Be Relevant, Non-Redundant, and Timely: Deep Reinforcement Learning for Real-Time Event Summarization

Min Yang,[1] Chengming Li,[1*] Fei Sun,[2] Zhou Zhao,[3] Ying Shen,[4] Chenglin Wu[5]
[1]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
[2]Alibaba Group, [3]Zhejiang University
[4]Peking University Shenzhen Graduate School, [5]Deep Wisdom
{min.yang, cm.li}@siat.ac.cn, ofey.sunfei@gmail.com, zhaozhou@zju.edu.cn
shenying@pkusz.edu.cn, alexanderwu@fuzhi.ai

Abstract

Real-time event summarization is an essential task in natural language processing and information retrieval areas. Despite the progress of previous work, generating relevant, non-redundant, and timely event summaries remains challenging in practice. In this paper, we propose a Deep Reinforcement learning framework for real-time Event Summarization (DRES), which shows promising performance for resolving all three challenges (i.e., relevance, non-redundancy, timeliness) in a unified framework. Specifically, we (i) devise a hierarchical cross-attention network with intra- and inter-document attentions to integrate important semantic features within and between the query and input document for better text matching. In addition, relevance prediction is leveraged as an auxiliary task to strengthen the document modeling and help to extract relevant documents; (ii) propose a multi-topic dynamic memory network to capture the sequential patterns of different topics belonging to the event of interest and temporally memorize the input facts from the evolving document stream, avoiding extracting redundant information at each time step; (iii) consider both historical dependencies and future uncertainty of the document stream for generating relevant and timely summaries by exploiting the reinforcement learning technique. Experimental results on two real-world datasets have demonstrated the advantages of DRES model with significant improvement in generating relevant, non-redundant, and timely event summaries against the state-of-the-arts.

1 Introduction

Real-time event summarization aims to select an only relevant, non-redundant and timely subset from an overwhelming time-ordered stream of documents to summarize an event of interest. Different from the static event summarization (Saggion and Poibeau 2013) that only generates event summaries once without updating the summaries when new information emerges, real-time event summarization generates updating summaries when documents are coming in real time. It becomes an extremely important research topic with the increase of streaming applications.

[*]Chengming Li is corresponding author.

Recently, great efforts have been devoted to building effective and efficient real-time event summarization systems (Aslam et al. 2015). One representative method is to select or skip the documents in the text streams using the locally optimal learning method with greedy search (Kedzie, Diaz, and McKeown 2016). As deep learning models have achieved promising performance in text matching, there have been increasing attempts to apply deep learning models to real-time event summarization (Tan, Lu, and Li 2017). Despite the effectiveness of previous studies, there are several technical challenges (i.e., text relevance, non-redundancy, and timeliness) that are critical to generate high-quality real-time event summaries.

Text Relevance Most real-time event summarization approaches focus on learning inter-document relations between the query and input document. However, the intra-document features within the query and input document are complementary to the inter-document relations between the query and input document (Vaswani et al. 2017; Liang et al. 2019). It is necessary to explore both intra- and inter-document features to capture the relevance between the query and input document for real-time event summarization. In addition, most methods consider relevance prediction and real-time event summarization as a sequential pipeline or treat the relevance score as a feature of the real-time event summarization model. However, these two subtasks are often mutually dependent on each other and shall be considered simultaneously. Concretely, for each specific query, knowing the relevance degree of each query-document pair in advance enables to filter out quite non-relevant documents. Meanwhile, the existence of the sentence in the summary context might further improve the performances of the relevance prediction as the part of the summary sentences is generally intensely relevant to the query.

Non-redundancy There exist long-term dependencies among the document stream for real-time event summarization, i.e., the later decision making will be affected by the previous actions so as to avoid repeating the redundant information. Considering the rapidly growing document stream, it is challenging for the model to scan through the complete historical sequence at each prediction time. Thus, most existing methods merely consider recent historical documents, which leaves huge space for capturing long-term, especially lifelong sequential patterns to facilitate the model to generate relevant and non-redundant event summaries.

Timeliness Timeliness is critical for the quality of real-time event summaries, where the updates should be provided as soon after the actual event occurrence as possible. Paying more attention to timeliness could hurt relevance since at early stages information is prone to be uncertain and hence less relevant. On the contrary, taking more time to hone relevance tends to sacrifice the timeliness of the information, resulting in the information having low utility for summarizing the event. Hence, real-time event summarization is a long-term process and the potential trade-offs between the timeliness and relevance of information should be considered. It is a real-time forward decision process in which the current action may affect future decisions (dependency), and the future streaming documents may generate uncertainty on the current decision (Tan, Lu, and Li 2017). However, most of the existing methods consider merely historical dependencies.

In this study, we aim to fully resolve all the three challenges in a unified framework. **First**, we propose a hierarchical cross-attention network (HCAN) with intra- and inter-document attentions to integrate important semantic features within and between the query and input document, which is expected to capture the text relevance effectively. In addition, we leverage relevance prediction as an auxiliary task to strengthen the document representation learning and help to recognize relevant documents. **Second**, we devise a multi-topic dynamic memory network (MDMN) to maintain merely the latest multi-topic memories through an incremental updating mechanism along with the evolving document stream. MDMN model updates memorization from newly coming document stream for different topics belonging to the event so as to capture multi-topic lifelong sequential patterns during the lifetime of the event. Hence, MDMN can avoid repeating the redundant information even if that information was pushed many steps away by using the multi-topic memories to modulate an adaptive attention for text matching. **Third**, we model the real-time event summarization as a long-term optimization problem considering both historical dependencies and future uncertainty of the document stream. A deep reinforcement learning (RL) algorithm is then proposed as the solution. The RL reward is estimated on a complete sequence produced by the model, taking relevance, non-redundancy, and timeliness of event summaries into consideration simultaneously.

