Temporal Search in Document Streams

Dimitrios Kotsakos *

This PhD thesis addresses different challenges in searching temporal document sequences, where documents are created and/or edited over time, and the contents of documents are strongly time-dependent. Examples of temporal document collections are web archives, news archives, blogs, social networking platforms, and personal emails. The main focus of this dissertation is how to exploit temporal information provided in documents and combine it with textual information with the goal of improving the effectiveness of searching temporal document collections.

This summary describes the motivation and research questions addressed in the thesis. In addition, we explain our research context and methods. Our contributions to this thesis are composed of different approaches to solving the addressed research questions. In the end of this chapter, the organization of the rest of the thesis is presented.

1.1 Motivation

The ease of publishing content on social media sites brings to the Web an ever increasing amount of content captured during various types of events and/or before/after these events take place. Event content shared on social media sites such as blogs, Twitter, Facebook, YouTube, and others varies widely, ranging from planned, known occurrences such as a concert or a parade, to unplanned incidents such as an earthquake, floods or death of a celebrity. By exploring and proposing techniques to automatically identify and characterize these events and the relevant user-contributed social media documents (e.g., blog posts, photographs, videos, messages, status updates), we can enable rich search and presentation of all event content. In this dissertation we present approaches for leveraging the wealth of social media documents available on the Web for search purposes and content filtering and characterization.

In this work, we address major challenges in searching temporal document collections. In such collections, documents are created and/or edited over time. Examples of temporal document collections are web archives, news archives, blogs, personal emails and enterprise documents. Unfortunately, traditional IR approaches based on term-matching only can give unsatisfactory results when searching temporal document collections. The reason for this is twofold: the contents of documents and queries are strongly time-dependent, i.e., documents discuss events that took place at particular time periods, and a query representing an information need can be time-dependent as well, i.e., a temporal query.

---

* Dissertation Advisor: Dimitrios Gunopoulos, Professor

One problem faced when searching temporal document collections is the large number of documents possibly accumulated over time, which could result in the large number of irrelevant documents in a set of retrieved documents. Therefore, a user might have to spend more time in exploring retrieved documents in order to find documents satisfying his/her information need. A possible solution for this problem is to take into account the time dimension, i.e. extending keyword search with the creation or published date of documents.

1.2 Document Dating

During the recent years, the amount of user-contributed and digitized content on the Internet has dramatically increased, and makes web search even more challenging. Although well-known search engines (e.g. Google, Bing, etc) deliver very good results for pure keyword searches, they still do not take full advantage of the temporal dimension that characterizes most document collections.

However, in order for temporal text-containment search to give good enough and actually useful results, it is obvious that the timestamps of indexed documents have to be as accurate as possible. In the case of local document archives, trustworthy metadata that includes time of creation and last update is available. However, in the case of web search and web warehousing, having an accurate and trustworthy timestamp is a serious challenge. Another motivational example for research in the area of estimating a document's focus or creation time is that of old digitized documents or of partially failing optical character recognition applications (OCR). Moreover, a web page/document can be relocated and discovery time in this case will be very inaccurate. In some cases metadata about documents on the web can be retrieved but they can also in general not be trusted and often are simply just wrong. Thus, our research challenge is: for a given document with uncertain timestamp, can the contents of the document itself be used to determine the timestamp with a sufficient high confidence? To our knowledge, the only previous work on this topic is the work by de Jong, Rode, and Hiemstra [3], which is based on a statistic language model. In this work, we present approaches that extend the work by de Jong et al. and increases the accuracy of determined timestamps. Our main contributions are 1) a semantic-based preprocessing approach that improves the quality of timestamping, 2) extensions of the language model and incorporating more internal and external knowledge, and 3) an experimental evaluation of our proposed techniques illustrating the improved quality of our extensions.

Several related research efforts have focused on estimating a document's focus or creation time, mostly by the information retrieval community. Purely statistical methods have been proposed [1]. Other approaches have tried to deal with the problem by utilizing information from linguistic constructs with clear references to time periods or moments,

by mentioning, for example, a specific date or year. Another line of work cosiders the entire vocabulary used used in a document in order to reason about when it was created [2]. Kanhabua and Nørvåg in [3, 4] propose a document-dating method that extends the one proposed by De Jong et al. Specifically, the authors propose the application of semantic-based preprocessing of the reference collection, and apply a term-weighting scheme based on their previous work on temporal entropy [3]. The authors further enhance their approach by considering search statistics from Google Zeitgeist.

