Multimed Tools Appl (2019) 78:857–875
https://doi.org/10.1007/s11042-018-5749-3

Abstractive text summarization using LSTM-CNN based deep learning

Shengli Song[1,2] · **Haitao Huang**[2] · **Tongxiao Ruan**[2]

Received: 23 November 2017 / Revised: 12 January 2018 / Accepted: 2 February 2018 /
Published online: 16 February 2018

Abstract Abstractive Text Summarization (ATS), which is the task of constructing summary sentences by merging facts from different source sentences and condensing them into a shorter representation while preserving information content and overall meaning. It is very difficult and time consuming for human beings to manually summarize large documents of text. In this paper, we propose an LSTM-CNN based ATS framework (ATSDL) that can construct new sentences by exploring more fine-grained fragments than sentences, namely, semantic phrases. Different from existing abstraction based approaches, ATSDL is composed of two main stages, the first of which extracts phrases from source sentences and the second generates text summaries using deep learning. Experimental results on the datasets CNN and DailyMail show that our ATSDL framework outperforms the state-of-the-art models in terms of both semantics and syntactic structure, and achieves competitive results on manual linguistic quality evaluation.

Keywords Text mining · Abstractive text summarization · Relation extraction · Deep learning

1 Introduction

Text summarization (TS) is a task of generating a short summary consisting of a few sentences that capture salient ideas of a text [16]. Overcoming this task is an important step towards natural language understanding. Additionally, a concise and good summary can help humans comprehend the text content better in a very short time.

Based on previous studies, the text summarization can be broadly classified into two different categories [6]. One is Extractive Text Summarization (ETS), which uses traditional

✉ Shengli Song
xidiansls@163.com

1 Software Engineering Institute, Xidian University, Xi'an 710071, China

2 School of Computer Science and Technology, Xidian University, Xi'an 710071, China

approaches to generate summaries by cropping important segments of the original text and combining them to form a coherent summary. The other is Abstractive Text Summarization (ATS), which generates more qualitatively human-written sentences to generate summaries from scratch without being constrained to phrases from the original text.

ETS models were firstly proposed to mine the core semantic information in the original text and summarize it. More recently, with the improvement of computer performance and the development of deep-learning theory, ATS models that map an input sequence into another output sequence, called sequence-to-sequence models, have been proposed. ATS models have been successful in many problems such as machine translation and text summarization. In the framework of sequence-to-sequence models, a very relevant model for our task is the long short-term memory (LSTM) encoder-decoder model proposed in [11], which has produced state-of-the-art performance in machine translation, a natural language task. Despite the high similarities with machine translation, TS is a very different problem. In machine translation, the target output sequence is approximately as long as the original text. However, in text summarization, the output sequence is typically very short and does not depend very much on the length of the original text. In other words, a key challenge in text summarization is to optimally compress the original text in a lossy manner such that the key concepts in the original text are preserved, whereas, in machine translation, the translation is expected to be loss-less. In machine translation, there is a strong notion of almost one-to-one word-level alignment between source and target, but in text summarization, the alignment is less obvious.

Although the present TS models have achieved very good results in many recognized datasets, these models are not resolved all the problems. Semantics and syntactic structure are two important factors to evaluate TS models, but these two kinds of models focus on only one factor alone. The existing ETS models are coarse-grained and model summaries with some isolated sentences. The advantage of ETS models is that the sentences in the summaries must be in accordance with the requirements of the syntactic structure. However, the disadvantage of ETS models is that the sentences in the summaries may not be semantically coherent. The reason for this disadvantage is that the adjacent sentences in the summaries are not necessarily adjacent in the original text. The existing ATS models are fine-grained and model summaries with semantic items (word, phrases, etc.). The advantage of ATS models is inclusive semantics, because these models learn the collocation between words, and will generate a sequence of keywords based on the collocation between words after training. The disadvantage of ATS models is that this sequence of keywords is difficult to meet the requirement of syntactic structure. The other key problem of mainstream ATS models is rare words. The importance of a rare word will be determined by the number of occurrences of that word and the collocation of the words, but humans will use more factors to judge whether a word is important. Therefore, in some scenarios, some words that rarely appear would be considered not important, but a part of these words is important for summary construction in human’s view.

In order to solve the problems, we propose an LSTM-CNN based ATS framework, named ATSDL, and make the following main contributions in this work: (i) we apply LSTM model that was originally developed for machine translation to summarization, and combine CNN and LSTM together to improve the performance of TS. ATSDL considers both semantics and syntactic structure, which is different from existing ETS models and ATS models. ATSDL firstly applies phrase extraction method to extract key phrases from sentences, and then uses the LSTM model to learn the collocation of phrases. After training, the new model will generate a sequence of phrases. This sequence is the text summary that is composed of natural sentences. (ii) In order to solve the key problem of rare words, we use phrase location information, so we can generate more natural sentences. (iii) The experiment results show that ATSDL outperforms state-of-the-art abstractive and extractive summarization systems on both two different datasets.

The rest of the paper is organized as follows. In Section 2, we present both the sequence-to-sequence model and our novel model and then introduce the key idea of each model. Section 3 contextualizes the whole process and discuss the results of our experiments on two different data sets. Related work is discussed in Section 4. Finally, we conclude the paper with remarks on our future direction in Section 5.

2 ATS deep learning model

The sequence-to-sequence model is reviewed with detail analysis in this section, and then we describe our proposed ATSDL framework, the phrase based LSTM-CNN model.

