Memory-based Extractive Summarization

Chong Feng[1], Zhiqiang Pan[3], Jianming Zheng[4]
Systems Engineering College
National University of Defense Technology
Changsha, China
{[1]fengchong16, [4]zhengjianming12}@nudt.edu.cn,
[3]panzhiqiang15@gmail.com

Ying Xu[2]
The 28th Research Institute of China Electronics
Technology Group Corporation
Nanjing, China
cele_xy@qq.com

***Abstract*—Recurrent neural networks (RNNs) have been widely used in previous work on extractive text summarization. However, the information stored by RNNs structure is typically limited and is not compartmentalized enough to accurately record facts from the past. That makes it difficult to obtain relationships of the sentences in a document and select salient sentences with important information for a summary. In order to rectify this problem, we propose a memory-based extractive summarization (MES) model which is mainly constructed by memory generalization and sentence extractor. Our model can store more information of features extracted from sentences, relationships between sentences and implications of document, thus giving richer representations for selecting sentences for summary. Our experimental results show that MES model outperforms the baselines. We obtain improvements of up to 3.8%, 8.1%, 8.5%, 3.6% and 5.6% in terms of R-1, R-2, R-3, R-4 and R-L, respectively, over a relevant state-of-the-art baseline.**

Keywords-summarization; memory generalization; sentence extractor

I. INTRODUCTION

Text summarization means generating a short text summary for a document or a set of documents, which is generally done by so-called extractive and abstractive models. Extractive summarization models, aiming to produce a summary by extracting the salient sentences in a document [1], can generate summaries with more fluency than abstractive approaches. Fundamentally, extractive summarization systems should fulfill two tasks: storing and generalization. It first converts the text into some internal representations (i.e., memory or knowledge in our brains) so as to store them conveniently; and then the systems will have the ability to generalize these representations. Both two tasks, storing and generalization, are crucial to the success of summarization systems. Previous extractive approaches generally rely on the recurrent neural networks (RNNs) [2][3][4] and some models are based on encoder-decoder architecture which is also constructed by RNNs. These models can capture some important information and select the salient sentences of a document for summarization. However, the memory that RNNs structure stores (encoded by hidden states and weights) tends to be too small, and not compartmentalized enough to accurately record facts from the past. This leads to information loss, making it difficult to find important sentences. In order to rectify this problem, a class of models proposed by [5], which is called memory networks, were introduced. The models are trained to learn how to operate with the memory component effectively, which are more suitable for the two tasks in extractive summarization systems.

Thus, in this paper, we introduce a general framework for single-document summarization called memory-based extractive summarization (MES) which is aimed at performance improvements on text summarization. It mainly consists of two stages, i.e., memory generalization and sentence extractor. In memory generalization, the MES model first reads and generalizes all sentence embeddings in a document to form the memory slots that belong to the corresponding document. And then, in sentence extractor, the MES reads all sentence embeddings successively and produces sentence labels based on the memory slots. This architecture can record more information of a document, and as a result, it can capture more features and relationships of sentences in a document and generate richer document representations, making the summary more informative.

We evaluate the performance of our MES model on a public dataset consisting of the CNN news articles. Our experimental results show that MES outperforms the baselines in terms of the ROUGE scores. In particular, MES presents a significant improvement of up to 3.8%, 8.1%, 8.5%, 3.6% and 5.6% in terms of R-1, R-2, R-3, R-4 and R-L, respectively, over a relevant state-of-the-art baseline.

The main contributions are summarized as follows: (1) We propose a general framework for text summarization, named memory-based extractive summarization. (2) We use memory networks to store the generalized information of a document which can reduce information loss and help to capture rich features of sentences and select the correct sentences for the summary. (3) We investigate the performance of our model with various length of documents and summaries required, and find that MES model can generate better summaries for long documents and is more effective when generating longer summaries.

