WILSON: A Divide and Conquer Approach for Fast and Effective News Timeline Summarization

Yiming Liao
The Pennsylvania State University
State College, PA, US
yiming@psu.edu

Shuguang Wang
The Washington Post
Washington, D.C., US
shuguang.wang@washpost.com

Dongwon Lee
The Pennsylvania State University
State College, PA, US
dongwon@psu.edu

ABSTRACT

Major news media frequently uses the method of news timeline summarization to summarize important daily news over major events across the timeline. While various sophisticated methods have been proposed to generate both concise and complete news timelines, in practice, generating timelines from a large number of news articles not only faces quality issues but also encounters the challenge of generation speed, which all existing methods have neglected. To mitigate these issues, in this work, we propose to speed up timeline generation by dividing the whole summarization task into sub-summarization tasks, adopting the "divide and conquer" philosophy: (1) date selection and (2) text summarization.

Furthermore, since existing methods in news timeline summarization pay less attention to the date selection than text summarization, in this paper, we re-examine the role of date selection in news timeline summarization and demonstrate that accurate date selection "alone" can significantly contribute to the task of news timeline summarization. Leveraging on the explicit date selection, then, we propose a simple yet fast and effective news timeline summarization method, named WILSON (nEWS tImeline Summarization). Experimented on two widely used timeline summarization benchmark datasets, timeline17 and crisis, empirical evaluation shows that WILSON outperforms state-of-the-art approaches in both speed and ROUGE scores, significantly improving ROUGE-2 F1 scores by 9.5%–17.7% and reducing generation time by two orders of magnitude. A further user study with professional journalists also validates the superiority of WILSON. Finally, we build a real-time news timeline summarization system and achieve encouraging results on an industrial-level corpus.

1 INTRODUCTION

Along with the rapid development of web services, an increasing number of news articles are published daily, describing both major and minor events worldwide. Due to the tremendous amount of news articles being produced every day, readers easily get lost in this information flood. Fortunately, news timeline, which summarizes each event with primary messages in a chronological order, makes it easy for readers to gain key insights and understand the evolution of news events. As such, many major news media has adopted the idea and have frequently produced news timelines of major news events. For example, Table 1 describes how 2018 North Korea-United States Singapore summit finally became a reality. Note that as the example illustrates, creating a news timeline requires the resolution of two sub-problems: (1) choosing of an ideal number of days among hundreds or thousands of candidate days, and (2) generating succinct text summaries per days.

1.1 Industrial Use Case

Combined with visual or interactive interfaces, news timelines can provide a convenient way to compress overloaded news to audience. Figure 1 illustrates two real timeline example on two major US newspapers. Figure 1 (a) is an interactive timeline summarization about Trump-Russia investigation from The Washington Post, while Figure 1 (b) is a text-based timeline summarization about China-US Trade War from The New York Times. To help readers better understand the evolution of each news event, journalists take time to collect and organize related news articles, figure out major events and story lines, and "manually" summarize them in a chronological order. As events such as natural disasters and political issues can span from several months to multiple years and involve thousands of news articles, such a manual process cannot scale well. As this process is both time-consuming and labor-intensive, currently, despite the popularity of the concept, not all newspapers are able to quickly produce such news timelines.

To address this challenge, several automatic news timeline summarization methods have emerged in recent years [4, 12, 21, 22, 25, 27, 29]. By and large, these methods have been proposed to generate both concise and complete news timeline generation methods. One is aimed at separating
different stories from a whole news corpus, such as using variants of topic modeling [8, 31] and neural networks [30]. Another category focuses on generating a series of chronological summaries for one specific event from only relevant news articles [12, 22, 28], where the first categories can serve as pre-processing to find relevant news articles for each event. In this paper, our focus is on the latter category in an unsupervised manner.

However, majority of existing methods focus only on the quality of generated timelines and neglect the generation speed. For example, the state-of-the-art unsupervised approach adopts submodular framework [12] and requires the pairwise similarities for all tokenized sentences, which could be over 100,000 per timeline. This yields extremely slow running time, as clearly demonstrated in the comparison of running times in Figure 2.

As the compression rates of timeline summarization vary with the relationships among article contexts while paying less attention to date selection. For example, some models [14, 26, 27] just treat date information the same as text information and include it as one of the features, while others [4, 19] simply use date frequency to resolve events. Although simply modeling text correlation shows good performance on both timeline summarization and date selection [12], it is not clear how date selection will contribute to news timeline summarization. In addition, existing state-of-the-art unsupervised approaches mostly include global optimization, which helps daily summaries to be relevant to the topic. However, using global optimization also makes daily summaries less specific per each day and very time-intensive to generate timelines. Therefore, considering both the quality and speed of news timeline summarization, this paper makes the following main contributions:

1. We re-examine the role of date selection in timeline summarization and show that, even without considering contextual correlation across different dates, accurate date selection is sufficient to generate high-quality news timelines. More importantly, although ignoring contextual correlation across dates leads to a lower empirical upper bound than other models, all of the previous approaches still fail to reach this lower upper bound, and they are not even close.