2.1 Sequence-to-sequence model

The sequence-to-sequence model can be seen as the first and simplest ATS model, and many of the ATS models presented in this paper improve upon this base model [15]. According to Fig. 1, the architecture of the sequence-to-sequence model mainly consists of two parts - an encoder and a decoder - each of which is a Recurrent Neural Networks (RNN).

The encoder is fed as inputting the text of a news article one word of one time. Each word first passes through an embedding layer that transforms the word into a distributed representation. This distributed representation is then combined using a multi-layer neural network with the hidden layers generated after feeding in the previous word, or all 0’s for the first word in the text.

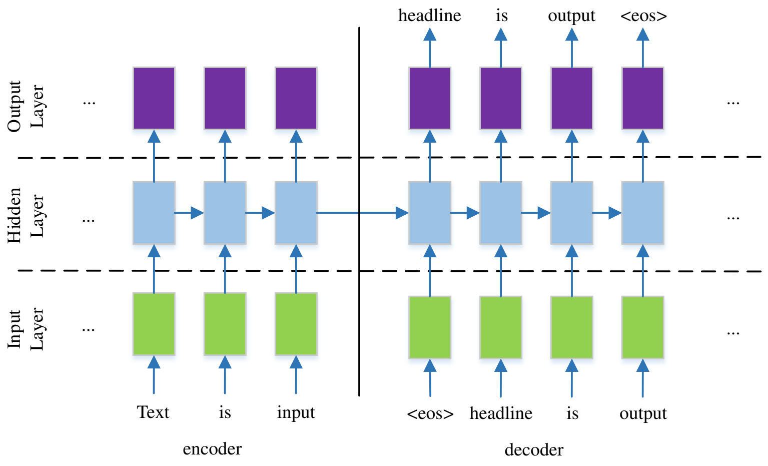


Fig. 1 Sequence-to-sequence model

The decoder takes as input the hidden layers generated after feeding in the last word of the input text. First, an end-of-sequence symbol is fed in as input, again using an embedding layer to transform the symbol into a distributed representation. Then, the decoder generates the text summaries, using a *softmax* layer and the attention mechanism described in the next section, each of the words of the headline, ending with an end-of-sequence symbol. After generating each word, the same word is fed in as input when generating the next word.

The loss function we use is the log loss function:

$$-\log p(y_1, \dots, y_T | x_1, \dots, x_T) = -\sum_{t=1}^{T} \log p(y_t | y_1, \dots, y_{t-1}, x_1, \dots, x_T) \tag{1}$$

Where y represents output words and x represents input words.

2.2 Phrases based LSTM-CNN model

In this section, we describe LSTM-CNN model from three perspectives: input and output, convolutional phrase encoder, and recurrent document decoder. The structure of proposed model is present in Fig. 2.

Input and Output: According to the sequence-to-sequence model, our model takes phrases rather than words as input and output sequences. The phrases are composed of several words to express the overall meaning, and it is a key concept in the field of phrase filling [2, 9] and phrase extraction [1]. The use of phrases enables our model to produce natural sentences. Phrases can be roughly divided into three major types: subject phrases, relational phrases and object phrases. Relational phrases are sentence fragments that express the relational information in the sentence, such as "is", "win prize for" and "discovery of". It is not confined to the predicate (verb phrase) in grammar. It may be comprised of a noun phrase, a copula (be), or a preposition, etc. Every relational phrase will be associated with two entities named subject phrase and object phrase respectively. Subject/object phrases are often composed of entities, noun phrases, adjectives, and objects that are associated with its relational phrase, such as "I," "Mary" and "Beijing". Subject phrases, relational phrases and object phrases can constitute a natural sentence, for example in "Mary wants to go home", the subject phrase is "Mary", the relational phrase is "wants to go" and the object phrase is "home".

In Fig. 2, before we input phrase sequences into ATSDL model for training, we will use phrase extraction method to decompose the sentences in the original text into the phrase sequence. Then, we input the phrase sequence in order and learn the collocation relation between phrases.

Convolutional phrase encoder: We opted for a convolutional neural network model for representing phrases for two reasons. Firstly, single layer CNNs can be trained effectively (without any long-term dependencies in the model) and secondly, CNNs have been successfully used for sentence-level classification tasks such as sentiment analysis. Let d denote the dimension of word embedding, and s a document phrase consisting of a sequence of n words ($w_1, \dots, w_n$) which can be represented by a dense column matrix $W \in \mathfrak{R}^{n \times d}$. We apply a temporal narrow convolution between W and a kernel $K \in \mathfrak{R}^{c \times d}$ of width c as follows:

