Query Based Timelines Generation in News Archives

Yi YU†, Adam JATOWT†, and Masatoshi YOSHIKAWA†

† Graduate School of Informatics, Kyoto University
Yoshida-honmach, Sakyo-ku, Kyoto, 606–8501, Japan
E-mail: yuyi@db.soc.i.kyoto-u.ac.jp, adam@dl.kuis.kyoto-u.ac.jp, yoshikawa@i.kyoto-u.ac.jp

Abstract This research tackles the problem of timeline generation. With the rapid growth in the number of digital archives, timeline generation plays an important role in information retrieval that helps users quickly locate their information needs. However, most of the previous works focus on the generation of a single timeline. The result of single timeline creation is the summary of the collection of timestamped news articles, which does not take into account the notion of information diversity. Thus, we present a novel timeline generation framework that extracts events relevant to a query from a collection, and places these events into different timelines such that timelines are mutually different as much as possible. After generating multiple timelines, we also rank them based on their utility scores.

Key words Timeline, Diversification, Information Retrieval, Affinity Propagation

1 Introduction

Over the past decades, along with the wide use of the World Wide Web, a increasing number of archives have been digitized. Digital News Archives System can help users, including historians, journalists, and general users, to find news articles conveniently. Given a document collection, a user issues a query to the system, then the search system returns the ranked results to him/her according to the relevance between documents and queries. If the number of results returned is huge, users likely get lost in the tons of documents. Timeline summarization, to some extent, can help alleviate this problem. Timeline summarization is the task to automatically generate evolutionary summaries of documents in chronological order of events. An example case is shown in Figure 1, where the timeline is manually created by The Guardian [1].

At first look, timeline summarization is quite close to the task of multi-document summarization (MDS). The difference is that the former one makes good use of temporal information of the documents while MDS does not. Basically, one can create a timeline by first clustering documents with respect to certain dates, then extracting/abstracting the summaries of the clusters. After obtaining summaries of each cluster, then one can put them in chronological order to show the result. In this way, a timeline of a collection of documents is created. This is the intuition of timeline generation. Based on this idea, many sophisticated models [3,4] have been proposed, aiming to produce a much more coherent and precise storyline to describe events. Nevertheless, our work is not focusing on improving such performances of the timeline.

BP oil spill Timeline
2010.4.20 Explosion and fire on the BP-licensed Transocean drilling rig Deepwater Horizon in the Gulf of Mexico. Eleven people are reported missing and approximately 17 injured. A blowout preventer, intended to prevent release of crude oil, failed to activate.
2010.4.22 Deepwater Horizon rig sinks in 5,000ft of water. Reports of a five-mile-long oil slick. Search-and-rescue operations by the US National Response Team begin.
2010.4.24 Oil is found to be leaking from the well. A homeland security report on critical infrastructure says the problem has “no near-term impact to regional or national crude oil or natural gas supplies.”
2010.4.25 US coast guard remote underwater cameras report the well is leaking 1,000 barrels of crude oil per day (bpd). It approves a plan for remote underwater vehicles to activate a blowout preventer and stop the leak..
2010.4.26 BP’s shares fall 2% amid fears that the cost of cleanup and legal claims will hit the London-based company hard. Roughly 15,000 gallons of dispersants and 21,000ft of containment boom are placed at the spill site.
...

Figure 1 : manually created timeline for BP oil spill.

In this paper, we present a novel framework of timeline summarization, called “multiple timelines summarization” (MTS). Taking a general query (e.g.Japan) issued by users and the returned collection as input, the system automatically outputs several timelines with summaries that represent different aspects of the query. For instance, a user inputs the search query “Japan”, the system should return him/her several timelines rather than a single timeline, because the query “Japan” is such a vague term that it contains many aspects representing different intents of the user, e.g., the sports of Japan, the history of Japan, the politics of Japan and so on. Next, we will illustrate the reasons why we think of MTS better than traditional timeline summarization.

The inspiration of MTS comes from the concept of information diversity. To our understanding, information diversity can be interpreted from two different standpoints. For example, since the search engine cannot know the intents of users without prior knowledge, in order to meet most users' search needs, the utilization of diversification should be considered. Besides, the undeniable fact says that not every user has the ability to construct a qualified search query when performing searches. Therefore, users usually perform searches by first giving a vague query to search engine, only after they get returned results from the system, then they will know what they want and how to construct detailed queries. That's our understanding of the concept of information diversity. Our framework aims to provide comprehensive knowledge by returning multiple timelines representing various possible aspects of the query.

