Neural Computing and Applications
https://doi.org/10.1007/s00521-019-04638-3

ORIGINAL ARTICLE

CRHASum: extractive text summarization with contextualized-representation hierarchical-attention summarization network

Yufeng Diao[1,2] · Hongfei Lin[1] · Liang Yang[1] · Xiaochao Fan[1,3] · Yonghe Chu[1] · Di Wu[1] · Dongyu Zhang[4] · Kan Xu[1]

Received: 10 May 2019 / Accepted: 22 November 2019

Abstract
The requirements for automatic document summarization that can be applied to practical applications are increasing rapidly. As a general sentence regression architecture, extractive text summarization captures sentences from a document by leveraging externally related information. However, existing sentence regression approaches have not employed features that mine the contextual information and relations among sentences. To alleviate this problem, we present a neural network model, namely the Contextualized-Representation Hierarchical-Attention Summarization (CRHASum), that uses the contextual information and relations among sentences to improve the sentence regression performance for extractive text summarization. This framework makes the most of their advantages. One advantage is that the contextual representation is allowed to vary across linguistic context information, and the other advantage is that the hierarchical attention mechanism is able to capture the contextual relations from the word-level and sentence-level by using the Bi-GRU. With this design, the CRHASum model is capable of paying attention to the important context in the surrounding context of a given sentence for extractive text summarization. We carry out extensive experiments on three benchmark datasets. CRHASum alone can achieve comparable performance to the state-of-the-art approach. Meanwhile, our method significantly outperforms the state-of-the-art baselines in terms of multiple ROUNG metrics and includes a few basic useful features.

Keywords Extractive text summarization · Contextualized representation · Hierarchical attention mechanism · Bi-GRU

1 Introduction

With the rapid increase in the volumes of textual data that are available both online and offline, the need for automatic document summarization that can be implemented in practical scenarios is increasing [1, 2]. Text summarization is meant to quickly locate the significant sentences of an article, and the methods are usually divided into extractive and abstractive models. As a framework for extractive summarization, the goal is to produce a short text summarization by extracting the salient sentences in a

✉ Hongfei Lin
hflin@dlut.edu.cn

Yufeng Diao
diaoyufeng@mail.dlut.edu.cn

Liang Yang
liang@dlut.edu.cn

Xiaochao Fan
fxc1982@mail.dlut.edu.cn

Yonghe Chu
yhchu@mail.dlut.edu.cn

Di Wu
wudi@dlut.edu.cn

Dongyu Zhang
zhangdongyu@dlut.edu.cn

Kan Xu
xukan@dlut.edu.cn

[1] Department of Computer Science and Technology, Dalian University of Technology, Dalian, China

[2] School of Computer Science and Technology, Inner Mongolia University for Nationalities, Tongliao, China

[3] Xinjiang Normal University, Xinjiang, China

[4] School of Software, Dalian University of Technology, Dalian, Liaoning, China

Published online: 02 January 2020

document, which will generate more fluent summaries than abstract approaches. Such methods are popular and widely used for practical applications because they are computationally cost-effective and less complex. In the field of extractive text summarization, the sentence regression achieves the state-of-the-art performance [3–6], and it has been widely used in practical systems [7, 8].

A sentence regression is composed of two major components: sentence scoring and sentence selection. The former computes the score of a sentence in order to measure its importance. The latter chooses a few sentences to produce a summary according to the importance scores and redundancy. Sentence scoring has been extensively investigated in extractive text summarization. Traditional scoring methods, such as the sentence length, the sentence position and TF-IDF-based features, incorporate feature engineering as a necessary and labor-intensive task. Here, we list the scores of the t-SR method [7] and the upper bound method in Table 1. The first is a traditional feature engineering sentence regression approach that obtains the state-of-the-art performance for extractive text summarization. The second is achieved by scoring the sentences according to human-written summaries. The results demonstrate the sizable performance gap between the t-SR and the upper bound approaches. The reason is that the t-SR should introduce more semantic information to encode and understand text summarization.

In recent years, neural network-based approaches for text summarization have rapidly advanced [5, 6, 9]. They have received increased attention by extracting semantic features via neural networks for extractive summarization. These semantic latent features have proven their effectiveness in the field of natural language processing. CRSum [6] is a recent example that obtains the best performance for extractive summarization on the three datasets that are shown in Table 1. However, the traditional methods may ignore the relationships of the contexts between sentences that are able to improve the sentence selection decisions in the final summarization. We argue that sentence importance also depends on the contextual relations with its surrounding sentences.

Table 1 Multidocument summarization

Dataset	Approach	ROUGE-1	ROUGE-2
DUC 2001	t-SR	34.82	7.76
	CRSum	35.36	8.30
	Upper bound	40.82	14.76
DUC 2002	t-SR	37.33	8.98
	CRSum	37.10	9.29
	Upper bound	43.78	15.97
DUC 2004	t-SR	37.74	9.60
	CRSum	38.19	9.66
	Upper bound	41.75	13.73

ROUGE (%) is used to measure the sentence regression approach using greedy-based sentence selection. The upper bounds are determined by scoring the sentences using human-written summaries

Motivated by the above intuition, we present a hybrid neural model, namely the Contextualized-Representation Hierarchical-Attention Summarization (CRHASum) network, for extractive text summarization to fully learn the contextual semantic information and relation features. In this structure, we first expand the contextualized embedding according to the linguistic context, which has the ability to capture the syntax and semantic relationships to effectively extract the text summarization. Then, we design a two-level hierarchical attention mechanism (word-level and sentence-level) to differentially focus on more and less significant context with a given sentence when constructing the sentence and context representations using the Bi-GRU model. Finally, CRHASum jointly learns the sentence and context representations between the given sentence and its previous/following context by considering the capacity of this sentence in order to summarize its context. We carry out extensive experiments on the DUC 2001, 2002 and 2004 multidocument summarization datasets. The experimental results demonstrate that our proposed CRHASum model is able to achieve comparable performance to the state-of-the-art baselines. By using a combination of a few basic surface features (SF), CRHASum + SF bests the other approaches on the ROUGE metrics.

In summary, the main contributions of our work are as follows.