II. APPROACH

Before introducing our model, we first officially define the extractive text summarization task studied in this paper. Given a document D which consists of a sequence of n sentences $s_1, s_2, \cdots, s_n$, we aim to generate a summary by selecting a subset consisting of $l (l < n)$ sentences. To achieve this, we score and sort each sentence s_t $(1 \leq t \leq n)$ and predict its label

978-1-5386-8481-8/18/$31.00 ©2018 IEEE
DOI 10.1109/ICMCCE.2018.00121

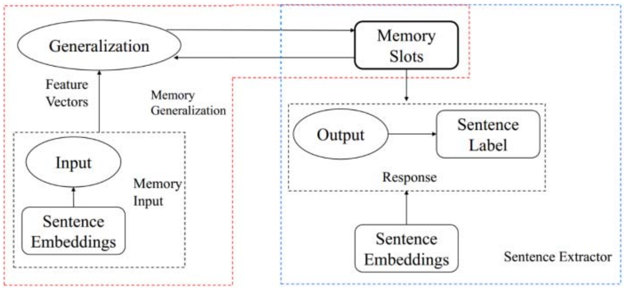


Figure 1. The framework of memory-based summarization.

$\hat{y}_t \in \{0,1\}$, where 1 indicates that s_t should be selected into the summary and 0 indicates the opposite. In a supervised training setup, we aim to maximize the likelihood of all predicted sentence labels $\hat{y} = (\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_n)$ given the input document D and a model parameters θ:

$$p(\hat{y} \mid D; \theta) = \prod\nolimits_{t=1}^{n} p(\hat{y}_t \mid D; \theta). \tag{1}$$

Firstly, we intend to get sentence embeddings for the following computation. In order to reflect the semantic relationships of all elements in a sentence, we adopt the convolutional neural networks (CNNs) to get the sentence embeddings [6]. Given a sentence $(w_1, w_2, ..., w_r)$ with r words, we first convolute the sentence using different filter operators $W_h \in \mathbb{R}^{h \times \dim}$ with $(h = 1, 2, ..., u)$ on word level to get the feature map $c^h \in \mathbb{R}^{r-h+1}$, where $\dim$ is the dimension of the word embeddings, u is the number of filter operators. After that, we employ the max-over-time pooling operation [7] to different feature maps c^h to capture the most important feature $\widehat{c^h}$. Then we concatenate these feature vectors generated by different filter operators so that we can get the final sentence embedding.

Next, we will illustrate the MES model from the following two modules, i.e., memory generalization and sentence extractor.

A. Memory Generalization

As shown in Fig. 1, memory generalization can be divided into the following three components, i.e., memory input, memory slots and generalization.

- **Memory input:** The process of memory generalization is based on the sentence level. Therefore, the component uses each sentence embedding $\boldsymbol{s_1}, \boldsymbol{s_2}, \cdots, \boldsymbol{s_n}$ as the input and feed them through a one-layer Multi-Layer Perception (MLP) to get the internal feature representation $\boldsymbol{I} = (\boldsymbol{I}_1, \boldsymbol{I}_2, ..., \boldsymbol{I}_n)$, i.e.

$$\boldsymbol{I}_1 = \tanh(W_I \boldsymbol{s}_i + b_i), \tag{2}$$

where W_I and b_i are the reshape matrix and bias term, respectively.

- **Memory Slots:** The memory slot $\boldsymbol{m}$ consist of an array of objects, i.e., $\boldsymbol{m} = (\boldsymbol{m}_1, \boldsymbol{m}_2, ..., \boldsymbol{m}_k)$, where k is the number of the memory slots.
- **Generalization:** By interacting with the memory slots, this component decides whether to store the new input into or modify/delete any earlier memory based on this new information. Generalization takes the internal feature representation $\boldsymbol{I}$ and current memory $\boldsymbol{m}$ as input to update the memory information, i.e.

$$\boldsymbol{m}_i = G(\boldsymbol{m}_i, \boldsymbol{I}, \boldsymbol{m}), \tag{3}$$

where G is a update function.