What's more, another benefit of our framework is that the highly comprehensive search results can help enhance users' understanding of event evolution. As we can see from Figure 1, it's quite clear for users to grasp the evolution of the BP oil spill from the start point,i.e., 2010.4.20 to now. It benefits users, especially those who want to get a quick overview of events because it can greatly reduce the time they spend in browsing unnecessary information.

The rest of the paper is organized as follows. We first retrospect the related work in Section 2, then in Section 3, We introduce the problem definition. We explore the significant features of MTS framework in Section 4, and in Section 5 we illustrate our proposed model based on Affinity Propagation algorithm. Finally, in Section 6, we conclude our current work and future work.

2 Related Work

In this part, we will review previous works from two kinds of researches, namely, Timeline Generation and Search Result Diversification.

Timeline Generation : Timeline is a kind of tool to visualize search results according to the chronological order of events. [2] explores the utility of timeline representation in the information system. Then, the early studies [3,5] share the similar approach of extracting the most relevant sentences from the document collection, then listing the clusters in the chronological order. [6] proposed a framework called "Evolutionary Timeline Summarization", which considers the properties of Relevance, Coverage, Coherence and Diversity of timeline. Based on the above constraints, the author gives a balanced framework by an optimization algorithm. Also, timeline generation system can be divided to two classes,i.e., extractive and abstractive methods. Most of them are the former that choosing important sentences with high weights from the collection and use them as labels. Recently, abstractive timeline generation approaches have been proposed [4,7]. Instead of extracting sentences, they use graph-based algorithms to produce sentences as the summary. However, all of these models are to generate single timeline. By contrast, our task is to generate multiple timelines so as to realize the diversity. The most similar framework to our work is [8], which creates a map consisting of events. In this model, the system generates an information metro map , where multiple timelines are correlated showing the relation and interaction between different events.

Search Result Diversification : Search Result Diversification is a task to make search results different from the previous results to realize the high coverage of search intents of users. In the probabilistic ranking model, the results are ranked by the relevance between documents and the queries, while the diversity of search results alleviates the consequences derived by information redundancy. In 1988, [9] first put forward the notion of Maximal Marginal Relevance (MMR). They rank documents according to the relevance as well as the information diversity. Then the works [10,11,12] have proposed the methods to explicitly diversify search results. They use intents or categories as external knowledge to compare the similarity between the documents and deem the ratio of uncovered intents or categories as the degree of novelty. [13] gives a definition of Search Result Diversification from three aspects, that are content-based definition, novelty-based definition, and coverage-based definition. Nevertheless, in our case, the unit of diversification we exploit is no longer a document but a timeline.

To best of our knowledge, the combination of timeline generation and diversification is a novel idea that previously little work has been done. Although [6] explicitly mentioned the conception of diversity, they regard the diversity as the difference in date. Our framework is to produce many isolated timelines that maximize the inter-timeline difference and minimize the intra-timeline difference.

3 Problem Formulation

We give a formal definition of the MTS task as follows:

Input: Given a general query $q = \{w_1, w_2, ..., w_{|q|}\}$ from users, where w_i is a term of query, and a timestamped document collection D from which results are to be collected.

Output: Returned a timeline collection $\tau = \{T_1, T_2, ..., T_{|\tau|}\}$, where T_i consists of a series of correlated events, i.e., $T_i = \{e_1, e_2, ..., e_{|T_i|}\}$ and the generated timelines are mutually different yet relevant to q. Here, e is the important event related to q and it is essentially a clustering of document that describes the similar topics. Also, the timelines are ranked according to their utility score.

4 Features

In this section, we will explore the features of MTS framework.

Coherence. Timeline shouldn't be just a list of clustering of documents. Ideally, it should have a good performance in showing the evolution of the same topics. For example, we know there are various famous people named "Michael Jordan", such as an American basketball player, or a Machine Learning scientist. For the query "Michael Jordan", different timelines standing for different persons can be shown. The events which happened to the athlete Michael Jordan shouldn't show in the timeline of scientist Michael Jordan. Otherwise, it results in meaningless timelines. The events that make up the timeline should be highly coherent in terms of narrative.

