Learning towards Abstractive Timeline Summarization

Xiuying Chen[1,2,*], **Zhangming Chan**[1,2,*], **Shen Gao**[2], **Meng-Hsuan Yu**[2], **Dongyan Zhao**[1,2] and **Rui Yan**[1,2,†]

[1]Center for Data Science, Peking University, Beijing, China

[2]Institute of Computer Science and Technology, Peking University, Beijing, China

{xy-chen, zhangming.chan, shengao, yumenghsuan, zhaody, ruiyan}@pku.edu.cn

Abstract

Timeline summarization targets at concisely summarizing the evolution trajectory along the timeline and existing timeline summarization approaches are all based on extractive methods. In this paper, we propose the task of *abstractive timeline summarization*, which tends to concisely paraphrase the information in the time-stamped events. Unlike traditional document summarization, timeline summarization needs to model the time series information of the input events and summarize important events in chronological order. To tackle this challenge, we propose a *memory-based timeline summarization* model (MTS). Concretely, we propose a time-event memory to establish a timeline, and use the time position of events on this timeline to guide generation process. Besides, in each decoding step, we incorporate event-level information into word-level attention to avoid confusion between events. Extensive experiments are conducted on a large-scale real-world dataset, and the results show that MTS achieves the state-of-the-art performance in terms of both automatic and human evaluations.

1 Introduction

Timeline summarization aims at concisely summarizing the evolution trajectory of input events along the timeline. Existing timeline summarization approaches such as [Li and Li, 2013; Ren *et al.*, 2013] are all based on extraction methods. However, these methods rely on human-engineered features and are not as flexible as generative approaches. Herein, we propose the *abstractive timeline summarization task* which aims to concisely paraphrase the event information in the input article. An example case is shown in Table 1, where the article consists of events of a greatest entertainer in different periods, and the summary correctly summarizes the important events from the input article in order.

Abstractive summarization approaches including [See *et al.*, 2017; Hsu *et al.*, 2018] have been proven to be useful recently thanks to the development of neural networks. However, unlike traditional document summarization, timeline summarization dataset consists of a series of time-stamped events, and it is crucial for timeline summarization model to capture this time series information to better guide the chronological generation process. Besides, as the example in Table 1 shows, bad summary confuses the birthplace and the residence, the first album and the best-selling album of the celebrity. As we found in experiment, such infidelity problem is a commonly-faced problem in summarization tasks.

Events	Michael Jackson (dubbed as "King of Pop") was born on August 29, 1958 in Gary, Indiana.
	In 1971, he released his first solo studio album "Got to Be There".
	In late 1982, Jackson's sixth album, "Thriller", was released, where videos "Beat It", "Billie Jean" in it are credited with breaking racial barriers and transforming the medium into an art form and promotional tool
	In March 1988, Jackson built a new home named Neverland Ranch in California.
	In 2000, Guinness World Records recognized him for supporting 39 charities, more than any other entertainer.
Bad summary	Michael Jackson on August 29, 1958 in Gary, California. In 1971, his first album "Thriller" was released. In 2000, Guinness World Records recognized him for supporting 39 charities.
Good summary	Michael Jackson was born on August 29, 1958 in Gary, Indiana. His sixth album "Thriller" was released in 1982. In 2000, Guinness World Records recognized him for supporting 39 charities.

Table 1: Example of timeline summarization. The text in red demonstrates time stamp, and text in blue demonstrates wrong event description. Events are split by lines.

To tackle above challenges, we come up with a *memory-based timeline summarization* (MTS) model. Specifically, we first use an event embedding module with selective reading units to embed all events. Then, we propose a key-value memory module storing time series information to guide the summary generation process. Concretely speaking, the key in memory module is the time position embedding that represents the time series information, and the value is the corresponding event representation. Keys together forms a timeline and we use the time position of events on the timeline to guide generation process. Finally, in each decoding step, we introduce event-level attention and use it to determining word-level attention so as to avoid confusion between events.

Overall, our contributions can be summarized as follows:

- We propose the generative timeline summarization task.

[*]Equal contribution. Ordering determined by dice rolling.

[†]Contact Author.

• To tackle this task, we first come up with a time-event memory modeling time series information to guide chronological generation process.

• In each decoding step, we incorporate event-level information to assist in determining word-level attention so that the generated summary can avoid confusion between events.

• We also release the first real-world large-scale timeline summarization dataset[1]. Experimental results on the dataset demonstrate the effectiveness of our framework.

2 Related Work

We detail related work on timeline summarization, abstractive summarization, and key-value memory network.