2. Leveraging the explicit date selection, we propose a simple but fast and effective unsupervised news timeline summarization method, named WILSON. Experimented on two widely used timeline summarization datasets, WILSON outperforms state-of-the-art approaches in both speed and ROUGE scores, significantly improving ROUGE-2 F1 score by 9.5%–17.7% and reducing generation time by two orders of magnitude.

3. To our best knowledge, WILSON is the first work to include an evaluation by professional journalists in news timeline summarization. Through manually comparing the machine-generated news timelines with corresponding human-generated ones, journalists confirm that our approach produces better timelines than competing methods.

4. Based on the proposed WILSON, we build a real-time news timeline summarization system on an industrial-level news corpus.

2 THE PROPOSED METHOD: WILSON

In this section, we introduce our proposed method, named WILSON (neWs tImeLIne SummarizatiON), also illustrated in Figure 3. Besides the pre-processing modules such as temporal tagging...
and search engine indexing, WILSON mainly consists of two components – explicit date selection and text summarization for each selected date.

2.1 Problem Formulation

A news timeline can be viewed as a series of chronologically ordered daily summaries over main events, denoted by \((d_i, S_i)\), where \(d_i\) and \(S_i\) stand for \(i_{th}\) date and \(i_{th}\) summary. Thus, news timeline summarization can be formulated as:

**Definition 1 (News Timeline Summarization).** Given a corpus of articles \(C_q\), which is associated with a topic query \(q\) and a time window \([t_1, t_2]\), the process of automatic timeline generation is to produce a series of daily summaries \((d_1, S_1), \ldots, (d_T, S_T)\), where \(t_1 \leq d_i \leq t_2\).

For both readability and reliability of generated news timelines, we follow existing works and utilize extractive summarization, which directly selects sentences from the corpus as summaries. More specifically,

**Definition 2 (Extractive News Timeline Summarization).** Given a corpus of articles \(C_q\), which is associated with a topic query \(q\) and a time window \([t_1, t_2]\), the corpus is first tokenized to dated sentences \(((date_i, sentence_i), date_i \in [t_1, t_2], sentence_i \in C_q)\) by a date expression in the sentence and/or by the publication date, then the timeline generation is to produce a series of daily summaries \((d_1, S_1), \ldots, (d_T, S_T)\), where \(t_1 \leq d_i \leq t_2\) and \(S_i = \{sentence_{i1}, \ldots, sentence_{IN}\}\).

The number of selected dates \(T\) and sentences \(N\) are hyper-parameters and chosen by users to control the compression rate of the generated timelines. Date selection is evaluated by f1 scores and summaries are evaluated by ROUGE [10].

2.2 Date Selection

We use HeidelTime [20] to tag temporal expressions in sentences during pre-processing stage and start with an unsupervised date selection algorithm [23] to select the most salient dates: (1) we build a date reference graph with dates as nodes and reference relationships as edges; (2) then, we run the PageRank algorithm [16] on the graph and select the top \(T\) ranked nodes as the most salient dates. Date references refer to the sentences \(s_{ij}\) that are published on \(date_j\) while mentioning \(date_j\). We experiment with 4 types of edge weights as follows:

- W1: the number of reference sentences \(|s_{ij}|\)
- W2: temporal distance \(|date_j – date_i|\) in days
- W3: \(W1 + W2\), which considers both frequency and temporal distance.
- W4: We adopt BM25 [18] to estimate the relevance of sentences to the query, and use \(\max BM25(s_{ij}, q)\) as edge weight for each reference.

For example, considering \(date_i=2018-06-01, date_j=2018-06-12\), and \(s_{ij}\) composed of only two sentences, i.e. Trump says summit with North Korea will take place on June 12 and The summit will take place on June 12. Then, W1 is the number of sentences and equals 2, while W2 is the difference between 2018-06-01 and 2018-06-12 in days and equals 11. Accordingly, W3 equals W1 + W2 and is 22. For W4, we treat each sentence as a document, use topic query \(q\) to score each document with BM25, and take the maximum BM25 score as W4.