Relevance. Generated timelines should be relevant to the query as much as possible. According to our study, it's often the case that the search system returns many irrelevant documents when we search for news articles. Thus, the feature of Relevance guarantees that the timeline does not deviate too much from the user's needs. In the document search system, the Okapi BM25 [14] is often adopted to rank documents balancing relevance and novelty. Users can adapt the search strategy according to their individual search preferences. In the study of timeline generation, there is no golden rule to measure the relevance of the timeline to the query. ROUGE score is generally chosen to evaluate the goodness of the timeline. In our work, we regard the mean average of document relevance as the timeline relevance.

Novelty. Based on the idea of diversification of search results, a timeline should provide new information as much as possible. In other words, the feature of Novelty is reflected in the difference between timelines, which can be expressed by calculating the distance between two timelines. As each timeline is the set of events, We can calculate the cosine similarity between the events, and then use the average of values as the distance between timelines.

Salience. Salience or Importance, of a timeline, is interpreted as how many events it covers among all the events. The timeline with a high Salience score could be deemed as a good instance. On the contrary, the timeline with only one or two events is regarded as a bad one. For simplicity, we calculate the value of Salience as the number of events in a timeline divided by the total number of events.

Uncertainty. Last but not least, unlike other frameworks, Uncertainty is a unique feature of MTS. It means that the number of timelines we will generate as well as the number of events of each timeline are uncertain. Given a certain number of events, we cannot arrange them in a single timeline, otherwise, the information would be seriously messy. On the other hand, we cannot either put the events to that many timelines. If we do so, the coherence and salience of the timeline would be very low. Therefore, the algorithm which we use should be able to resolve the property of Uncertainty.

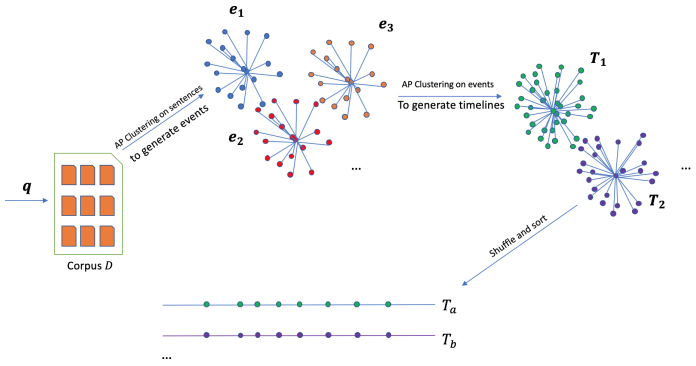

Figure 2 : Overview of our proposed model.

To sum it up, the characteristics we've listed above are the key factors to determine how well the multi-timeline summarization system performs. Besides, one thing should be noted that these features are not necessarily independent. For example, the salience to some extent could affect the coherence of the timeline.

5 Methodology

Taking all factors into consideration, we will adapt the Affinity Propagation algorithm to our framework. We describe our model as the three-step approach and give an overview of the system in figure 2.

Affinity Propagation

Affinity Propagation (AP) is a clustering algorithm based on the concept of "message passing" between data points. Unlike other clustering algorithms, AP algorithm doesn't need to manually determine the number of clusters in advance. The basic idea of AP algorithm is to treat all samples as nodes of the network, then calculate the clustering center of each sample through the message passing of each edge in the network [15]. In the process of clustering, there are two kinds of messages transmitted among nodes, namely, responsibility and availability, representing the trend of others choosing candidates as an exemplar and the attractiveness of the candidate, respectively. By iterating these messages many times and the algorithm stops when they converge into constant values. Inspired by the work [7], we find AP algorithm is effective and performs well in clustering sentences to generate events. What's more, another vital reason we choose AP algorithm is that AP algorithm can automatically determine the number of clusters, which coincides with the feature of

Uncertainty we mentioned in Section 4. Specifically, if tons of articles are returned according to the query, perhaps dozens of or even hundreds of timelines can be produced possibly. If the query leads to a small number of articles returned, little timelines should be returned.

