ECAI 2010
H. Coelho et al. (Eds.)
IOS Press, 2010
© 2010 The authors and IOS Press. All rights reserved.
doi:10.3233/978-1-60750-606-5-517

From bursty patterns to bursty facts: The effectiveness of temporal text mining for news

Ilija Subašić and Bettina Berendt [1]

Abstract. Many document collections are by nature dynamic, evolving as the topics or events they describe change. The goal of temporal text mining is to discover bursty patterns and to identify and highlight these changes to better enable readers to track stories. Here, we focus on the news domain, where the changes revolve around novel, previously unpublished, "facts" that have an effect on the story developments. However, despite intense research activities on bursty patterns, a lack of common procedures today makes it impossible to compare methods in a principled way. To close this gap, we (a) investigate how different temporal text mining methods discover novel facts and (b) present an evaluation framework for methods assessment, consisting of a set of procedures and metrics for cross-evaluating models. Bursty patterns are transformed into queries for sentence retrieval, either with or without taking into account internal pattern structure, and these sentences are compared with a set of editor-selected ground-truth reference sentences. Our experiments on different classes of temporal text mining show that different methods perform at similar levels overall, but provide distinctive advantages in some settings. The experiments also demonstrate the benefits of using patterns' internal structure for query generation.

1 Introduction

One of the frequent use cases of today's Internet is *story tracking*: the use of search engines, archives or other sources for following the developments of a topic over time. Usually, this is done by executing a series of searches, often with the same search query such as the name of a person ("Amy Winehouse"), an event ("Tsunami", "Winter Olympics"), or a scientific area ("text mining"). The information need behind story-tracking searches differs from that behind one-off searches: search results should not only be relevant, but also novel. Novelty is generally conveyed by documents with *bursty* content elements: words, n-grams, terms highlighted by LDA distributions, or similar elements appearing significantly more frequently in a time window of search than in other times. This calls for new analysis, search and interaction methods.

In the past decade, a number of methods for story tracking have been proposed. Text-oriented versions of the story-tracking task have been described in the Document Understanding Conference (DUC) Update Summarization task [18] and in the TREC Novelty Detection task [22], and several text-summarisation or sentence retrieval methods have been applied to these tasks. These methods re-use sentences from the original documents to produce a description of the novel developments, either as a summary or as a set of retrieved sentences. The tasks are associated with a well-defined evaluation procedure suited to natural-language texts. In contrast to this, more recent methods have focused on mining for lower-level elements or patterns such as keywords or n-grams/term sequences. We refer to these approaches collectively as *temporal text mining* (TTM) [11, 16, 8]. These methods bear much promise in terms of the additional flexibility afforded by the discovery of sub-sentential patterns. However, the diversity of patterns and the absence of standardised tasks and evaluation procedures such as in DUC or TREC have rendered it basically impossible to compare their quality for the task.

To close this gap, we (a) investigate how different TTM methods discover novel or bursty "facts" and (b) present an evaluation framework for methods assessment. We concentrate on the news domain, in which most novel developments can be expressed in sentential form (e.g., "The ski slalom ended with ... winning the gold medal"). Our basic assumptions are that users construct their own description out of the patterns they are presented with and that this description is sentential, such that its quality can be assessed by the degree to which these constructed sentences (the presumed novel "facts") resemble "true" sentences. The challenge is to measure an aggregate re-construction quality over the possible/plausible fact constructions. We therefore present procedures and metrics for (i) focusing on patterns or pattern combinations that a TTM method highlights; (ii) turning these into "fact" sentences; (iii) inspecting and comparing the degree of resemblance between the "fact" and the ground-truth sentences. We do this by (i) formulating new pattern-specific ordering and combination operations, drawing on method-specific burstiness scores for the former; (ii) employing sentence retrieval methods in a new way; (iii) proposing new metrics that capture "recall" and "precision" characteristics and build on the ROUGE evaluation framework for summarisation evaluation [15]. We illustrate this cross-method evaluation method by (iv) comparing three typical TTM methods [11, 16, 23] on two corpora consisting of online news documents and an editor-selected set of ground-truth sentences.

To the best of our knowledge, this is the first systematic cross-methods evaluation framework for TTM methods. Our contributions are thus the formulation of such a framework and the demonstration of its use and of the interpretability of its results. After a brief overview of related work given by Section 2, Sections 3–6 describe the above steps (i)–(iv), and Section 7 concludes.

2 Related work

In this section, we give a short overview of related evaluation approaches and frameworks, and we explain why they are not directly applicable to our questions for comparing TTM methods.