To evaluate the performance of the proposed DRES model, we conduct extensive experiments on two widely used real-life datasets from TREC 2016 Real-time Summarization track and TREC 2017 Real-time Summarization track. Experimental results show that the proposed model is able to generate more relevant, non-redundant, and timely real-time event summaries than the state-of-the-art baseline methods.

2 Related Work

Automatic real-time summarization has been an active research field. Recently, many approaches have been proposed to deal with real-time event summarization. Guo, Diaz, and Yom-Tov (2013) presented a temporal summarization approach, which extracts event summaries from real-time news stream for specific events. Thereafter, temporal summarization has been operationalized in the TREC Temporal Summarization (TS) Track from 2013 to 2015 (Xu, Oard, and McNamee 2013; Aslam et al. 2015; Aliannejadi et al. 2015). The top performers at TREC TS included the Multi-Document Summarization (MDS) combination model (McCreadie, Macdonald, and Ounis 2014) and the affinity propagation clustering model (Kedzie, Diaz, and McKeown 2016), etc. In addition, the real-time document pushing on document stream has attracted much attention from the TREC-2015 Microblog Track (Lin et al. 2015) and the TREC 2016-2017 Real-Time Summarization (RTS) Track (Lin et al. 2016; 2017). For example, Tan et al. (2016), the winner of the TREC-2015 Microblog Track, proposed a dynamic emission method to maintain an adaptive threshold for filtering the non-relevant tweets.

Subsequently, there are some works treating the real-time or temporal summarization task as a dynamic forward decision process by employing the reinforcement learning algorithms. Kedzie, Diaz, and McKeown (2016) proposed a locally optimal learning method, which learns a policy to select or skip each document from the document stream. In (Kedzie, Diaz, and McKeown 2016), the real-time summarization task is performed by using greedy search, and the model is trained to simulate the oracle summarization system. Tan, Lu, and Li (2017) employed a deep Q-Network to maximize the long-term rewards for real-time event summarization. The Q-Network is composed of an LSTM layer and a three-layer fully-connected neural network. Yang et al. (2019) proposed a multi-task learning method for web-scale event summarization, which learns the relevance prediction and real-time document filtering subtasks simultaneously.

Different from previous methods, we propose a unified framework to resolve all three challenges (relevance, non-redundancy, timeliness) in real-time event summarization. Previous methods mainly focused on one of these problems, but an algorithm that assumes and resolves all three problems together has not been proposed yet.

3 Our Methodology

Problem Formulation

Given the document stream and queries, the real-time event summarization task aims at selecting or skipping each document d as it is observed such that the users are provided a set of filtered documents that are relevant, non-redundant, and timely. Formally, when a user issues a query q that contains n words $[w_1^q, w_2^q, \ldots, w_n^q]$, DRES generates a summary by choosing documents from a document stream $D = \{d_1, d_2, \ldots, d_t, \ldots\ d_T\}$, where each document d consists of l words $[w_1^d, w_2^d, \ldots, w_l^d]$, and has a relevance label $\mathbf{y}^r$ (one-hot vector). It is noteworthy that, for simplicity, we assume the size of the stream window is fixed with size T. However,

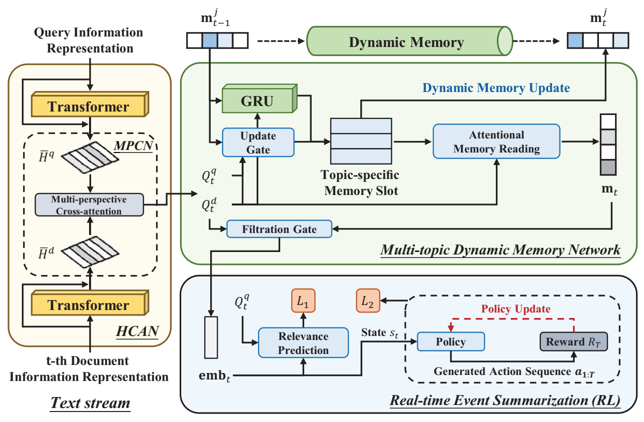

Figure 1: Architecture of our DRES model.

this is not strictly necessary. To avoid conceptual confusion, we adopt superscripts "r", "q" and "d" to represent the variables that are related to the relevance prediction, the query, and the input document.

As illustrated in Figure 1, DRES is composed of the following four components: (1) information representation layer to convert each word into a vector representation, (2) a hierarchical cross-attention network (HCAN) to integrate important semantic features within and between the query and input document, (3) a multi-topic dynamic memory network (MDMN) to maintain multi-topic memories and retain long-term dependencies for each event, (4) a deep reinforcement learning model to consider both historical dependencies and future uncertainty of the document stream. In addition, we treat relevance prediction as an auxiliary task to improve the relevance modeling in real-time event summarization. Next, we will describe each component of DRES in detail.