Timeline summarization. Timeline summarization task is firstly proposed by [Allan *et al.*, 2001] which extracts a single sentence from each event within a news topic. Later, a series of works [Yan *et al.*, 2011b; Yan *et al.*, 2011a; Yan *et al.*, 2012; Zhao *et al.*, 2013] furthur investigate timeline summarization task. There are also works focusing on tweets summarization that are related to timeline summarization. For example, [Ren *et al.*, 2013] considered the task of time-aware tweets summarization, based on a user's history and collaborative social influences from "social circles". However, all above works are based on extractive methods, which are not as flexible as abstractive approaches.

Abstractive summarization. Recently, with the emergence of strong generative neural models for text [Bahdanau *et al.*, 2014], abstractive summarization is also becoming increasingly popular [Nallapati *et al.*, 2017; See *et al.*, 2017]. Most recent work includes [Hsu *et al.*, 2018], where they use sentence-level attention to modulate the word-level attention such that words in less attended sentences are less likely to be generated. Their sentence-level attention is static during the generation process, while in our model, the high-level attention changes in each decode step depending on current generated word which is more reasonable.

Key-value memory network Key-value memory proposed by [Miller *et al.*, 2016] is a simplified version of Memory Networks [Weston *et al.*, 2015] with better interpretability and has been applied in document reading [Miller *et al.*, 2016], question answering [Pritzel *et al.*, 2017], language modeling [Grave *et al.*, 2017], and neural machine translation [Kaiser *et al.*, 2017]. In our work, we apply the key-value memory network on timeline summarization task and fuses it into the generation process.

3 Problem Formulation

MTS takes a list of events $X = \{x_1, ..., x_{T_e}\}$ as inputs, where T_e is the number of events. Each event x_i is a list of words: $x_i = \{w_1^i, w_2^i, ..., w_{T_w^i}^i\}$, where T_w^i is the word number of event x_i. The goal of MTS is to generates a summary $\hat{Y} = \{\hat{y}_1, ..., \hat{y}_{T_y}\}$ that is not only grammatically correct but also consistent with the event information such as occurrence place and time. Essentially, MTS tries to optimize the parameters to maximize the probability $P(Y|X) = \prod_{t=1}^{T_y} P(y_t|X)$, where $Y = \{y_1, ..., y_{T_y}\}$is the ground truth answer.

[1] http://tiny.cc/lfh56y

4 Model

4.1 Overview

In this section, we introduce our memory-based timeline summarization model in detail. The overview of MTS is shown in Figure 1 and can be split into three modules: (1) Event Embedding Module (See § 4.2): We employ a recurrent network with Selective Reading Units (SRU) to learn representation of each event. (2) Time-Event Memory (See § 4.3): we propose a time-event memory to establish a timeline, and use the time position of events in the timeline to guide generation process. (3) Summary Generator (See § 4.4): Eventually, we use an RNN-based decoder to generate the answer incorporating memory information, event-level information, and word-level information.

4.2 Event Embedding Module

To begin with, we use an embedding matrix e to map one-hot representation of each word in x_i into a high-dimensional vector space. We then employ a bi-directional recurrent neural network (Bi-RNN) to model the temporal interactions between words:

$$h_t^i = \text{LSTM}_{\text{enc}}([e(w_t^i); p^i], h_{t-1}^i) \tag{1}$$

where ";" denotes the concatenation between vectors, h_t^i denotes the hidden state of t-th word in Bi-RNN for event x_i. To capture the sequential information of events, we randomly initialize a *time position encoding* vector p^i of i-th event to be included in the Bi-RNN input. Apart from gaining word representation h_t^i, we also need to gain event representation. Simply taking the final state of Bi-RNN $h_{T_w}^i$ as the representation of the whole event cannot fully capture the feature of the whole event. Thus, we establish a second RNN made of SRU proposed in [Chen *et al.*, 2018] to gain new event representation a^i:

$$s_t^i = \text{SRU}(h_t^i, h_{T_w}^i) \tag{2}$$

$$a^i = s_{T_w^i} \tag{3}$$

Generally speaking, SRU replaces the update gate in original GRU with a new gate taking each input h_t^i and coarse event representation $h_{T_w}^i$ into consideration. We omit the details here due to limited space and readers can refer to [Chen *et al.*, 2018] for details.

So far, we gain the representation of i-th event a^i and t-th word in a^i, *i.e.*, h_t^i.

4.3 Time-Event Memory

As stated in Introduction, in timeline dataset, the generated summary should capture the time series information to guide the chronological generation process. Hence, we propose a key-value memory module where keys together forms a timeline, and this time series information is used to guide generation process as shown in Figure 2.