Evaluation frameworks for the DUC update and TREC novelty tasks. Introduced in 2007 as a part of the DUC workshop series

[1] K.U. Leuven, Belgium; email: firstname.lastname@cs.kuleuven.be

[18], the evaluation framework for update summarization combines human and automatic evaluation. In this paper, we only address the automatic evaluation part. It has been shown that the automatic evaluation using ROUGE framework is highly correlated with the human evaluation [15]. The ROUGE framework measures the recall of n-grams between the human and machine summaries. Most commonly used ROUGE scores are measured on bigrams ($ROUGE.2$) and skip-4 bigrams ($ROUGE.SU4$). An evaluation framework for sentence retrieval methods has been standardized through the TREC Novelty track, which ran from 2002 until 2004 [22]. Among the different tasks of the Novelty track, the one most similar to story tracking is novel-sentence retrieval, in which the goal is to retrieve sentences previously judged as "new" by human editors.

The main problem one faces when trying to adopt one of these two frameworks for the evaluation of TTM methods, is the limited number of documents they use (10 for DUC and 25 per topic for TREC). TTM methods rely on data mining techniques which require a larger document set for pattern extraction. Another problem are the differences between the patterns that TTM methods extract. Summaries for DUC and sentences for TREC methods are standardized outputs which are directly comparable. In contrast, the output of TTM methods differs, and it is hard to compare it without an intermediate step that creates comparable representations out of the different patterns.

Temporal text mining evaluation procedures. Most of the evaluation procedures in this research field were developed to evaluate a single TTM method. Roy et al. [19] presented a method for semi-automatically detecting and naming emerging topics and compared these topics with an editor-created list. Wang and McCallum [25] modified LDA using time as one of the latent variables for bursty-pattern detection. They compared this modified LDA to standard LDA and showed differences in the distribution of bursty patterns over time. The idea is that more bursty words should have different distributions over topics in different time periods, while the less or non-bursty patterns should have more similar distributions over topics in different periods. To test the accuracy of their measures of burstiness defined on word-topic distribution, Knights et al. [12] created an artificial set by drawing words from a set of word-topic distributions. In selected periods, the words were drawn from a subset of topics, making these topics bursty. The authors measured whether their method captures this artificial burst. The STORIES method [23], which produces "story graphs", was evaluated with a user test in which participants were asked to connect bursty graph elements with sentential ground-truth events. This gave rise to precision and recall scores. In a second study, participants were given the story graphs and asked to make a true/false decisions on an event. Wang et al. [24] tested their method by comparing bursts discovered in multilingual corpora on the same topic.

Most of these evaluation procedures are tailored to evaluating only one method, assessing how well it discovers bursts in text streams. We wish to measure not only if a burst is discovered, but also whether and how this burst's representation helps users discover the ground truth sentences that created the burst.

Topic detection and tracking (TDT). In addition to the foregoing, TDT tasks such as new event detection, on-line new event detection, and story-link detection also address the same problem [2]. The TDT tasks decide whether a newly arrived document reports on an already existing story/topic or on a novel (emerging) one. However, all of the TDT tasks operate at a document level, and they only decide between "old" and "new". In contrast, we want to evaluate novelty detection at a more fine-grained level, both in terms of the syntactical units and in terms of a content characterisation.

3 Patterns, representations, and TTM groups

TTM methods output bursty patterns that point to the changes in the story they track, and the subjects arising from these changes. *Subjects* constitute the high-level story; they can be for example events (e.g., a specific ski slalom in the Winter Olympics) or topics (e.g., doping). The patterns consist of *story elements*, syntactical units extracted from the underlying documents. For example, an element could be a term, and the pattern this term plus some score assigned to it. We also define a *story representation* as a set of bursty story elements used to represent a subject. Story elements have different levels of expressiveness, i.e. different amounts of information about subjects conveyed. TTM methods operate on sub-sentence story elements, and we distinguish the following elements: tokens, n-grams, and n-gram groups. Further filters on these types are possible, giving rise to elements such as terms with above-threshold frequencies or other weights, or n-grams identified as named entities or similarly semantic entities.

Token is used in a computational-linguistics sense: a series of characters not containing any of a set of pre-defined delimiters.

N-grams are content-bearing tokens. Basic n-grams are unigrams (1-grams), where every token is a unigram. More advanced n-grams are sequences of n contiguous (or not) tokens extracted from the text. Non-consecutive, or skip-m n-grams, contain n tokens appearing in a window of m tokens. For example, in a text *[big bad wolf sleeps]*, skip2 bigrams are *[big-bad], [bad-wolf],[big-wolf], [big-sleeps], [bad-sleeps]*, and *[wolf-sleeps]*.

