Identifying Evolutionary Topic Temporal Patterns Based on Bursty Phrase Clustering

Yixuan LIU† Zihao GAO‡ and Mizuho IWAIHARA‡
†早稲田大学大学院情報生産システム研究科
‡808-0135 福岡県北九州市若松区ひびきの 1-15-4212
E-mail: †liuyixuan@ruri.waseda.jp, ‡gao_jihao@suou.waseda.jp, ‡iwaihara@waseda.jp

Abstract Temporal text mining has wide applications in many domains, such as topic trajectories and summarization of opinion trends online. In Wikipedia, one of the most popular websites, article topics cover quite broad and also highly specific themes. This paper studies on a particular temporal text mining task: finding evolutionary patterns of topics from a collection of articles. Topics in Wikipedia articles grow and fade over time. In order to reveal the evolution of topics in one category, we propose a time series clustering method based on burstiness of phrases in article revisions, where additions and deletions are both considered. Our work presents a novel way to simplify phrase time series by burst detection，thus simplified phrase addition and deletion time series are attained. As for clustering, we use dynamic time warping to measure the distance between phrases, in addition to classical Euclidean distance. Our experiments provide reasonable results showing that the proposed method can identify interesting evolutionary topic temporal patterns effectively. Research so far can be applied in diverse domains like finding temporal association between key phrases, discovering emerging topics, detecting synonym, and so on.

Keyword Topic evolution, Temporal pattern, Burst detection, DTW, Clustering

1 Introduction

Over the past decade, numerous online collaboration systems have appeared and thrived on the Internet. Prime examples include Wikipedia, Linux, Yahoo! Answers, Mechanical Turk-based systems [1]. They enlist a large number of people to supply information and solve problems. Users become the main strength of contributors. More development of such systems will be witnessed in the near future.

It is important to note that these platforms have brought a large amount of information. Wikipedia, one of the most popular websites, is a free, multilingual Internet encyclopedia written collaboratively by volunteers around the world [9]. Users who browse in it can also edit articles. Each edit version will be saved online and all the edit history is available to public. Therefore, article topics cover quite broad and also highly specific themes, and also topics are changing and evolving as time goes by.

As a large-scale repository of structured knowledge and edit history, Wikipedia has become a valuable resource for temporal text mining work [8]. Actually, a collection of Wikipedia articles in one category have more than one topics. Although the temporal patterns of articles are not obvious, there often exist temporal patterns in these evolutionary patterns. As we can see in Figure 1, the edit history of “Obama” in Wikipedia article “Barack Obama” has a huge burst since the 18th week from the beginning of revisions. If patterns can be found, it will be much easier for us to find the reasons and relationship behind these patterns and evolution.

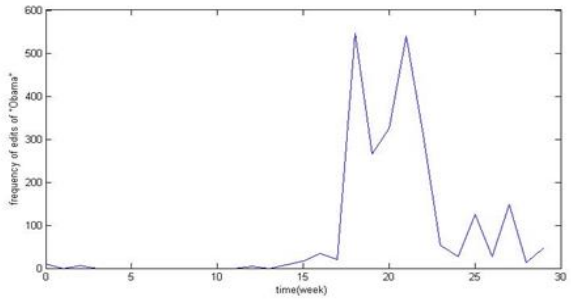

Figure 1 the edit trend of Obama

In this paper, we investigate the problem of temporal text mining and time series simplification. The goal is revealing the evolutionary patterns of topics in one whole category in Wikipedia. Each output cluster contains key phrases that refer to real topics. Topics in one cluster share similar temporal patterns in edit history. One cluster may contain multiple events which are related each other.

Usually people try to identify topic patterns from semantic level, such as word2vector and so on. Actually the semantic method performs well at this field. However, few work finish this job from temporal angle. Unlike traditional text clustering works, we solve the problem

with temporal information, by collecting the frequency and time point of each key phrase candidate, we avoid the semantic fuzzy problem. According to time series, the phrases in one cluster share similar temporal patterns, which semantic method cannot find. What is more, we combine the addition and deletion information to track the dynamic situation of a phrase, instead of just quantity statistics, which is never implemented in previous work.