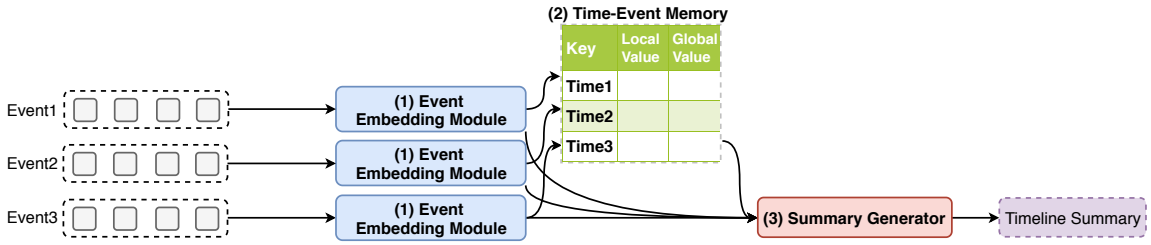

Figure 1: Overview of MTS. We divide our model into three ingredients: (1) *Event Embedding Module* learns event representation; (2) *Time-Event Memory* stores highlevel structral event information; (3) *Summary Generator* fuses the result from previous stages and generates a summary.

The key in this memory is the time position encoding p^i introduced in § 4.2. In § 4.4, we will use this key as time guidance to extract information from memory. As for the value part, it stores event information of local aspect in local value v_1 and global aspect in global value v_2. v_1 simply stores the event representation a^i as v_1^i, which means that v_1 only captures event information from current input article. On the other hand, v_2 is responsible for learning the global characteristics of events in different time position. Thus, we first randomly initialize v_2^i for i-th event in the same way as time position encoding. Then we establish a gate ν to combine average event representation in current batch $\dot{a}$ as a sub-global information:

$$\nu^i = \sigma(W_e[v_2^i; \dot{a}^i] + b_e) \tag{4}$$

$$v_2^i = \nu^i \cdot v_2^i + (1 - \nu^i) \cdot \dot{a}^i \tag{5}$$

where σ is the sigmoid function and $\cdot$ is dot product. In this way, the memory learns itself the global feature of each event in different position and stores it in v_2.

4.4 Summary Generator

To generate a consistent and informative summary, we propose an RNN-based decoder which incorporates outputs of time-event memory module and event representation as illustrated in Figure 2.

Following [Li *et al.*, 2018], we randomly initialize an LSTM cell taking the concatenation of all event representations as input, and use the output as decoder initial state:

$$h_0^{'} = \text{LSTM}\left(h_c, [a^1; ...; a^{T_e}]\right) \tag{6}$$

where h_c is a random variable. Next, following traditional attention mechanism in [Bahdanau *et al.*, 2015], we summarize the input document into context vector $c_{t-1}{}'$ dynamically, and the t-th decoding step is calculated as:

$$h_t^{'} = \text{LSTM}_{\text{dec}}\left(h_{t-1}^{'}, [c_{t-1}^{'}; e(y_{t-1})]\right) \tag{7}$$

where $h_t^{'}$ is the hidden state of t-th decoding step, and will be modified by output from memory module in Equation 22. Context vector $c_{t-1}^{'}$ is calculated as:

$$\alpha_{t,i,j}^{'} = W_a^{\intercal} \tanh\left(W_b h_{t-1}^{'} + W_h h_j^i\right), \tag{8}$$

$$\alpha_{t,i,j} = \exp\left(\alpha_{t,i,j}^{'}\right) / \sum_{k=1}^{T_e}\left(\sum_{l=1}^{T_w} \exp\left(\alpha_{t,k,l}^{'}\right)\right), \tag{9}$$

$$c_{t-1}^{'} = \sum_{i=1}^{T_e}\left(\sum_{j=1}^{T_w} \alpha_{t,i,j} h_i^d\right). \tag{10}$$

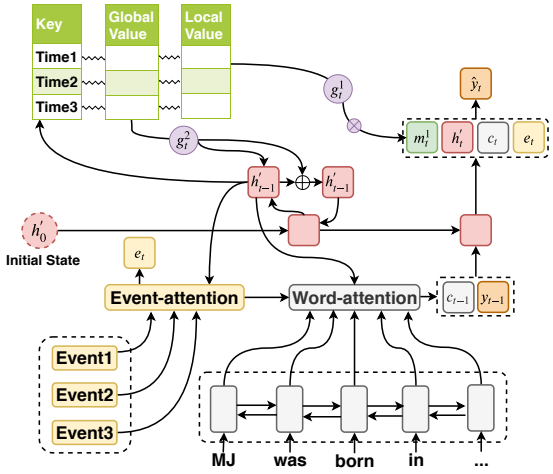

Figure 2: An overview of the summary generator.