N-gram groups are collections of n-grams pointing to the same subject. These groups can be n-gram cluster center values, latent variables' probability distributions over n-grams, or some other way of grouping by similarity.

Every element of a story representation has some weight assigned to it. This weight can be a specific "burstiness score", a probability of an element appearing in a bursty subject, the relative importance in a bursty subject cluster center, or a weight in a latent component. While the detailed mathematical properties of these weights vary, they all induce some order of importance on the elements; we therefore regard all these weights as *burst scores* of the respective story element and use these scores in the same way in the query-generation processes that are explained in Section 4.

Based on the differences in their story representations, we divide TTM tracking methods into three groups: (a) keyword representation, (b) group representation, and (c) combo representation methods. Group (a), presented in [11, 4, 5, 6, 21], uses a list of bursty n-grams ranked by their burst scores. Group (b) [4, 25, 16, 20, 9] joins bursty n-grams into groups which point to subjects. Group (c) methods use a combination of the previous two approaches [23, 1].[2]

4 From patterns to sentential facts

For news, the "real-life" changes in a text corpus can be pinpointed using sentences. Due to differences in the expressiveness levels of story representations, comparing the patterns directly with sentences would be biased towards the patterns with sentence-like structure.

[2] Examples of representations, methods, model-specific query generation, and the corpora can be found at `https://sites.google.com/site/subasicilija/ttm-evaluation`.

Therefore, to make direct comparisons possible, we developed a process for obtaining the sentences that story elements best resemble to. This task is very akin to that of sentence retrieval, a task defined as follows: Given a query, rank sentences based on some measure of their similarity to that query. Our approach is therefore to transform the patterns into queries and then use sentence retrieval. Direct comparisons are then possible on the retrieved sentences.

The large number of developed sentences retrieval alorithms and the lack of a benchmark method for sentence retrieval in the TREC sentence-retrieval task make it difficult to decide which specific retrieval method to use. However, a detailed analysis of sentence retrieval [17] used Query-likelihood retrieval method (QL) with Jelinek-Mercer topic smoothing on a pseudo-document index of sentences as a baseline. Therefore, we consider the QL method a sensible choice for our framework. The inputs for this model are an index of pseudo-documents and a set of queries used for retrieving. An index is created from sentences of a document set, and queries are obtained from story representations of evaluated tracking methods.

Generating the pseudo-document index We first parse the complete document set, creating a set of sentences S. Then we store every sentence $s \in S$ as pseudo-document, creating an index I.[3]

Query generation for sentence retrieval For generating the queries used for sentence retrieval, we combine story elements. The combination greatly depends on the internal structure of the story representation and the relations between its story elements.

To obtain the same number of queries and to limit their length, we impose two parameters: maximum query length ($maxQ$) and maximum representation size ($maxR$). For every group, we consider two approaches. The first, *generic query generation*, is the same for all groups, and uses the top $maxR$ story elements from story representation, where the order used to determine the top elements is determined by the burst score. The second approach, *specific query generation*, takes into account the semantics of different story representations. The idea is to combine the basic story elements into more complex queries.

Specific query generation based on the output of keyword representation methods uses the following procedure. First, we extract the $maxR$ highest ranked story elements from the method's story representation. Then we combine them by creating all possible combinations not larger than $maxQ$. We rank these newly formed combination based on the average burst scores of their elements and use the top $maxR$ as queries for retrieving sentences.

Group representation methods output groups of elements describing subjects. The procedure for query generation based on the story representation with i groups is as follows. For each group, we extract $maxR/i$ story elements with the highest in-group burst score and combine them into all possible combinations not larger than $maxQ$. Then we rank the new in-group element combinations based on the average burst scores of their elements. Then, for each group, we use the top $maxR/i$ combinations as queries for retrieving sentences. We assume that all groups are equally important and use $maxR/i$ story elements from each group. If an evaluated method gives preference to some groups, this could, in future work, be incorporated by choosing more elements from the more important groups.

For combo representations, we utilize the pattern structure of the specific method to determine $maxR$ elements of size $maxQ$. For example, the STORIES method that relies on graphs is a combo method [23]. This pattern structure arises from a skip-bigram network model; combined patterns are subgraphs with at most $maxQ$ edges, scored by the edges' average burst score.