$$f_j^i = \tanh\left(W_{j:j+c-1} \otimes K + b\right) \tag{2}$$

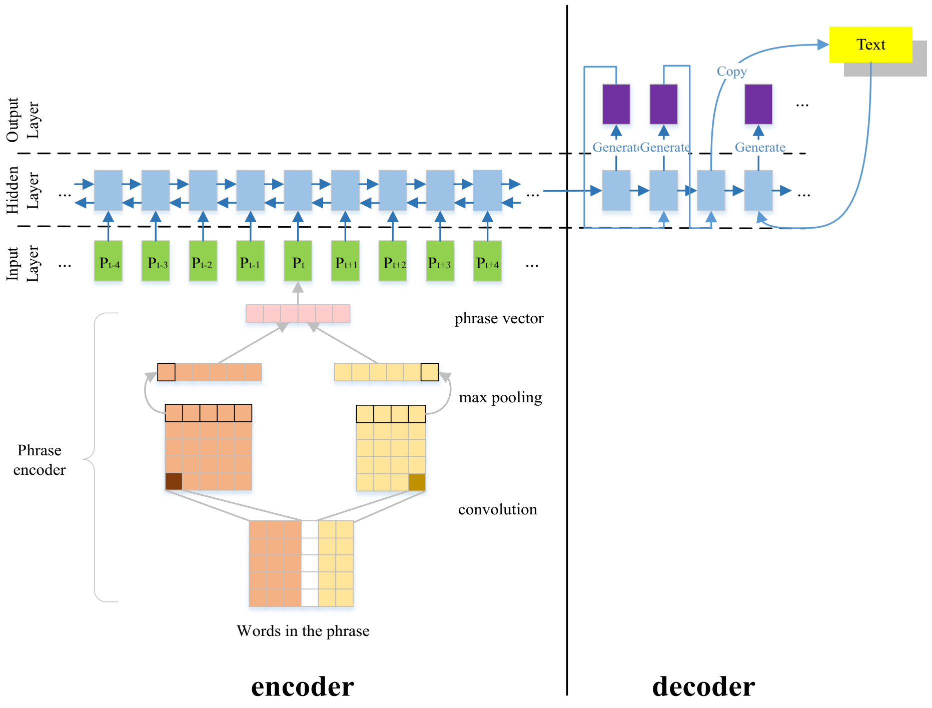


Fig. 2 Semantic units based LSTM model

Where ⊗ is the Hadamard Product followed by a sum over all elements. f^i_j denotes the j-th element of the i-th feature map f^i and b is the bias. We perform max pooling over time to obtain a *single* feature (the i-th feature) representing the phrase under the kernel K with width c:

$$s_{i,K} = \max_j f^i_j \tag{3}$$

In practice, we use multiple feature maps to compute a list of features that match the dimensionality of a phrase under each kernel width. In addition, we apply multiple kernels with different widths to obtain a set of different phrase vectors. Finally, we sum these phrase vectors to obtain the final phrase representation. The CNN model is schematically illustrated in Fig. 2. In the example, the phrase embeddings have six dimensions, so six feature maps are used under each kernel width. The yellow feature maps have width two and the orange feature maps have width three. The phrase embedding obtained under each kernel width are summed to get the final phrase representation (denoted by pink).

Recurrent document encoder: On document encoding, we do not restrict the encoders' architectures as if it is an RNN. The recent literature shows LSTM and gated recurrent units (GRU) are both good architectures and perform better than RNN. Although GRU has an advantage in training time, our model uses LSTM instead of GRU, because LSTM is easy to tune parameters and has a stronger theoretical guarantee.

In the simplest bi-directional setting, when reading input phrase sequences from left to right, the forward propagation of LSTM is computed as follows:

$$f_t = \sigma\left(W_f{}^*[h_{t-1}, x_t] + b_f\right) \tag{4}$$

$$i_t = \sigma\left(W_i{}^*[h_{t-1}, x_t] + b_i\right) \tag{5}$$

$$\underset{t}{\tilde{C}} = \tanh\left(W_C{}^*[h_{t-1}, x_t] + b_C\right) \tag{6}$$

$$C_t = f_t{}^* C_{t-1} + i_t{}^* \underset{t}{\tilde{C}} \tag{7}$$

$$o_t = \sigma\left(W_o{}^*[h_{t-1}, x_t] + b_o\right) \tag{8}$$

$$h_t = o_t{}^* \tanh(C_t) \tag{9}$$

Where W_f, W_i, W_C, $W_o \in R^{n \times m}$ weight matrices, n is the number of hidden units, * is matrix multiplication, σ is the sigmoid function and h_t is the output of LSTM.

Decoder: Based on the sequence-to-sequence model, our model divides decoder part into two different modes: generate mode and copy mode. According to Eq. (10),

$$\mathrm{y}^* = \left\{ \underset{\mathrm{y}}{\mathrm{argmax}} \prod_{\mathrm{t=1}}^{\mathrm{T}} \mathrm{p}(\mathrm{y_t}|\mathrm{y_1}, \ldots, \mathrm{y_{t-1}}, \mathrm{h}), \mathrm{if}|\max \prod_{\mathrm{t=1}}^{\mathrm{T}} \mathrm{p}(\mathrm{y_t}|\mathrm{y_1}, \ldots, \mathrm{y_{t-1}}, \mathrm{h})| > \delta\, \mathrm{x_i}, \mathrm{if}|\max \prod_{\mathrm{t=1}}^{\mathrm{T}} \mathrm{p}(\mathrm{y_t}|\mathrm{y_1}, \ldots, \mathrm{y_{t-1}}, \mathrm{h})| <= \delta \mathrm{and} \mathrm{x_{i-1}} = \mathrm{y_{t-1}} \right. \tag{10}$$

our model needs to calculate the conditional probability $\max \prod_{\mathrm{t=1}}^{\mathrm{T}} \mathrm{p}(\mathrm{y_t}|\mathrm{y_1}, \ldots, \mathrm{y_{t-1}}, \mathrm{h})$ first like the sequence-to-sequence model. When the absolute value of $\max \prod_{\mathrm{t=1}}^{\mathrm{T}} \mathrm{p}(\mathrm{y_t}|\mathrm{y_1}, \ldots, \mathrm{y_{t-1}}, \mathrm{h})$ is bigger than threshold δ, our model enters generate mode. In generate mode, the decoder predicts the next phrase y_t given all annotations obtained in encoding $h = h_1, \cdots, h_T$, as well as all previously predicted phrases $y_1, \cdots, y_{t-1}$. On the other hand, when the absolute value of $\max \prod_{\mathrm{t=1}}^{\mathrm{T}} \mathrm{p}(\mathrm{y_t}|\mathrm{y_1}, \ldots, \mathrm{y_{t-1}}, \mathrm{h})$ is no more than threshold δ, our model enters copy mode. In copy mode, we think that the phrase generated by generate mode may not be suited to match the current phrase, so we find the location of the current phrase in the original text and copy the next phrase into the summary. In this mode, the next phrase in the original text may be better than the phrase generated by generate mode from the human's view.