In order to solve the problems above, we propose a novel clustering method for bursty phrases. More specifically:

(1) We utilize burst detection technologies to detect burst phrases in a collection of articles of edit history. The time series of edit additions and deletions are attained, where only important timestamps and burst levels are labeled. Thus we dump the minor details and the time series are simplified by burst detection.

(2) We use k-means to cluster key phrases, by similarities on temporal patterns. Dynamic time warping (DTW) [7] is applied in time series distance measuring, where temporal similarities reflect factors of edit history, such as bursts, additions and deletions.

The rest of the paper is organized as follows. Section 2 covers related work on term burst detection and temporal clustering. Section 3, 4 and 5 presents our proposed method on bursty phrase clustering. Section 3 presents the algorithm of selecting historically significant phrases. Section 4 abstracts time series of bursty phrases. Our clustering method with DTW is demonstrated in Section 5. In Section 6, comparative experimental results are presented, and finally, Section 7 provides our conclusions and further research.

2 Related work

Kalogeratos et al.[5] proposed an algorithm to improving text clustering algorithm by term co-burstiness. The algorithm first focuses on tern burstiness and co-burstiness, then creates a bursty term correlation graph. Then it partitions the graph into k groups and compute a synthetic prototypes of each group. Next, it reduces the number of clusters by an agglomerative approach that merges similar groups based on cosine similarity of prototypes. Finally, spherical k-means is used to obtain the final clusters with the initialization of the synthetic prototypes. This method introduces bursts and inter-burst relationships into the traditional text clustering. Temporal information is directly used and the algorithm performs well.

The fundamental problem in [5] is that it is irrespective of bursts of edits. Kleinberg [6] proposed a burst model for extracting meaningful structures from document streams. The work develops a model to identify bursts that imposes a hierarchical structure over all the streams. A burst is defined as a rapid increase of a term's frequency of occurrences. If the term frequency is encountered at an unusual high rate, then the term is labeled as 'bursty'. The work is used to identify bursts in text stream and produce state labels of bursts.

3 Selecting historically significant phrases

To find evolution of topics in a Wikipedia category, we need to monitor a relatively long period of editing time. There are often more than one topic in an article, thus the first step of our work is to extract typical topics in articles. We utilize key phrases to represent candidate topics. According to Kleinberg [6], the larger the number of articles containing the candidate phrases, the larger the probability of detecting it as bursty. Obviously the most important and shifty articles in Wikipedia are edited more times, therefore, key phrases are more likely to be shifty and could be identified as bursty phrases. Then what we need to do is observing the time series of these phrases and abstracting them by burst detection. Additions and deletions need to be distinguished. Dynamic time warping method is used to measure the distance between time series of key phrases. The DTW distance will replace classical Euclidean distance in k-means clustering method according to Jang [4], and capture aforementioned features of edit history. Finally clusters and centroids are obtained, where similar temporal patterns are kept together and more detailed semantic relationships can be discovered by these evolution patterns.

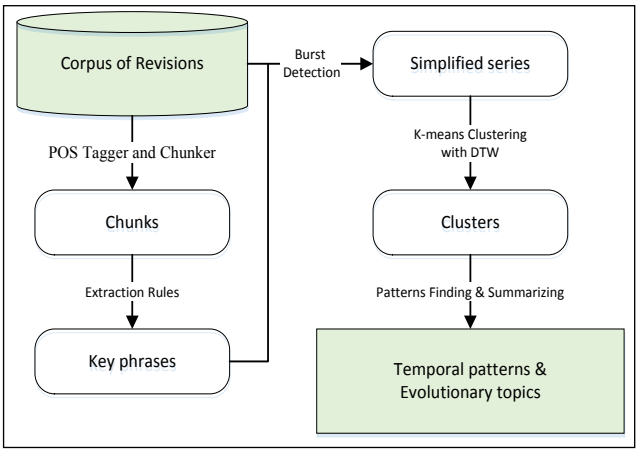

Figure 2

All the procedures are shown in the flow diagram in .

Flow diagram of Bursty Phrase Clustering Method

In this section, we discuss about abstracting bursty phrase time series problem. The clustering algorithm will be discussed in Section 4.