Sentence retrieval procedure For every given method $M_\bullet$ and its story representation of documents of sentences S indexed in I, we generate a query set $Q_\bullet$ based on the generic query generation procedure, and a query set $Q^*_\bullet$ for the specific query generation procedure. Then, for every $q \in Q_\bullet$ and $q^* \in Q^*_\bullet$, we retrieve the top ranked sentence using the QL retrieval method. This creates two sets of retrieved sentences R and R^*.

5 Evaluation framework

Evaluation consists of comparing the sentences obtained by the procedure described in the previous section, with a set of ground-truth sentences. This requires a corpus (news-article documents and a ground truth), measures, and a procedure including a statistical test.

Corpus We divide the document set into time periods $T = \{t_1, t_2, \ldots, t_N\}$ of equal length. For each $t \in T$, we (a) build an index I_t as described in the previous section; and (b) obtain a set of ground-truth sentences $G_t = \{g_{t1}, g_{t2}, \ldots, g_{tN}\}$, where N is the index number of ground truth in t. G is the union of all these sets.

Evaluation measures We wish to capture novelty at the sentence (fact) level, and therefore we define the performance measure on the same level. We compare a set of retrieved sentences with set of ground truth sentences in a same time frame. As atomic-level measure, we adopt $ROUGE$ metrics [15], and create a set of aggregate measures (X) to show (a) to what extent the retrieved sentences capture the ground truth ("recall-oriented" measures) and (b) how many sentences are needed to obtain a satisfactory ground-truth match ("precision-oriented" measures).

For each $t \in T$, let $R_t = \{r_{t1}, r_{t2}, \ldots, r_{tK}\}$ be the set of retrieved sentences, and $G_t = \{g_{t1}, g_{t2}, \ldots, g_{tN}\}$ the set of ground truth sentences. For every retrieved sentence r_{tk} $(1 \leq k \leq K)$, we calculate the *atomic measures* $ROUGE.2$ and $ROUGE.SU4$ scores against every ground truth sentence in the same time period: g_{tj} $(1 \leq j \leq N)$. This will give us a Cartesian product of R_t and G_t, where each element has attached scores.

Based on this, we define *aggregate measures* to quantify how well a retrieved sentence set matches ground truth set, what percentage of the best possible ground truth match is obtained from the retrieved set, and how the number of retrieved sentences influences these scores. In the following, these three measures will be explained in their form based on $ROUGE.2$; their forms based on $ROUGE.SU4$ are directly analogous.

To find the best possible match in the set of retrieved sentences, we define the $maxM$ measure as the maximum score that any retrieved sentence in a period t has for the ground truth j:

$$maxM.2_{tj} = max_{r_{tk} \in R_t} \mathit{ROUGE.2}(r_{tk}, g_{tj}). \quad (1)$$

Since the ground truth sentences do not come from the same corpus as the retrieved sentences, it is hard, if not impossible, to obtain the maximum match of 1. The maximum $maxM$ score that a ground truth can obtain, varies from one ground truth to the other. So, in order to normalize the score, we introduce the $maxMR$ measures to capture how much of the possible maximum match between the ground truth and all sentences is captured in the retrieved sentences

[3] We used the Lemur Toolkit `www.lemurproject.org`.

set. We derive the maximum $ROUGE.2$ scores between G_t and S_t, where $S_t = \{s_{t1}, s_{t2}, \ldots, s_{tH}\}$ is the set of *all* sentences from t.

$$maxMR.2_{tj} = (max_{r_{tk} \in R_t} ROUGE.2(r_{tk}, g_{tj}))/(max_{s_{th} \in S_t} ROUGE.2(s_{th}, g_{tj})). \quad (2)$$

$maxM$ and $maxMR$ measures give us "recall oriented" measures, telling us what is the proportion of the best possible match between the ground truth and the entire corpus obtained by the retrieved sentences. However, different methods may retrieve different numbers of sentences, and the ones with larger retrieved sentence set increase the chance of having a better match. We take this into account and define a new measures:

$$maxMP.2_{tj} = maxMR.2_{tj} \times min(|G_t|, |R_t|)/|R_t| \quad (3)$$

$maxMP$ rewards the methods that produce a good fit with a small number of retrieved sentences (matching the usually small number of ground-truth sentences). In this sense, the measures correspond to "top heavy precision-oriented" measures like precision@k or discounted cumulative gain.

Procedure and test In a cross-evaluation testing procedure, we compare the performance of different methods.[4] Given time periods T, methods M, ground truths G, metrics X, and a set of indices I: For every $t \in T$ and for every $M_\bullet \in M$, we calculate all three measures from the previous paragraph. This gives rise to, for every metric $x \in X$, $|M|$ sets $F_{mx} = \{f_{m11}, \ldots, f_{mtj}\}$ where j is a number of ground truths in a period t. For every x, we then test the results of different methods using Friedman's and Tukey's multiple comparison test [10].

