1}^{K} M_{\mathcal{C}_i} = M$ and $M_{\mathcal{C}_i} \cap M_{\mathcal{C}_j} = \emptyset, \forall i \neq j, j = 1 : K$.

For the subtopic extraction (clustering at the 2nd level), we refine the feature space $f(M)$ by TF–IDF weighting in the subcollection $M_{\mathcal{C}_i}$. Then, we apply *fuzzy c-means* over $M_{\mathcal{C}_i}$ in this cluster-specific feature space $f(M_{\mathcal{C}_i})$. The result of this step is a set of 2nd level clusters $\{c_{i1}, c_{i2}, \ldots, c_{ik}\}$ in each 1st level cluster $\mathcal{C}_i$, where k is the number of 2nd level clusters to be discovered for each 1st level cluster. As before, we crisply assign each document to its most similar 2nd level cluster, hence the clusters at this level are also disjoint.

The result is a 2-level hierarchy of clusters Θ containing, at the 1st level, the generic clusters $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$ and, at the 2nd level, the specific subclusters $\{c_{i1}, c_{i2}, \ldots, c_{ik}\}$ for

each $\mathcal{C}_i$. The 1st level clusters correspond to broad topics, whereas the 2nd level clusters correspond to more specific subtopics. Each (sub)topic in the hierarchy is described in terms of its label (cf. Definition 1).

The reason for choosing *fuzzy c-means* is that it is considered more robust and stable than *k-means*, e.g., [5]. This is important in our case, since the quality of the initial hierarchy affects novelty evaluation and consequently, hierarchy reconstruction.

3.4 Online Hierarchy Maintenance

When a new document $d \in D_i$ arrives, we try to merge it to the existing hierarchy first, into some generic topic (1st level clusters) and subsequently, into some more specific subtopic (2nd level clusters).

The merging decision is based on *document novelty* (cf. Definition 2), which evaluates the cosine similarity between the new document and (sub)topics in the hierarchy. The merging procedure works as follows: We first try to merge d to its most similar generic topic. To this end, the most similar topic $\mathcal{C}$ is computed among all topics $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$. The merging though is possible only if the similarity between d and C is higher than the similarity threshold δ. Otherwise, d is novel with respect to the existing topics in the hierarchy and it is added to the container $\mathcal{Z}$ at the topics level.

If d is in the boundary of a topic $\mathcal{C}$, we further check whether it can be absorbed by one of its subtopics $\{c_1, \ldots, c_k\}$. The most similar subcluster c for d is located as before and if the similarity between d and c is higher than the threshold δ, then d is absorbed by c. Otherwise, d is novel with respect to the subtopics of its relevant topic and thus, it is added to the subcontainer $\mathcal{Z}_\mathcal{C}$ for the specific topic $\mathcal{C}$. $\{c_1, \ldots, c_k\}$. So, there are three cases for d: i) it does not fit to the existing topics in the hierarchy ii) it fits to an existing topic but not to some of its subtopics iii) it fits to both a topic and one of its subtopics. Cases i) and ii) indicate that d presents some novelty for the existing hierarchy. If this is real novelty or noise, however, cannot be judged by a single document, it can rather be induced based on the next arriving documents. So, we cannot decide on this until more documents are accumulated in the (sub)container.

In particular, we rely upon the number of documents that exhibited novelty within a (sub)container in order to decide whether the hierarchy should be updated. If this number exceeds the size threshold σ, then there is the necessary accumulated novelty in the stream which might change the hierarchy. Note that we maintain novelty containers at different levels of granularity: a container at the topics level and a subcontainer for each topic. Based on the arriving documents either the container or a subcontainer might reach the size limit. In the first case, the whole hierarchy (topics and subtopics) needs adjustment (*global re-organization*). In the second case, only some topic in the hierarchy exhibits enough novelty and thus, only its subtopics should be adjusted. So, there is no need though to re-adjust the whole hierarchy, but only the branch that corresponds to the specific topic (*local re-organization*). The local re-organization involves less runtime than the global re-organization, since the complete 1st level topic remains unchanged and almost all subtopics, except the one in question, remain the same as well. This is possible since the subtopics of each topic are extracted independently and upon their own feature space.