3 Experiment

In this section, we will first describe the experiment process in detail, and then briefly introduce the experiment datasets and evaluation methods. Last, we will conduct a comprehensive analysis of our experimental results and give some points of our model that can be improved in the future.

3.1 Experiment procedure

This part details the entire process flow of our model. According to Fig. 3, our model can be divide into three steps: text pre-processing, phrase extraction and text summary generation.

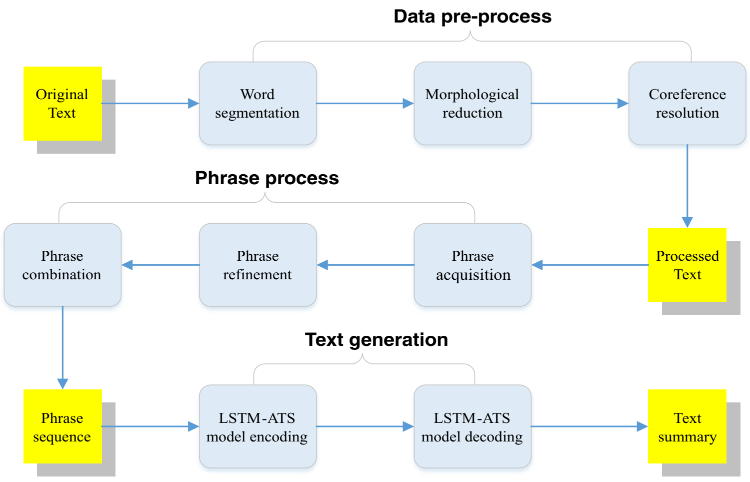
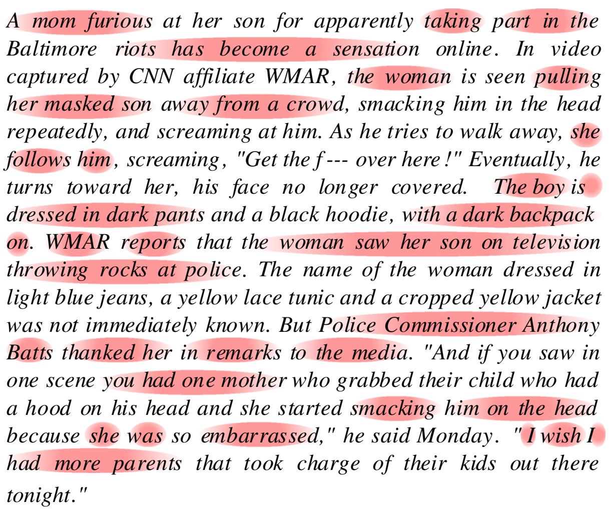


Fig. 3 Summary generation process using ATSDL

Here we use a news text selected from CNN dataset to illustrate our approach, and the original text is shown in Fig. 4 where we use red background to mark phrases.

3.1.1 *Text pre-processing*

We use the Stanford natural language process tool CoreNLP to pre-process original text because CoreNLP makes it very easy to apply a bunch of linguistic analysis tools to process text content. Data pre-processing includes word segmentation, morphological reduction, and coreference resolution. This phase is simple but necessary because reasonable text pre-processing is helpful for the following steps. For example, our model will treat "be", "is", "am" "are" "was" "were" and "been" as different phrases without morphological reduction, but these phrases have the same semantics, so it is better to use morphological reduction to



A mom furious at her son for apparently taking part in the Baltimore riots has become a sensation online. In video captured by CNN affiliate WMAR, the woman is seen pulling her masked son away from a crowd, smacking him in the head repeatedly, and screaming at him. As he tries to walk away, she follows him, screaming, "Get the f--- over here!" Eventually, he turns toward her, his face no longer covered. The boy is dressed in dark pants and a black hoodie, with a dark backpack on. WMAR reports that the woman saw her son on television throwing rocks at police. The name of the woman dressed in light blue jeans, a yellow lace tunic and a cropped yellow jacket was not immediately known. But Police Commissioner Anthony Batts thanked her in remarks to the media. "And if you saw in one scene you had one mother who grabbed their child who had a hood on his head and she started smacking him on the head because she was so embarrassed," he said Monday. "I wish I had more parents that took charge of their kids out there tonight."

Fig. 4 Original text

merge these phrases into one. In addition, there are many pronouns and demonstratives in the original text, and these words will cause ambiguity during our model training. In order to eliminate ambiguity, it is necessary to carry on coreference resolution before our model is trained. Using the original text in Fig. 4 as input, the pre-processing result is shown in Fig. 5.

3.1.2 Phrases extraction

Phrase extraction includes three sub-steps: Phrase acquisition, phrase refinement, and phrase combination. In Fig. 5, we also use red background to mark phrases.