As Table 2 shows that all four edge weights yield comparable results, date reference relationship alone can extract as accurate date selections as topical information. Since constructing topical relationships across dates takes extra time, we finally adopt W3 as the edge weight to select the most salient dates in the rest of this paper. Note that, for completeness, we also generate daily summaries to obtain a complete news timeline per each date.

![Figure 3: Workflow of our proposed method–WILSON.](image)

<table>
<thead>
<tr>
<th>Edge Weight</th>
<th>Date F1</th>
<th>Rouge-1 F1</th>
<th>Rouge-2 F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>0.3512</td>
<td>0.3905</td>
<td>0.0969</td>
</tr>
<tr>
<td>W2</td>
<td>0.3528</td>
<td>0.4029</td>
<td>0.1002</td>
</tr>
<tr>
<td>W3</td>
<td>0.3628</td>
<td>0.4099</td>
<td>0.0995</td>
</tr>
<tr>
<td>W4</td>
<td>0.5068</td>
<td>0.3934</td>
<td>0.0934</td>
</tr>
</tbody>
</table>

For example, considering \(date_i=2018-06-01, date_j=2018-06-12\), and \(s_{ij}\) composed of only two sentences, i.e. Trump says summit with North Korea will take place on June 12 and The summit will take place on June 12. Then, W1 is the number of sentences and equals 2, while W2 is the difference between 2018-06-01 and 2018-06-12 in days and equals 11. Accordingly, W3 equals W1 + W2 and is 22. For W4, we treat each sentence as a document, use topic query \(q\) to score each document with BM25, and take the maximum BM25 score as W4.

Although the occurrence of an event signals its importance within the news timeline [4] and is well leveraged in existing timeline summarization algorithms, we note that the occurrence of events is also correlated with the recency of events, where past events occur earlier and are more heavily reported than recent events. Consequently, existing approaches may suffer from this issue. For example, approaches that optimize the summaries to
recover the whole corpus, such as ETS [29] and TILSE [12], will generate more summaries on past events.

In addition, as most references in articles refer to past events, the current date selection algorithm tends to give too much weight on old dates and will also result in timelines that lack recent dates. For a better illustration, we present the Cumulative Distribution Function (CDF) of the date duration between selected dates and the start date in Figure 4. As expected, both TILSE (Submodular) and date selection via PageRank (Tran et al.) tends to select more old dates, while the date distribution of ground-truth timelines is generally more uniform. Thus, we use the standard deviation of differences between consecutive dates to measure the uniformity of date distribution:

**Definition 3 (Uniformity of Date Selection).** Given a series of selected dates \( \{d_1, d_2, \ldots, d_T\} \) in chronological order, we regard the differences between consecutive dates as \( \{\text{diff}_i = d_{i+1} - d_i\} \), then define its standard deviation as \( \sigma = \sqrt{\frac{1}{T} \sum_{i=1}^{T} (\text{diff}_i - \overline{\text{diff}})^2} \), as the uniformity of date selection.

2.2.1 Recency Adjustment. To add more weights on recent dates, we leverage the Personalized PageRank algorithm [1], where the restart distribution is not uniform. More specifically, we weight each date node \( d_i \) by \( W_i = e^{-\alpha d_i} \), where \( d_i = |d_i - date_{start}| \). \( \alpha \) ranges from 0 to 1 and is used to control the restart distribution. In practice, we use a grid search to find the \( \alpha \) that gives the most uniform distribution in the date selection, then use the chosen dates for news timeline generation.

2.2.2 Date Coverage. To better check the coverage of generated timelines, besides f1 score on date selection, we also measure the date coverage, e.g., if any day of ground-truth date \( g_i \pm 3 \) days lies in the generated timeline, \( g_i \) will be considered to be covered and we will measure what percentage of ground-truth dates are covered per timeline. For comparison, we also generate news timelines on truly uniformly distributed dates and present the results in Table 3. As we can see, although truly uniformly distributed dates cover the most ground-truth dates, due to the low accuracy in the date selection, the generated daily summaries are poor. However, adding recency adjustment with uniformity contributes to date selection in coverage, thus yields better timeline summarization.

2.3 Daily Summarization

Having selected the most salient dates, next, we divide timeline summarization into sub-summarization tasks. Although daily summarization tasks can be accomplished by any supervised or unsupervised document summarization algorithms, we intend to use a simple daily summarization method to validate the effectiveness of our explicit date selection, as complicated summarization techniques may introduce extra improvements in the performance. Specifically, we utilize the classic TextRank algorithm [13] to generate daily summaries. Similar to the task of date selection, TextRank constructs a sentence graph with sentences as nodes and similarity scores as edge weights. In particular, we use BM25 [18] to compute edge weights and run PageRank on this directed graph to select the most important sentences as daily summaries.