B. Sentence Extractor

In the first module, we complete memory generalizations to get the memory slot that belongs to a document, i.e., $\boldsymbol{m} = (\boldsymbol{m}_1, \boldsymbol{m}_2, ..., \boldsymbol{m}_k)$. In order to extract the most representative sentence, we use the memory slot and sentence embeddings as input, and propose a sentence extractor algorithm as follows.

Algorithm: Sentence Extractor

Input: The memory slot $\boldsymbol{m} = (\boldsymbol{m}_1, \boldsymbol{m}_2, ..., \boldsymbol{m}_k)$ and all sentence embeddings $(\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n)$ in a document D; the number of sentences that will be extracted, i.e., l.

Output: The label for each sentence.

```
Initialize the distance representation $d_i = [0, 0, ..., 0]_k$ for each sentence $s_i$; the label for extractive sentence, i.e., $\boldsymbol{ext} = [0, 0, ..., 0]_l$; the label for each sentence, i.e., $\boldsymbol{lab} = [0, 0, ..., 0]_n$.
$i = 0$
while $i < k$ do
    $j = 0$, $\boldsymbol{dis} = [0, 0, ..., 0]_n$, $\boldsymbol{pos} = [0, 0, ...0]_n$
    while $j < n$ do
        $\boldsymbol{dis}[j] = \mathrm{ED}(\boldsymbol{m}_i, \boldsymbol{s}_j)$
    $\boldsymbol{pos} = \mathrm{sort}(\boldsymbol{dis})$
    $j = 0$
    while $j < k$ do
        $d_{ij} = \boldsymbol{pos}[j]$
$\boldsymbol{ext} = \mathrm{k\_nn}(\boldsymbol{d}_i, l, i = 1, 2, ..., n)$
return $\boldsymbol{lab} \leftarrow \boldsymbol{ext}$
```

Table 1. Datasets Statistics

Variables	*Train*	*Test*
#Documents	83568	1093
Maximal # words in a sentence	1341	1426
Maximal # sentences in a document	125	105
Average # words per sentence	23.6	23.1
Average # sentences per document	29.8	30.2
Average # highlights per document	3.5	2.6

In the algorithm, ED(x, y) is the Euclidean distance function that compute the distance between x and y , sort(x) is a function that sorts each element in x from small to large and returns the corresponding position information, k_nn$(\boldsymbol{a}, l)$ is a k-nearest neighbor algorithm that divides the vector set $\boldsymbol{a}$ into l groups and returns the respective central vector for each group, and the last step means to set the value in ***lab*** as 1 for the position which is in ***ext*** , and the other value as 0. 1 represents the extracted sentence, while 0 indicates the sentence that has not been extracted.

After following the procedure in sentence extractor algorithm, we can get the extracted sentences as the summary.

III. EXPERIMENTS

A. Research Questions

(**RQ1**) How is the performance of our proposal, memory-based extractive summarization(MES) when compared to those of others start-of-the-art baselines? (**RQ2**) How does MES model perform on different length of documents? (**RQ3**) What is the impact on summarization performance of our model under different lengths of generated summary, i.e., 75 bytes vs. 275 bytes vs. full length (three sentences)?

B. Model Summary

As our MES model generates a summary by selecting the salient sentences from an original document, we compare our models with baselines of extractive text summarization: (1) LEAD: a standard text summarization baseline selecting the leading three sentences from each document as the summary [1][2][8]; (2) NN-SE: the state-of-the-art neural extractive text summarization model composed of a hierarchical document encoder and an attention based sentence extractor [8]. The model we propose in this paper is MES, a general framework for extractive summarization that use memory networks store information and select sentences for summaries.

C. Datasets and Experimental Setup

We train and evaluate our summarization models on a publicly available dataset created from the CNN news [8]. Each document in the dataset contains its highlights which are genuinely created by the news editors, so we can use the highlights as the ground truth summary. Every sentence of a document has been labeled as 1 (selected into the summary) or 0 (otherwise). Details of the dataset are shown in Table. 1. All sentences are padded to a fixed length of 60 words with dimension of 150. To get sentence embeddings, we use a list of kernels of widths 1 to 7, each with output channel size of 50. This leads the sentence embedding size to be 350.