When the hierarchy update is necessary, *fuzzy c-means* is applied over the new dataset to extract the topics (in case of global re-organization) or the subtopics of a topic in case of local re-organization. In case of global re-organization, the new dataset M consists of all documents in the container $\mathcal{Z}$ plus the W latest documents from the stream, where W is a window parameter. In case of local re-organization, the new dataset $M_\mathcal{C}$ contains all documents in the subcontainer of the corresponding topic $\mathcal{Z}_\mathcal{C}$ plus the W latest documents from the stream referring to the specific topic.

The reason for considering a certain amount of latest documents is that the container contains only documents that exhibit novelty whereas from the latest documents we can also derive already existing and still developing (sub)topics. Both of them though are necessary for establishing the new set of (sub)topics that are representative of the underlying population. Note that in both global and local re-organization cases, we keep in main memory only the contents of the container and of the current window, while we forget the rest of the data. The feature space is recomputed based on the new dataset (TD–IDF weighting) and the new (sub)topics are extracted. The updated hierarchy is used hereafter for the assignments of the new incoming documents. The contents of the (sub)container that issued the re-arrangement are discarded.

4. EXPERIMENTS

In this section, we evaluate our approach and the effect of the different parameters in a real stream of news data. We show that the learning of global and local topics based on the contents of the incoming stream is both effective and efficient. We further show how the detection of local bursts reduces the need for re-clustering all data from scratch without loosing much quality.

4.1 Dataset

For the evaluation, we use a stream of news articles. We collected them with *Google Search* from BBC by launching keyword-based queries and picking the top hits on four 1st level topics, namely *Egyptian revolution*, *2011 Cricket World Cup*, *Earthquake in Japan* and *Libyan civil war*. The statistics about each topic thread, including the monitoring period of crawling are shown Table 1. We then mixed the threads (respecting the timestamps!), and thus devised a stream of real data with known labels for the 1st level.

Table 1: Description of the News Threads

Topic	*keywords e.g.*	Duration	Docs/day	Total
Egypt	*protests, violence*	25/01-04/04	≈ 250	17,067
Libya	*rebels, revolution*	04/02-04/04	≈ 250	12,421
Japan	*earthquake, tsunami*	11/03-02/04	≈ 300	6,812
Cricket	*worldcup 2011, cricket*	16/02-04/04	≈ 100	4,208
				41,340

It must be stressed that the ground truth we have devised for the evaluation of our 1st level topics is not of doubtless veracity. The reason is that the results of the keyword-based searches may still contain documents that do not fit in the intended thread - this holds most of all for the threads on Egypt and Libya. Hence, the quality achieved by our clustering algorithm is a best attempt on very noisy data.

4.2 Parameter Settings and Measures

We evaluate the quality of the 1st level topics against the ground truth as average purity. For each cluster $\mathcal{C}$ found at some timepoint, we identify the majority class Y in it. Then, cluster purity for $\mathcal{C}$ is the ratio of documents in $\mathcal{C}$ that belong to Y. We define as (average) *purity* of a clustering Θ the average over the purity values for its clusters.

Table 2 summarizes our parameter settings for the 1st level (global) and the 2nd level (local). We study how these parameters affect purity directly (1st level only) and indirectly, through the number of re-clusterings at 1st and 2nd level: a re-clustering ensures that the new clusters fit the most recent data best, hence re-clustering is expected to have a positive effect on purity.

Table 2: Parameters and experimental settings

Hierarchy level	Num of clusters K	Container σ	Similarity δ
Global	5	{200,300}	[0-1]
Local	2	{100,200}	[0-1]

4.3 Global purity results

We experimented with the effect of the container on the global purity (Recall that we have the ground truth only for the global topics). The results are presented in Figure 2 where the evolving purity of the global topics under different container size thresholds is depicted. We experimented with different hierarchy update strategies:

- *no re-clustering*: The new documents are absorbed by the existing topics. Practically, each new document is assigned to its closest topic and thus, the containers are empty. This is the lower baseline.
- *continuous re-clustering*: Re-clustering is performed at each time point. This is the upper baseline.
- *re-clustering when container is full*: Re-clustering is performed only when the container is full of documents that display novelty w.r.t. the existing topics.