A serious drawback and disadvantage of most proposed methods, that is being addressed in this dissertation, is that most methods initially pre-segment the timeline of study into intervals of the same fixed length (e.g. a week) and afterwards choose the interval that is most likely to be the temporal origin of the query document, by comparing its vocabulary with the model built for each of the candidate intervals. The drawback of this approach is obvious: it limits the choices of possible time intervals.

1.2.1 Problem Definition

1.2.2 Preliminaries

The problem we address here is defined in the context of a collection of documents $\mathcal{D}$, spanning a timeline of $Y = t_1, t_2, ..., t_n$ of n distinct timestamps. We define a function $t(d)$ to return the timestamp of a given document $d \in \mathcal{D}$. Given a query document $q \notin \mathcal{D}$, for which the timestamp $t(q)$ is unknown, our goal is to find the best possible interval of size ℓ, $I = t_i, ..., t_{\ell+i}, 1 \leq i, j \leq n$ within T, so that $t(q)$ most likley falls within I. We refer to $\mathcal{D}$ as the *reference corpus*

Among other things, our approach considers the *burstiness* of the terms in the query document q. Given a term $x \in q$, we use $\mathcal{B}(x, \mathcal{D})$ to represent the set of non-overlapping bursty intervals for x, as computed over the given corpus $\mathcal{D}$. Each bursty interval is defined within the timeline T spanned by $\mathcal{D}$. In addition, we define $s(b)$ to return the burstiness score of a given bursty interval $b \in \mathcal{B}(x, \mathcal{D})$. Since we are only considering a single corpus, we henceforth refer to $\mathcal{B}(x, \mathcal{D})$ simply as $\mathcal{B}(x)$.

In order to evaluate the effectiveness of our method, we compare its precision against state of the art methods. Given the variety of our datasets as well as their uniform distribution of the time periods examined, we believe that the ability to place a document within a desired timeframe, in other words the precision we achieve, to be the most accurate valuation. At a high-level, the problem can be defined as follows:

Problem 1 [Document Dating]: *Let $\mathcal{D}$ be a collection of documents spanning a timeline of $T = t_1, t_2, ..., t_n$ of n discrete timestamps (e.g. days). Each document $d \in \mathcal{D}$ is associated with exactly one timestamp from T. Let $q \notin \mathcal{D}$ be a query document for which*

the timestamp is unknown. Then, we want to find we want to find the smallest possible timeframe within T during which the document was written.

The definition of the problem assumes that the query document was written within the timeline spanned by the corpus $\mathcal{D}$. No constraints are placed on the size or nature of $\mathcal{D}$. We observe that the presence of such a reference-corpus is necessary, otherwise it would be impossible to arbitrarily assign a timestamp to q.

1.2.3 Our approach

In this section, we introduce our algorithm for the Document Dating problem. As mentioned in the introduction of this chapter, our approach considers (i) the lexical similarity of the query document with the documents in the reference collection $\mathcal{D}$ (ii) the burstiness of the significant terms of the query document q, e.g., top-k terms ranked by *tf-idf*.

The use of lexical similarity captures the intuition that similar documents are more likely to discuss similar topics and events, and are thus more likely to originate in the same timeframe. In practice, however, similar documents may appear on different timestamps across the timeline. We address this, by introducing term burstiness. When an event or topic is recorded in a textual corpus, its characteristic terms exhibit atypically high frequencies. We refer to these timeframes as *bursty intervals*. Our algorithm is orthogonal to the actual mechanism used for computing non-overlapping bursty intervals. By identifying the bursty intervals of different terms, we can identify the timeframe of relevant events, as well as relevant documents that discuss them.

Figure 1.2 illustrates the architecture of our approach. In this section we describe our steps in detail.

We are given a query document q, discussing a disastrous fire in the Jackson theater in Chicago. The figure shows the 7 most lexically similar documents to q: $d_1, d_2, d_3, d_4, d_5, d_6$ and d_7 . Each of these documents has a subset of the following four terms in common with q: *Fire, Chicago, Jackson, Disaster*. The figure shows the bursty intervals for each of these terms. In this example, there are three visible sets of neighboring documents: $\{d_1\}$, $\{d_3, d_4, d_5\}$ and $\{d_5, d_6, d_7\}$. The documents in the second set overlap with multiple bursty intervals from the four characteristic terms, and are thus more likely to discuss the the actual event. Therefore, our approach will report the interval that starts with the first document on the (d_2) and ends with the last one (d_4) as the most likely timeframe for the query document.