where we first use the decoder state $h_{t-1}^{'}$ to attend to each states h_j^i and resulting in the attention distribution $\alpha_{t,i,j} \in \mathbb{R}^{T^d}$, shown in Equation 9. h_j^i denotes the representation of j-th word in event x_i. Then we use the attention distribution $\alpha_{t,i,j}$ to get the weighted sum of document states as the context vector $c_{t-1}^{'}$.

Context vector $c_{t-1}^{'}$ here only takes the word-level attention into consideration without considering event-level information. However, in timeline summarization, it is important for the model to be aware of which event it is currently describing, or it may confuse information from different events and results in an unfaithful summary. Hence, we introduce an event-level attention β similar to the calculation of word-level attention and use it to adjust word-level attention:

$$\beta_{t,i}^{'} = W_c^{\intercal} \tanh\left(W_d h_{t-1}^{'} + W_h a^i\right), \tag{11}$$

$$\beta_{t,i} = \exp\left(\beta_{t,i}^{'}\right) / \sum_{j=1}^{T_e} \exp\left(\beta_{t,j}^{'}\right), \tag{12}$$

$$\gamma_{t,i,j} = \alpha_{t,i,j} \beta_{t,i} \tag{13}$$

The new context vector c_t (replacing $c_t^{'}$ in Equation 10) is now calculated as:

$$c_t = \sum_{i=1}^{T_e}\left(\sum_{j=1}^{T_w} \gamma_{t,i,j} h_i^d\right) \tag{14}$$

Apart from using event-level attention to directly guide word-level attention, we also use it to gain the weighted sum of event representation to be concatenated in the projection layer in Equation 23:

$$e_t = \sum_{i=1}^{T_e} \beta_{t,i} a^i \tag{15}$$

So far, we have finished the calculation of context vector. Next, we introduce how to incorporate the guidance from memory. We first use hidden state $h_t^{'}$ to attend to each key in memory. As stated in § 4.3, keys, *i.e.*, time position embeddings, conform the timeline that represents the time series information. Thus, we let the model take advantage of this sequential information, and calculate the relevance between position encoding and current state as time-attention $\pi(p^i, h_t^{'})$:

$$\pi(p^i, h_t^{'}) = \exp(h_t^{'} W_e p^i) / \sum_{j=1}^{T_e} \exp(h_t^{'} W_e p^j) \tag{16}$$

Time-attention is then used to gain the weighted sum of local value v_1 and global value v_2 in the memory:

$$m_t^{1'} = \sum_{i=1}^{T_e} \pi(p^i, h_t^{'}) v_1^i \tag{17}$$

$$m_t^{2'} = \sum_{i=1}^{T_e} \pi(p^i, h_t^{'}) v_2^i \tag{18}$$

$m_t^{1'}$ and $m_t^{2'}$ stores information from different level, thus should play different roles in generator.

By a fusion gate, local value $m_t^{1'}$ is changed to m_t^1 and will be incorporated into the projection layer in Equation 23.

$$g_t^1 = W_o([h_{t-1}^{'}; c_t; m_t^{1'}]) \tag{19}$$

$$m_t^1 = g_t^1 \cdot m_t^{1'} \tag{20}$$

We place the local value in the projection layer since $m_t^{1'}$ stores the detailed information rather than the global feature in the input, thus should play an important part when generating each word.

As for the global value $m_t^{2'}$, it stores the global feature of event in different position, thus should influence the whole generation process. Concretely, information from $m_t^{2'}$ is fusioned into the decoding state $h_{t-1}^{'}$ by a gate:

$$g_t^2 = W_n([h_{t-1}^{'}; c_t; m_t^{2'}]) \tag{21}$$

$$h_{t-1}^{'} = g_t^2 \cdot h_{t-1}^{'} + (1 - g_t^2) \cdot m_t^{2'} \tag{22}$$

Finally, an output projection layer is applied to get the final generating distribution P_v over vocabulary:

$$P_v = \text{softmax}\left(W_v[m_t^1; h_t^{'}; c_t; e_t] + b_v\right) \tag{23}$$

We concatenate the output of decoder LSTM $h_t^{'}$, the context vector c_t, and memory vector m_t as the input of the output projection layer.