- The contextualized representation is capable of capturing the syntax and semantic relationships to understand and learn the linguistic contexts by using word representations, which can effectively extract the text summarization.
- The hierarchical attention mechanism enables our proposed CRHASum model to pay more attention to the significant context at the word-level and sentence-level in the surrounding context of a given sentence.

2 Related works

We introduce the related work on extractive summarization in three categories that are discussed as follows.

2.1 Unsupervised methods

Sentences were scored using unsupervised techniques in early studies on extractive summarization. Radev et al. [10]

built cluster centroids and proposed the MEAD summarization system. A centroid-based method was presented by Mihalcea [11] that applied the sentence centrality to measure the importance. The LexRank approach calculated the sentence importance in a graph of sentence similarities according to the eigenvector centrality [12, 13]. Wan and Yang [14] proposed a centroid-based method for text summarization. The maximum marginal relevance (MMR) was applied by Goldstein et al. [15] to generate extractive text summarizations that incorporated additional information about the document and relations between documents. Lin and Bilmes [16] presented a variant of the MMR architecture that introduced the linear trade-off of the coverage and redundancy in order to maximize an objective function.

2.2 Feature engineering-based methods

Feature-based approaches rely on discriminative features to build statistical models, which have been widely used to better measure sentence importance. To decide whether to include a sentence in the final summarization, Kupiec et al. [17] built a Naïve Bayes classifier to estimate the importance of the given sentence. Li et al. [18] mainly used the support vector regression to identify the criticality, and then leveraged the rules to remove the redundant phrases when extracting the summarization. A PPSGen approach was proposed by Hu and Wan [19] in order to automatically extract presentation slides that selected and aligned critical phrases and sentences. Gillick and Favre [20] employed bi-gram features to obtain the importance and then applied the scores to measure the sentence importance and redundancy according to different linear combinations. Lin and Bilmes [16] presented a structural SVM training method to get the weights of features and used a MMR-like submodularity function to extract a summarization. These above approaches mainly depend on human-constructed features. However, these features are mostly surface features that do not consider contextual information and the relations of summarizations.

2.3 Neural network-based methods

With the development of deep learning models, some researchers have applied neural networks to text summarization. We mainly focus on the applications of neural network models for extractive summarization. Kobayashi et al. [21] leveraged the sum of pretrained word representations to represent sentences or documents, which considered the summarization task as a problem of maximizing a submodular function according to the similarities of representations. Yin and Pei [22] proposed a convolutional neural network model called the CNNLM to build the sentences into dense distributed semantic embeddings, and then determine the sentence redundancy based on the cosine similarity function. Cao et al. [3] constructed a summarization PriorSum system that applied enhanced convolutional neural networks to capture the summarized prior features from variable phrases. Ren et al. [6] developed a contextual relation-based summarization (CRSum) model, which used a two-level attention to pay more attention to important content, to improve the sentence regression performance. A general framework was built by Isonuma et al. [9] for single-document summarization, and a multitask learning approach was proposed to settle with document classification. Feng et al. [5] introduced an attentive encoder-based summarization model to generate article summarizations, and the method produced a rich document representation by considering the global information and the relationships of sentences in the text. However, the above methods do not introduce the context information from linguistic embedding and the relationships from the sentences in a document.

Neural network approaches based on attention mechanisms are also associated with our work. They have attracted significant research interest and attention for capturing the entire semantic relationships to improve the natural language processing performance. Attention-based models have been widely used in a few domains, such as machine translation [23], text classification [24, 25] and question answering [26, 27] and generation [28, 29]. Hence, an attention-based neural network is also applied to extract text summarizations.

In this work, we present an end-to-end approach Contextualized-Representation Hierarchical-Attention Summarization (CRHASum) network for extracting text summarizations. It considers the contextual embedding and applies the hierarchical attention mechanism to generate extractive text summarizations.

3 Methodology

In this section, we describe the details of our presented CRHASum architecture that is shown in Fig. 1. This working CRHASum model is composed of two components, the word-level attention understanding module and the sentence-level attention understanding module, as follows.

3.1 Word-level attention understanding module

This module uses three parts to extract attractive word-level features, including the Contextualized Word Representation, the Word Encoding Layer and the Word-level Attention Layer.

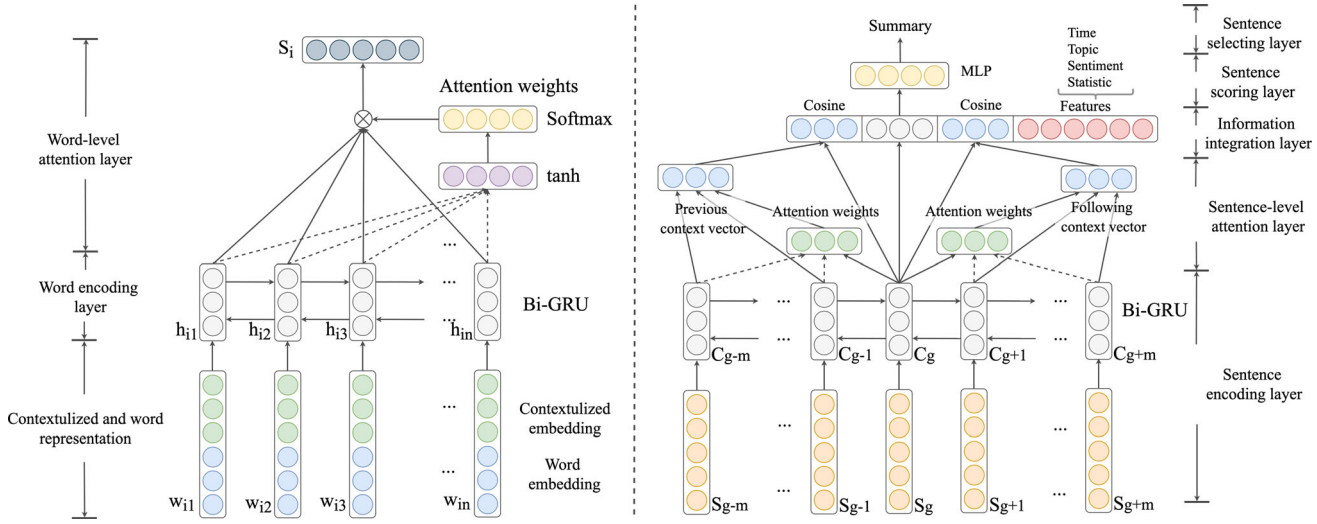

Fig. 1 The overall architecture of our proposed Contextualized-Representation Hierarchical-Attention Summarization approach

3.1.1 Contextualized and Word representation

We use the linguistic context to learn and expand the contextual embedding that is combined with the word embedding.