We refer to our algorithm as `BurstySimDater`. The pseudocode is given in Algorithm 3.1.

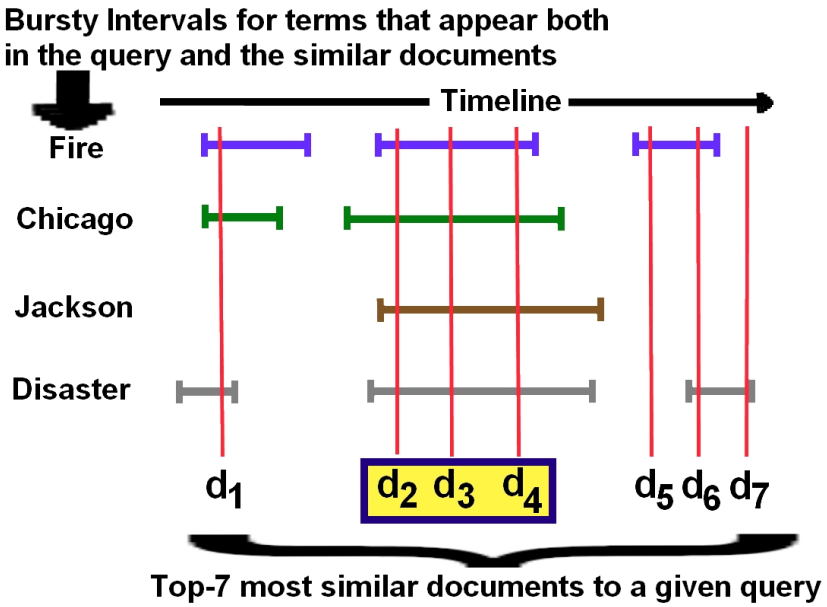

Figure 1.1: An example of how `BurstySimDater` identifies the appropriate timestamp for a given query document. In this case, the three documents d_2, d_3, d_4 will be selected by our algorithm, since they are both close to each other and overlap with multiple bursty intervals of the considered terms.

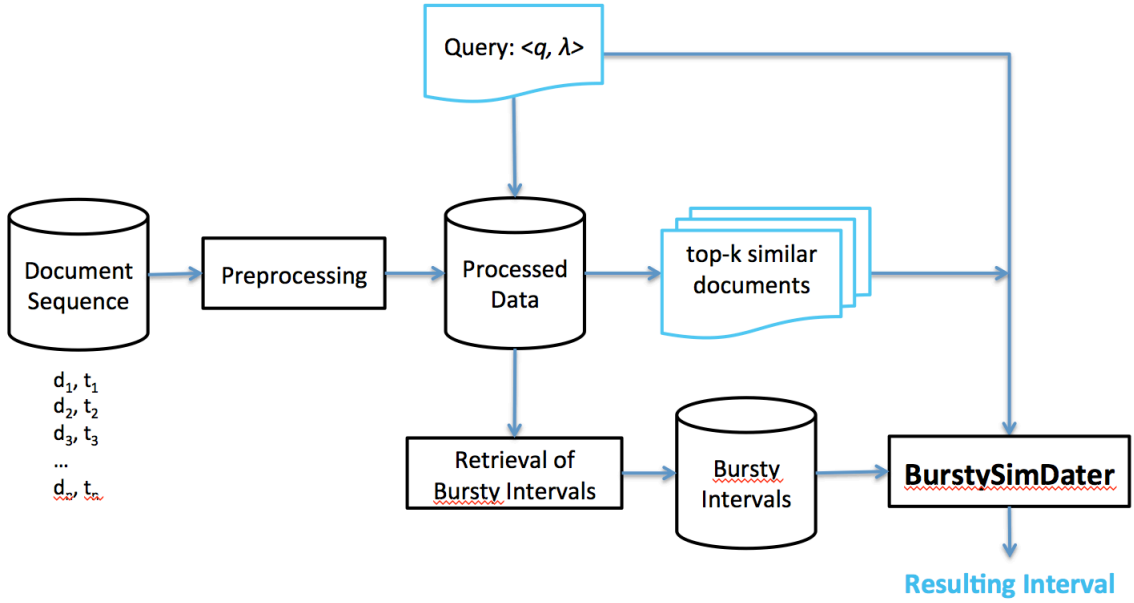

Figure 1.2: The architecture of our approach

The input to the algorithm consists of the query document q, the reference corpus $\mathcal{D}$, the