Information Representation Layer

The words which share common parts (e.g., root, prefix, suffix) may be related potentially. Therefore, we design a hybrid word embedding layer to take advantage of both the character-level and word-level representations. For the word-level embedding model, we adopt the popular word2vec (Mikolov et al. 2013) embeddings which are widely used in NLP domain. For the character-level embedding model, the ELMo (Peters et al. 2018) language model is used because of its superior performance in a wide range of NLP tasks. Each word is then represented as a concatenation of the character-level embedding and word-level embedding, resulting in a hybrid word embedding vector $\mathbf{e}_i \in \mathbb{R}^{d_w}$ for each token w_i. Here, d_w represents the size of the hybrid word embedding. The context representations of query q and input document d thus can be represented as $E^q = [\mathbf{e}_1^q, \ldots, \mathbf{e}_n^q]$ and $E^d = [\mathbf{e}_1^d, \ldots, \mathbf{e}_m^d]$, respectively.

We employ gated recurrent unit (GRU) (Cho et al. 2014) to learn the hidden states of tokens in the input query and document. Specifically, the learned feature representations E^q and E^d are passed to GRU encoder to obtain the context-dependent hidden state matrices $H^q = \{\mathbf{h}_i^q | \mathbf{h}_i^q \in \mathbb{R}^{d_h}, i = 1, 2, \ldots, n\}$ for query q and $H^d = \{\mathbf{h}_i^d | \mathbf{h}_i^d \in \mathbb{R}^{d_h}, i = 1, 2, \ldots, l\}$ for input document d, where d_h is the dimension of the hidden state of GRU.

Hierarchical Cross-Attention Network

Context-aware Transformer In order to model the temporal interactions and long-term dependency between words within a document, we employ a multi-head self-attention Transformer (Vaswani et al. 2017) to learn essential intra-document knowledge. We do not design a position embedding used in original Transformer. Instead, we have employed a GRU before the Transformer for capturing the sequential features of the documents (see Section 3.1). Formally, we feed the hidden representations H^q and H^d to Transformer to obtain the self-attentive representations for query q and document d as $\tilde{H}^q = \{\tilde{\mathbf{h}}_i^q | \tilde{\mathbf{h}}_i^q \in \mathbb{R}^{d_w}, i = 1, 2, \ldots, n\}$ and $\tilde{H}^q = \{\tilde{\mathbf{h}}_i^d | \tilde{\mathbf{h}}_i^d \in \mathbb{R}^{d_w}, i = 1, 2, \ldots, l\}$, respectively. The reader can refer to the paper (Vaswani et al. 2017) for the implementation details of Transformer.

After the Transformer layer, the sequential information captured with GRU layer will be discarded since the mean and variance of the feature vector will be changed. Inspired by (Li et al. 2018), we combine the features before the transformation and the self-attentive features, and pass them together to the next layer. A gate ξ_i for the i-th token is devised to control the proportions of the features before the transformation, which is computed as:

$$\xi_i = \text{sigmoid}(W_{trans}\mathbf{h}_i + b_{trans}) \tag{1}$$

Then, we can get the context-aware representations by combining context-dependent hidden states and self-attentive representations based on the gate:

$$\bar{\mathbf{h}}_i = \xi_i \cdot \mathbf{h}_i + (1 - \xi_i) \cdot \tilde{\mathbf{h}}_i \tag{2}$$

Thus, we can take advantage of the context information and global information of the input text. We denote the final intra-document representations as $\bar{H}^q = \{\bar{\mathbf{h}}_i^q | \bar{\mathbf{h}}_i^q \in \mathbb{R}^{d_w}, i = 1, 2, \ldots, n\}$ for query q and $\bar{H}^d = \{\bar{\mathbf{h}}_i^d | \bar{\mathbf{h}}_i^d \in \mathbb{R}^{d_w}, i = 1, 2, \ldots, l\}$ for document d.

Multi-perspective Cross-attention We propose a **m**ulti-**p**erspective **c**ross-attention **n**etwork (MPCN) to capture the semantic interaction between query q and document d. MPCN produces a 2-dimensional attention weight matrix. Given the intra-document query and document representations (i.e., $\bar{H}^q$ and $\bar{H}^d$), the attention matrix Σ^q for the document-aware query representation is computed as:

$$\Sigma^q = [\Sigma_1^q, \cdots, \Sigma_n^q], \quad \Sigma_i^q = \frac{\exp(\delta([\bar{H}_i^q, \mu(\bar{H}^d)]))}{\sum_{j=1}^k \exp(\delta([\bar{H}_j^q; \mu(\bar{H}^d)]))} \tag{3}$$

$$\delta([\bar{H}_i^q, \mu(\bar{H}^d)]) = W_1^c \tanh(W_2^c[\bar{H}_i^q, \mu(\bar{H}^d)]) \tag{4}$$

where $\Sigma_i^q \in \mathbb{R}^b$ denotes the i-th row of attention matrix, b is the number of hops of attention, μ is average pooling operation, W_1^c and W_2^c are learnable parameters. Similarly, we can calculate the attention matrix Σ^d for the query-aware document representation.

After obtaining the interactive query and document representations, we compute the interactive representations for query q and document d as:

$$O^q = \text{flat}(\Sigma^q \cdot \bar{H}^q), \quad O^d = \text{flat}(\Sigma^d \cdot \bar{H}^d) \tag{5}$$

where $flat$ is a function flattening matrix into vector form.

Multi-topic Dynamic Memory Network

In this section, we propose a multi-topic dynamic memory network (MDMN) to memorize input facts temporally from historical document stream and consider long-term dependencies (sequential patterns) within document stream with extremely large length. MDMN model is proposed based on two considerations of the motivation.