After obtaining the result of pre-processing the original text, we use the phrase extraction method to acquire phrases. The task of this step is to extract as many phrases as possible. Therefore, a novel phrase extraction method called Multiple Order Semantic Parsing (MOSP) is proposed to construct a binary semantic multilayer structure to analyze multiple order semantics.

The process of MOSP for scattering and restructuring is shown in Fig. 6. Firstly, sentences are split into fragments with semantic association information which contains lexical and syntactic features in sentences. The Stanford NLP Parsing tools can do this step. Then, with the aid of the dependency parser, the semantic information is reorganized into a binary tree structure. Then, we identify an initial root node as a starting point, which is the relational phrase in the main clause. From the initial root, its associated relational argument pairs are assembled into a tree structure as a child node. If the current root has no child, construction of the semantic tree is completed. Otherwise, the child node will be as a new root node for recursive construction. Next, the dependency parsed finds if there is a compound clause. The relational phrase of other clauses are considered as “brothers” of the relational phrase of the main clause. For each “brother”, a conjunction node will be generated as a parent node to connect the its root and its brothers with its conjunction semantics. Then, the relational phrase of the brother node will be treated as a root node to construct the semantic tree for the compound clause. The constructed semantic tree in front may contain a single child subtree (only left child or only right child). Therefore, after the construction of the binary semantic tree, this tree will be traversed to upgrade to a strict binary tree, which is our MOS Tree. In the process of traversing, the optimization work mainly composes

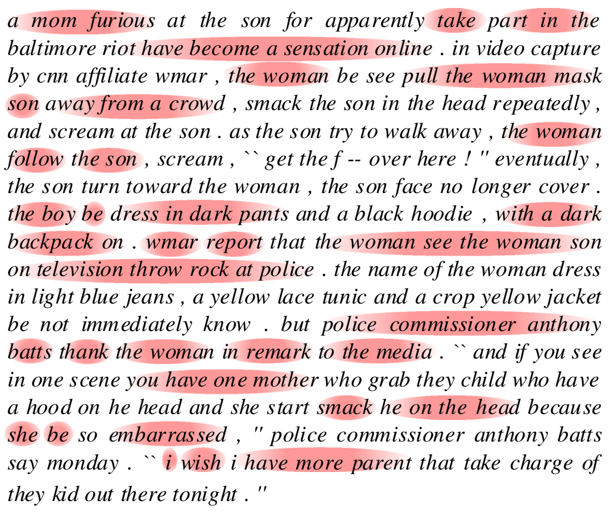

a mom furious at the son for apparently take part in the baltimore riot have become a sensation online . in video capture by cnn affiliate wmar , the woman be see pull the woman mask son away from a crowd , smack the son in the head repeatedly , and scream at the son . as the son try to walk away , the woman follow the son , scream , `` get the f -- over here ! '' eventually , the son turn toward the woman , the son face no longer cover . the boy be dress in dark pants and a black hoodie , with a dark backpack on . wmar report that the woman see the woman son on television throw rock at police . the name of the woman dress in light blue jeans , a yellow lace tunic and a crop yellow jacket be not immediately know . but police commissioner anthony batts thank the woman in remark to the media . `` and if you see in one scene you have one mother who grab they child who have a hood on he head and she start smack he on the head because she be so embarrassed , '' police commissioner anthony batts say monday . `` i wish i have more parent that take charge of they kid out there tonight . ''

Fig. 5 Processed text

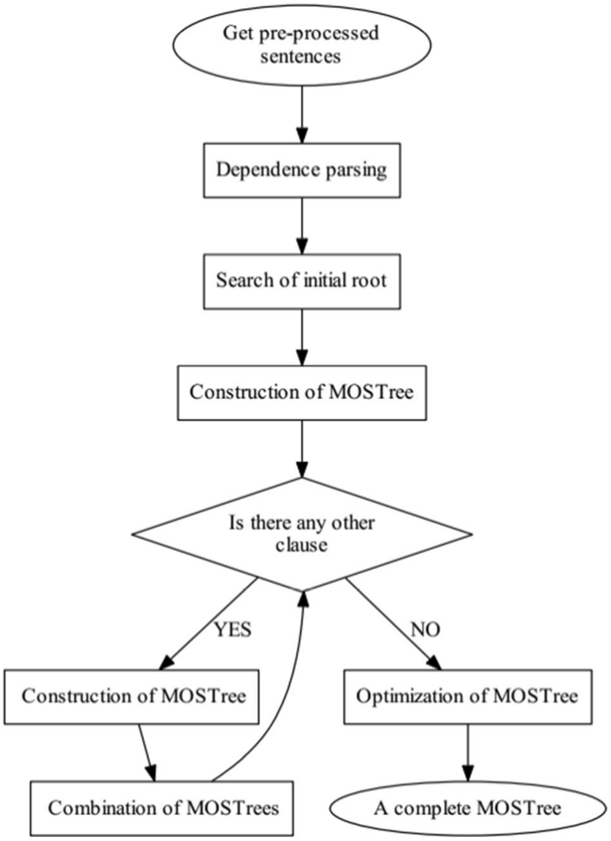


Fig. 6 Process of MOSP for scattering and restructuring

of two parts. We determine whether the subtree has complete semantic information. Different solutions to these two cases are designed to adjust the structure of the subtree. If it has binary semantics, the subtree will be processed to a strict binary tree. For example, some noun phrases may be constructed as a distinct right child structure. In this case, the root node will be broken down into two parts to add a left child of it. Otherwise, the semantics of these subtrees are not enough to split it into a binary structure. This situation often occurs in the subtree with a sole single left child. These subtrees with incomplete binary semantic information are processed to merge into a semantic object node as a "Leaf Node". After traversing and adjusting the tree, all non-strict subtrees are processed to be strict. In this MOS Tree, each subtree rooted by a non-conjunction node expresses a single semantic unit. Once again to traverse the tree, each semantic will be obtained as triples. For example, our phrase extraction method is applied to the source sentence "Madame Curie, born in Sarsaw in 1867, won the Nobel Prize for her discovery of Polonium and Radium." As shown in Fig. 7, MOSP provides a MOS tree of this sentence. Then, we traverse this tree and obtain six phrase triples.