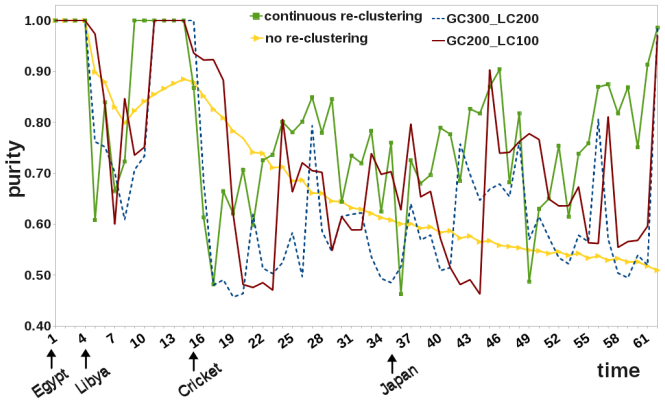

Figure 2: Global topics purity over time for different (sub)containers size (K=5, GSim=0.4,LSim=0.6)

The upper and lower baselines would help us to interpret the evolving purity of the global topics while varying the container size thresholds. The *lower baseline* (*no re-clustering*) shows an evolving purity over time varying from 1.0 to 0.6. These values imply that the initial global and local topics cannot represent the evolving structure of the news stream good enough. An average purity of almost 0.6 indicates that only the half of the documents contributing to the topic can be represented correctly by the topic. As such, the lower baseline shows a unstable structure of the global topics as the stream evolves. The *upper baseline* (*continuous re-clustering*) starts also at 1.0. It shows a decreasing tendency in the beginning which is inverted afterwards and ends with almost 1.0. Hence, continuous re-clusterings ensure a stable structure of the global topics as the stream evolves. However, re-clustering at each timepoint is an expensive process since a new feature space should be computed and the new clusters and their labels should be extracted upon it.

The notion of the (sub)containers acts as a trade-off between the no-reclustering and the continuous re-clustering approaches. The idea is to apply re-clustering only when it is necessary, that is, when enough novelty has been introduced in the stream. We experimented with different container size thresholds σ and we found that a size between 200 and 300 for the global container and a size between 100 and 200 for the local container exhibit stable global topics. The window parameter W, i.e., the number of latest documents from the stream to consider during re-clustering, was set to 300. The evolving purity for these two settings, (300,200) and (200,100), is depicted in the corresponding curves in Figure 2. Both curves show an increasing tendency of the purity, starting at different timepoints though: the curve with the sizes (200,100) shows an increasing purity ongoing from timepoint 32 by 13 global reclusterings and 27 local reclusterings while the sizes (300,200) results in an increasing purity ongoing from timepoint 41 by only 12 global and 10 local reclusterings. We also observe that the curves oscillate: downward trends indicate that the clusters get outdated, upward trends are the result of re-clustering. Oscillations are inevitable since the stream exhibits concept shifts and drifts.

Summarizing, our method detects local and global bursts and adapts well to evolving topics in the stream.

4.4 Influence of local bursts

The impact of local bursts is not shown by the purity of the global topics. So, we study separately how the detection of local bursts influences the number of global re-clusterings and thus, the quality of the global topics. To this end, we varied the *local similarity threshold* δ from 0.3 - 0.8, while keeping the global δ constant at 0.8. The size threshold σ was set to 200 (for the global container) and 100 (for the local container).

Table 3 shows the number of local and global re-clusterings under an increasing local δ . The higher δ is , the more local re-clusterings are performed. This is intuitive, since the local similarity threshold δ controls the process of growing the local containers: a small value for the local δ implies a slow growth while a larger value implies that many documents are characterized as novel and thus, the local containers exceed the size threshold earlier.

Figure 3 depicts the size of the global container over time under different values for the *local similarity threshold* δ. If we combine this chart with Table 3, it becomes obvious that the higher δ is, the more local re-clusterings are performed

Table 3: Number of global and local reclusterings for different local similarity thresholds δ

global δ	local δ	# global reclusterings	# local reclusterings
0.2	0.3	9	10
0.2	0.5	5	37
0.2	0.8	3	55

and also, that many local re-clusterings reduce the need of global re-clustering. This entails a very good interaction between the two levels of the topics hierarchy.