In order to handle the out-of-vocabulary (OOV) problem, we equip the pointer network [Gu *et al.*, 2016; See *et al.*, 2017] with our decoder, which enables the decoder capable of copying words from the source text. The design of the pointer network is the same as the model used in [See *et al.*, 2017], thus we omit this procedure due to limited space.

Our objective function is the negative log likelihood of the target word y_t, shown in Equation 24:

$$\mathcal{L} = -\sum_{t=1}^{T_s} \log P_v(y_t) \tag{24}$$

The gradient descent method is employed to update all parameters to minimize this loss function.

5 Experimental Setup

5.1 Research Questions

We list four research questions that guide the experiments: **RQ1** (See § 6.1): What is the overall performance of MTS? Does it outperform other baselines? **RQ2** (See § 6.2): What is the effect of each module in MTS? **RQ3** (See § 6.3): Is the time position embedding useful so that time-event memory can correctly guide generation process? **RQ4** (See § 6.4): Can event-level attention correctly guide word-level attention in decoding process?

5.2 Dataset

We collect a large-scale timeline dataset from the world's largest Chinese encyclopedia[2]. The character subsection of this website consists of celebrities at all times and in all countries or lands. On each website page, there is a timeline summary for each character, and in the character experience section of this page, each event is set as a paragraph with explanation and details, which is selected as input article. We filter out irrelevant content such as cited sources and figures. In total, our training dataset amounts to 169,423 samples with 5,000 evaluation and 5,000 test samples. On average, there are 352.22 words and 61.16 words in article and summary respectively.

5.3 Comparison Methods

We first conduct ablation study to prove the effectiveness of each module in MTS. Then, to evaluate the performance of our proposed dataset and model, we compare it with the following baselines:

(1) **Pointer-Gen**: Sequence-to-sequence framework with pointer mechanism proposed in [See *et al.*, 2017]. (2) **FT-Sum**: A summarization model proposed in [Cao *et al.*, 2018]. Since there is no open information extraction tool in Chinese, we use POS tagging to extract entities and verbs to replace it. (3) **Unified**: State-of-the-art generative summarization model proposed in [Hsu *et al.*, 2018]. (4) **LEAD3**: a commonly used baseline, which selects the first three sentence of document as the summary. (5) **TextRank**: [Mihalcea and Tarau, 2004] propose to build a graph, then add each sentence as a vertex and use link to represent semantic similarity. (6) **ITS**: One of state-of-the-art extractive summarization models proposed in [Chen *et al.*, 2018].

5.4 Evaluation Metrics

For evaluation metrics, we adopt ROUGE score in [Lin, 2004] which is widely applied for summarization evaluation [Sun *et al.*, 2018; Chen *et al.*, 2018]. The ROUGE metrics compare generated summary with the reference summary by computing overlapping lexical units, including ROUGE-1 (unigram), ROUGE-2 (bi-gram) and ROUGE-L (longest common subsequence).

[Schluter, 2017] notes that only using the ROUGE metric to evaluate summarization quality can be misleading. Therefore, we also evaluate our model by human evaluation. Three highly educated participants are asked to score 100 randomly

[2]https://baike.baidu.com/

	ROUGE-1	ROUGE-2	ROUGE-L
Pointer-Gen	36.61	21.35	34.51
FTSum	37.84	21.47	35.37
Unified	38.24	21.95	36.42
MTS	**39.78**	**22.24**	**37.69**
LEAD3	32.36	17.96	30.99
TextRank	32.27	15.34	30.86
ITS	34.03	18.20	31.24

Table 2: RQ1: ROUGE scores comparison between baselines.

sampled summaries generated by Unified and MTS. Statistical significance of observed differences between the performance of two runs are tested using a two-tailed paired t-test and is denoted using $^{\blacktriangle}$(or $^{\blacktriangledown}$) for strong significance for $\alpha = 0.01$.

5.5 Implementation Details

We implement our experiments in TensorFlow [Abadi *et al.*, 2016] on NVIDIA GTX 1080 Ti GPU. The word embedding dimension is set to 128 and the number of hidden units is 256. For time-event memory, the dimension of key, global value, and local value is 128, 512, and 256 respectively. We initialize all of the parameters randomly using an uniform distribution in [-0.02, 0.02]. The batch size is set to 16, and the event number is set to 8. We use Adagrad optimizer [Duchi *et al.*, 2010] as our optimizing algorithm and the learning rate is 0.15. In decoding, we employ beam search with beam size 4 to generate more fluency summary sentence.