2.3.1 Post-processing. Dividing large text summarization tasks into smaller ones greatly speeds up timeline generation, and these sub-tasks can naturally be further accelerated through parallel processing. Conducting text summarization on a daily basis rather than on the whole corpus, however, ignores temporal correlation and thus introduces redundancy in summarization. To remove redundancy across dates, therefore, we incorporate post-processing to re-rank daily summaries based on the whole summarization. Similar to MMR [3], instead of directly using daily summaries, we add sentences into timeline summarization by their daily ranks and only accept sentences whose maximum cosine similarity with selected ones is smaller than a threshold (e.g., < 0.5).

2.4 News Timeline Generation Algorithm

The generation algorithm of WILSON is summarized in Algorithm 1. First, we build a date reference graph based on the date
Algorithm 1: Algorithm for WILSON

Input: temporally tagged sentences \( C = \{ (date_i, sent_i) \} \)

Output: a series of daily summaries \( (d_1, S_1), ... , (d_T, S_T) \)

1. Build a date reference graph based on date co-occurrence \((\{date_i, date_j\}) | (\{date_i, sent_k\} \in C \& (date_j, sent_k) \in C))\);
2. Compute edge weight according to W3 in Section 2.2;
3. \( selected\_dates \leftarrow \emptyset \);
4. for Grid search \( a \in (0, 1) \) do
5. \( \text{Compute personalized node weight for each date } date_i \) using \( W_i = e^{-a(date_i-min(date_i))} \);
6. \( \text{Run personalized PageRank to select the top } T \text{ ranked dates as a date selection candidate} \);
7. Based on Definition 3, compute the uniformity score of this date selection candidate;
8. \( selected\_dates \leftarrow \text{save the date selection with the best uniformity score as } (d_1, d_2, ..., d_T) \);
9. end for
10. for \( d_i \in selected\_dates \) do
11. \( \text{Find all sentences on } d_i \)
12. \( C_i \leftarrow \{sent_k | (date_k, sent_k) \in C \& date_k = d_i \} \);
13. Run TextRank on \( C_i \) to rank all sentences by importance score in a max heap \( H_i \);
14. \( \text{Initialize selected sentences } S_i \leftarrow \emptyset \);
15. end for
16. repeat
17. \( \text{Currently selected sentences } S \leftarrow \bigcup_{i=1}^{T} S_i \);
18. \( \text{Top ranked sentence per day } H \leftarrow \bigcup_{i=1}^{T} H_i[0] \);
19. \( \text{Remove top sentences: heap_pop(H_i) for } i \in [1, T]; \)
20. \( \text{Remove sentences from } H \text{ that have maximum similarity > 0.5 with existing sentences in } S; \)
21. \( \text{Add remaining sentences in } H \text{ to the corresponding daily summary } S_i \text{ only if } |S_i| < N; \)
22. until (all \( |S_i| = N \)) or (all \( |H_i| = 0 \));
23. return \((d_1, S_1), ... , (d_T, S_T)\)

Table 4: Dataset overview

<table>
<thead>
<tr>
<th>Dataset</th>
<th># of topics</th>
<th># of timelines</th>
<th>average per timeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeline17</td>
<td>9</td>
<td>19</td>
<td>739</td>
</tr>
<tr>
<td>Crisis</td>
<td>4</td>
<td>22</td>
<td>5,130</td>
</tr>
</tbody>
</table>

O(T^2) while \( t \) daily summarization tasks take \( O(t \cdot N^2) \). Thus the total time complexity of WILSON is \( O(T^2 \cdot t \cdot N^2) \).

For submodular framework [12], which conducts global summarization, it takes \( O((TN)^2) \) to obtain pair-wise similarities for all sentences and takes \( O(t \cdot n \cdot N \cdot T) \) to iterate \( t \cdot n \) times to select each individual sentence in a greedy manner. Therefore, the total time complexity is \( O((TN)^2 + t \cdot n \cdot N \cdot T) \). In Figure 2, the corpus size is defined as the total number of sentences (i.e. \( T \cdot N \)). As expected, the submodular frameworks show quadratic growth with a time complexity \( O((TN)^2) \), while our approach is almost linear to the corpus size with a time complexity \( O(T^2 + t \cdot N^2) \).

Given the approximation that \( T \) and \( N \) are in the same order of magnitude (based on Table 4), WILSON runs faster than the submodular framework by a factor of \( O(T^2/T) \). Given around 10% date compression rate (\( f \)) and \( T \) in hundred level, theoretically, our approach could gain over three orders of magnitude improvement in generation speed. Note that, due to the scalability issue of the submodular framework, [12] filtered sentences with pre-defined keywords to reduce \( N \) by over one order of magnitude, reducing the time complexity in practice. Given around 10% filtering rate, our approach could still gain about two orders of magnitude in generation speed, which is consistent with experiments in Table 7.