3.1 POS Tagging and filtering

In order to cluster shifty topics, we first detect key phrases that could represent topics in one article. In this task, POS (part of speech) Tagger and Chunker provided by Apache openNLP [11] is used to apply POS tagging and chunking function upon the revisions of Wikipedia articles. In each revision, sentences are divided into chunks and tagged by POS, like noun groups, verb groups and so on.

The POS tagger produces various POS labels to each chunk, such as: NN(Noun, singular or mass), NNS(Noun, plural), NNP(Proper noun, singular), and NNPS(Proper noun, plural). Then we consider following features to extract our candidate phrases.

3.2 Decay phrase frequency

Next we introduce a number of temporal features to refine candidate phrases.

Revisions of one article are created over time, where various phrases are added or deleted, where *phrase frequencies*, namely the times one phrase occur in one revision, dynamically change over time. Temporal changes of phrases capture the evolution of topics and articles. Traditional TF-IDF does not capture such temporal factors of document revisions. Revert of a revision, which is to discard the latest revision and restore an older revision, sometimes happens due to a variety of reasons, but its main cause is vandalism, which appears as a total or massive deletion of contents. Since massive deletions discontinue topic evolution, we remove such deletions from our history analysis.

In order to give weights to phrases through the whole edit history, one way is to compute the term frequency with a decay factor, like Aji [3]. However, their model ignores revision quality, and decay factor is applied based on revision counts, instead of time interval. In real life, some revisions can be poorly edited and revised soon, while a burst of revisions can occur in a short time period due to frequent revises. Long-lived revisions are accepted by editors as trustworthy and high-quality through the edit history. On the other hand, short-lived revisions are likely to be often insignificant, erroneous, or having content overlapping due to unsupervised repeated additions. We should give a higher weight to phrases appearing in long-lived revisions.

Let $t(r)$ denote the time point of revision r, and we utilize the timespan the revision lived to indicate the quality of r. We propose our timespan-weighted phrase frequency over history as:

$$PFH = \frac{f(p, r_i)}{i^{\rho}} \cdot \frac{t(r_{i+1}) - t(r_i)}{t(r_n) - t(r_1)} \quad (1)$$

Here, $\rho > 0$ is a decay factor, and r_i is the i-th revision of the article, i=1,..,N.

IDF (inverse document frequency) of a phrase is widely used for measuring uniqueness of the phrase in the corpus. However, since we focus on detecting phrases that are bursting in most revisions of articles, we do not consider IDF. We can evaluate how widely a term occurs in the considered article set as below:

$$rf(p, D) = \frac{|\{r \in D : p \in r\}| + 1}{N} \quad (2)$$

Here, N is the total number of articles in the corpus D, $|\{r \in D : p \in r\}|$ is the number of revisions where the term p occurs. In case of the phrase is not in the corpus, we plus one on numerator so that *rf* has no chance to be zero.

3.3 Survival rate

When a phrase is added at a revision, it will be checked and sometimes revised by editors. If a phrase survives through a long sequence of revisions, it is an indicator of good quality. Such long surviving phrases can be regarded as significant, and they are useful for labeling bursting edit activities. We measure the number of phrase remain times to value the quality of phrase, since editors would delete poor quality texts.

We define the *survival rate* of a phrase p in an article a as:

$$SR(p) = e^{\frac{scale(p)}{|R|}} \cdot \frac{contain(p)}{scale(p)} \quad (3)$$

Here, *scale* (p) is the number of revisions in a from the first time p appears to the last time, and *contain*(p) is the number of a's revisions containing p. The first part measures how many revisions the phrase survived over the history, and the second part measures how long the phrase appears without interruptions during its lifespan. The survival rate is independent from text length. Phrases having long and non-interrupting lifespans are highly

scored by the survival rate.

Combining the features we proposed so far, we define the weight function for selecting historically-significant phrases:

$$W_{p_i} = PFH(p_i, R_x) \cdot SR(p_i) \cdot rf(p, R_x) \quad (4)$$

4 Abstracting time series of bursty phrases

4.1 Time series modeling

Let *S* be a corpus, which is a sequence of target articles $s_1, ..., s_n$. The corpus can be chosen based on certain topics, such as from a Wikipedia category. Each article has a sequence of revisions, which are considered as text stream:

$$S = [s_1, ...s_n] \quad (5)$$

$$s_i = [r_{11}, ...r_{mn}] \quad (6)$$

Here, *S* is the corpus, s_i represents the sequence of revisions of the *i*-th article in *S*. The *j*-th revision of s_i is denoted as *rij*.