Although MOSP is an excellent phrase extraction method, this method cannot guarantee that extracted phrases have the correct semantics and syntactic structure. Accordingly, we have to refine the extracted phrase that is redundant in semantics or syntactic structure before training model. After the analysis of the experimental results, we use some simple rules to delete wrong and repeated phrases. These rules include: (i) The compound-complex sentence usually includes

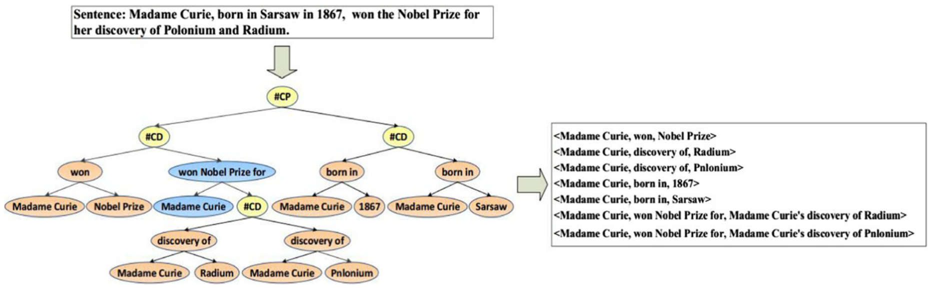

Sentence: Madame Curie, born in Sarsaw in 1867, won the Nobel Prize for her discovery of Polonium and Radium.
#CP
#CD
#CD
won
won Nobel Prize for
born in
born in
Madame Curie
Nobel Prize
Madame Curie
#CD
Madame Curie
1867
Madame Curie
Sarsaw
discovery of
discovery of
Madame Curie
Radium
Madame Curie
Pnlonium
<Madame Curie, won, Nobel Prize>
<Madame Curie, discovery of, Radium>
<Madame Curie, discovery of, Pnlonium>
<Madame Curie, born in, 1867>
<Madame Curie, born in, Sarsaw>
<Madame Curie, won Nobel Prize for, Madame Curie's discovery of Radium>
<Madame Curie, won Nobel Prize for, Madame Curie's discovery of Pnlonium>

Fig. 7 MOS tree of the sample sentence

more semantics than simple sentence. In order to simplify the process, we need to extract the semantic information including more semantics. So, if we can extract multiple phrase triples from one sentence, we only use the topmost phrase triples because they have the complete semantics. (ii) The noun always includes more conceptual information that others in common sense. If the subject phrase or object phrase contains several words, but no word is a noun, then we delete this phrases triples because the sentence may not be complete. (iii) If the relational phrase contains several words, but no word is a verb, then we delete these phrase triples. In addition to these three rules, there are some other additional rules that we do not enumerate in this paper.

After phrase refinement, the correct proportion of phrases will improve a little. Then, when the study was conducted, we found an interesting phenomenon that there would be a lot of phrases with similar semantics in the same text. For example, our model will treat "Sherlock Holmes", "Holmes" and "Sherlock" as three different phrases, but these three phrases will refer to the same person in the human view. If we can combine different phrases with similar semantics into one phrase, the redundancy of phrases is improved and this will benefit the training of LSTM-CNN model. Thus we refer to the English dictionary tool WordNet to complete this task. WordNet is a cognitive-linguistic English dictionary designed by psychologists, linguists and computer engineers at Princeton University. Here we provide some simple heuristics to help us determine whether two different phrases can combine into one phrase or not. (i) If phrase A is a part of phrase B, then we combine these two phrases into one phrase and use phrase B to replace phrase A. (ii) If phrase A is a hypernym of phrase B, then we combine these two phrases into one phrase and use phrase B to replace phrase A. (iii) If phrase A and phrase B have the same hypernym, then we combine these two phrases. In addition to these artificial rules, Additional ones that we do not enumerate in this paper. Table 1 shows the results of phrase extraction.

3.1.3 Text summary generation

After phrase extraction, we need to generate a summary by our LSTM-CNN model. We implement the model in the deep learning framework Torch and encode phrases as one-hot. On average, our model takes about 8 min to train a text (approximately 3 KB). Training this model can be tricky, so we provide some heuristics and tips for the optimization: (i) We use GPUs to accelerate the training of our model. (ii) the batch_size is an important parameter during model training. If the batch_size is too big, training will slow down. Otherwise, if the batch_size is too small, the model error may get stuck in a local optimum. Empirically, we find that model error will reduce to a minimum when we set the batch_size to make the ratio of the number of train batches and validate batches to 3:1. (iii) The text summarization dataset that we use is small, and we find that as the value of dropout increases, the training error decreases in small