3 EMPIRICAL VALIDATION

3.1 Set-Up

3.1.1 Datasets. We run experiments on timeline17 [24, 25] and crisis [22]. Both datasets consist of journalist generated timelines from major news media such as CNN, BBC and Reuters, and a corresponding corpus of articles per topic (e.g. H1N1 flu and Egypt war). More specifically, timeline17 contains 19 timelines from 9 topics, while crisis involves 22 timelines from 4 topics. An overview of the two datasets is shown in Table 4.

3.1.2 Competing methods.

- **Random**: The system generates daily summaries by randomly selecting sentences from the corpus.
- **MEAD** [17]: a classic centroid-based multi-document summarization system.
- **Chieu et al.** [4]: a multi-document summarization system that uses date related TFIDF scores to measure sentence importance among corpus.
- **ETS** [29]: an unsupervised timeline summarization algorithm via simultaneously optimizing multiple heuristic metrics, including relevance, coverage, coherence, and diversity.
- **Tran et al.** [25]: a supervised timeline summarization algorithm, which extracts various features from sentences and leverages learning to rank techniques.
- **Regression** [26]: a supervised approach that formulates sentence selection as a linear regression problem.
- **Wang et al.** [27]: a supervised approach that formulates sentence selection as a matrix factorization problem.
### 3.1.3 Measurement
Among all the baselines, TILSE is the only one with source code available. Consequently, for all the other baselines, we follow the existing works [9, 25, 27], which conduct experiments on timeline17 with settings mentioned at the beginning of Section 5.2 in [25] and directly report the baseline results from previous papers. More specifically, in the generated timeline, the number of selected dates \( T \) is set to the number of dates in each ground-truth timeline, while the number of sentences per day \( N \) is forced to be the rounded value of the average number of sentences per date from the ground-truth timeline.

To fairly compare with TILSE, we re-run the their code, follow all their pre-processing, including text cleaning and keywords filtering, and conduct experiments on exactly the same sentence corpus per timeline generation. Note that, [12] used a slightly different setting from previous papers: 1) for Timeline17 dataset, they mixed articles of the same topic from different news agencies while our method mainly focuses on local dependencies.

#### 3.1.4 Evaluation Metrics
The commonly used summarization metrics, ROUGE scores [10], including ROUGE-1, ROUGE-2 and ROUGE-S* F1 scores, are adopted to evaluate the agreement between machine-generated and journalist-generated timelines. Moreover, to be consistent with TILSE comparison, we also include time-sensitive ROUGE scores as additional measurements [11]. More specifically, concat ROUGE scores totally ignore the time information by directly concatenating all texts together, while agreement ROUGE scores only consider the generated daily summaries on the ground-truth dates, and align ROUGE scores discount the quality of generated daily summaries by their distance to the corresponding ground-truth date. Last but not least, we test for significant improvements using an approximate randomization test [15] with a p-value of 0.05.

### Table 5: Results on Timeline17

<table>
<thead>
<tr>
<th>Methods</th>
<th>ROUGE-1</th>
<th>ROUGE-2</th>
<th>ROUGE-S*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.128</td>
<td>0.021</td>
<td>0.026</td>
</tr>
<tr>
<td>Chieu et al.</td>
<td>0.202</td>
<td>0.037</td>
<td>0.041</td>
</tr>
<tr>
<td>MEAD</td>
<td>0.208</td>
<td>0.049</td>
<td>0.039</td>
</tr>
<tr>
<td>ETS</td>
<td>0.207</td>
<td>0.047</td>
<td>0.042</td>
</tr>
<tr>
<td>Tran et al.</td>
<td>0.230</td>
<td>0.053</td>
<td>0.050</td>
</tr>
<tr>
<td>Regression</td>
<td>0.303</td>
<td>0.078</td>
<td>0.081</td>
</tr>
<tr>
<td>Wang et al. (Text)</td>
<td>0.312</td>
<td>0.089</td>
<td>0.112</td>
</tr>
<tr>
<td>Wang et al. (Text + Vision)</td>
<td>0.331</td>
<td>0.091</td>
<td>0.115</td>
</tr>
<tr>
<td>Liang et al.</td>
<td>0.334</td>
<td>0.105</td>
<td>0.103</td>
</tr>
<tr>
<td>WILSON (Ours)</td>
<td>0.370</td>
<td>0.083</td>
<td>0.144</td>
</tr>
</tbody>
</table>