The *revision frequency* of a phrase is the times the phrase occurs in one revision. As new revisions are created over time, certain phrases increase their revision frequency, while others keep stable for a time span, or decrease, and sometimes fluctuate. The revision frequencies of the same phrase can be diverse between articles, but we can expect bursting and simultaneous increase of revision frequencies on multiple articles, due to a significant real world event that triggers edits of these articles. To detect edit activities, we should focus only on changes of revision frequency, which is caused by additions and deletions of a phrase. The frequency difference between two revisions is given by:

$$\delta(p, r_i) = pf(p, r_i) - pf(p, r_{i-1}), \quad (7)$$

where, *p* is a phrase and *pf* is the revision frequency of phrase *p* in one article.

When an event happens, a number of revisions will be created on one article, because each editor repeatedly improves the article, and multiple editors could try to describe the same event. Also, there can be multiple articles related to the event. In our burst modeling, we apply a decay function to consider lingering effects from preceding edits.

For adding edits, where $\delta(p, r_i) > 0$:

To keep the encyclopedia clear, editors tend to reduce repeated words. When new event happens, editors tend to add new contents to the related articles, the first editor will add the most content, which cover main event. The later editor will supplement less contents. The effect of added contents in the early revision will last for a few revision and decreases as time goes by. We define the phrase frequency added in *ri* will have an adding effect *af* to revision *rj*:

$$af(p, r_i, r_j) = \begin{cases} \delta(p, r_i) \cdot e^{-\lambda_1 (j-i)}, \delta(p, r_i) > 0 and j > i \\ 0, \delta(p, r_i) \le 0 or j \le i \end{cases} \quad (8)$$

For deleting edits, where $\delta(p, r_i) < 0$:

Negative frequency added into article means revise to previous revisions, which indicate that there was some errors in previous revisions. Usually these errors would stay for several revisions. Such Incorrect frequency should be removed, thus the phrase frequency deleted in *ri* will have an deleting effect *df* to revision *rj*:

$$df(p, r_i, r_j) = \begin{cases} \delta(p, r_i) \cdot e^{-\lambda_2 (i-j)}, \delta(p, r_i) < 0 and j < i \\ 0, \delta(p, r_i) \ge 0 or j \ge i \end{cases} \quad (9)$$

Therefore, the time series model is defined as follows:

$$PF(p, S) = [Pf(p, r_{11}), ...Pf(p, r_{nm})], where$$

$$Pf(p, r_j) = \delta(t, r_j) + \sum_{i=1}^{n} [af(p, r_i, r_j) + df(p, r_i, r_j)] \quad (10)$$

Where r_{ij} is the *j-th* revision in *i-th* revision in certain category.

4.2 Burst detection

Now the time series of each phrase is defined. However, the revisions are too many to find rules or patterns. Therefore, a simplified method is considered on time series—burst detection.

The goal of burst detections to develop a model and the "bursts" can be identified robustly and efficiently. According to [6], the rapid increase of a phrase's frequency of appearance, defines a phrase burst in the text stream. The bursts are identified with multiple states, where rate depends on the current state.

First time series $PF(p, S) = [Pf(p, r_{11}), ...Pf(p, r_{nm})]$ are split in T non-overlapping time windows:

$$PF(p,S)=[Pf(p,R_{11}),...Pf(p,R_{t_0 j_0})], where$$
$$Pf(p,R_{pj})=\delta(p,R_{pj})+\sum_{i=1}^{n}[af(p,R_{pi},R_{pj})+df(p,R_{pi},R_{pj})] \quad (11)$$

In this equation, R_p is a batch of revisions arriving at the *p*-th window, and R_{pj} denotes the revisions of *p-th* batch which contain the phrase p_j. The time window is the time span divided by finite number, in other word, is time unit. Usually we take one day or one week as a time unit.