Table 2. ROUGE evaluations (%) on the CNN dataset. The results produced by the best baseline and the best per- former in each column are underlined and boldfaced, respectively. Statistical significance of pairwise differences between MES and the best baseline (↑ / ↓) is determined by a t-test at the level of p < 0.05

Models	*R-1*	*R-2*	*R-3*	*R-4*	*R-L*
LEAD	43.5	13.3	8.1	5.3	40.1
NN-SE	44.5	13.6	8.2	5.6	39.4
MES	**46.2↑**	**14.7↑**	**8.9**	**5.8**	**41.6↑**

D. Evaluation Metric

We evaluate the performance of the models using the ROUGE metrics [9], where ROUGE-1 (R-1), ROUGE-2 (R-2), ROUGE-3 (R-3) and ROUGE-4 (R-4) indicate the informativeness of a summary, and ROUGE-L (R-L) captures its fluency.

IV. RESULTS AND DISCUSSION

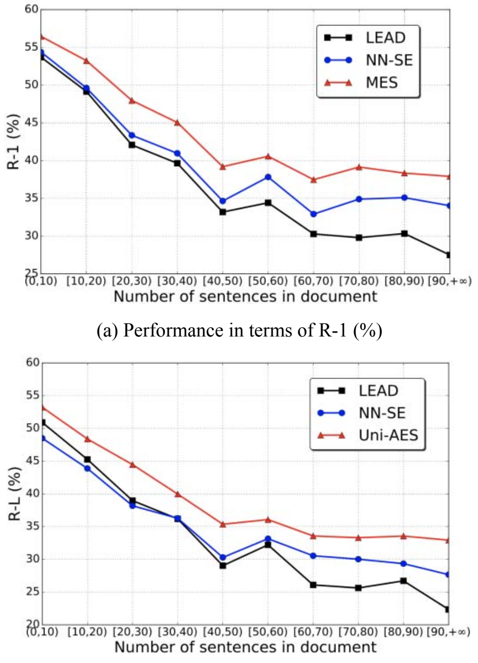
A. Performance of Summarization Models

For **RQ1**, we examine the ROUGE scores of full length summaries produced by our MES model as well as the baselines. The results are shown in Table. 2. In the baseline group, we can find that NN-SE has a higher R-N (N=1, 2, 3, 4) score than LEAD but a lower R-L score. This means that NN-SE can generate more informative but less fluent summaries than LEAD. This may be because that LEAD selects three consecutive sentences and NN-SE could choose sentences from different places.

Compared our model with the baselines, we can see that MES model shows obvious improvement over both two baselines in terms of all ROUGE scores. In particular, compared to NN-SE, MES presents improvement of 3.8%, 8.1%, 8.5%, 3.6% and 5.6% in terms of R-1, R-2, R-3, R-4 and R-L, respectively. In addition, improvements on R-1, R-2 and R-L are significant. That indicates that summaries generated by MES contain more information and are more fluent. These results confirm the effectiveness of MES model. The memory network we use in MES can indeed store rich information of a document and use them to capture the features of the document and obtain the salient sentences for summarization

B. Impact of the Length of Document

To answer **RQ2**, we group the test documents according to their lengths, i.e., the number of sentences they contain. For simplicity, we examine the performance of discussed models in terms of R-1 and R-L for generating the full-length summaries. We plot the results in Fig. 2. The figures show that the overall R-1 and R-L scores of the four summarization model go down as the length of documents increases. This could be explained by the fact that for long documents, the key sentences may exist in more places of a document, which increases the difficulty to locate them for summarization. Additionally, the LEAD model may be more sensitive to document length than



(a) Performance in terms of R-1 (%)

(b) Performance in terms of R-L (%)

Figure 2. Performance for different document lengths.

the other two models as the ROUGE scores present a more obvious change when the number of sentences varies.