### Table 6: Results on Crisis

<table>
<thead>
<tr>
<th>Methods</th>
<th>ROUGE-1</th>
<th>ROUGE-2</th>
<th>ROUGE-S*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>0.207</td>
<td>0.045</td>
<td>0.039</td>
</tr>
<tr>
<td>Wang et al. (Text)</td>
<td>0.211</td>
<td>0.046</td>
<td>0.040</td>
</tr>
<tr>
<td>Wang et al. (Text + Vision)</td>
<td>0.232</td>
<td>0.052</td>
<td>0.044</td>
</tr>
<tr>
<td>Liang et al.</td>
<td>0.268</td>
<td>0.057</td>
<td>0.054</td>
</tr>
<tr>
<td>WILSON (Ours)</td>
<td>0.352</td>
<td>0.074</td>
<td>0.123</td>
</tr>
</tbody>
</table>

![Figure 5: Concat Rouge 2 f1 scores when adding more sentences on each date on Crisis.](image)

### 3.2 Performance Comparison
Table 5 and 6 shows that our unsupervised approach WILSON outperforms all baselines in ROUGE-1 and ROUGE-S* f1 scores by a significant margin, and is only second to [27], a supervised approach, and [9] in ROUGE-2 f1 score on Timeline17 dataset.

In addition, Table 7 illustrates that WILSON outperforms the state-of-the-art unsupervised framework TILSE in all ROUGE metrics. Averagely, our method outperforms the submodular approaches by 12.9% in concatenate ROUGE-2 scores, by 58.3% in agreement ROUGE-2 scores, and by 40.1% in alignment ROUGE-2 scores. More importantly, our method also gains two orders of magnitude improvement in generation speed, making it possible to generate news timelines in a real-time manner.

In Table 7, We also include multiple variants of WILSON for ablation analysis. WILSON-uniform simply adopts uniform date selection, while WILSON-Tran directly uses W3 as edge weight without recency adjustment. As expected, selecting uniformly distributed dates results in the worst summarization, while including recency adjustment improves time-sensitive ROUGE-2 scores by 9.0%∼21.6%.

Overall, comparing with all competing approaches, the performance improvement of our method is higher in Crisis dataset. One explanation is that Crisis dataset contains more articles and spans a longer period, making it difficult for those competing approaches to correctly identify the long-term event dependencies, while our method mainly focuses on local dependencies.

#### 3.2.1 Effectiveness of Post-processing
In Table 7, we observed that considering correlation across different dates and reducing redundant daily summaries are seemingly minor, especially on Crisis datasets. Different from Timeline17 datasets, Crisis datasets consist of more compact daily summaries, where more than 90% dates contain only 1 sentence. Although reducing redundancy across dates is not necessary for timelines with compact daily summaries, we intend to verify the effectiveness of...
post-processing for timelines with abundant daily summaries. Instead of using the exact number of sentences per date in the ground-truth timelines, we generate timelines with more sentences per date, which is more practical as the true numbers are unknown. As demonstrated in Figure 5, simply adding daily summaries together suffers from the redundancy issue and using post-processing indeed helps. Note that we use the ROUGE-2 f1 score, so the overall scores going down with more sentences is because generated texts lead to lower ROUGE accuracy.

3.2.2 Empirical Bounds. Empirical bounds of our two-stage method are given in Table 8, where we use ground-truth dates as date selections for daily summarization. Note that, besides using ground-truth dates, the upper bounds of the submodularity framework [12] also employ ground-truth summaries and are obtained by directly optimizing ROUGE f1 scores in a supervised way, but we only use ground-truth dates and never touch ground-truth summaries, making us aware of how date selection will contribute to news timeline summarization. As demonstrated, even without considering contextual correlation across different dates in text summarization, it is still possible to generate reasonable news timelines with accurate date selection. Although the upper bound of our two-stage framework is much lower than that of the submodular framework, it is worth mentioning that all existing approaches fail to reach our upper bound, not even to 2 or 3 sentences, determining the number of dates requires understanding for the whole corpus, making it difficult to select. To solve this issue, we aim at automatically detecting the number of dates for news timelines. Motivated by the fact that news timelines consist of major events within the duration, we propose to consider major event coverage to determine the number of dates.