3.1.2 Word encoding layer

We subject the words of each sentence to a unified semantic representation using a Bi-GRU for the left-to-right and right-to-left directions.

3.1.3 Word-level attention layer

We capture the attractive and meaningful information to generate the extractive summarization using a word-level attention mechanism.

3.2 Sentence-level attention understanding module

This module uses five parts to leverage the meaningful sentence-level features and useful designed features to evaluate the importance of given sentence, and it then selects a set of sentences in order to produce the extractive summarization. The module includes the Sentence Encoding Layer, the Sentence-level Attention Layer, the Information Integration Layer, the Sentence Scoring Layer and the Sentence Selecting Layer.

3.2.1 Sentence encoding layer

We leverage the given sentence representation to incorporate the contextual relationships between the previous sentences and the following sentences.

3.2.2 Sentence-level attention layer

We extract the semantic and attentive sentences to produce the final summarization using the sentence-level attention mechanism.

3.2.3 Information integration layer

We exploit the given sentence representation, the cosine similarity between the previous/following context and the current sentence, and the surface features to measure the importance of this certain sentence.

3.2.4 Sentence scoring layer

We adopt a multilayer perceptron (MLP) to compute the score that evaluates the sentence importance using the input integrated information.

3.2.5 Sentence selecting layer

We apply the greedy method to choose the sentences to generate the extractive text summarization.

The detailed Algorithm 1 (Contextualized-Representation Hierarchical-Attention Extractive Summarization Algorithm) is given as follows.

Algorithm 1 Contextualized-Representation Hierarchical-Attention Extractive Summarization Algorithm

Input: Let the word representation E_w and contextualized representation E_c be the input features, which will be computed by Eqs.(1), (2) and (3);

Output: return the extractive text summarization.

1: **for** iteration t **do**

2: The word encoding features h_t are extracted from the left-to-right and right-to-left directions using the GRU and Eqs.(4) and (5) are used to combine the word representation and contextualized representation inputs.

3: Use Eq.(6) to extract the attractive word-level information j_t according to the word-level attention.

4: The sentence encoding semantic contextual-level features c_t are captured by the Bi-GRU.

5: Eq. (7) is used to obtain the sentence-level weighted attention feature vectors p_t.

6: The given sentence representation, cosine similarity and surface features are integrated.

7: Eq.(8) is used to get the score s_t and measure the importance for each given sentence.

8: The greedy algorithm is used to choose the final sentences to generate the summarization.

9: Update the parameters of the model using the loss function Eq.(9) and the MSE method.

10: **end for**

3.3 Word-level attention understanding module

3.3.1 Contextualized and Word representation

As we know, a pretrained word embedding [30, 31] is an important component to natural language processing performance. In the following, we mainly introduce the word representation and contextualized representation.

3.3.2 Word representation

Word embedding is capable of discovering the latent dense distributed regularities in the vector space among words. We also apply word embeddings as the surface features to describe the extractive summarization. GloVe [31] is applied to initialize the word embedding and fine-tune the whole learning procedure for our summarization task. Hence, we produce the word representation E_w and its dimension is d_w.

3.3.3 Contextualized representation

The semantic relationship in the context is a critical challenge for extracting text summarizations, which requires a high-quality representation of the linguistic context to introduce sufficient syntax and semantic elements. Therefore, ELMo [32] is applied to form the contextualized representation as a function of the entire input sentence. This model is composed of two-layer Bi-LSTM combined with feature convolutions from a large pretrained scale.

We first define a forward language model in order to obtain the probability for the sequence $(t_1, t_2, \ldots, t_N)$, which builds the given history sequence $(t_1, t_2, \ldots, t_{k-1})$ with token t_k. The formula is as follows.

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k | t_1, t_2, \ldots, t_{k-1}) \quad (1)$$

Recently, the neural language model has been found to be capable of representing a contextual independent representation and transmitting the representation by adopting L layers for the forward LSTM. Each LSTM model is capable of outputting a contextual-dependent representation $\vec{h}_{k,j}$ for each position k, $j = 1,\ldots, L$. The output of the top layer LSTM $\vec{h}_{k,L}$ is mainly used to predict the next token t_{k+1} based on a softmax function.

Meanwhile, a backward LM layer predicts the previous token according to the given following context, except that it reverses the sequence, which is similar to forwarding the LM in the following.

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k | t_{k+1}, t_{k+2}, \ldots, t_N) \tag{2}$$

It can be obtained similar to the forward LM, which leverages each backward LSTM in a L layer deep model to produce the representations $\overleftarrow{h}_{k,L}$ of token t_k according to a given context (t_{k+1}, t_{k+2}, …, t_N). Hence, the forward LM and backward LM are the components of biLM that are used in order to maximize the log likelihood.

Generally, intermediate layer representations are combined into ELMo in a biLM. For each token t_k, an L layer biLM can generate a set of $2L + 1$ representations to present the context information as follows.

$$E_{\mathrm{C}} = \{x_k, \vec{h}_{k,j}, \overleftarrow{h}_{k,j} | j = 1, \ldots, L\} = \left\{h_{k,j}^{\mathrm{LM}}\right\} \tag{3}$$

where $\vec{h}_{k,L}$ is a forward biLSTM layer and $\overleftarrow{h}_{k,L}$ is a backward biLSTM layer.

Therefore, the contextualized representation E_{c} is used to settle the complex characteristics of the word usage in large linguistic contexts to extract text summarizations.