Algorithm 1 `BurstySimDater`

Input: reference corpus $\mathcal{D}$, bursty intervals $\mathcal{B}$, query document q, max timeframe length ℓ
Output: timeframe of q

1: $\mathcal{S} \leftarrow$ top-k most similar documents to q from $\mathcal{D}$
2: $W_{\mathcal{S}} \leftarrow \emptyset$
3: **for** $d \in \mathcal{S}$ **do**
4: $\quad w_d \leftarrow 0$
5: $\quad \mathcal{Y} \leftarrow d \cap q$
6: $\quad$ **for** $x \in \mathcal{Y}$ **do**
7: $\quad\quad w_d \leftarrow w_d + |\{I \in \mathcal{B}(x) : t(d) \in I\}|$
8: $\quad w_d \leftarrow w_d / |\mathcal{Y}|$
9: $\quad W_{\mathcal{S}} \leftarrow W_{\mathcal{S}} \cup \{w_d\}$
10: $A_{\mathcal{S}} \leftarrow (d \in \mathcal{S}, W_{\mathcal{S}})$
11: $\mathcal{I} \leftarrow \texttt{GetMax}(A_S, \ell)$
12: **Return** $\mathcal{I}$

set of precomputed bursty intervals $\mathcal{B}$ and the upper bound on the reported timeframe ℓ. The output is an interval of length at most ℓ, within the timeline T spanned by $\mathcal{D}$.

First, the algorithm retrieves the top-k most similar documents to q from $\mathcal{D}$. In our own evaluation, we experimented, among others, with the *tf-idf* measure and the Jaccard similarity. We use the latter in our experiments, since it led to the best results. We refer to the retrieved set of the k most similar documents as $\mathcal{S}$.

In steps 2-9, we assign a weight w_d to each document in $d \in \mathcal{S}$, based on its overlap with the burstiness patterns of its terms. Initially, w_d is set to zero. Let $\mathcal{Y}$ be the overlap of d's vocabulary with the vocabulary of the query document q. For each term $x \in \mathcal{Y}$, let $\mathcal{B}(x)$ be the precomputed set of bursty intervals for x. We then increment w_d by the number of the intervals from $\mathcal{B}(x)$ that actually contain $t(d)$. After the iteration over all terms in $\mathcal{Y}$ is complete, we normalize w_d by dividing it by $|\mathcal{Y}|$. Conceptually, the weight w_d of a document d is the average number of bursty intervals that it overlaps with, computed over all the terms that it has in common with the query q. The computed weights are kept in the set $\mathcal{W}_{\mathcal{S}}$.

We want to identify the interval when the most terms from the top-k similar documents are *simultaneously* bursty. This period is the intersection of intervals with the maximum sum of weights. To do this, in steps 10-11, we create an array $A_{\mathcal{S}}$ of size T, where cell i equals to the sum of weights w_d for all documents $d \in \mathcal{S}$ that were written at t_i. Next, we find the interval $\mathcal{I}$ of length ℓ with the maximum sum. By tuning ℓ, we tune the level of desired accuracy. In order to compute the sets of bursty intervals we use the `GetMax` algorithm [6]. Given a discrete time series of frequency measurements, `GetMax` returns a set of non-overlapping bursty intervals with respect to the frequencies.

A *burst* on the timeline is marked whenever the popularity of a specific term dramatically and unexpectedly increases. In order to compute the sets of bursty intervals we use the `GetMax` algorithm, introduced in [5]. Given a discrete time series of frequency measurements for a given term, `GetMax` returns a set of non-overlapping bursty intervals with respect to the number of frequency measurements.

This chapter reviews the literature on the document timestamping problem and proposes a new approach for document dating that overcomes the drawbacks of previous methods: it doesn't depend on temporal linguistic constructs and it can report timeframes of arbitrary length. The proposed method outperforms the previous state-of-the-art in precision and computational efficiency in most of the cases, while being the most versatile of all, since it performs well for many reporting intervals and throughout a variety of datasets. This is achieved by taking into consideration the burstiness of the terms and lexical similarity of testing documents with the timestamped training corpus. An extensive experimental evaluation on real datasets demonstrated the efficacy of the algorithm and its advantage over the state of the art.