- There exist long-term dependencies among the document stream for real-time event summarization, i.e., the later decision making will be affected by the previous actions so as to avoid repeating the redundant information. Considering the rapidly growing document stream, it is challenging for the model to scan through the complete historical sequence at each prediction time. Hence, we devise MDMN to maintain merely the latest memories through an incremental updating mechanism along with the evolving document stream, which is expect to capture the lifelong dependencies of the document stream.
- Each critical event may contain several evolving topics that span various time distances. To analyze various evolution patterns that emerge from multiple topics, it requires multi-topic sequential pattern mining. MDMN model deals with this by maintaining multi-topic memory slots with different update periods.

Dynamic Memory Update For each query, there is a multi-topic memory network containing L memory slots, where each topic-specific memory slot $\mathbf{m}_t^j \in \mathbb{R}^p$ denotes the memory slot of the j-th topic at the t-th time step, which will be transmitted to the next time step. This external memory is used to better memorize the historical document stream and capture temporal sequential patterns for different topics belonging to a specific event.

MDMN model maintains the memory in real time through an incremental updating mechanism along with the evolving document steam. The memory slot at each layer would be updated as:

$$\mathbf{m}_t^j = g_t^j \text{GRU}(\mathbf{m}_{t-1}^j, O_t^d) + (1 - g_t^j)\mathbf{m}_{t-1}^j \tag{6}$$

$$g_t^j = \text{sigmoid}(W_m[O_t^d, O_t^q, \mathbf{m}_{t-1}^j]) \tag{7}$$

where W_m indicates learnable parameters, g_t^j is a gate that dynamically decide whether to update the memory slot of j-th topic at time step t. In Eq. (6), the memory writing in each layer is based on the GRU. O_t^q and O_t^d represent the query and document representations at time step t, respectively.

Attentional Memory Reading After conducting the long-term memorization of the multi-topic temporal dynamics along with the document stream, we implement the attentional memory usage similar to the standard memory networks (Sukhbaatar et al. 2015). Formally, we compute the comprehensive event memory $\mathbf{m}_t$ up to time step t as:

$$\mathbf{m}_t = \sum_{j=1}^{L} \mathbf{w}_t^j \cdot \mathbf{m}_t^j \tag{8}$$

where the weight w_t^j indicates the contribution of the memory slot $\mathbf{m}_t^j$ to the long-term memory representation $\mathbf{m}_t$ up to time step t, which is calculated as:

$$\mathbf{w}_t^j = \frac{\exp(\omega_t^j)}{\sum_{k=1}^{L}\exp(\omega_t^k)}, \ \omega_t^j = \tanh(W_a[O_t^d; \mathbf{m}_t^j]) \tag{9}$$

where ω_t^j is an energy function that measures the relevance between the document representation O_t^d and the long-term memory $\mathbf{m}_t^j$, W_a is a learnable parameter.

Distilled Non-redundant Document Representation Looking back at previous timesteps will allow DRES to extract more novel information and avoid repeating the redundant information even if that information was pushed many steps away. To achieve this, we devise a filtration gate $\mathbf{s}_t$ to filter out the redundant features contained in the document representation O_t^d at time step t. Specifically, we compute new fact-specific document representation emb_t at time step t based on the event memory $\mathbf{m}_t$ and the document representation O_t^d as:

$$\mathbf{s}_t = \text{Relu}(W_{s1}\mathbf{m}_t + W_{s2}O_t^d + b_{s1}) \tag{10}$$

$$emb_t = \mathbf{s}_t(\tanh(W_{s3}O_t^d + b_{s2})) \tag{11}$$

where W_{s1}, W_{s2}, W_{s3}, b_{s1}, and b_{s2} are parameters to be learned. emb_t is the novel features after filtering out the redundant information by the filtration gate.

Relevance Prediction

We treat relevance prediction as an auxiliary task to enhance the document representation learning, which helps to identify relevant documents from the document stream. Given the document stream and queries, the relevance prediction task aims at assigning a relevance label (i.e., "*highly-relevant*", "*relevant*" or "*non-relevant*") to each coming document, indicating whether it is relevant to the corresponding query. Relevance prediction can be treated as a multi-class classification problem.

We feed the concatenation of the query representation Q_t^q and new fact-specific document representation emb_t into a fully-connected layer and a softmax layer (for probabilistic classification) to obtain the predicted relevance label $\hat{y}_t^r$ of document d_t corresponding to query q_t at time step t:

$$\hat{\mathbf{y}}_t^r = \text{softmax}(V_3^r \cdot \tanh(V_1^r emb_t + V_2^r Q_t^q + b^r)) \tag{12}$$

where V_1^r, V_2^r, and V_3^r are learnable parameters for relevance prediction, and b^r is the bias term.

The parameters used for the relevance prediction task is learned via supervised learning. Specifically, given the labeled training data $\{(d_1, q_1, \mathbf{y}_1^r), \ldots, (d_T, q_T, \mathbf{y}_T^r)\}$, we learn the model by minimizing the cross-entropy between the ground truth relevance distribution $\mathbf{y}^r$ and the predicted relevance distribution $\hat{\mathbf{y}}^r$:

$$L_1 = -\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{K}\mathbf{I}\{\mathbf{y}_t^r = i\}\log(\hat{\mathbf{y}}_t^r), \tag{13}$$

where $\mathbf{I}\{\cdot\}$ represents an indicator such that $\mathbf{I}\{\text{true}\}$=1 and $\mathbf{I}\{\text{false}\}$=0. K denotes the number of relevance categories.