6 Case study

To investigate our measures and how they behave in the different scenarios, we undertook a case study. We compared 3 TTM methods, one from each group discussed in Section 3, using 2 well-known news stories.

Data. We refer to the corpora consisting of news article documents and ground-truth sentences as A and B, and to the methods as M_1, M_2, and M_3. We re-used the corpora from [23], see the reference for a detailed description of sources, features and construction.

Methods. The question is how to choose the methods which both represent the groups described in Section 4 and are of high quality. We decided to use the methods that received the most attention by the community over the last few years.[5] Namely, for keyword representation we chose Kleinberg's burst detection algorithm (M_1) [11], for group representation, we used a probability mixture method developed by Mei and Zhai (M_2) [16], and for the combo group we used the STORIES method (M_3) [23].

Method M_1 [11] discovers bursty words by minimizing their state-transition cost between bursty and non-bursty states. The cost is defined as a lift in proportion of relevant document to an observed word in a data set. The method takes a scaling factor value as a parameter, which determines the lift intensity needed for word to reach the bursty state. We use the parameter value 2, as set in [11]. In the original paper, the algorithm was applied to scientific publications. The same method was later used to find bursts in blogs [13] and to track news [14]. We coded the algorithm following the description in the original paper.

An extension of probabilistic mixture method for topic discovery presented described in [26] was used in [16]. This method, M_2, outputs set of bursty topics represented by word distributions. Each of the different distributions points to a subject. The original papers reports on a number of parameters set on document-topic, topic-time, and word-topic distributions. We used the same values as reported by the authors. However, the authors report on a threshold value set by empirical testing which is not described in detail. We were therefore not able to replicate this, and manually set the threshold to 5 subjects ("topics"). We coded the algorithm employing a modification of the implementation of [26] in DRAGON.[6]

M_3 [23] defines burstiness via a skip bigram's co-occurrence frequency normalized by the count of documents. A skip bigram is bursty if this frequency in a period exceeds a lift threshold relative to the overall frequency. Skip bigrams are created by combining unigrams as "story basics". A story representation is built by joining skip bigrams into a graph ("story graph"). In the original paper, the 150 most frequent terms (not in the stopword list) were used to extract skip bigrams. We keep this settings and add another 3 choices of story basics: (1) the 150 top-tf.idf terms; (2) 100 most frequent terms plus 50 most frequent named entities (with no overlap); and (3) all terms from the document collection. We refer to the different versions of the method as M_3^{tf}, $M_3^{tf.idf}$, M_3^{ne}, and M_3^{all}, depending on the story basics used in graph creation. We modified the original code accordingly.

Procedure We re-used the period partitioning first used with these corpora. Corpus A was split into 18 periods, and corpus B into 10 periods. Corpus A contained 24 and corpus B 12 ground-truth sentences.

As discussed in Section 4, for each method we generated two sets of retrieved sentences, with generic and with specific query generation. This yields 12 sets (2 for both M_1 and M_2, and 2 for every version of M_3). We set the query-generation parameter $maxQ$ to 5, and we varied $maxR$ from 5 to 30 in increments of 5. The value of $maxQ$ was chosen based on query length in major search engines [7]. Variations of the second parameter simulate the situation in which different number of top story elements are used. The query generation process for M_1 and M_2 follows the procedure described in Section 4, while for M_3 we first extract all the paths up to size $maxQ$ from the story graph, and then sort them based on the average edge weight, following the evaluation procedure described in [23].

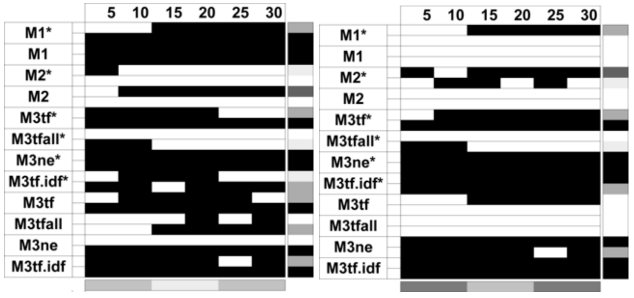
We calculated $maxM$, $maxMR$, and $maxMP$ for all 12 settings and for both $ROUGE.2$ and $ROUGE.SU4$, and tested them as described in Section 5, resulting in a total of 72 tests.