In burst detection model [5], each (R_p, R_{pj}) can be seen as an output symbol that is produced probabilistically according to the internal state of a hidden Markov model. The whole model considers a cost of passing in more burst states. The goal of it is to find the minimal cost for each state. The cost of a state sequence $q=(q_{i1},...q_{in})$ in finite-state automaton is defined as follows:

$$\sigma(i,|R_{pj}|,|R_p|)=-\ln[\binom{|R_{pj}|}{|R_p|}p_i^{|R_{pj}|}(1-|p_i|)^{|R_{pj}|-|R_p|}] \quad (12)$$

Here, p is the probability of a state change, $|R_p|$ is the number of revisions in the article at the *p-th* window, and $|R_{pj}|$ denotes the number of revisions of *p-th* batch which contains the phrase p_j.

With this burst detection model the state transition sequence that minimizes the sot function is attained. The life span of the burst is a distance from first and last burst appearance moment. Using this model, we ignore small spikes and abnormal phenomenon and focus on common phrase frequency trend.

4.3 Time series abstraction

Now the time series are detected by Kleinberg model, we can just simplify the time series with burst and timestamp. Then only state-transition point are recorded:

$$TS_j=[(s_1,x_1),...(s_n,x_n)] \quad (13)$$

Where s_i is the state of x_i time point to phrase *j*.

According to phrase extraction and burst detection model we simplify phrase frequency time series, which can be applied in time series mining domain.

5 Clustering by DTW

5.1 Dynamic time warping measure

In order to uncover the temporal dynamics of topics in category in Wikipedia, we use clustering method. It allows us to find diverse temporal patterns. However, the classic Euclidean distance metric of traditional clustering method is not suitable enough for times series distance. That is because the topic bursts are sparse and random. Sequences vary in time and different in length. For example, in

, two series are quite matched in patterns but different in time span. In Euclidean metric, the distance is too large. How to align the peaks? The time scaling and normalization work, but the overall volume of distinct time series is too diverse to be directly compared. Therefore, we introduce dynamic time warping (DTW) to measure the similarity between two sequences.

DTW is implemented using dynamic programming. [2] By forming local cost matrix where each element denotes the distance between two points with corresponding indices of two sequences, the method aligns two sequences

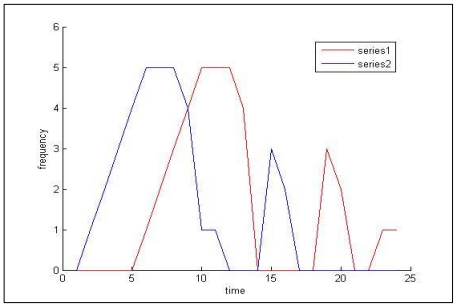

Figure 3

together. Let $d(i,j)$ be the local distance cost associated with the i-th component of sequence TS_1 and the j-th component of TS_2. Then the distance measure in local cost matrix is as follows:

$$d(i,j)=\mathrm{dis}((s_{i1},x_{i1}),(s_{j2},x_{j2}))=\sqrt{(s_{i1}-s_{i2})^2+(x_{j1}-x_{j2})^2} \quad (14)$$

Then the warping path is constructed by building a global cost matrix *D(i, j)*:

$$D(i,j)=d(i,j)+M_{ij} \quad (15)$$

Where M_{ij} is the minimal value of three elements.

$$M_{ij}=\min[D_{i-1,j},D_{i,j},D_{i,j-1}] \quad (16)$$

Therefore, the global distance matrix can be formed:

$$D = \begin{bmatrix} 0 & \infty & \dots & \infty \\ \infty & d_{1,1} + M_{2,2} & \dots & d_{1,n} + M_{2,n+1} \\ \infty & \dots & \dots & \dots \\ \infty & d_{m,1} + M_{m+1,2} & \dots & d_{m,n} + M_{m+1,n+1} \end{bmatrix} \quad (17)$$

In [2], as we can see, using DTW method, the peaks are aligned under optimal procedure. After that the distance is calculated. Though there is time delay, the distance is still small, that is what we want: focus on temporal patterns and forms instead of exact time point. While by the Euclidean metric (Figure 4a), the time series distance is larger because of the time lag.