Real-time Event Summarization

Relevance-aware Document Representation Inspired by (Cao et al. 2017), we develop a relevance-aware transformation process to make the transformed document representation hold the relevance information, and therefore learn better document representation. Formally, we transform the document representation emb_t into a relevance-aware document representation $\tilde{emb}_t$ by

$$\tilde{emb}_t = \tanh(W_\mu \times emb_t) \tag{14}$$

where $W_\mu \in \mathbb{R}^{d_r}$ is the transformation matrix, d_r is the dimensionality of the relevance-aware document representation. Note that we define the same dimensionality for both emb_t and $\tilde{emb}_t$. To make $\tilde{emb}_t$ capture relevance information, we introduce 3 sub-matrices $\{W_\mu^1, W_\mu^2, W_\mu^3\}$, with each directly corresponding to one relevance label (i.e., "*highly-relevant*", "*relevant*" or "*non-relevant*"). Based on the predicted relevance label derived from Eq. (12), the relevance-aware transformation matrix W_μ is computed as the weighted sum of the sub-matrices: $W_\mu = \sum_{i=1}^{3} \hat{\mathbf{y}}_t^r W_\mu^i$. In this way, W_μ is automatically biased to the sub-matrix of the predicted relevance label.

Reinforcement Learning for Long-term Optimization In this paper, we formulate the real-time event summarization as a long-term optimization problem. Specifically, a policy gradient reinforcement learning method is employed to maximize the long-term rewards by considering both historical dependencies and future uncertainty. Different from (Tan, Lu, and Li 2017) which learns an approximation of the value function, the proposed DRES model searches the policy space with the policy gradient algorithm directly. In this way, DRES can express stochastic optimal policy and is robust to small change in function approximation. Next, we introduce the state, action, policy and reward of the reinforcement learning algorithm we used.

State At timestep t, a state s_t is the concatenation of new fact-specific document representation emb_t and relevance-aware document representation $\tilde{emb}_t$, i.e., $s_t = [emb_t; \tilde{emb}_t]$. The learned state s_t is expected to capture important information from both the current input document and the historical document steam.

Action At time step t, the action is $a_t \in \{0, 1\}$ with 1 representing we select the t-th document to the real-time event summary, and 0 representing we jump it.

Policy We use the stochastic policy gradient algorithm to learn the approximation of the policy π_θ. An approximation function with parameters θ is employed, which takes the state s_t as input, and outputs the probability of choosing the action a_t. Formally, we define the policy function as follows:

$$\pi_\theta(a_t|s_t) = \text{softmax}(V_2^p \cdot \tanh(V_1^p \cdot s_t + b^p)) \tag{15}$$

where V_1^p and V_2^p are learnable weight parameters, and b^p is the bias term.

Reward When we reach the end of the sequence of document-query pairs, the expected reward (R_T) will be calculated from the predicted distribution, which represents the score for producing the global action sequence $a_{1:T}$ given document stream and the update summary. Here, the delayed final reward R_T can be computed as:

$$R_T = r(a_{1:T}) = \lambda_1 EG(a_{1:T}) + \lambda_2 nCG(a_{1:T}) + \lambda_3 Latency \tag{16}$$

where $r(\cdot)$ is the reward function; λ controls the effect of EG, nCG and Latency, and we empirically demonstrate that we can achieve best results when $\lambda_1 = 0.15, \lambda_2 = 0.8, \lambda_3 = 0.05$. EG and nCG can capture the output quality (relevance and non-redundancy), while Latency ensures the timeliness.

Objective Function We optimize the parameters θ of our model using REINFORCE algorithm (Williams 1992) and policy gradient method, aiming to maximize the expected reward as shown below:

$$\begin{aligned} L_2(\theta) &= \mathbb{E}_{a_t \sim \pi_\theta(a_t,|s_t)} r(a_1 \dots a_T) \\ &= \sum_{a_1 \dots a_T} \prod_t \pi_\theta(a_t|s_t) R_T, \end{aligned} \tag{17}$$

Note that this reward is computed over just one time interval, say $D = [d1, \dots, d_T]$. The reader can refer to the study (Williams 1992) for details of policy gradient theorem.

Joint Training

DRES is a multi-task learning method, which contains two subtasks. and each task has an objective function for training. To strengthen the learning of the share document-query representations, we train these two related tasks simultaneously by minimizing the joint multi-task objective function:

$$L = \gamma_1 L_1 + \gamma_2 L_2, \tag{18}$$

where γ_1 and γ_2 are hyper-parameters that determines the weights of L_1 and L_2. Here, we set $\gamma_1 = 0.2$ and $\gamma_2 = 0.8$ by performing the grid search on a validation set.

4 Experimental Setup

Datasets

We evaluate the proposed DRES model on two publicly available corpora.