3.3.4 Word encoding layer

The goal of this layer is mainly to extract the informative and semantic word-level features for extractive summarization. As we know, the Recurrent Neural Network (RNN) is an effective way to largely improve the sequence modeling performance; it has been widely used in the field of natural language processing. With the combination of the contextualized representation E_{c} and word representation E_{w} as the input, a particular effective implementation of the sequence model RNN called the Gated Recurrent Units (GRU) [33] is applied in order to determine the word sequence to model the word-level features.

In this framework, this GRU is composed of a reset gate z_t and an update gate r_t at each step t, and it generates a current input x_t and previous hidden state h_{t-1} to produce the immediate hidden state h_t = GRU (x_t, h_{t-1}) as follows.

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z) \\ r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r) \\ \tilde{h}_t &= \tanh\left(W_h x_t + r_t^{\circ} U_h h_{t-1} + b_h\right) \\ h_t &= z_t^{\circ} h_{t-1} + (1 - z_t)^{\circ} h_t \end{aligned} \tag{4}$$

where z_t denotes the reset gate, r_t denotes the update gate, σ is the sigmoid activation function, tanh is the tanh activation function, x_t is the combined input of the contextualized representation and word representation at step t, $\{W_z, W_r, W_h, U_z, U_r, U_h\}$ are the training weights that are updated during the learning process, $\{b_z, b_{\mathrm{r}}, b_h\}$ are the training bias terms that are used in order to randomly initialize the vectors, and $^{\circ}$ represents the element-multiplication operation.

Generally, the bidirectional Gated Recurrent Unit (Bi-GRU) is more suitable for understanding and capturing the semantic information in the context from two directions, and it consists of a left-to-right direction GRU and a right-to-left direction GRU. The formulas are as follows.

$$\begin{aligned} \overrightarrow{h_t} &= \mathrm{GRU}_{\mathrm{ltr}}\left(x_t, \vec{h}_{t-1}\right) \\ \overleftarrow{h_t} &= \mathrm{GRU}_{\mathrm{rtl}}\left(x_t, \overleftarrow{h}_{t-1}\right) \\ h_t &= \left[\overrightarrow{h_t}; \overleftarrow{h_t}\right] \end{aligned} \tag{5}$$

where $\overrightarrow{h_t}$ is the left-to-right GRU model, $\overleftarrow{h_t}$ is the right-to-left GRU model, and h_t is the combination of the two directions of the GRU states.

At last, we employ the Bi-GRU to encode a sentence in order to capture the word-level features as the combined contextualized embedding and word embedding input.

3.3.5 Word-level attention layer

Generally, all the words in a sentence do not contain equal information. Certain words are more meaningful and significant for the extractive summarization. Therefore, our model would set different weights according to the importance of each word. To learn and assign the weights among words in a sentence, we leverage word-level attention to capture the word-level features. The formulas are as follows.

$$\begin{aligned} u_t &= (V_w)^{\mathrm{T}} \tanh(W_w \cdot [h_t] + b_w) \\ \alpha_{ti} &= \mathrm{softmax}(u_t) = \frac{\exp(u_{ti})}{\sum_j \exp(u_{tj})} \\ j_t &= \sum_i \alpha_{ti} h_i \end{aligned} \tag{6}$$

where h_t is the word encoding vector of the Bi-GRU, W_w is the weight matrix and b_w is the bias terms that need to be updated during the training procedure, V_w is a weight

vector, $(V_w)^{\mathrm{T}}$ denotes its transpose vector, and j_t is the hidden state vector in the word encoding layer that is used for text summarization to extract the word-level features.

3.4 Sentence-level attention understanding module

3.4.1 Sentence encoding layer

As we know, the context of the given sentence is an important clue when constructing and summarizing text [34, 35]. This layer aims to aggregate the word-level features into sentence-level representations to capture the contextual relationships in a summary. Here, we also employ the word-level attractive representation j_t to incorporate the contextual information. As a result, we transform the sentence vector j_t into the contextual sentence-level vector c_t using the Bi-GRU model, which are then the inputs of the sentence-level attention layer.

3.4.2 Sentence-level attention layer

Different sentences make different contributions to the final extractive summarization, which does not rely on the Bi-GRU to mine the important parts of the summary. To address this problem, we apply a sentence-level attention mechanism to capture the significant and informative elements in each sentence vector c_t and the given sentence vector g_t. The formulas are as follows.

$$\begin{aligned} u_t &= (V_w)^{\mathrm{T}}\tanh\left(W_w \cdot \left[c_t; c_g\right] + b_w\right) \\ \alpha_{ti} &= \mathrm{softmax}(u_t) = \frac{\exp(u_{ti})}{\sum_j \exp\left(u_{tj}\right)} \\ p_t &= \sum_i \alpha_{ti} c_i \end{aligned} \tag{7}$$

where c_t is the sentence encoding vector from the Bi-GRU, c_g is the given sentence vector, W_w is the weight matrix and b_w is the bias terms that are used to update the parameters during the learning procedure, V_w is a weight vector, $(V_w)^{\mathrm{T}}$ is its transpose vector, and p_t is the sentence-level representation based on the attention vectors α_i.

According to the relative position between this given sentence and its context, we define the context before this sentence as the previous context and define the context after it as the following context. Then, we build the sentence-level attention mechanism between the previous/following context and this given sentence. Finally, we obtain the sentence-level representations $\overrightarrow{p_t}$ and $\overleftarrow{p_t}$ for the previous context and following context, respectively.

3.4.3 Information integration layer

This layer concatenates this given sentence vector with the cosine similarity and basic surface features in order to evaluate the importance of this given sentence from different dimensions.

The given sentence vector. It is mainly the current vector based on the sentence encoding layer, where c_g represents the semantic information.

Cosine similarity. We compute the cosine similarity between the previous context $\overrightarrow{p_t}$ and the given sentence vector c_g. Meanwhile, we also calculate the cosine similarity between the following context $\overleftarrow{p_t}$ and this current sentence vector c_g.