3.2.3 Automatic Date Compression. As defined in Section 2.1, existing news timeline summarization works only use a preset number of dates and length of daily summaries to generate news timelines. Unlike the length of daily summaries, which only implies the compression rate for a single day and is usually set to 30 or 60 sentences, determining the number of dates requires understanding for the whole corpus, making it difficult to select. To solve this issue, we aim at automatically detecting the number of dates for news timelines. Motivated by the fact that news timelines consist of major events within the duration, we propose to consider major event coverage to determine the number of dates. Specifically, we use the daily summarization to generate major events for each date and encode daily summaries with BERT.
3.3 Evaluation by Journalists
In addition to ROUGE scores, we also consult two professional journalists at the Washington Post, which is one of the leading daily American newspapers, to manually evaluate the quality of machine-generated news timelines. Among 41 timelines from the two datasets, we sample 10 timelines (20%) from 6 topics, including H1N1 flu, BP oil spill, Egypt crisis, Libya war, Yemen war, and Syria war. For each sampled timeline, we present the human-generated ground-truth timeline and three machine-generated timelines from ASMDS, TLSCONSTRINTS, and WILSON (Ours) to journalists. The ground-truth timeline is labeled as a reference, while the other three are given in random order and the order is independent for each evaluation. The evaluation is based on the comprehensiveness and readability of the generated timelines compared with the ground-truth timelines.

For each evaluation, the two journalists are asked to review ~80 daily summaries from ~50 distinct dates, which adds up to ~800 daily summaries from ~500 distinct dates in total, and collaborate to provide one final ranking of the three machine-generated timelines. To measure the ranking performance of each method, we adopt two common rank-aware measurements, Mean Reciprocal Rank (MRR) and Discounted Cumulative Gain (DCG), and present the results in Table 9. As shown, when evaluated by professional journalists, our method outperforms the state-of-the-art unsupervised framework TLSCONSTRAINS and achieves slightly better or comparable results with ASMDS. Considering our method gains two orders of magnitude improvement in generation speed, the results are very encouraging. More interestingly, although TLSCONSTRAINS generally achieves higher ROUGE scores than ASMDS in Table 7, TLSCONSTRAINS receives unexpectedly lower rank scores than ASMDS in this evaluation by journalists. This may imply a warning that automated measures may not be enough for news timeline summarization and human evaluation could be beneficial at times.

<table>
<thead>
<tr>
<th>Method</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>MRR</th>
<th>DCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASMDS</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>0.72</td>
<td>7.39</td>
</tr>
<tr>
<td>TLSCONSTRINTS</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>0.56</td>
<td>6.29</td>
</tr>
<tr>
<td>WILSON (Ours)</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0.76</td>
<td>7.63</td>
</tr>
</tbody>
</table>

Table 9: Results of journalist evaluation on the quality of machine-generated timelines. Best and second best scores are highlighted by bold and underscore respectively.

5 REAL-TIME SYSTEM FRAMEWORK

The framework of our real-time news timeline generation system is shown in Figure 7. This framework applies our proposed method WILSON on a 4-year news corpus of over 1 million news articles4 from the Washington Post and can generate timelines by event keywords in seconds. Firstly, we tokenize all the news articles into sentences and conduct temporal tagging to label each sentence. Then, to query relevant news content in real-time, we build a search engine on tagged sentences and index both date and content information. Specifically, we use ElasticSearch [7] as our backend search engine. Note that, we can easily include newly published news articles into our system by inserting them into the existing search engine. Finally, given both the keywords and duration time of a query event, our system will fetch related news sentences and run WILSON to generate a complete news timeline.

For example, we can generate a timeline about how the United States and North Korea reached the summit in seconds by setting query keywords to "Trump, North Korea, Kim, summit, united states" and time duration between 2018-01-02 to 2018-06-12. We set the timeline length to 10 and present the output in Table 11. Taking journalist generated timeline5 as a reference, we highlight manually generated timeline was collected from BBC. As different approaches generate timelines with different date selections, we only consider the dates that appear in all 4 timelines and show the first a few dates and their summaries in chronological order. We highlight the overlaps between manually generated and automatic generated timelines in colors and observe that the output of our approach is aligned better with the handcrafted one.

Interestingly, more summaries of our outputs are closer to the main events on each date than those of TILSE’s, though they are all relevant to this topic. We think it may be because more important daily events are reported more heavily on each date, while existing models try but fail to effectively capture the evolution clues across dates, thus simple daily summarization can work well. Apparently, how to balance local and global summarization and effectively capture event evolution could be one potential direction for news timeline summarization.
Table 10: Summary examples on the Death of Michael Jackson. To save space, we only select the first a few dates in chronological order, which appear in all 4 timelines. Main coverage with groundtruth is colored: red texts highlight the overlaps between groundtruth and TILSE/ours while blue texts highlight the distinct overlaps between groundtruth and ours. Note that all summarization approaches use exactly the same sentence candidates pool.
Besides ROUGE scores, [19] is the only existing work to include human in the evaluation, but they just assess the readability of daily summaries as they utilize the abstractive summarization. Since none of the previous studies utilize user study to measure the generation quality of the whole news timelines, we are the first work to include user study in timeline evaluation and consult journalists to assess the generation quality of the entire news timelines.