1.3 Memes and Events

As a motivational example, consider the part of the homepage of many social media sites that is devoted to *Trending Topics*. Most modern platforms like Twitter, Yahoo!, Facebook, etc. offer such a functionality, where different types of algorithms are used to identify the most popular topics in the platform during a current time window. *Trending Topics* lists may include popular items people search for in an e-commerce site like Amazon.com, trending queries in a web search engine like Google.com, popular topics of interest people write about in a micro-blogging platform like Twitter.com or trending tags people use to annotate their blog posts in a blogging site like Wordpress.com. Most of the items that appear in this lists have been caused by real-life events that triggered the interest of the users and they wrote or posted about them in the social media. They main functionality of the *Trending Topics* lists is that of facilitating search and discovery of new content. However, not all trending items are related to real life events. A significant percentage of popular content has become strategically popular, especially within microblogging environments like Twitter and Tumblr, where fan- and sports-related communities thrive and dominate the usage of the media. Thus, social media platforms and search engines would benefit from better understanding *why* a specific item became popular, and offer a variety of landing pages for different types of content. Specifically, Figure 1.3 illustrates the trending topics on December 28, 2014 in Twitter and Yahoo! respectively. A user clicking on a news-related trending topic would expect to land on a page with a series of news articles, maybe chronologically ordered, describing the timeline and the current state of that specific topic. On the other hand, a user clicking *'iPhone 6 Plus'*, which is a consumer product would expect to find offers, technical specs or reviews of the item. Last, users interested in a celebirty named *Nash*, would expect to find fanpages, photos and videos of the celebrity when clicking on `#HappyBirthdayNash` trending topic.

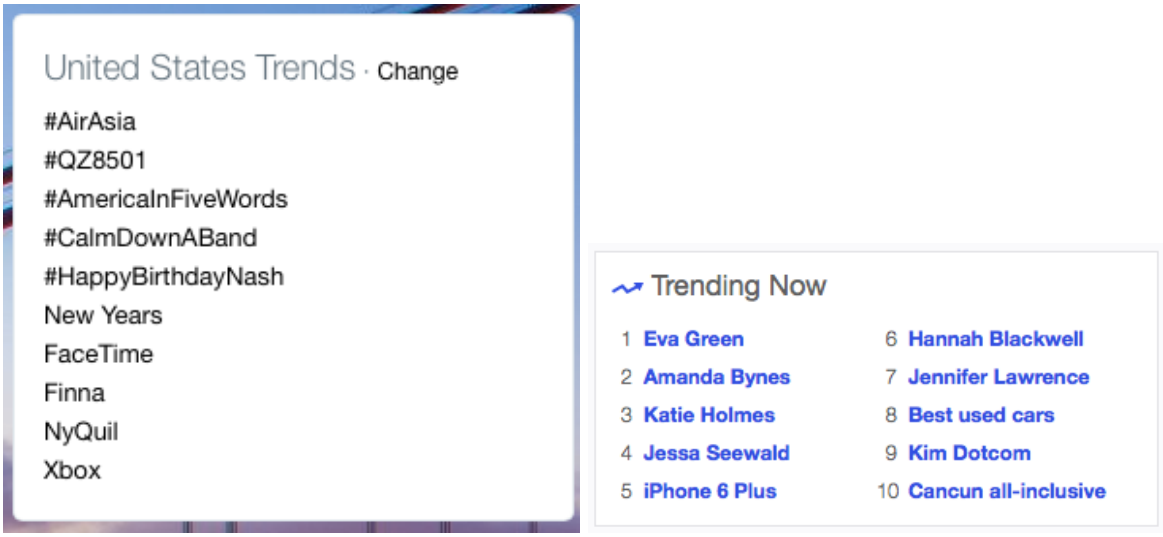

Figure 1.3: Trending Topics in Twitter.com and Yahoo.com on *December 28, 2014*

For our analysis of trending topics, we specifically focus on one social media site, namely, Twitter, due to its transient, large-scale publicly available content. In particular, we collected a vast amount of Twitter messages from various locations posted during various periods of time and focus on the analysis of the most popular hashtags. We show that event-detection methods can not only be based on the detection of terms and phrases that exhibit trending behavior on Twitter, characterized by an unusual increase in message frequency during a particular time period in a Twitter message stream. While some of these trends might refer to actual real-world events, others might include non-event information, triggered by strategically planned advertising campaigns or by user communities trying to promote themselves. Unlike related efforts in this area, which focused on characterizing or analyzing content from individual events on Twitter, or characterizing aggregate trend characteristics for manually identified terms, the features we use in this study in order to discriminate between the various classes of content can be used for more than the two classes that are defined in this dissertation.