TREC-RTS-16[1] This dataset consists of 61,519 documents which are collected with Twitter streaming API (about 1% instances of all data) during the period from August 2, 2016 to August 11, 2016. It is used by TREC 2016 RTS

[1]http://trecrts.github.io/TREC2016-RTS-guidelines.html

Method	EG-0	nCG-0	EG-1	nCG-1	GMP	Latency
IPS	0.033	0.039	0.201	0.213	-0.324	192344
AP	0.037	0.031	0.232	0.235	-0.103	134077
CST	0.048	0.063	0.262	0.254	-0.321	91456
DES	0.036	0.035	0.229	0.219	-0.351	128857
LS	0.070	0.081	0.271	0.297	-0.185	85665
NNRL	0.069	0.085	0.282	0.288	-0.236	84343
MP-HCNN	0.073	0.077	0.285	0.294	-0.233	89647
CIG	0.069	0.079	0.276	0.288	-0.247	90472
COMP2016	0.048	0.069	0.269	0.291	-0.205	91539
DRES	**0.087**	**0.102**	**0.308**	**0.315**	**-0.080**	**72264**
DRES+Bert	**0.089**	**0.105**	**0.306**	**0.311**	**-0.078**	**71983**
w/o Transformer	0.079	0.091	0.299	0.305	-0.085	73425
w/o MPCN	0.078	0.093	0.302	0.307	-0.087	72958
w/o MDMN	0.073	0.085	0.291	0.299	-0.091	73536
w/o RP	0.076	0.089	0.297	0.304	-0.097	74376
w/o RL	0.071	0.082	0.295	0.298	-0.102	80143

Table 1: Event summarization results on TREC-RTS-16. The bottom five rows show the ablation test of DRES.

Method	EG-0	nCG-0	EG-1	nCG-1	GMP	Latency
IPS	0.051	0.054	0.202	0.188	-0.225	119273
AP	0.047	0.049	0.197	0.220	-0.183	126575
CST	0.063	0.061	0.251	0.243	-0.089	68112
DES	0.057	0.053	0.243	0.237	-0.064	77427
LS	0.073	0.065	0.283	0.255	-0.087	48392
NNRL	0.068	0.062	0.274	0.257	-0.115	44758
MP-HCNN	0.072	0.067	0.278	0.261	-0.092	65371
CIG	0.066	0.063	0.273	0.256	-0.094	61824
HLJIT (winner)	-	-	0.281	0.126	-0.156	119374
DRES	**0.086**	**0.075**	**0.302**	**0.278**	**-0.056**	**35319**
DRES+Bert	**0.085**	**0.077**	**0.298**	**0.283**	**-0.061**	**35362**
w/o Transformer	0.080	0.067	0.289	0.272	-0.083	37631
w/o MPCN	0.083	0.071	0.295	0.275	-0.074	36219
w/o MDMN	0.073	0.065	0.276	0.261	-0.067	37946
w/o RP	0.082	0.069	0.283	0.265	-0.097	39189
w/o RL	0.077	0.066	0.281	0.266	-0.094	40182

Table 2: Event summarization results on TREC-RST-17. The bottom five rows show the ablation test of DRES.

Track. Each instance has a relevance label ("*highly relevant*", "*relevant*" or "*non-relevant*") conditioned on the particular interest profile. At testing phase, 56 interest profiles are monitored, which represent the information needs of users. Similar to (Tan, Lu, and Li 2017), we choose instances of 41 interest profiles for training, instances of 5 interest profiles for validation, and the rest instances of 10 interest profiles are utilized for testing.

TREC-RTS-17[2] This corpus consists of 203,344 documents which are provided by Twitter API from July 29, 2017 to August 5, 2017, which is used by TREC 2017 RTS Track. There are 97 interest topics being monitored during the evaluation period. We choose the tweets of 80 interest topics as the training set, the tweets of 7 interest topics as the validation set, and the rest tweets are utilized for testing.

Baseline Methods

We evaluate and compare the proposed DRES with several state-of-the-art event summarization methods, including Interest Profile Similarity (IPS) (Han et al. 2014), Affinity Propagation (AP) (Frey and Dueck 2007), Cosine Similarity TDREShold (CST, Team WATERLOOCLARKE at TREC 2015) (Kedzie, McKeown, and Diaz 2015), Learning to Search (LS) (Kedzie, Diaz, and McKeown 2016), and Neural Network based Reinforcement Learning (NNRL) (Tan, Lu, and Li 2017). We also compare DRES with several advanced text matching methods, including Multi-Perspective Hierarchical Convolutional Neural Network (MP-HCNN) (Rao et al. 2019) and Concept Interaction Graph (CIG) (Liu et al. 2019). In addition, we replace the information representation layer and Transformer layer of DRES with the pre-trained BERT (Devlin et al. 2019) (denoted as **DRES+Bert**), and take DRES+Bert as a compared method. The hyperparameters are chosen with the validation data.

Implementation Details

In the experiments, we utilize 100-dimensional word2vec (Mikolov et al. 2013) vectors to initialize the word embeddings. The tokens that did not appear in the pre-trained word embeddings are initialized to zero. We set the initializations of bias vectors to zero, and all weight matrices are randomly sampled from orthogonal matrices. The hidden state sizes of GRUs in both representation layer and memory network are set to 300. We use mini-batch (batch size=64) training with Adam (Kingma and Ba 2014) optimization algorithm to learn the parameters of DRES. l_2 regularization (weight decay = 0.001) and dropout strategy (dropout rate = 0.2) are used to avoid overfitting.

Evaluation Metrics

The relevance of the documents returned by event summarization systems are evaluated by NIST assessors (Roegiest, Tan, and Lin 2017). Each document is assigned a relevance label from {"*not relevant*", "*relevant*", or "*highly-relevant*"}. Then, the NIST assessors conduct semantic clustering merely on the "*relevant*" and "*highly-relevant*" documents. The documents are grouped into clusters, and each cluster contains the documents that convey similar information. As discussed previously, the update summaries should be relevant, timely, and non-redundant, thus we adopt the Expected Gain (EG), Normalized Cumulative Gain (nCG), Gain Minus Pain (GMP) to evaluate the output quality (redundancy and relevance) and use Latency to evaluate the output timeliness, which are the official metrics of 2016-2017 TREC RTS track. We compute these metrics for each event cluster per day and average the values.