Surface features. Here, we mainly use four types of features, including the time, topic, sentiment and statistical features. (1) Time feature. Text summarization of news is a greater concern for people, and strong timeliness is necessary. Therefore, we extract the time information. (2) Topic feature. The title is a key clue to representing this news. We obtain the topic feature using the LDA model. (3) Sentiment feature. News always expresses some opinions and sentiment. We apply SenticNet to identify the sentiment features, which provides the sentiment polarity and sentics. (4) Statistical feature. We use the sentence length, position, average term frequency and average document frequency as the effective statistical features.

3.4.4 Sentence scoring layer

This layer is used to measure the score of a given sentence. As the inputs of the information integration layer, we adopt the MLP as a decoder and convert the information into a score to evaluate the importance of the current sentence. The formula is as follows.

$$f(S_t|\theta) = \mathrm{MLP}\left(\begin{bmatrix} \cos\left(\overrightarrow{p_t}, c_g\right) \\ \cos\left(\overleftarrow{p_t}, c_g\right) \\ g_t \\ \text{features} \end{bmatrix}\right) \tag{8}$$

where s_t is the current sentence that needs to measure the importance score, and θ is the parameters of neural network. The MLP is a three-layer hidden network that adopts tanh as active function, and the sizes of the layers are 100, 30 and 1. We increase the dimension and size of layers, but it has little influence on improving the performance based on our experiments. The features are the basic and surface ones, such as the time, topic, sentiment and statistic features.

Following existing works [3, 4], we apply the traditional Mean Square Error (MSE) as the loss function to train our proposed CRHASum model. The formulas are as follows.

$$L(\theta) = \frac{1}{|C| \cdot |D|} \sum_{D \varepsilon C} \sum_{S_t \varepsilon D} \mathrm{Err}(S_t)$$
$$\mathrm{Err}(S_t) = (f(S_t|\theta) - \mathrm{ROUGE} \sim 2(S_t|S_{\mathrm{ref}}))^2 \quad (9)$$

where c is the set of the whole documents, and s_t is the current sentence that needs to have its importance score measured.

3.4.5 Sentence selection layer

This layer is used to choose the final sentences to generate the summarization. Thus, we apply the Greedy algorithm to measure the trade-off between the performance and computation costs. First, this method uses the sentence that obtains the highest score. We add a new sentence s_i to the summary R if this sentence satisfies the following two conditions in each step.

(1) It achieves the highest score in the remaining sentences.

(2) $\frac{\mathrm{bi\_gram\_overlap}(S_i,R)}{f_{\mathrm{len}}(S_i)} \leq 1 - \lambda$, where $\mathrm{bi\_gram\_overlap}(S_i, R)$ is the count of the bi-gram overlap between the candidate sentence s_i and the current summary R.

When the length constraint of the summarization is reached, the method is terminated. We discuss the setting of parameter λ in our experiments.

4 Experiments and evaluation

We list the datasets, experimental implementation details and metrics in Sect. 4.1; we evaluate the performances of our proposed model in Sect. 4.2, and we introduce the detailed analysis in Sect. 4.3.

4.1 Experimental settings

In this subsection, we introduce the datasets, experimental implementation details and evaluation metrics in detail.

4.1.1 Dataset

To evaluate the performance, we apply a popular dataset that was made available by the Document Understanding Conference (DUC).1[1] The DUC 2001, 2002 and 2004 corpora are multidocument summarization datasets, which are almost all from the news domain and grouped into thematic clusters. We concatenate the whole articles and then split them into sentences via the tool that was provided with the DUC 2003 dataset for each document cluster. As with existing studies [3, 6], we also train our models using 2 years of data and test them using the third. The statistical information is shown in Table 2.

4.1.2 Evaluation metrics

The ROUGE metrics are the official metrics of the DUC extractive summarization tasks, which were proposed by Rankel et al. [36]. We leverage the standard ROUGE tool[2] [37] to measure the performance of other baselines and our method. It uses the n-gram information between the standard summary and candidate summary as the evaluation basis. Meanwhile, the length constraint is "–l 100" for DUC 2001 and DUC 2002 and "–l 665" for DUC 2004. The ROUGE-2 recall is used as the main comparison metric since it can reflect the effectiveness and advancement when evaluating automatic extractive summarization systems [38]. The formula is as follows.

$$\mathrm{ROUGE\_N} = \frac{\sum_{S \in \{\mathrm{ReferenceSummaries}\}} \sum_{\mathrm{gram}_n \in S} \mathrm{Count}_{\mathrm{match}}(\mathrm{gram}_n)}{\sum_{S \in \{\mathrm{ReferenceSummaries}\}} \sum_{\mathrm{gram}_n \in S} \mathrm{Count}(\mathrm{gram}_n)} \quad (10)$$

where n is the length of the n-gram, {Reference Summaries} is the set of standard summaries, $\mathrm{Count}_{\mathrm{match}}(\mathrm{gram}_n)$ denotes the concurrent length of the n-gram between the standard summary and the candidate summary, and $\mathrm{Count}(\mathrm{gram}_n)$ is the length of the n-gram of the standard summary.

4.1.3 Experimental implementation details

In our experiments, we apply the Stanford CoreNLP[3] to tokenize the sentences. The GloVe model is used to initialize the word embeddings. We select the dimension of 200 and compare the results the dimensions of 50 and 100. For an unknown word, we adopt random initialization to set its vector according to a uniform distribution U(-0.25, 0.25). All the word representations need to be fine-tuned during the training process. Moreover, ELMo [32] is applied to determine the contextualized representation and its dimension is 512. The number of units of the Bi-GRU model is set to 150. The MLP is a three-layer hidden layer, and tanh is used as the activation function. The hidden unit sizes are 100, 30 and 1. We perform the dropout operation as a regularization approach in order to reduce overfitting during training, and the dropout rate is set to 0.5 [39]. To learn the weights of our model, we employ the AdaGrad

[1] http://duc.nist.gov/.

[2] ROUGE-1.5.5 with options: $-n$ 2 $-m$ $-u$ $-c$ 95 $-x$ $-r$ 1000 $-f$ A $-p$ 0.5 $-t$ 0.

[3] http://stanfordnlp.github.io/CoreNLP/.