Overall, we show that social media sites contain substantial, useful information about different types of popular content that can be exploited and utilized in order to provide more effective and useful services to users of social media sites. With the features we propose in this dissertation, we can effectively identify different types of popular content and their associated social media documents across various social media sites. Regardless of the classifier we use, the type of event/meme, or the social media site, any single popular topic might have hundreds or thousands of associated social media documents. While some of these associated documents might contain interesting and useful information (e.g., event time and location in case of events, participants and opinions in case of memes), others might provide little value (e.g., using heavy slang, incomprehensible language without ac-

companying media) to people interested in learning about an event or meme. Techniques for effective selection of quality event content may then help improve applications such as event browsing and search. Therefore, we propose a noise-filtering mechanism for selecting a subset of the social media documents associated with the significant real-world events.

1.3.1 Hashtag-Based Event Detection: A Proof of Concept Use Case of Meme-Filtering

In order to show the utility of the proposed methodology we applied a hashtag-based event detection approach to our data. As mentioned in the introduction, due to the limited text length of tweets, hashtag analysis is a common approach for micro-blogs mining. For example, most event detection methods in social media rely on time-series analysis of hashtags, inspecting terms that appear bursty for specific time-periods or generally popular terms. The assumption is that the bursty keywords/hashtags will be related to emerging events.

In this section, we argue that these event detection methods can lead to mixed results, since memes are also popular and bursty. In fact, memes appear to have a very well defined popularity period, just like events, so time-series approaches will fail distinguishing one from the other. We applied a burstiness algorithm for event detection in order to study the insufficiency of this type of approach. At a high level, a time-frame is considered bursty if the term exhibits atypically high frequencies for its duration. Bursts in terms of frequency capture the trends in vocabulary usage during each corresponding time-frame and can thus prove useful in event detection. When an event takes place in real life (e.g. an earthquake, sports finals), the event's characteristic terms (e.g. *earthquake, shooting, overtime*) appear more frequently in social media. Unfortunately, memes demonstrate a similar behaviour.

1.3.2 Burstiness Results

In our experiment we split the *Germany* dataset in two sets, one including months *April, June, July* and *August* which served as the training set and one including only *September* which was our testing set. Table 1.1 lists some bursty intervals computation examples along with a short description for the corresponding hashtags. The last column shows the classification result when using meme filtering with the Random Forest classifier, trained over labeled data from the first four months of the dataset. It is apparent that while the bursty intervals computed by `GetMax` algorithm precisely match the actual dates of excessive popularity of the corresponding hashtags, it is not enough to reason about significant real life events that affected the Twitter community. Hence `GetMax` identifies memes and events.

Table 1.1: Bursty Intervals for popular hashtags in *Germany* during September, 2014

hashtag	Bursty Intervals	Description	Meme Filtering
#ff	Sep 5, 12, 19, 25	"Follow Friday" Twitter meme	meme
#eaie2014	Sep 16 - Sep 19	Conference held in Prague during Sep 16 - Sep 19	event
#jaykingslandto60k	Sep 11 - Sep 12	Bot account posting thousands of tweets	meme
#nominateavrillavigne	Sep 11, 15	Celebrity fan campaign	meme
#h96hsv	Sep 14	Soccer match: Hannover 96 vs. Hamburger SV	event
#iphone6	Sep 9, Sep 19	Announcement and release of iPhone 6	event
#iphone6plus	Sep 9-10, 19	Announcement and release of iPhone 6 Plus	event

A closer look at Fig. 1.4 reveals an even further similarity between the different types of popular hashtags in terms of behavior in time. On *Friday, September 19* four hashtags exhibit similarly bursty behavior, being simultaneously and unexpectedly popular. Two of them, namely #iPhone6 and #iphone6Plus, correspond to the event of the release of the new iPhones, #eaia2014 is the hashtag used to annotate discussions and reports from the European Association for International Education held in Prague, while the #ff is a viral Twitter meme with the aim of suggesting people for other users to follow. While the reader would argue that the distinction between an event and a meme in this case is rather trivial, since #ff is a periodically popular hashtag, this is not the case with hashtags like #nominateavrillavigne that have similarly bursty behavior, but only not periodic.