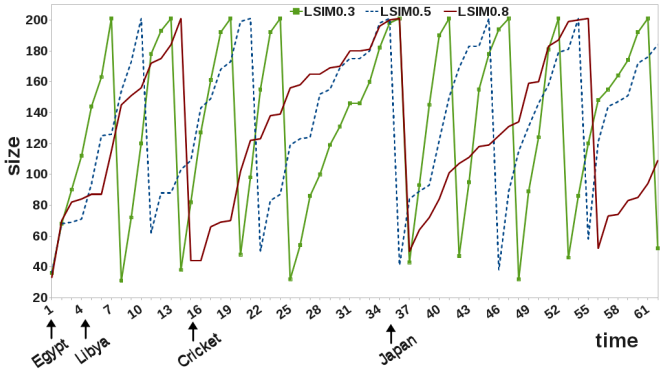

Figure 3: Global container size over time for different local similarity thresholds δ (K=5, GC=200, LC=100)

In Figure 2, we showed that less global re-clusterings result in lower purity of the global topics. We want to see whether this quality loss can be offset by local re-clusterings. The results are presented in Figure 4 where the purity of the global topics under different local δ's is depicted over time. As one can observe, the run (*LSIM0.8*) with the higher number of local re-clusterings (55) has the highest purity at the end of the recorded time even if it has the lower number of global re-clusterings (3). So, it seems that due to the local re-clusterings, the need for global re-clusterings is reduced while the quality of the topics remains good.

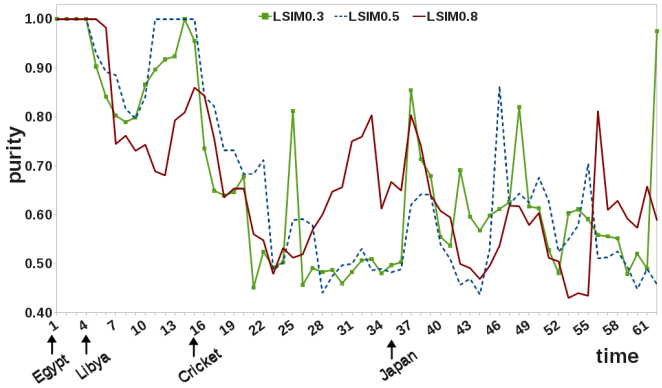

Figure 4: Global topics purity over time for different local similarity thresholds δ (K=5, GC=200, LC=100)

5. CONCLUSIONS

In this paper, we proposed a stream clustering approach for the detection of small and major bursts in a fast stream of news. In contrast to existing methods, our method is able to detect both global and local changes and adapt the two-level hierarchy of topics accordingly. Our experiments over a real stream of news show that the distinction between local and global topics triggers changes at either local or global level and allows for local or global model adaptation, respectively. So, changes are detected, and re-clustering is applied only over the affected part of the model which is essential when dealing with data streams.

As a result, this approach reduces the overhead of re-clustering over the whole collection of documents while maintaining a good quality of (sub)topics. Our future work is on the dynamic learning of the correct number of topics and subtopics in the hierarchy over time.

6. REFERENCES

[1] C. C. Aggarwal and P. S. Yu. A framework for clustering massive text and categorical data streams. In *SDM*, 2006.

[2] L. AlSumait, D. Barbara, and C. Domeniconi. On-line LDA: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *ICDM*, 2008.

[3] A. Bifet and E. Frank. Sentiment knowledge discovery in Twitter streaming data. In *DS*, 2010.

[4] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *ICML*, 2006.

[5] C. Borgelt and A. Nurnberger. Document clustering using cluster specific term weights. In *TIR*, 2004.

[6] Y. Chi, X. Song, D. Zhou, K. Hino, and B. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. In *KDD*, 2007.

[7] A. Gohr, A. Hinneburg, R. Schult, and M. Spiliopoulou. Topic evolution in a stream of documents. In *SDM*, 2009.

[8] H. Gu, X. Xie, Q. Lv, Y. Ruan, and L. Shang. ETree: Effective and efficient event modeling for real-time social networks. In *WIC*, 2011.

[9] D. He and S. D. Parker. Topic Dynamics: An alternative model of 'bursts' in streams of topics. In *KDD*, 2010.

[10] Y.-B. Liu, J.-R. Cai, J. Yin, and A. Fu. Clustering text data streams. *Journal of Computer Science and Technology*, 23(1):112–128, 2008.

[11] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *SIGMOD*, 2010.

[12] Q. Mei and C. Zhai. Discovering Evolutionary Theme Patterns from Text - An Exploration of Temporal Text Mining. In *KDD*, 2005.

[13] R. Schult and M. Spiliopoulou. Discovering emerging topics in unlabelled text collections. In *ADBIS*, 2006.

[14] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult. MONIC – modeling and monitoring cluster transitions. In *KDD*, 2006.

[15] X. Wang and A. McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *KDD*, 2006.