6 Experimental Results

6.1 Overall Performance

For research question **RQ1**, we examine the performance of our model and baselines in terms of ROUGE as shown in table 2. Firstly, generative models outperform extractive models by a substantial margin, demonstrating the necessity of generative timeline summarization approaches. Secondly, the state-of-the-art model on CNN/DailyMail summarization dataset, Unified, still gets the best performance among baseline models on our timeline summarization dataset and outperforms the Pointer-Gen by 4.45% in ROUGE-1, which demonstrates the effectiveness of baselines. Finally, MTS achieves better performance with 4.02%, 1.32% and 3.48% increment over Unified and 8.65%, 4.16% and 9.21% over Pointer-Gen in terms of ROUGE-1, ROUGE-2 and ROUGE-L respectively, which proves the superiority of our model.

As for human evaluation, we ask three highly educated participants to rank generated summaries in terms of fluency, informativity, and fidelity. We pick FTSum and Unified as baselines since their performance is relatively high compared to other baselines. The rating score ranges from 1 to 3 and 3 is the best. The result is shown in Table 3, where MTS outperforms Unified by 5.44%, 3.61% and 18.09% in terms of fluency, informativity, and fidelity. It is worth noticing that the infidelity problem is a serious problem existing in timeline summarization, and MTS greatly alleviates such problem. We also conduct the paired student t-test between our model and Unified (row with shaded background), and result demonstrates the significance of the above results. The kappa statistics is 0.54 and 0.57 respectively, which indicates moderate agreement between annotators[3].

	Fluency	Informativity	Fidelity
Faithful	2.43	2.29	2.36
Unified	2.57	2.49	2.21
MTS	**2.71**$^{\blacktriangle}$	**2.58**$^{\blacktriangle}$	**2.61**$^{\blacktriangle}$

Table 3: RQ1: Human evaluation comparison with main baseline.

	ROUGE-1	ROUGE-2	ROUGE-L
MTS w/o EA	38.75	21.43	36.34
MTS w/o LV	37.95	21.01	35.76
MTS w/o GV	38.93	21.84	36.96
MTS	39.78	22.24	37.69

Table 4: RQ2: ROUGE scores of different ablation models.

6.2 Ablation Study

Next, we turn to research question **RQ2**. We conduct ablation tests on the usage of event-level attention, global and local value in time-event memory, corresponding to MTS w/o EA, MTS w/o LV, MTS w/o GV respectively. The ROUGE score result is shown in Table 4. Performances of all ablation models are worse than that of MTS in terms of all metrics, which demonstrates the necessity of each module in MTS. Concretely, local value and global value both make great contribution to overall performance, demonstrating that time series information is indeed helpful in extracting information to guide generation process. Besides, event-level attention also plays an important part. Without guidance from this level, word level attention has difficulty in focusing on input article and that leads to a 3.71% drop in ROUGE-L.

6.3 Analysis of Time Position Embedding

We then address **RQ3**, the usefulness of time position embedding is reflected by time-attention. We visualize the attention map of two randomly sampled example as shown in Figure 3. The figure above is the attention map in the first decoding step, and the figure below is in the final decoding step. The darker the color is, the higher the attention is. Due to limited space, we omit the corresponding event descriptions. When decoding starts, MTS learns to pay attention to the first two events, which always consist of parallel information such as the birthplace and birth date of the character. The attentions on last several events are low since it does not need this information in advance. When decoding ends, MTS focuses more on the last several events. However, it also pays attention to the first few events, since timeline summarization is a process of information accumulation, and latter sentences should consider previous information. Above example demonstrates the effectiveness of time position embedding.

[3][Landis and Koch, 1977] characterize kappa values < 0 as no agreement, 0-0.20 as slight, 0.21-0.40 as fair, 0.41-0.60 as moderate, 0.61-0.80 as substantial, and 0.81-1 as almost perfect agreement.

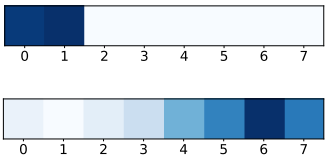

Figure 3: RQ3: Visualizations of time-attention. The figure above is the attention map in the first decoding step, and the figure below is in the final decoding step.

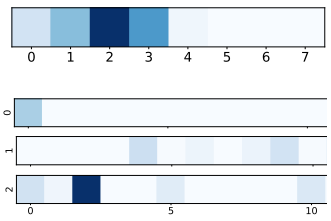

Figure 4: RQ4: Visualizations of two level attentions. The figure above is the event-level attention and the three figures below are the word-level attentions of first lead three events.