7 CONCLUSION
This paper shows that, with accurate date selection, we can generate high-quality news timelines without considering the temporal correlation of text summarization. Leveraging the explicit date selection, we propose a fast and efficient unsupervised timeline summarized method named WILSON. Specifically, WILSON outperforms state-of-the-art approaches in both ROUGE scores and speed, significantly improving concatenate ROUGE-2 F1 scores by 9.5%–17.7%, time-sensitive ROUGE-2 F1 scores by 17.1%–123.1% and reducing generation time by two orders of magnitude, which allows us to develop a real-time news timeline generation system for the news room. More importantly, a user study with professional journalists also confirms that the outputs of WILSON are closer to human-generated ones than outputs of other methods. Last but not least, this work also suggests two potential directions for future works, i.e. considering both occurrence and recency of events for better salient date selection and reducing contextual correlation across dates by balancing local and global summarization to improve daily summarization.

ACKNOWLEDGMENTS
The authors would like to thank the anonymous referees for their valuable feedback and helpful suggestions. This work was in part supported by NSF awards #1742702, #1820609, #1909702, #1915801, and #1934782. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors. Last but not least, we truly appreciate Everdeen Mason, Sophie Ho and their journalist team at the Washington Post for the valuable feedback and support.

REFERENCES
If one sentence contain multiple date expressions, we consider all distinct date-sentence pairs in generating dated sentences {{{date, sentence}}}. Besides, each sentence is also paired with the publication date of the article it appears in.

Evaluation. As suggested at the beginning of Section 5.2 in [25], we set the number of selected dates $T$ to the number of dates in each ground-truth timeline, and the number of sentences per day $N$ to the rounded value of the average number of sentences per date from the corresponding ground-truth timeline. In Table 4 and Table 5, we follow existing works and use ROUGE-1.5.5 to get concatenate ROUGE scores, including ROUGE-1, ROUGE-2 and ROUGE-S*, which ignores date selection in the generated summarization and concatenate all daily summaries together. For comparison with TILSE, we use the evaluation library from the authors 8 for time-sensitive ROUGE scores in Table 6. But different from previous papers, for Timeline17 dataset, TILSE [11] mixed articles of the same topic from different news agencies together and uses filtered sentence corpus for both datasets. Thus, for a fair comparison, we dump their sentence candidate pool through TILSE code and run our daily summarization on the same sentence candidate pool for each timeline. In speed evaluation, we do not consider the temporal tagging in the pre-processing, and only measure the speed of generation on the tagged sentences for both TILSE and WILSON. The wall time is measured on a 24-core machine.

Implementation details of WILSON. For daily summarization, we group dated sentences {{{date, sentence}}} by the date to obtain the sentence candidates for each date. Since one sentence can have multiple paired dates, it may appear in multiple daily summaries. When utilizing TextRank [13] to generate daily summaries, we use BM25 [18] scores as edge weight. More specifically, when calculating the edge weight of one sentence to other sentences, we treat the source sentence as query and other sentences as documents, and use its BM25 relevance scores as edge weights. BM25 weights are unsymmetrical, so we build a directed graph for each date and then run the PageRank algorithm to select top sentences as daily summaries. For PageRank algorithm in both date selection and daily summarization, we use the implementation of NetworkX 9 with default damping parameter $\alpha = 0.85$. Code is available at https://github.com/wilson-nts/WILSON.

Implementation details of baselines. Among all the baselines, TILSE is the only one with source code available. Therefore, for all the other baselines, we follow the existing works [9, 25, 27], adopt the conventional experiment setting and directly report the results from previous papers. For the news timeline outputs of TILSE [12] (both TLSCONSTRAINTS and ASMDS), we use the author implementation 10 and their provided configurations 11. Note that, the TILSE implementation uses the same processing (e.g. caches sentence similarity calculation) to generate multiple timelines that use the same news corpus, therefore, we add the processing time back in measuring the generation time per timeline.

A REPRODUCTION

We present the experiment details to reproduce our results.

Datasets and pre-processing. Both timeline17 and crisis are available at http://B3s.de/~gtran/timeline/. We use spaCy 6 to tokenize news articles into sentences. For temporal tagging, we use HeidelTime 7 to detect all temporal expressions in each sentence.

---

6https://spacy.io
7https://github.com/HeidelTime/heidetime