In this chapter we defined the problem of distinguishing a popular topic of interest in a social network between network-generated topics of discussion, denoted as *Memes* and real-life events that triggered the interest of the social network users, denoted as *Events*. We provided a detailed study of the features that affect the classification, applying our experiments on the Twitter network using two different real-life datasets with 27.8 and 6.8 million tweets each and 1.1 million and 491,043 unique hashtags respectively. We evaluated multiple classification methods, among of which the Random Forest classifier performed always best, having been able to reach an accuracy of 89% in its prediction on whether a topic is a *meme* or an *event*. Our study reveals interesting characteristics of the two classes of hashtags, some expected and some not. To demonstrate the utility

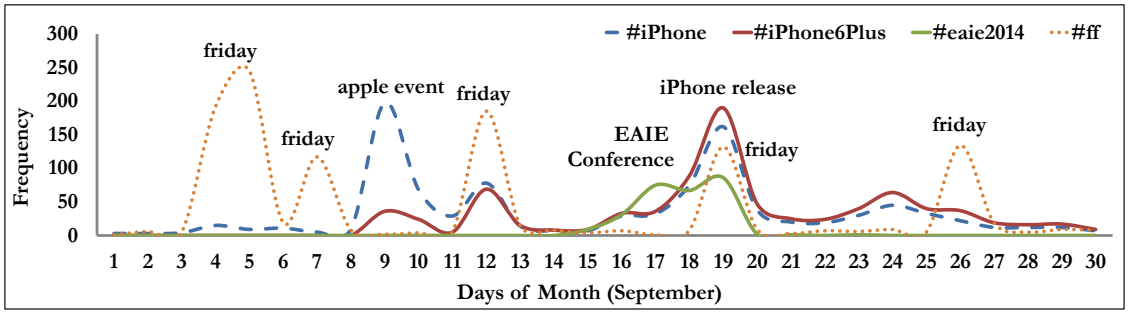

Figure 1.4: Frequency curves of popular hashtags of various kinds as they appeared in *Germany* during *September, 2014*

of our approach we enhance a hash-tag based event detection with meme-filtering and comment on the improved results.

1.4 Contributions and roadmap of this thesis

This dissertation addresses research problems in searching temporal document collections. We have proposed different approaches to solving the addressed research questions. In summary, the contributions of this thesis are:

- We exploited term burstiness in order to detect events in social media document streams.
- We proposed a state of the art technique for for determining the creation time of non-timestamped documents. The proposed approach outperforms the methods of the relevant literature. We improved the quality of document dating by incorporating term burstiness information and textual similarity methods into the algorithm. By conducting extensive experiments, we showed the evaluation of our proposed approach and the improvement over the baseline.
- We formally defined the difference between memes and events in social media and thoroughly examined the differences between the two different types of popular content along various descriptive characteristics, proposing a set of features to aid the classification of various types of content. We showed the usefulness of our method via a burstiness-based event detection approach.

BIBLIOGRAPHY

[1] Nathanael Chambers. Labeling documents with timestamps: Learning from their time expressions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 98–106, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[2] F.M.G. de Jong, H. Rode, and D. Hiemstra. Temporal language models for the disclosure of historical text. In *Humanities, computers and cultural heritage: Proceedings of the XVIth International Conference of the Association for History and Computing (AHC 2005)*, pages 161–168, Amsterdam, The Netherlands, September 2005. Royal Netherlands Academy of Arts and Sciences. Imported from EWI/DB PMS [db-utwente:inpr:0000003683].

[3] N. Kanhabua and K. Nørvåg. Improving temporal language models for determining time of non-timestamped documents. *Research and Advanced Technology for Digital Libraries*, pages 358–370, 2008.

[4] Nattiya Kanhabua and Kjetil Nørvåg. Using temporal language models for document dating. In *ECML/PKDD (2)*, pages 738–741, 2009.

[5] T. Lappas, B. Arai, M. Platakis, D. Kotsakos, and D. Gunopulos. On burstiness-aware search for document sequences. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 477–486. ACM, 2009.

[6] Theodoros Lappas, Benjamin Arai, Manolis Platakis, Dimitrios Kotsakos, and Dimitrios Gunopulos. On burstiness-aware search for document sequences. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 477–486. ACM, 2009.