6.4 Analysis of Event-level Attention

We now turn to **RQ4**, whether event-level attention can guide word-level attention. We first conduct a case study to visualize the two level attentions, as shown in Figure 4. The figure above is the event-level attention, and three figures below are word-level attention corresponding to the first three events. We only show first 11 words in an event. The result shows that the third event is the most important event in this decoding step, and weights of the words in this event are also greater than other words on average. Above observation demonstrates that event-level attention gives the correct guidance for word-level attention.

Apart from the visualization, we also conduct quantitative analysis to measure how greatly the word-level attention is influenced by event-level information, which is reflected by inconsistency loss. We adjust the inconsistency loss proposed in [Hsu *et al.*, 2018] into MTS, and the new consistency loss at t-th decoding step is the negative log-likelihood of the product of attention value of most attended three words and their corresponding event-level attention. The intuition is to verify whether the event-level attention is high too when word-level attention is high. When training starts, the inconsistency loss is around 4.8, and when training ends, the loss drops to 2.6. This means that event-level information greatly influences the word-level attention and the model learns to unify these two attentions. We did not directly add inconsistency loss to training because we found that made MTS perform worse. Instead, we let the model learn by itself to unify these two attentions.

We also show a case study in Table 5. We can observe that baseline Unified confuses the description of events for twice. It is the movie "The Flowers Of War" that wins Golden Globe Award instead of the actor. While in summaries generated by MTS, the important events and their corresponding descriptions are all correctly included.

	2006年，佟大为出演《奋斗》、《我们无处安放的青春》和《与青春有关的日子》三部热门电视。他于2007年入围第57届柏林国际电影节主竞赛单元，角逐最佳男演员奖。2008年出演电影《赤壁》。2011年，出演张艺谋执导的电影《金陵十三钗》，凭借《金陵十三钗》荣获第十二届华语电影传媒大奖观众票选最受瞩目男演员奖，《金陵十三钗》入围第69届美国电影电视金球奖"最佳外语片"奖。2012年，凭借《中国合伙人》提名最佳男配角。2013年，在电视剧《门第》中饰何春生。(In 2006, Tong Dawei starred in three popular TV shows: "Struggle", "Youth We Can't Place", and "Days Related to Youth". He was also included in the main competition unit of the 57th Berlin International Film Festival in 2007 to compete for the Best Actor Award. In 2008, he starred in the movies "Red Cliff". In 2011, he starred in the film "The Flowers Of War" directed by Zhang Yimou, and won the most popular actor award in the audience of the 12th Chinese Film and Media Award for his role in "The Flowers Of War". "The Flowers Of War" was awarded the 69th Golden Globe Award for Best Foreign Language Film. He nominated "Best Supporting Actor" for a young artist in "Chinese Partner" in 2012. In 2013, he decorated He Chunsheng in the TV series "Men Di".)
reference	2007年，佟大为出演电视剧《奋斗》《我们无处安放的青春》和电影《苹果》。2008年，参演电影《赤壁》。2011年出演电影《金陵十三钗》。2013年主演电视剧《门第》。(In 2006, Tong Dawei played in the TV drama "Struggle", "Youth We Can't Place" and the movie "Apple". In 2008, he acts in the film "Red Cliff". He starred in the film "The Flowers Of War'. In 2013, he starred in TV play "Men Di".).
Unified	他于2007年入围第57届柏林国际电影节主竞赛单元，角逐最佳男演员奖。2008年，参演电影《赤壁》。2011年 凭借电影《金陵十三钗》获得第69 届美国电影电视金球奖" 最佳外语片"。(He entered the main competition unit of the 57th Berlin International Film Festival in 2007 for Best Actor Award. In 2008, he appeared in the film "Red Cliff". In 2011, he won the 69th Golden Globe Award for Best Foreign Language Film in American Film and Television for "The Flowers Of War".
MTS	佟大为2006年 出演《奋斗》、《我们无处安放的青春》和《与青春有关的日子》。2008年，参演电影《赤壁》。2011 年 出演电影《金陵十三钗》入围第十二届华语电影传媒大奖观众票选最受瞩目男演员奖；2013年，参演电视剧《门第》。(In 2006, Tong Dawei appeared in "Struggle", "Youth We Can't Place", and "Days Related to Youth". In 2008, he appeared in the film "Red Cliff". In 2011, he starred in the film "The Flowers Of War" and won the 12th Chinese Film and Media Award. In 2013, he shot the TV drama "Men Di".)

Table 5: Examples of the generated answers by MTS and Unified.