Table 1 phrases sequence

Subject phrase	Relational phrase	Object phrase
a mom furious	take	part in riot have become a sensation
the woman	pull the women masked son	away from a crowed
the woman	follow	the son
the boy	be	dress in pants with a dark backpack on
wmar	report	the woman sees the woman son on television throw rock at police
police commissioner Anthony batt	thank the woman to media	in remark
you have one mother	smack	he on head
she	be	embarrassed
i	wish	i have more parent

datasets. This phenomenon is clearly not in line with the actual situation because when the value of dropout goes to the maximum, the neural network doesn't get trained essentially. Therefore, we set the value of dropout directly to 0.5. (iv) 512 or 1024 hidden units are recommended in the LSTM with two or three layers. More than three layers will lead to overfitting. During LSTM-CNN model training, we set the number of neurons in hidden layers to be 512 and the number of hidden layers to be 3. Under this circumstance, there are more than 10 million weight parameters in LSTM-CNN model that need to be trained. (v) We train our model for 50 epochs. Training for more than 50 epochs did not decrease the training error.

After the completion of our model training, we need to generate a summary by our model. If we calculate the next phrase by conditional probability as in previous ATS models, we will encounter the same rare words problem. Therefore, we divide the text generation into two different stages. We set a threshold δ, If the absolute value of conditional probability is more than threshold δ, our model will enter generate mode. In generate mode, we directly treat y^* as the next phrase in the summary, because high conditional probability represents the close relationship between these two phrases. On the other hand, if the absolute value of conditional probability is not more than threshold δ, our model will enter copy mode. In copy mode, we find the location of current phrase in the original text and copy the next phrase into the summary, because low conditional probability represents little connection between these two phrases and the next phrase in the original text may be more suitable as the next phrase in the summary from a human point of view. We have no prior knowledge about a good value to set y^*. If this value is set too high or too low, our model will put some unimportant phrases into summaries. After many experiments, we find that setting this threshold to 0.5 is a good choice.

Table 2 gives a comparison of the LSTM-CNN framework (system summary) and state-of-the-art summary system (reference summary). We can see that the summary generated by

Table 2 System summary and reference summary

System summary	Reference summary
wmar report the woman see the woman son on television throw rock at police. You have one mother smack he on head. She be embarrassed .	the mom sees the mom son on tv throwing rock at police, cnn affiliate report . police praise the mom action .

LSTM-CNN model is composed of natural sentences. At the same time, it is clear that this summary is very similar to the reference summary in semantics.

3.2 Experimental datasets

The existing famous abstractive text summarization corpora including Gigaword and DUC consist of only one sentence in each summary, and our model can achieve good results on these datasets. Compared with single-sentence summaries, multi-sentence summaries are more complex and since our model has a greater advantage in multi-sentence summaries, so we decide to do experiments about multi-sentence summaries. In our work, we use the human generated abstractive summary bullets from new stories in CNN and DailyMail websites as questions (with one of the entities hidden), and stories as the corresponding passages from which the system is expected to answer the fill-in-the-blank question. With a simple modification of the bullets, we restored all the summary bullets of each story in the original orders to obtain a multi-sentence summary, where each bullet is treated as a sentence. The CNN dataset is composed of more than 92,000 texts and corresponding summaries, while the DailyMail dataset is composed of more than 219,000 texts and corresponding summaries.

3.3 Evaluation method

In our work, we use the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) toolkit [13], which has been widely adopted by DUC for automatic summarization evaluation. ROUGE measures summary quality by counting overlapping units such as the n-gram, word sequences, and word pairs between the system summary and the reference summary. We choose automatic evaluation methods ROUGE-N in our experiment, which is an n-gram recall between a candidate summary and a set of reference summaries. ROUGE-N is computed as follows:

$$ROUGE\text{-}N = \frac{\sum\limits_{S \in \mathrm{Ref}} \sum\limits_{gram_n \in S} Count_{match}(gram_n)}{\sum\limits_{S \in \mathrm{Ref}} \sum\limits_{gram_n \in S} Count(gram_n)} \tag{11}$$

Where n stands for the length of the n-gram, Ref is the set of reference summaries. $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries, and $Count(gram_n)$ is the number of n-grams in the reference summaries. ROUGE toolkit reports separate scores for 1-g and 2-g. Among these different scores, the unigram-based ROUGE score (ROUGE-1) and the bigram-based ROUGE score (ROUGE-2) have been shown to agree with human judgment most.

3.4 Experimental results

Date pre-processing is a very simple step, and many natural language processing tools such as NLTK, CoreNLP can help us complete this step. Different natural language processing tools have their own advantages, and we find the correctness of word segmentation will be higher if we use CoreNLP rather than other natural language processing tools. In order to verify the effectiveness of data preprocessing, we calculate an average number of occurrences of each word in one text before and after data preprocessing, the result is described in Fig. 8. Before data preprocessing, the average number of occurrences of each word in one text is 1.512. While after data preprocessing it is 1.798. Therefore, we find that data preprocessing reduces word redundancy.