Results: Overview The large number of test settings and methods made it difficult to aggregate the results of the tests, so we created multiple aggregations. With these aggregations we aimed at capturing the following information: (a) which methods are robust to different $maxR$ settings, (b) whether there is a difference between "precision-like" and "recall-like" performance, and (c) whether specific query generation process improves retrieval.

[4] An alternative would be to compare all of them to a baseline, but such a baseline would yet need to be defined.

[5] By no means we are arguing that the chosen methods are the best among all possible methods, nor trying to diminish the quality of other work.

[6] http://dragon.ischool.drexel.edu/features.asp



(a) maxMR (b) maxMP

Figure 1. Top group method comparison matrix of $maxMR$ (a), and $maxMP$ (b) measures based on $ROUGE.2$.

Results: Test settings First, we investigated how different test settings influence the results of the methods. Figure 1 shows the results for $ROUGE.2$. The results for $ROUGE.SU4$, not shown for reasons of space, were very similar; in particular, the order of method quality was the same. We restrict ourselves to $maxMR$ (Fig. 1 (a)) and $maxMP$ (b). The reason is that the $maxM$ results were similar to $maxMR$, with the difference that $maxM$ was, as expected, due to the different numbers of ground-truth sentences less robust over different number of $maxR$. Each row shows a combination of one method and one corpus; grouped by methods to highlight similarities over corpora. The first row is for corpus A and the second for corpus B. Rows without an asterisk show the results of the method with generic query generation, rows with an asterisk show the results for specific query generation. Columns show test settings: the use of only the top $maxR$ story elements or story-elements combinations for sentence retrieval.

A cell is black if, according to the Friedman/Tukey test, this method-setting combination was in the group with the best results of the measure, i.e. there was no statistically significant difference in results between all the methods with a black cell in one column, but all of these significantly outperformed all of the others (the ones with a white cell in this column). Further significant differences between these lower-quality methods existed, but are not shown.

The "heatmap" under the table shows how much this setting differentiates between methods, ranging from low (= all methods are the same, light grey) to high (= only a small fraction of methods were in the best group, black). Differentiation is measured by the total number of white cells. The heatmap to the right of the table shows the robustness of the method quality over settings, ranging from low (= only in good group for few settings, light grey) to high (= always in good group, black). Differentiation is measured by *number of blocks - (number of white cells + number of block holes)*. Values for both heatmaps were binned into four categories of grey shades.

Figure 1 (a) and (b) show that the setting M_3^{ne} performs the best for both A and B, having only 3 settings in which they are not in the top group. Slightly less compact blocks are created by M_3^{tf} and $M_3^{tf.idf}$. This can be explained by the nature of A and B, which are both stories revolving around persons, such that tracking named entities makes more sense. Story basics in the $tf.idf$ (tf) based set overlap 72% (69%) with the named-entity set. M_3^{all} performs worse than the top group in most cases; for corpus A even in all cases. Thus, choosing story basics for M_3 has an influence on the results. For M_1, performance improves as $maxR$ rises, suggesting that capturing bursts with keyword lists needs lists of certain size. The M_2 results vary the most of all 3 methods, and they are much better for A than for B. We believe that the cause for this is the "group" story representation that M_2 uses. Each group should point to one of the events, but the number of ground-truth sentences for B is small and averages at 1.56 (A has on average 2.72 events) for a time period. This means that there more groups than subjects.

Results: "Precision" and "recall" The "precision-oriented" measure $maxMP$ shows that the settings M_3^{tf} and $M_3^{tf.idf}$ have the same results as M_3^{ne}. This suggests that the difference in overall results comes from the "recall" oriented measures. The results of M_1 show that the method performs well for the recall oriented measure $maxMR$, being in the best category in over 80% of the settings, and poorly for precision oriented measures, being in the best category in only 22% of the settings. This clearly shows that M_1 outputs patterns that are related to the ground truth, but also some that are the results of the changes in language use, vocabulary, etc. over time. For M_2, specific query generation clearly improves the results.

Results: Internal pattern structure for query generation We also investigated whether specific query generation improves the results. Observing patterns from Fig. 1 reveals that settings that use specific query generation have, on average, a larger number of black blocks for M_1 and M_3, while for M_2 specific query generation diminishes "recall" results by 20% (in block counts), while it slightly improves precision by 6.1%. However, that figure only differentiates between a method being in the top group or not. We wish to directly compare settings with and without the use of specific query generation procedure.

Figure 2 investigates the effect of query generation procedures. Each row describes the same basic method (e.g., M_3^{tf} with 150 top keywords in row 3) with or without combination, for a given corpus. Columns are the same as in Fig. 1. A cell is black if the method with combination significantly outperformed its counterpart without combination (regardless of whether any of them was in the top group or not), white if it significantly underperformed the latter, and grey if there was no significant difference.