7 Conclusion and Future Work

In this paper, we propose a framework named MTS which aims to generate summaries that concisely summarize the evolution trajectory along the timeline. we first propose an event embedding module with selective reading units to embed all events. Then we propose a time-event memory module storing structral evolutinary event information to guide generation process. Finally, in each decoding step, we unify the current sentence-level attention and word-level attention together to avoid confusion between events. Our model outperforms state-of-the-art methods in terms of ROUGE and human evaluations by a large margin. In the near future, we aim to propose a time-aware timeline summarization that can summary the a specific time period of an whole article.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive comments. This work was supported by the National Key Research and Development Program of China (No. 2017YFC0804001), the National Science Foundation of China (NSFC No.61876196, No. 61672058), Alibaba Innovative Research (AIR) Fund. Rui Yan was sponsored by CCF-Tencent Open Research Fund and Microsoft Research Asia (MSRA) Collaborative Research Program.

References

[Abadi *et al.*, 2016] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[Allan *et al.*, 2001] James Allan, Rahul Gupta, and Vikas Khandelwal. Temporal summaries of new topics. In *SIGIR*, pages 10–18. ACM, 2001.

[Bahdanau *et al.*, 2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[Cao *et al.*, 2018] Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. Faithful to the original: Fact aware neural abstractive summarization. In *AAAI*, 2018.

[Chen *et al.*, 2018] Xiuying Chen, Shen Gao, Chongyang Tao, Yan Song, Dongyan Zhao, and Rui Yan. Iterative document representation learning towards summarization with polishing. *EMNLP*, 2018.

[Duchi *et al.*, 2010] John C. Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12:2121–2159, 2010.

[Grave *et al.*, 2017] Edouard Grave, Armand Joulin, and Nicolas Usunier. Improving neural language models with a continuous cache. *ICLR*, 2017.

[Gu *et al.*, 2016] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. Incorporating copying mechanism in sequence-to-sequence learning. *CoRR*, abs/1603.06393, 2016.

[Hsu *et al.*, 2018] Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. A unified model for extractive and abstractive summarization using inconsistency loss. pages 132–141. ACL, 2018.

[Kaiser *et al.*, 2017] Łukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. *ICLR*, 2017.

[Landis and Koch, 1977] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[Li and Li, 2013] Jiwei Li and Sujian Li. Evolutionary hierarchical dirichlet process for timeline summarization. In *ACL*, volume 2, pages 556–560, 2013.

[Li *et al.*, 2018] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. A survey on deep learning for named entity recognition. *arXiv preprint arXiv:1812.09449*, 2018.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[Mihalcea and Tarau, 2004] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *EMNLP*, 2004.

[Miller *et al.*, 2016] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *EMNLP*, pages 1400–1409. ACL, 2016.

[Nallapati *et al.*, 2017] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, 2017.

[Pritzel *et al.*, 2017] Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adria Puigdomenech, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. *arXiv preprint arXiv:1703.01988*, 2017.

[Ren *et al.*, 2013] Zhaochun Ren, Shangsong Liang, Edgar Meij, and Maarten de Rijke. Personalized time-aware tweets summarization. In *SIGIR*, pages 513–522. ACM, 2013.

[Schluter, 2017] Natalie Schluter. The limits of automatic summarisation according to rouge. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 41–45. ACL, 2017.

[See *et al.*, 2017] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. pages 1073–1083. ACL, 2017.

[Sun *et al.*, 2018] Min Sun, Wan Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, and Jing Tang. A unified model for extractive and abstractive summarization using inconsistency loss. In *ACL*, 2018.

[Weston *et al.*, 2015] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *ICLR*, abs/1410.3916, 2015.

[Yan *et al.*, 2011a] Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. Timeline generation through evolutionary trans-temporal summarization. In *EMNLP*, pages 433–443. ACL, 2011.

[Yan *et al.*, 2011b] Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. Evolutionary timeline summarization: a balanced optimization framework via iterative substitution. In *SIGIR*, pages 745–754. ACM, 2011.

[Yan *et al.*, 2012] Rui Yan, Xiaojun Wan, Mirella Lapata, Wayne Xin Zhao, Pu-Jen Cheng, and Xiaoming Li. Visualizing timelines: Evolutionary summarization via iterative reinforcement between text and image streams. In *CIKM*, pages 275–284. ACM, 2012.

[Zhao *et al.*, 2013] Xin Wayne Zhao, Yanwei Guo, Rui Yan, Yulan He, and Xiaoming Li. Timeline generation with social attention. In *SIGIR*, pages 1061–1064. ACM, 2013.