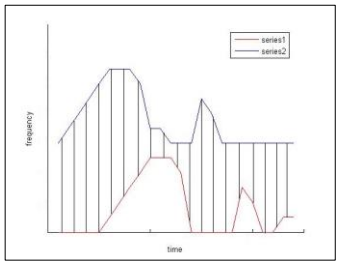

Figure 5a

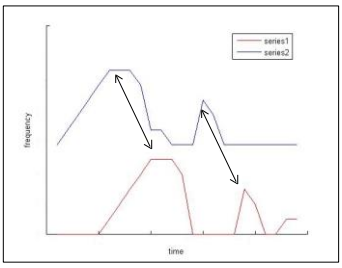

Figure 4b

Each similarity between pair of sequences can be measured in DTW now. The distance part is calculated by 2d coordinates instead of value in sequence with unified time interval, because the sequences are sparse, unifying time axis does not work out.

5.2 Improving K-means clustering

In classical k-means clustering, the method accomplishes work in two steps. [10] The first one is assignment step: it selects initial centroids randomly and assigns each item to the closest cluster. Next is refinement step, the cluster centroids are recalculated by current situation and the items are reassigned to the cluster closest to them. The second step is repeated until the sum of the Euclidean distances between the items inside one cluster minimize.

However, Euclidean metric is not suitable for sequence distance measuring. So DTW metric is introduced. Now the Goal become minimizing the function *F*:

$$F = \sum_{k=1}^{K} \sum_{x \in C_k} d_{DTW}(Pf_i, \mu_k)^2 \quad (18)$$

Where μ_k is the centroid of cluster *k*, C_k represents the cluster *k*.

6 Experiments

In order to analyze the temporal patterns of content in Wikipedia, we focus on several themes as categories. Usually the entertainment shows and politic events include enough contents, with useful information and involving widely fields. Therefore, these themes will be selected as experiment samples.

In our experiments, we select 10 different categories. In each category, articles are filtered by these criteria: English language, text volume is more than one page, having more than 200 revisions. After filtering, we manually review the articles and remove articles with repeat content or irrelevant contents. Finally, in each category 10 articles are filtered as datasets. As shown below. Then we retrieve all the revisions of each article.

Since the data is too big to present, we just show one category as an example and explain it. In this category we chose 10 articles: Barack Obama, Donald Trump, Democratic Party (United States), Donald Trump presidential campaign, 2016, Hillary Clinton, Republican Party (United States), United States Presidential election, 2016, United States Presidential election, 2012, United States Presidential election, 2008. The revisions of these articles are sorted by timeline and then processed by POS tagging and filtering strategy. 100 key phrases candidates are picked out by the key phrase extraction step, and phrase frequency on each time point to responding phrase Now we could process the datasets by our method in this paper.

We first consider the POS tagging and filtering problem. When detecting key phrases that could represent topics in one article, POS (part of speech) Tagger and Chunker provided by Apache openNLP [11] is used to apply POS tagging and chunking function upon the revisions of Wikipedia articles.

Since articles contain quite a large number of chunks, but not all of them are useful, we reduce the amount of calculation, by filtering chunks through linguistic features. As we know, the most frequent but least useful POS in an article is preposition. We remove prepositions, as well as adjective and adverbs from the articles. There are also stop words and phrases, such as “let’s”, “for example”, “December”, which are also removed from candidate phrases. After the filtering step, the majority of remaining chunks are nouns, followed by verbs and others.

After collecting phrases of revisions, we utilize the method in this paper to extract key phrases and cluster them according to time series.