Table 2 The statistic information on DUC dataset

Dataset	Article	Sentence	Limitation
2001	30	11,295	100
2002	59	15,878	100
2004	50	13,070	665

[40] as an optimization method, and the mini-batch size is set to 20.

4.2 Experimental results

The baseline approaches are employed to comparatively evaluate the performance of the extractive summarization. Here, they are listed as follows.

- Upper bound: This method uses the upper limit result of the sentence regression, which is computed using the standard artificial summarization and is provided by the DUC dataset.
- LexRank: LexRank is a centroid-based lexical centrality method that is salient in text summarization.
- T-SR: It is an extractive summarization method based on the sentence regression, which uses the sentence length, position and TF/IDF statistic information as the features.
- PriorSum: It is a deep learning approach that adopts the CNN to encode the feature vector for each sentence, and combines it with the basis features to regress sentences for extractive summarization.
- CRSum: This approach leverages an attention-based neural network model in order to understand the context for generating summarizations.
- Ours: We propose a Contextualized-Representation Hierarchical-Attention Summarization network (CRHASum) that introduces the context information and relationships between sentences to extract text summarizations.

We compare the above approaches with CRHASum in Table 3. We find the following.

(1) The LexRank approach obtains worse performance than the other feature engineering methods and deep learning methods, meaning that the unsupervised method has some limitations when extracting summarizations. Meanwhile, it shows that traditional features and semantic features are both helpful. The other approaches based on neural networks are largely better than LexRank and t-SR, which verifies that semantic features are able to automatically produce a comprehensive summary.

Table 3 Comparison with existing methods

Dataset	Method	ROUGE-1	ROUGE-2
DUC 2001	LexRank	33.43	6.09
	t-SR*	35.41	7.76
	PriorSum*	35.98	7.89
	CRSum*	35.36	8.30
	CRSum + SF*	36.54	8.75
	CRHASum	36.07	8.82
	CRHASum + SF	36.99	8.94
	Upper bound*	40.82	14.76
DUC 2002	LexRank	35.29	7.54
	t-SR*	37.49	8.95
	PriorSum*	36.63	8.97
	CRSum*	37.10	9.29
	CRSum + SF*	38.90	10.28
	CRHASum	38.25	9.57
	CRHASum + SF	39.72	10.63
	Upper bound*	43.78	15.97
DUC 2004	LexRank	37.87	8.88
	t-SR*	38.36	9.98
	PriorSum*	38.91	10.07
	CRSum*	38.19	9.66
	CRSum + SF*	39.53	10.60
	CRHASum	38.84	9.89
	CRHASum + SF	39.92	10.77
	Upper bound*	41.75	13.73

The results with superscript * are reported in [6]. The best results in each type are highlighted

(2) The CRSum method achieves better performance than the PriorSum baseline. The reason is that it uses the contextual information from the left-to-right direction and the right-to-left direction. Further, CRSum + SF stably outperforms the standard CRSum method because of the basis and useful features that represent the characteristics of the summary.

(3) Our proposed CRHASum model takes a further step toward emphasizing the importance of the contextualized representation by learning a pretrained large corpus and the contextual relationship between sentences via the hierarchical attention mechanism. It is clear that CRHASum + SF gets the best performance among all the baseline methods, which shows the effectiveness and advancement of the combination of semantic features and traditional features. With this design, we can well learn the representations with sufficient context and words whose collocations are helpful for extractive summarization. Moreover, the attention mechanism is an

efficient method for the context to improve the performance. It also inspires our future work to try more effective ways. Meanwhile, there is still a gap between our proposed model and the upper bound method with human constructions. These findings indicate that there are some challenges for extracting summarizations and that this task should receive more attention from research and industry.

4.3 Detailed analysis

We conduct extra experiments to perform a more detailed analysis of this task as follows.

4.3.1 Impacts of different representations

In this subsection, we use a series of models to verify the advancement and effectiveness of our contextualized representation ELMo. To measure the performance of the different embeddings, we apply the same Bi-GRU neural network and experimental parameters to conduct the following experiments. First, we only employ the random initialized embedding based on the Bi-GRU classifier. Then, we implement the second model GloVe to represent the dense semantic space. Next, we apply ELMo to form the contextualized embedding. Finally, we apply the combination of ELMo + GloVe to represent the dimensional semantic vector. Table 4 demonstrates the performances of all these methods.

From Table 4, we can see that all the pretrained word embedding methods obtain better performance than the random initialization method, which proves that a pretrained embedding matrix is necessary for extractive text summarization due to the scale of the dataset.

Table 4 Impact of different representations

Dataset	Method	ROUGE-1	ROUGE-2
DUC 2001	Random initialized	33.13	6.49
	GloVe	34.07	7.23
	ELMo	34.82	7.64
	ELMo + GloVe	35.01	7.87
DUC 2002	Random initialized	35.10	7.58
	GloVe	36.40	8.22
	ELMo	36.79	8.41
	ELMo + GloVe	37.13	8.57
DUC 2004	Random initialized	36.17	7.34
	GloVe	37.01	8.04
	ELMo	37.35	8.36
	ELMo + GloVe	37.52	8.51

For the contextualized representation ELMo, it outperforms random initialization but it is worse than GloVe. Compared with random initialization, ELMo uses the complex semantic characteristics of words' utility across linguistic contexts as contextualized embeddings.

As we expect, the combination of ELMo and GloVe obtains the best performance among all the approaches. The reason is that this hybrid structure adequately considers the effects of the contextualized representation and word representation, which contribute to generating the extractive summarization.

4.3.2 Impact of different CRHASum components

In this subsection, we assess the different roles of model components by comparing several subnetworks to validate the effectiveness of our proposed CRHASum model. Our proposed CRHASum model introduces the contextualized representation, word representation and two-layer attention mechanism to generate summarizations. All-CWSA only adopts the word representation. All-WSA denotes that it just explores the word representation and contextualized representation. All-WA does not contain the word encoding unit and word-level attention mechanism. All-SA neglects the sentence encoding part and sentence-level attention mechanism.

The performance is shown in Table 5. From the results, we have several observations.