The First Step. In this step, because the clustering process is similar to the work in [7], we don't need to change too many. The input is document collection and query, and output is the clusters of sentences. Similarity we considered not only lies in the cosine similarity between two sentences, but also take the similarity on temporal information. The intuition is that two sentences are very likely to describe the same event when they mention the same date. For example, when we talk about the 2020 Summer, or 2020 July, it's natural for us to think about the event of 2020 Tokyo Olympics. Thus, as for the similarity matrix in AP clustering, we make small modification that similarity is computed by the combination of cosine similarity and date similarity, that is

$$s(i,j) \leftarrow \gamma cosine(i,j) + (1-\gamma) date\_sim(i,j) \qquad (1)$$

, where $s(i,j)$ means the similarity between sentence i and sentence j.

The Second Step. In this step, the input is events collection, and output are the clusters of events. Although it seems highly similar to the last step, it indeed has a completely different meaning. For example, in the first step clustering, the intuition is that two sentences that have high cosine similarity and similar dates are describing the same event. By this means, we can call the output cluster of sentences a "event". However, in second step clustering, that's not the case. Think about a question : do two events describing different entities perform a good storyline? The answer is of course no. For instance, event A : Alice graduated from Oxford University when she was eighteen ; event B : Bob graduated from Oxford University when he was eighteen. They should not be in the same timeline even if these two events are highly similar because they describe very different named entities. Thus, considering this fact into consideration, in order to meet the requirement of **coherence** of timeline, we should add some elements into the process when performing AP clustering algorithm. By using Named Entity Recognition techniques, we discount the similarity value between events that contain different named entities. Therefore, in this step, the similarity matrix is adapted as follows:

$$s(i,j) \leftarrow \alpha cosine(i,j) + (1-\alpha) sim(entity_i, entity_j) \quad (2)$$

, where $s(i,j)$ means the similarity between event i and event j, and $entity_i$,$entity_j$ stand for the named entities recognized respectively in event i,j.

The Third Step. After finishing the first two steps, we now obtain many clusters of events and we call every cluster a timeline. Such a timeline makes sense as it is the assembly of events, where all the events of it describe the similar/same entity, as well as they are arranged in chronological order. In the third step, we will take operations on these generated timelines to obtain optimal performance. Specifically, the task is to shuffle the events and rank the timelines.

To begin with, a timeline is a cluster of events. It's straightforward to arrange them in chronological order by their timestamp. We simply sort them based on their date. When it happens that more than one events refer to the same date, we consider merging these events.

Then, the next problem is to sort timeline. Based on the principle of **novelty**, we adapt MMR algorithm to balance the timeline utility and their similarity by an equilibrium parameter λ, that is,

$$MMR \stackrel{\text{def}}{=} \text{Arg} \max_{T_i \in R \setminus S} [\lambda \text{Utility}(T_i) - (1-\lambda) \max_{T_j \in S} \text{Sim}(T_i, T_j)] \qquad (3)$$

where R is the initial timeline set which is returned by clustering,S is the set of reranked timelines. Utility of a timeline is the consideration of the features of **relevance**, **coherence**, and **salience** which are mentioned above. Then the utility score of timeline T is computed by,

$$\text{Utility } (T) = \beta Rel(T,q) + (1-\beta) Coh(T) \qquad (4)$$

function $Rel(T,q)$ stands for the relevance between timeline and query, which could be interpreted as the combination of text similarity and the salience of the timeline. For similarity between timeline and query, as we mentioned in Section 3, we regard the mean average of event relevance as the timeline similarity. For salience, we represent it simply by a ratio number. that is,

$$\text{Rel}(T,q) = \alpha Sim(T,q) + (1-\alpha) Sal(T) \qquad (5)$$

$$\text{Sim}(T,q) = \frac{\sum_{i=1}^{n} Sim(e_i,q)}{n}, \quad for\ any\ e_i \in T \qquad (6)$$

$$\text{Sal}(T) = \frac{\#\ of\ events\ in\ T}{\#\ of\ total\ events} \qquad (7)$$

Then, the function $Coh(T)$ describes the continuity of the story, which we compute it as the average similarity of any two adjacent events pair, that is,

$$\text{Coh}(T) = \frac{\sum_{i=1}^{n} \text{Sim}\,(e_i, e_{i+1})}{n}, \text{for any } e_i \in T \qquad (8)$$