Figure 2 shows that for the "recall" oriented measure, specific query generation does not much improve the results. It improves them in 11.1% of the settings for $maxMR$, while it diminishes them in 8.3% settings. This suggests that methods output sentences with the best match to the ground-truth sentences without specific query generation. This is not the case for the "precision-oriented" measure: for M_1 and M_3, specific query generation always improves the results. This is to be expected due to the nature of the story representation these methods use. Both the keyword representation and the group representation have little structure built into them, and the specific query generation combines the story elements so that the queries become more similar to the top weighted story elements in the story representation. More similar queries limit the number of retrieved sentences and preserve the "good" ones. As for M_3, the specific query generation never diminishes the results, and for B it improves the results in almost all settings, except for $M3_{ne}$. The difference in the performance over different corpora most likely comes from the different number of ground truth sentences. If the story graph produced by M_3 is highly connected, the generated queries are more alike resulting in retrieving same sentences. The correlation of graph topology and number of events for M_3 was presented in [3].

These results indicate that specific query generation in most cases does not affect the results. In some cases, notably for M_1 and M_2 precision metrics, the specific query generation always performs better. This indicates that the retrieval process benefits from the use of specific query generation. One possible way of improving this process for M_3 would be to take into account the similarity of already

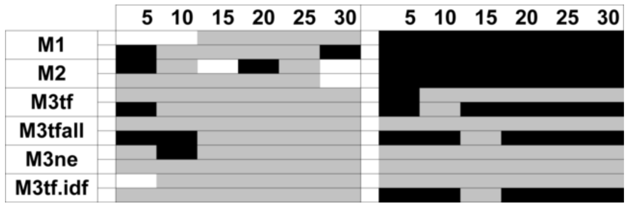


Figure 2. Query generation comparison for different test settings.

extracted queries, and in that way include more diverse queries rather than the ones with the highest weight.

Results summary. The main results of our case-study experiments are: (1) method M_3^{ne} is the most robust method to test settings changes; (2) keyword representation methods need larger $maxR$;(3) M_3 variations outperform M_1 and M_2 in "precision-oriented" measures; and (4) specific query generation improves "'precision-oriented" results, especially, for methods M_1 and M_2.

7 Conclusions and outlook

This research started with the aim of creating an evaluation procedure for evaluating different temporal text mining approaches. We concentrated on news stories, where changes over time can be discovered through "sentential" facts. An inspection of existing evaluation frameworks showed that most slightly differ from our aim of investigating how different patterns match to the ground-truth sentences. Therefore, the paper first proposed a process which connects patterns and sentences, and then evaluates the results against an editor-created ground truth. Following the experience in the similar fields we defined measures that capture both the "recall" and "precision" oriented performance of the methods. We presented a case study evaluating 3 methods over 2 corpora. The results suggest that the method presented in [23] is the most robust of all tested methods. The research also shows that using bursty patterns' internal structure for connecting them with ground truth sentences improves the results.

The evaluation framework and the query generation procedure presented here have some limitations, and overcoming them will be a challenging task for future research. First, a creation of larger, cleaner, and editor-annotated data sets comprising different stories would help avoid generalization errors. Second, we devised our performance measures based on text summarization frameworks, which are geared towards comparing longer text sequences. We miss a clear notion of how the scores generated by our measures transfer into "real-life" similarity between retrieved sentences and ground truth sentences. Therefore, we did a cross-method evaluation that, while suggesting which methods perform better than which others, cannot tell how "good" these matches really are. Thus, creating a baseline for evaluation would be an useful addition to the framework. Comparing against a baseline would simplify the testing procedure, and it would give clues as to whether extracting temporal patterns outperforms non-temporal patterns for creating story representation.

Different settings in which different TTM methods build and evaluate their results are not always compatible. Even with taking this into account, parts of our framework favour some methods in certain settings. However, we believe that any bias is distributed over the methods, not systematically favouring any of the evaluated methods. We consider this paper to be just a first step into defining a widely accepted test settings for testing TTM methods, and by no means consider our framework as the only solution for TTM evaluation. One of the messages we hope to have conveyed is that a larger effort by the community is needed to create a set of unified settings for evaluating different methods.

REFERENCES

[1] J. Allan, R. Gupta, and V. Khandelwal, 'Temporal summaries of news topics', in *SIGIR 2001*, pp. 10–18. ACM, (2001).

[2] J. Allan, *Topic Detection and Tracking*, Springer, Berlin etc., (2002).