As shown in Table 5, we can see the following. (1) The combination of word embedding and contextualized embedding can further improve the use of the contextual

Table 5 Impact of different CRHASum components

Dataset	Method	ROUGE-1	ROUGE-2
DUC 2001	All-CWSA	34.07	7.23
	All-WSA	35.01	7.87
	All-WA	35.48	8.04
	All-SA	35.61	8.19
	All (CRHASum)	36.07	8.82
DUC 2002	All-CWSA	36.40	8.22
	All-WSA	37.13	8.57
	All-WA	37.53	8.85
	All-SA	37.61	8.93
	All (CRHASum)	38.25	9.57
DUC 2004	All-CWSA	37.01	8.04
	All-WSA	37.52	8.51
	All-WA	37.99	8.97
	All-SA	38.06	9.02
	All (CRHASum)	38.84	9.89

pretrained information and the relationships between sentences for extracting summarizations. (2) Both the word-level and sentence-level attention mechanisms achieve good performance, which verifies that they can capture the significant parts to learn and generate the cores of the summarizations from word-level features and sentence-level features. Meanwhile, this mechanism is suitable for other natural language processing tasks. (3) It is clear that our proposed CRHASum model has the ability to extract good summarizations by using contextualized representations and the two-layer attention mechanism for this task.

4.3.3 Impacts of different features

We test the sentiment, time, topic and statistical features to assess the summarization performance. To evaluate the effectiveness of the different features, the Pearson correlation coefficient can represent the features to some extent. We examine the correlations of the basis surface features with the ground truth, as shown in Table 6.

From Table 6, we can find that our presented CRHASum model obtains the best Pearson correlation coefficient compared with other features, which verifies the effectiveness and advancement of our model for generating summarizations. For the four types of features, the sentiment and statistical features are important for extracting summaries, while the length, position, TF/IDF and sentiment are very helpful elements for this task.

4.3.4 Threshold parameter λ

For sentence selection, we greedily choose the sentence with the highest importance score to be used as the final summary until the length constraint is determined. During the learning process, λ is applied to avoid redundant sentences by abandoning sentences whose bi-gram overlap with already chosen sentences is larger than $1 - \lambda$. To investigate the sensitivity, we evaluate the performance of our proposed CRHASum + SF with the threshold parameter λ ranging from 0 to 0.9 with a step size of 0.05. We adopt ROUGE-2 to measure the results, which are shown in Fig. 2. Generally, it is obvious that the performance of our proposed model is not sensitive to the experimental setting of λ for values less than 0.8. When the parameter ranges from 0.65 to 0.75, our framework achieves the best performance at generating summarizations.

Table 6 Impact of different features

Feature	DUC2001	DUC2002	DUC2004
Time	0.14	0.19	0.22
Topic	0.16	0.21	0.24
Sentiment	0.17	0.23	0.30
Length	0.28	0.27	0.31
Position	0.27	0.30	0.39
Term frequency	0.17	0.23	0.38
Document frequency	0.32	0.35	0.40
CRHASum	0.43	0.41	0.52

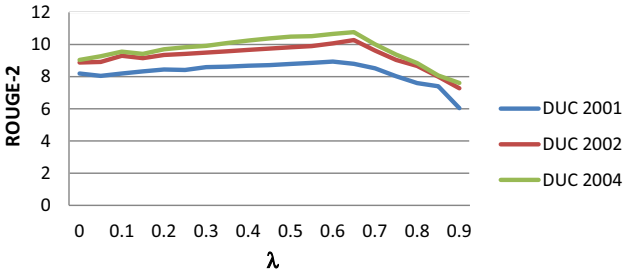

Fig. 2 Sensitivity to the parameter λ of CRHASum +SF during sentence selection layer

4.3.5 Case study

We present one example from the DUC 2002 dataset to illustrate our approaches. They are the upper bound, SF and CRHASum, which are shown in Fig. 3. Generally, the depth of the color corresponds to the importance of the sentences according to the certain method, where a red sentence is the most important and blue is the least important.

We can see that the combination of features in the SF method cannot significantly distinguish the importance of different sentences. This method wrongly estimates which of the two sentences is more critical, the fourth one or the fifth one. The reason is that SF does not use the dense semantic features to extract the keys of the summarization. Our proposed CRHASum model is better than SF since it considers the contextualized representation and context relationship using the two-layer attention mechanism. However, CRHASum is still limited when identifying the different degrees of importance compared with upper bound method, which needs to leverage more semantic information. This will be a topic for future work.

In the future, we will test suitable methods to incorporate external common sense knowledge bases to improve the performance of extracting text summarizations, which may be combined with different neural network architectures.

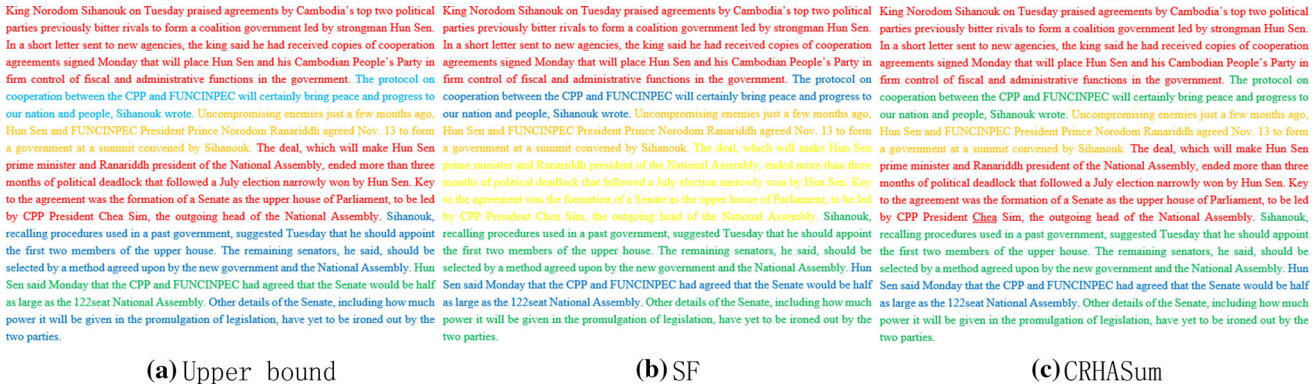

Fig. 3 An example among different methods

5 Conclusion and future work

In this paper, we propose a novel effective Contextualized-Representation Hierarchical-Attention Summarization network to deal with extractive text summarization. Based on the contextualized representation and word representation, this architecture models the word encoding, extracts the word-level features, builds the sentence encoding and then generates sentence-level features. With this design, it is capable of capturing the important information in order to extract the summarizations. The experimental results verify that our presented CRHASum model can learn effective semantic word-level and sentence-level features, and provide enough information to extract text summarizations.