According to the utility function we can get the utility score for each timeline. After knowing the utility of timeline, for

the sake of requirement of **novelty**, we have to compute the information redundancy between two timelines. We use function $Sim(T_i, T_j)$ to compute such measure between timeline T_i and T_j:

$$\mathrm{Sim}\left(T_i, T_j\right) = \sum_{i=1}^{m} \sum_{j=1}^{n} \mathrm{Sim}\left(e_i, e_j^{'}\right), \quad e_i \in T_i,\ e_j^{'} \in T_j \quad (9)$$

So far, we have completed all the steps of shuffling and sorting. The system will thus return a series of well-ordered and well-behaved timelines to users according to his/her search query.

6 Conclusion and Future Work

Conclusion. In this paper, we propose MTS framework to deal with users' fuzzy queries. Taking a general query (e.g. Japan) issued by users as an input, the system automatically outputs multiple timelines with summaries. For example, the users input the search query "Japan", the system will produce various timelines, such as one timeline for sports events, one timeline for economic events, and one for famous stars .etc. What's more, we put forward three-step model to realize this framework. First, we use AP clustering on sentences to get events relevant to the query. These sentences are from the document collection that is relevant to the query. Next, we perform AP clustering once more, however, not on sentences but on events we obtained by last step. Clustering on events could return an information nugget which describes the same entity, we call such information nugget a timeline. In the third step, we sort events and timelines based on MMR principle.

Future Work. Our work is based on the assumption that the returned document collection is abundant so that we can generate multiple timelines. In fact, many queries are not the case. So, our application scenarios are only useful for ambiguous queries. However, these is no strict criteria to tell a query is ambiguous or clear. In the future work, **1.** We will think of such a metric to judge the "ability" of the query to generate timelines. **2.** We will adjust the Affinity Propagation algorithm more elaborately, using more skillful approach to utilize temporal information. Also, we will add the information, such as date reference, timestamp of documents into comparison. It will not only help improve the precision of our model, but also benefit many other works related, such as predict date for events, date judgement for documents, and so on.

References

[1] https://www.theguardian.com/environment/2010/jun/29/bp-oil-spill-timeline-deepwater-horizon

[2] Kumar, Vijay, Richard Furuta, and Robert B. Allen. "Metadata visualization for digital libraries: interactive timeline editing and review." Proceedings of the third ACM conference on Digital libraries. ACM, 1998.

[3] Chieu, Hai Leong, and Yoong Keok Lee. "Query based event extraction along a timeline." Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2004.

[4] Chen, Xiuying, et al. "Learning towards abstractive timeline summarization." Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. 2019.

[5] Swan, Russell C, and James Allan. "TimeMine: visualizing automatically constructed timelines." SIGIR. 2000.

[6] Yan, Rui, et al. "Evolutionary timeline summarization: a balanced optimization framework via iterative substitution." Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval. ACM, 2011.

[7] Steen, Julius, and Katja Markert. "Abstractive Timeline Summarization." Proceedings of the 2nd Workshop on New Frontiers in Summarization. 2019.

[8] Shahaf, Dafna, Carlos Guestrin, and Eric Horvitz. "Trains of thought: Generating information maps." Proceedings of the 21st international conference on World Wide Web. ACM, 2012.

[9] Carbonell, Jaime G, and Jade Goldstein. "The use of MMR, diversity-based reranking for reordering documents and producing summaries." SIGIR. Vol. 98. 1998.

[10] Dang, Van, and W. Bruce Croft. "Diversity by proportionality: an election-based approach to search result diversification." Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval. ACM, 2012.

[11] Agrawal, Rakesh, et al. "Diversifying search results." Proceedings of the second ACM international conference on web search and data mining. ACM, 2009.

[12] Santos, Rodrygo LT, Craig Macdonald, and Iadh Ounis. "Exploiting query reformulations for web search result diversification." Proceedings of the 19th international conference on World wide web. ACM, 2010.

[13] Drosou, Marina, and Evaggelia Pitoura. "Search result diversification." ACM SIGMOD Record 39.1 (2010): 41-47.

[14] https://en.wikipedia.org/wiki/Okapi_BM25

[15] Frey, Brendan J., and Delbert Dueck. "Clustering by passing messages between data points." science 315.5814 (2007): 972-976.