The results are shown as Figure 6 temporal patterns of

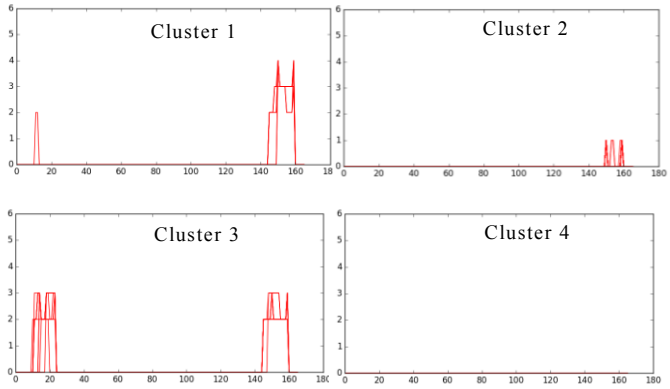

Figure 6 temporal patterns of evolutionary topics

evolutionary topics

Since the edits of a phrase of one day are too small, we choose “week” as a time unit. There are totally 167 weeks from Aug.1st, 2012 to Oct.6th, 2015. We test many times and 4 clusters are observed. The burst detection step helps us to remove most of spikes and noises so that onl y notable burst are remarked. In cluster 1, there is a small burst at October 2012, and main burst is at 2015 June to August. In this cluster, most key phrases are American news post, such as Washington post, Huffington post and so on. That is because in 2012 and 2016, political news emerges under the circumstances of presidential elections. Cluster 2 only has 3 topic peaks, most phrases are sudden events which we cannot predict though it may happen before, for example, an illegal immigrant. In 2015 Sept, a new policy for immigration is practiced, therefore the burst of this topic is obvious. Situations like this are many: ISIS, Syria, Mac Cain and so on. Cluster 3 is similar to cluster 1, but they are different. In cluster 3, two main bursts share the same level of bursts while cluster 1 does not. In cluster 3, most phrases are like “Trump”, “Rand Paul” and “Ted Cruz”. From 2012 October to 2013 January, the presidential election of US brought extensive influence and debate, which is displayed on Wikipedia edit history. Ted Cruz became famous for his Obamacare All-Nighter lasting 21 hours. Then in 2016, he joins the election again and the phease bursts at 2015 summer. In cluster 4, there is no obvious bursts, most phrases in this cluster are “actors”, “campaign”, “crime” and so on. These phrases displayed important role and are quite frequent in articles. Additions and deletions go with the revision editing all the time. But these edits are fluctuations, not bursts.

We try to do the experiments without bursts or additions and deletions, the results are as follows:

(1) Clusters without burst detection.

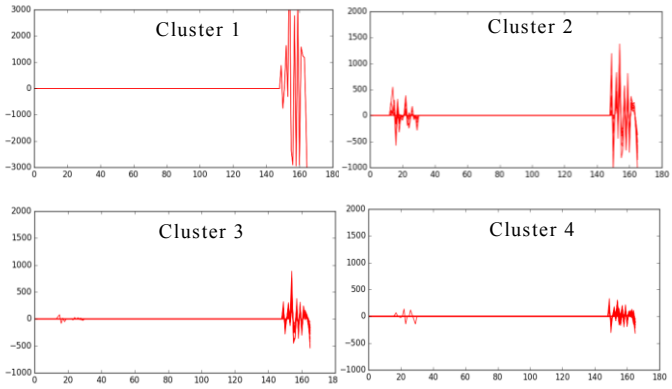

Figure 7 Temporal patterns without burst detection

In Figure 7, we still try to divide the datasets into 4 clusters, but the results are not obvious. Only 11 phrases are clustered into cluster 1 and cluster 4. The number of phrases in middle clusters is too many, which reduces the uniqueness of each cluster. The “news post” phrases distribute in every cluster. We cannot tell the difference between cluster 2 and 3.

That is because the frequency range is large so that details are neglected. The time series are only the frequency delta of each phrase, which leads to spikes and noises. Therefore temporal pattern of each topic is difficult to summarize.

(2) Clusters without dynamic time warping distance calculation.

In Figure 8, we calculate the distance between time series by Euclidean method, this time we cannot even distinguish temporal patterns and only 3 clusters are divided. Two clusters are similar and the other is zero. Only 14 phrases are clustered into cluster 1 and 3. The last 86 phrases are in cluster 2. It is difficult for us to find the topics in so many phrases.