[3] B. Berendt and I. Subasic, 'Measuring graph topology for interactive temporal event detection.', *KI*, **23**(2), 11–17, (2009).

[4] G. Pui Cheong Fung, J.X. Yu, P. S. Yu, and H. Lu, 'Parameter free bursty events detection in text streams', in *Proc. of the VLDB '05:*, pp. 181–192. VLDB Endowment, (2005).

[5] D. Gruhl, R. V. Guha, R. Kumar, J. Novak, and A. Tomkins, 'The predictive power of online chatter', In Proc. SIGKDD'05, pp. 78–87.

[6] Qi He, K. Chang, E.-P. Lim, and J. Zhang, 'Bursty feature representation for clustering text streams', in *Proc. SIAM 07*. SIAM, (2007).

[7] HitWise, Inc. Hitwise: Google's search share continues to grow. `http://www.bizreport.com/2009/05/hitwise_googles_search_share_continues_to_grow.html`.

[8] M. Hurst, 'Temporal text mining', in *AAAI Spring Symposium on Computational Approaches to Analysing Weblogs*, pp. 73-78. (2006). AAAI. `http://www.aaai.org/Papers/Symposia/Spring/2006/SS-06-03/SS06-03-015.pdf`

[9] F. A. L. Janssens, W. Glänzel, and B. De Moor, 'Dynamic hybrid clustering of bioinformatics by incorporating text mining and citation analysis', in *Proc. SIGKDD'07*, pp. 360–369. ACM, (2007).

[10] M.G. Kendall, 'Rank correlation methods', Oxford Press (1976)

[11] J. Kleinberg, 'Bursty and hierarchical structure in streams', in *Proc. SIGKDD '02*, pp. 91–101, (2002). ACM.

[12] D. Knights, M. Mozer, and N.Nicolov, 'Detecting topic drift with compound topic models' in *Proc. of the ICWSM'09*, (2009)., AAAI. `http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/191`

[13] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins, 'On the bursty evolution of blogspace', in *Proc. WWW 2003*, pp. 568–576, (2003). ACM.

[14] J. Leskovec, L. Backstrom, and J. Kleinberg, 'Meme-tracking and the dynamics of the news cycle', in *Proc. SIGKDD'09*, pp. 497–506, Paris, France, (2009). ACM.

[15] C.-Y. Lin and E. Hovy, 'Automatic evaluation of summaries using n-gram co-occurrence statistics', in *Proc. of the NAACL '03*, pp. 71–78, (2003). ACL.

[16] Q. Mei and C. Zhai, 'Discovering evolutionary theme patterns from text: an exploration of temporal text mining', In Proc. SIGKDD'05, pp. 198–207.

[17] V. Murdock and W. B. Croft, 'A translation model for sentence retrieval', in *: Proc. of the HLT '05*, pp. 684–691. ACL.

[18] National Institute of Standards and Technology. DUC 2007: Task, documents, and measures, 2007. `http://www-nlpir.nist.gov/projects/duc/duc2007/tasks.html#pilot`.

[19] S. Roy, D. Gevry, and W. M. Pottenger. Methodologies for trend detection in textual data mining, in em Proc. of the Second SIAM'02 TextMine Workshop (2002). SIAM. `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.5333`

[20] R. Schult and M. Spiliopoulou, 'Discovering emerging topics in unlabelled text collections', in *Proc. ADBIS*, pp. 353–366. Springer, (2006).

[21] D. A. Smith, 'Detecting and browsing events in unstructured text', in *Proc. SIGIR'02*, pp. 73–80. VLDB Endowment, (2002).

[22] I. Soboroff and D. Harman, 'Novelty detection: the TREC experience', in *Proc. of the HLT '05*, pp. 105–112, (2005). ACL.

[23] I. Subašić and B. Berendt, 'Discovery of interactive graphs for understanding and searching time-indexed corpora', *Knowledge and Information Systems*. DOI - 10.1007/s10115-009-0227-x

[24] X. Wang, C. Zhai, X. Hu, and R. Sproat, 'Mining correlated bursty topic patterns from coordinated text streams', in *Proc SIGKDD '07*, pp. 784–793, (2007). ACM.

[25] X. Wang and A. McCallum, 'Topics over time: a non-markov continuous-time model of topical trends', in *Proc SIGKDD '06*, pp. 424–433,ACM.

[26] C. Zhai, A. Velivelli, and B. Yu, 'A cross-collection mixture model for comparative text mining', in *In Proc. SIGKDD'04*, pp. 743–748, (2004).