Through the comparison between our MES model and the baselines, we can find that the improvements of MES are obvious. In general, both R-1 and R-L scores of MES decrease more slowly than those of baselines as document length increases. Also, the tendency of R-1 and R-L of MES are still slow when the baselines vary rapidly. This indicates that MES model is more stable for document length and can get better summaries for long documents. The reason may be that our memory network can record much more information of sentences as well as the whole document and use them to obtain the features of the document and find important sentences for summary more accurately.

C. Impact of the Length of Summaries

Finally, to answer **RQ3**, we compare the summarization results when summaries of varying length: 75 bytes vs. 275 bytes vs. full length (three sentences). The results for full-length summaries have been reported in Table 2; we show the ROUGE scores for 75 and 275 bytes in Table 3. We can see that NN-SE is superior to LEAD in general and our MES model again produces the best results.

For generating the 75-bytes summaries, MES only achieve slight improvement over other baselines in terms of ROUGE scores. Fortunately, for generating the 275-bytes summaries, MES achieve clearer improvement over the baselines. The improvement on R-1 and R-L is significant.

Table 3. ROUGE evaluations (%) on the CNN dataset with various length limits, i.e., 75 bytes and 275. The best baseline and the best performer in each column are underlined and boldfaced, respectively. Statistical significance of pairwise differences between MES and the best baseline (↑ / ↓) is determined by a t-test at the level of $p < 0.05$

Models	R-1	R-2	R-3	R-4	R-L
			75 bytes		
LEAD	12.2	2.6	1.1	0.6	11.1
NN-SE	12.4	2.8	1.3	**0.7**	11.2
MES	**12.9**	**3.0**	**1.4**	**0.7**	**11.6**
			275 bytes		
LEAD	34.6	11.0	5.1	3.2	32.3
NN-SE	35.5	11.2	5.3	3.3	31.1
MES	**38.2 ↑**	**12.3**	**6.1**	**3.8**	**33.4 ↑**

In summary, the proposed MES model works better in generating long summaries than short ones.

V. CONCLUSION AND FUTURE WORK

We have proposed MES, a general framework for single-document summarization, named memory-based extractive summarization (MES) where a memory network is used to store more information of a document to capture more important features and select salient sentences for summary. Our experimental results show that the MES model outperforms the baselines in terms of ROUGE scores and is more stable on long documents.

As to future work, we would like to test our models on different datasets to examine if they are suitable for text summarization. We also intend to incorporate other features of a document, e.g., topic, title and relevance between paragraphs, to capture richer information and generate better summaries.

REFERENCES

[1] Pengjie Ren, et al, "Leveraging contextual sentence relations for extractive summarization using a neural attention model," in proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2017, pp.95-104.

[2] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou, "SummaRuNNer: A recurrent neural Network based sequence model for extractive summarization of documents," in AAAI, 2017, pp.3075–3081.

[3] Cao, Z., Li, W., Li, S., Wei, F., and Li, Y., "Attsum: joint learning of focusing and summarization with neural attention," in COLING, 2016, pp.547-556.

[4] Narayan, S., Papasarantopoulos, N., Cohen, S. B., and Lapata, M. (2017). "Neural extractive summarization with side information," in CoRR, 2017.

[5] Weston J, Chopra S, and Bordes A, "Memory Networks," Eprint Arxiv, 2014.

[6] Kim Y, "Convolutional Neural Networks for Sentence Classification," in EMNLP, 2014, pp.1746-1751, in press.

[7] R. Collobert, J. Weston, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," in Journal of Machine Learning Research, 2011, pp.2493-2537.

[8] Cheng J, and Lapata M, "Neural Summarization by Extracting Sentences and Words," in ACL, 2016, pp.484-494, in press.

[9] Lin Chin-Yew, and E. Hovy, "Automatic evaluation of summaries using N-gram co-occurrence statistics," in HLT-NAACL, 2003, pp.71-78.