In the future, we will test the feasibility of various ways to improve the performance of text summarization extraction and incorporate useful common sense. Moreover, we will explore effective novel neural networks to capture the semantic relationship between sentences for text summarization.

Acknowledgements This work is partially supported by grant from the Natural Science Foundation of China (Nos. 61632011, 61572102, 61702080, 61602079, 61806038), the Ministry of Education Humanities and Social Science Project (No. 16YJCZH12), the Fundamental Research Funds for the Central Universities (DUT18ZD102, DUT19RC(4)016), the National Key Research Development Program of China (No. 2018YFC0832101) and China Postdoctoral Science Foundation (No. 2018M631788).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Human and animal rights This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Chopra S, Auli M, Rush AM (2016) Abstractive sentence summarization with attentive recurrent neural networks. In: Proceedings of the 2016 conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp 93–98
2. Takase S, Suzuki J, Okazaki N et al (2016) Neural headline generation on abstract meaning representation. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp 1054–1059
3. Cao Z, Wei F, Li S, Li W, Zhou M, Wang H (2015) Learning summary prior representation for extractive summarization. In: ACL
4. Wan X, Cao Z, Wei F, Li S, Zhou M (2015) Multi-document summarization via discriminative summary reranking. CoRR
5. Feng C, Cai F, Chen H et al (2018) Attentive encoder-based extractive text summarization. In: Proceedings of the 27th ACM International conference on information and knowledge management. ACM, pp 1499–1502
6. Ren P, Chen Z, Ren Z et al (2017) 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, pp 95–104
7. Ren P, Wei F, Chen Z, Ma J, Zhou M (2016) A redundancy-aware sentence regression framework for extractive summarization. In: COLING
8. Wan X, Zhang J (2014) CTSUM: extracting more certain summaries for news articles. In: SIGIR
9. Isonuma M, Fujino T, Mori J et al (2017) Extractive summarization using multi-task learning with document classification. In: Proceedings of the 2017 Conference on empirical methods in natural language processing, pp 2101–2110
10. Radev DR, Jing H, Budzikowska M (2000) Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In: NAACL-ANLP
11. Mihalcea R (2004) Graph-based ranking algorithms for sentence extraction, applied to text summarization. In: ACL
12. Mihalcea R, Tarau P (2004) TextRank: bringing order into texts. In: EMNLP
13. Erkan G, Radev DR (2004) LexRank: graph-based lexical centrality as salience in text summarization. JAIR 22(1):457–479
14. Wan X, Yang J (2008) Multi-document summarization using cluster-based link analysis. In: SIGIR

15. Goldstein J, Mittal V, Carbonell J, Kantrowitz M (2000) Multi-document summarization by sentence extraction. In: NAACL-ANLP
16. Lin H, Bilmes J (2011) A class of submodular functions for document summarization. In: NAACL-HLT
17. Kupiec J, Pedersen J, Chen F (1995) A trainable document summarizer. In: SIGIR
18. Li S, Ouyang Y, Wang W, Sun B (2007) Multi-document summarization using support vector regression. In: DUC
19. Hu Y, Wan X (2015) PPSGen: learning-based presentation slides generation for academic papers. TKDE 27(4):1085–1097
20. Gillick D, Favre B (2009) A scalable global model for summarization. In: ILP-NLP
21. Kobayashi H, Noguchi M, Yatsuka T (2015) Summarization based on embedding distributions. In: EMNLP
22. Yin W, Pei Y (2015) Optimizing sentence modeling and selection for document summarization. In: IJCAI
23. Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. Comput Sci
24. Shen T, Zhou T, Long G et al (2018) Disan: directional self-attention network for rnn/cnn-free language understanding. In: AAAI
25. Du J, Xu R, He Y et al (2017) Stance classification with target-specific neural attention. In: Twenty-sixth international joint conference on artificial intelligence, pp 3988–3994
26. Gui L, Hu J, He Y et al (2017) A question answering approach to emotion cause extraction. arXiv:1708.05482
27. Lu J, Yang J, Batra D et al (2016) Hierarchical question-image co-attention for visual question answering. In: Advances in neural information processing systems, pp 289–297
28. Kim J, Kong D, Lee JH (2018) Self-attention-based message-relevant response generation for neural conversation model. arXiv:1805.08983
29. Fan A, Lewis M, Dauphin Y (2018) Hierarchical neural story generation. arXiv:1805.04833
30. Mikolov T, Chen K, Corrado G et al (2013) Efficient estimation of word representations in vector space. In: Proceedings of ICLR:1301.3781
31. Pennington J, Socher R, Manning C (2014) Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
32. Peters ME, Neumann M, Iyyer M et al (2018) Deep contextualized word representations
33. Chung J, Gulcehre C, Cho KH et al (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555
34. Zhang H, Li J, Ji Y et al (2017) Understanding subtitles by character-level sequence-to-sequence learning. IEEE Trans Ind Inf 13(2):616–624
35. Zhang H, Wang S, Mingbo Z et al (2018) Locality reconstruction models for book representation. IEEE Trans Knowl Data Eng 99:1
36. Rankel PA, Conroy JM, Dang HT, Nenkova A (2013) A decade of automatic content evaluation of news summaries: reassessing the state of the art. In: ACL
37. Lin C-Y (2004) Rouge: a package for automatic evaluation of summaries. In: ACL
38. Owczarzak K, Conroy JM, Dang HT, Nenkova A (2012) An assessment of the accuracy of automatic evaluation in summarization. In: NAACL-HLT
39. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overing. JMLR 15(1):1929–1958
40. Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. JMLR 12:2121–2159