Expected Gain (EG) Following (Lin et al. 2016; 2017), we define the EG score in response to a specific interest profile at time t as: $EG = \frac{1}{N} \sum G(t)$, where N denotes the number of returned documents, $G(t)$ denotes the gain of each document at time t. Each document is assigned with a gain score from {0, 0.5. 1}, where 1 indicates "*highly-relevant*", 0.5 indicates "*relevant*", and 0 indicates "*not relevant*". Once a document from a cluster is selected, the other documents from the same cluster become non-relevant automatically. This strategy penalizes the real-time summarization model for extracting redundant information.

[2]http://trecrts.github.io/TREC2017-RTS-guidelines.html

Time	Tweet Text	Action
08-02 04:21:18	"Obama At Hiroshima: A World Without Nuclear Weapons šC Ours - American Thinker"	Keep
08-02 19:48:37	"Hiroshima nuclear bombing death toll: 146,000, World War II overall death toll: 65 to 80 million"	Keep
08-03 00:30:12	"Cockroaches survived atomic bombings at Hiroshima and Nagasaki but cannot survive a bug spray."	Skip
08-03 02:55:44	"But you the bomb, but you the bomb Hiroshima."	Skip
08-03 11:36:54	"This Saturday, August 6th, will mark the 71st anniversary of the #Hiroshima bombing. Time for abolition."	Keep
08-05 07:16:49	"Hiroshima prepares for A-bomb anniversary; peace activists meet: The Hiroshima city government sai"	Skip
08-05 09:08:05	"Lok Sabha pays homage 1945 atomic bomb victims, wishes Rio team well: New Delhi, Aug" 5	Skip
08-05 12:20:43	"The world says never again. Memorial events across the UK on #Hiroshima Day 6 August."	Skip
08-05 16:50:31	"Tomorrow: 71st anniversary of the U.S. dropping an atomic bomb on Hiroshima."	Skip
08-05 17:45:29	"How to Keep an Atomic Bomb From Being Smuggled Into New York City? Open Every Suitcase."	Skip

Table 3: Case study for topic "Hiroshima atomic bomb".

Normalized Cumulative Gain (nCG) Similar to previous work, we define the nCG score as: $nCG = \frac{1}{Z}\sum G(t)$, where Z denotes the maximum possible gain. Specifically, for a "*silent day*" (when there are only non-relevant documents), the EG-1 and nCG-1 have a score of 1 if the model does not select any documents, or 0 otherwise. In the EG-0 and nCG-0 variants, for a *silent day*, the gain is 0.

Gain Minus Pain (GMP) The goal of GMP is to punish the model for selecting non-relevant documents. Following (Lin et al. 2016; 2017), we define GMP as: $GMP = \tilde{\lambda}G - (1-\tilde{\lambda})P$, where G denotes the gain of each document defined above, P represents the number of non-relevant documents that are selected, and $\tilde{\lambda}$ determines the weights of G and P. Here, we set $\tilde{\lambda} = 0.5$ by tuning its value from [0,1] on the validation set.

Latency We calculate the latency of the model separately rather than apply a latency penalty to gain. The latency score only computed on the documents which are relevant. We report the mean difference between the time the target document was selected and the time the first relevant document in the same cluster emerged. We measure the mean latency of pushed tweets in seconds.

5 Experimental Results

Quantitative Evaluation

We first verify the performance of real-time summarization quantitatively. The experimental results are summarized in Tables 1-2. From the results, we can observe that DRES consistently and substantially surpasses the compared models by a large margin on all the evaluation metrics. The improvement from DRES is statistically significant over the compared models (t-test, p-value $<$ 0.05). Specifically, *DRES* achieves higher EG values than the compared baseline methods on the two corpora, indicating the novelty and non-redundancy of the produced summaries. From the relevance perspective, *DRES* obtains the highest GMP score, which implies that less "*non-relevant*" documents are pushed by DRES. In addition, the low latency score of DRES suggests the timeliness of DRES.

Ablation Study

In order to analyze the contribution of each part to the superiority of DRES, we also report the ablation test in terms of discarding the context-aware Transformer (w/o Transformer), multi-perspective cross-attention network (w/o MPCN), multi-topic dynamic memory network (w/o MDMN), relevance prediction task (w/o RP), and reinforcement learning algorithm (w/o RL), respectively. The ablation results are summarized in Tables 1-2 (bottom five rows). As expected, all the components contribute great improvements to the proposed DRES model. In particular, the performance of DRES decreases sharply when discarding the reinforcement learning and multi-task learning.

Case Study

To evaluate the proposed model qualitatively, we use an exemplary case for the topic "Hiroshima atomic bomb" from TREC-RST-16 to demonstrate the adaptive ability of *DRES* for real-time event summarization. Table 4 shows the summarization results for a snippet of the tweet sequence of 10 tweets given the query topic "Hiroshima atomic bomb". From the results, we can observe that *DRES* tends to generate a filtered stream of documents that are relevant, non-redundant, and timely to the given query. For example, our model skips the third and fourth tweets while it keeps the fifth tweet. Note that the fifth one is more comprehensive and relevant to the given query (topic), which shows the effectiveness of our method on selecting better documents. Moreover, after keeping the fifth one, our model skips the subsequent tweets (e.g., the eighth and the ninth tweets) that are relevant but redundant, which verifies the effectiveness of our method on avoiding redundancy.