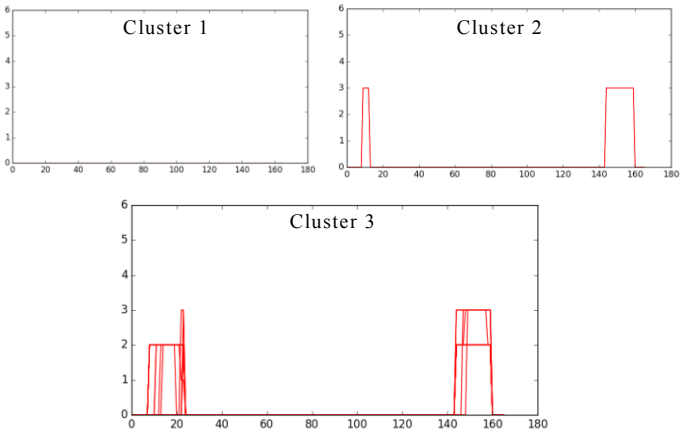

Figure 8 temporal patterns with Euclidean metric

This is because the addition and deletion delta is very small in time series, direct subtraction would make further

effect on reducing the difference, instead of expanding it. By DTW, we focus on the similar fragments of two sequences, which is proofed better in this experiment.

All in all we can see that we find different temporal patterns of topics by our method. These phrases are similar in semantic level, we cannot distinguish them from this angle. However, from the perspective of temporal information, we can find their similarities and differences. The reasons behind these patterns are multiple, some come from the common political events, and some come from famous person or festivals. The method helps us to identify these features and the rules behind these clusters.

7 Conclusion and future work

In this work, we proposed a novel approach for using bursty phrase information in text clustering. By the use of edit history we discover multiple evolutionary temporal patterns of topic in certain category in Wikipedia. The bursts of phrases are regarded as the way to simplify phrase time series. In addition to burst detection we cluster time series into multiple clusters, each of them represents a kind of evolutionary pattern of hot topics. To avoid the difference of phrase frequency, we consider the "change" of each phrase, that is, we calculate the addition and deletion of phrase frequency.

In this work we investigate the time series simplification and temporal patterns clustering. A novel way is proposed to simplify the time series, burst detection makes us focus on only burst part of time series. What is more, interesting evolutionary temporal patterns are identified, which helps us discover the semantic relationship from a temporal perspective. Phrases in one cluster share similar temporal patterns, which results from common events or festivals. These reasons are waiting for discovering.

Our experiments performs well. The reasonable results show that the algorithm can identify interesting temporal patterns effectively. Research so far can be applied in diverse domains like finding temporal association between key phrases, discovering emerging topics, detecting synonym and so on. Further work will focus on the clustering methods. DTW is also considered to be improved in order to stress the burst overlapping between time series.

8 Reference

[1] A. Doan, R. Ramakrishnan, and A. Y. Halevy. 2011. Crowdsourcing systems on the World-Wide Web. Commun. ACM 54, 4, 86-96, 2011. I. Tanaka and J. Suzuki, "Web and Database Technologies", Proc. of ACM SIGMOD, pp. 10-22, 2010.

[2] Adwan S, Arof H. On improving Dynamic Time Warping for pattern matching [J]. Measurement, 2012, 45(6): 1609-1620.

[3] Aji A, Wang Y, Agichtein E, et al. Using the past to score the present: Extending term weighting models through revision history analysis[C]//Proceedings of the 19th ACM international conference on Information and knowledge management. ACM, 2010: 629-638.

[4] Jang M, Han M S, Kim J, et al. Dynamic time warping-based k-means clustering for accelerometer-based handwriting recognition[M]//Developing Concepts in Applied Intelligence. Springer Berlin Heidelberg, 2011: 21-26.

[5] Kalogeratos A, Zagorisios P, Likas A. Improving Text Stream Clustering using Term Burstiness and Co-burstiness[C]//Proceedings of the 9th Hellenic Conference on Artificial Intelligence. ACM, 2016: 16.

[6] Kleinberg J. Bursty and hierarchical structure in streams[J]. Data Mining and Knowledge Discovery, 2003, 7(4): 373-397.

[7] Müller M. Dynamic time warping[J]. Information retrieval for music and motion, 2007: 69-84.

[8] Press A. Wikipedia and Artificial Intelligence: An Evolving Synergy[J].

[9] Wikipedia: http://en.wikipedia.org/wiki/Wikipedia

[10] Yang J, Leskovec J. Patterns of temporal variation in online media[C]//Proceedings of the fourth ACM international conference on Web search and data mining. ACM, 2011: 177-186.

[11] http://opennlp.apache.org/