6 Conclusion

This paper proposed a novel deep reinforcement learning framework for real-time event summarization. Benefiting from the hierarchical cross-attention, multi-task learning, multi-topic dynamic memory network, and RL strategy, the proposed model was able to resolve three main challenges (i.e., relevance, non-redundancy, timeliness) of real-time event summarization in a unified framework. Experimental results on two real-life datasets showed that our proposed approach significantly outperforms the baseline methods.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (NSFC) (No. 61602462 and No. 61906185), Guangdong Basic and Applied Basic Research Foundation (No. 2019A1515011705 and No. 2018A030313943).

References

Aliannejadi, M.; Bahrainian, S. A.; Giachanou, A.; and Crestani, F. 2015. University of lugano at trec 2015: Contextual suggestion and temporal summarization tracks. In *TREC*.

Aslam, J.; Diaz, F.; Ekstrand-Abueg, M.; McCreadie, R.; Pavlu, V.; and Sakai, T. 2015. Trec 2014 temporal summarization track overview. Technical report.

Cao, Z.; Li, W.; Li, S.; and Wei, F. 2017. Improving multi-document summarization via text classification. In *AAAI*, 3053–3059.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*.

Frey, B. J., and Dueck, D. 2007. Clustering by passing messages between data points. *science* 315(5814):972–976.

Guo, Q.; Diaz, F.; and Yom-Tov, E. 2013. Updating users about time critical events. In *ECIR*, 483–494. Springer.

Han, X.; Wang, L.; Park, S.; Cuevas, Á.; and Crespi, N. 2014. Alike people, alike interests? a large-scale study on interest similarity in social networks. In *ASONAM*, 491–496.

Kedzie, C.; Diaz, F.; and McKeown, K. 2016. Real-time web scale event summarization using sequential decision making. In *IJCAI*.

Kedzie, C.; McKeown, K.; and Diaz, F. 2015. Predicting salient updates for disaster summarization. In *ACL*, 1608–1617.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, X.; Bing, L.; Lam, W.; and Shi, B. 2018. Transformation networks for target-oriented sentiment classification. *EMNLP*.

Liang, D.; Zhang, F.; Zhang, W.; Zhang, Q.; Fu, J.; Peng, M.; Gui, T.; and Huang, X. 2019. Adaptive multi-attention network incorporating answer information for duplicate question detection. *SIGIR*.

Lin, J.; Efron, M.; Wang, Y.; and Sherman, G. 2015. Overview of the trec-2015 microblog track. Technical report.

Lin, J.; Roegiest, A.; Tan, L.; McCreadie, R.; Voorhees, E.; and Diaz, F. 2016. Overview of the trec 2016 real-time summarization track. In *TREC*.

Lin, J.; Mohammed, S.; Sequiera, R.; Tan, L.; Ghelani, N.; Abualsaud, M.; McCreadie, R.; Milajevs, D.; and Voorhees, E. M. 2017. Overview of the trec 2017 real-time summarization track. In *TREC*.

Liu, B.; Niu, D.; Wei, H.; Lin, J.; He, Y.; Lai, K.; and Xu, Y. 2019. Matching article pairs with graphical decomposition and convolutions. In *ACL*, 6284–6294.

McCreadie, R.; Macdonald, C.; and Ounis, I. 2014. Incremental update summarization: Adaptive sentence selection based on prevalence and novelty. In *SIGIR*, 301–310. ACM.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, 3111–3119.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. *NAACL*.

Rao, J.; Yang, W.; Zhang, Y.; Ture, F.; and Lin, J. 2019. Multi-perspective relevance matching with hierarchical convnets for social media search. In *AAAI*, volume 33, 232–240.

Roegiest, A.; Tan, L.; and Lin, J. 2017. Online in-situ interleaved evaluation of real-time push notification systems. In *SIGIR*, 415–424. ACM.

Saggion, H., and Poibeau, T. 2013. Automatic text summarization: Past, present and future. In *Multi-source, multilingual information extraction and summarization*. Springer. 3–21.

Sukhbaatar, S.; Weston, J.; Fergus, R.; et al. 2015. End-to-end memory networks. In *NIPS*, 2440–2448.

Tan, L.; Roegiest, A.; Clarke, C. L.; and Lin, J. 2016. Simple dynamic emission strategies for microblog filtering. In *SIGIR*, 1009–1012. ACM.

Tan, H.; Lu, Z.; and Li, W. 2017. Neural network based reinforcement learning for real-time pushing on text stream. In *SIGIR*, 913–916. ACM.

Vaswani, A.; Shazeer, N.; Parmar, N.; Jones, L.; Uszkoreit, J.; Gomez, A. N.; and Kaiser, Ł. 2017. Attention is all you need. *neural information processing systems* 6000–6010.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256.

Xu, T.; Oard, D. W.; and McNamee, P. 2013. Hltcoe at trec 2013: Temporal summarization. In *TREC*.

Yang, M.; Tu, W.; Qu, Q.; Lei, K.; Chen, X.; Zhu, J.; and Shen, Y. 2019. Mares: multitask learning algorithm for web-scale real-time event summarization. *World Wide Web* 22(2):499–515.