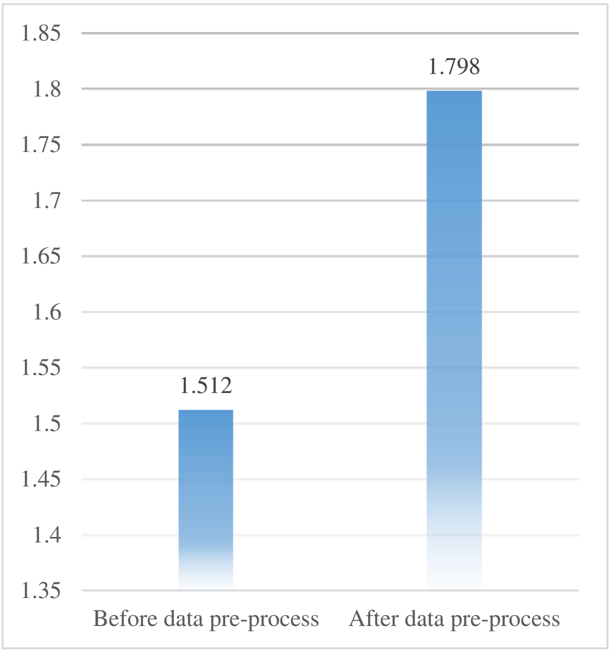


Fig. 8 Average number of occurrences of each word in one text

Nowadays, there are many methods to extract keywords from sentences. However, keyword is only a word and contain limited semantics. Compared with keywords, phrases have richer semantics and can construct a natural sentence, so it would be beneficial to extract phrases from sentences. Nevertheless, almost all of these methods have the disadvantage of incomplete extraction of phrases. Based on these phrase extraction methods, we use a more advanced phrase extraction method called MOSP. This method can help us extract more phrases than other existing phrase extraction methods.

Although MOSP can help us to extract more phrases, this method will also extract some wrong and repeated phrases. Therefore, it is necessary for us to delete some of these wrong or repeated phrases after acquiring phrases in the phrase acquisition step. In the phrase refinement step, we analyze the correct ratio of phrases before and after phrase refinement, and the result is represented in Fig. 9. Without phrase refinement, the correct ratio of phrases is 0.764, and this value can increase to 0.896 with phrase refinement. Therefore, it is obvious that phrase refinement can help us increase the correct ratio of phrases.

The sparsity of phrases or words is a major obstacle to neural network training, and each of new proposed neural network models is dedicated to solving this problem. In the phrase combination step, in order to prove that phrase combination can reduce the redundancy of phrases, we calculate the average number of different phrases in one text before and after phrase combination. We show the result in Fig. 10. We see that the average number of different phrases declines 13.3% (from 40.32 to 34.96). Hence, we draw the conclusion that phrase combination reduces the redundancy of phrases.

The results of ATS experiments on CNN and DailyMail datasets are shown in Table 3.

We compare the proposed model LSTM-CNN based ATSDL with various typical methods. First, we introduce the standard baseline called "LEAD". It simply selects the "leading" words from the source as the output. We also introduce the state-of-the-

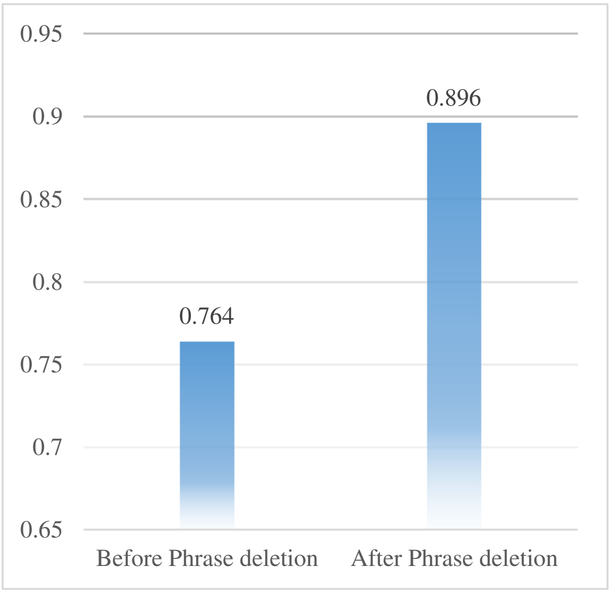


Fig. 9 Correct ratio of phrases

art statistical machine translation system Moses and the Seq2Seq model ABS. Moses is the dominant statistical approach to machine translation. It takes in the parallel data and uses co-occurrence of words and phrases to infer translation correspondences. For fair comparison, when implementing Moses, we also employ the alignment tool Fast Align and the language model tool SRILM. ABS is a Seq2Seq model with the attention mechanism. ABS has achieved promising performance on another one-sentence summarization benchmark.

The experiment shows that our ATSDL already outperforms state-of-the-art systems on CNN and DailyMail datasets. ATSDL achieves the ROUGE-1 of 34.9% (an increase of 4.4% over existing models) and the ROUGE-2 of 17.8% (an increase of 1.6% over existing models).

After getting the results of our experiments, we analyze the advantages of the ATSDL model. The advantages are as follows: (i) Our data preprocessing reduces word redundancy before phrase extraction process, this step improves the effect of phrase acquisition. (ii) We use a more advanced phrase extraction method called MOSP, and this method extracts more phrases. (iii) Our model takes phrases rather than words as input and output sequences of LSTM-CNN model. This process allows us to take both semantics and syntactic structures into account at the same time. If one phrase is selected to be a part of the system summary and one word in this phrase appears in reference summary, there is a possibility that the other words in this phrase also appear in this reference summary. Therefore, Rouge is beneficial to our model. (iv) We use WordNet to combine different phrases with similar semantics into one phrase, and this process reduces phrase redundancy and provides convenience for the LSTM-CNN model training. (v) We use phrase location information to help solve the problem rare words, and this greatly improves our final experimental results. (vi) Our model has an advantage in semantics according to experimental results. The summaries generated by our model are composed of natural sentences, but the summaries generated by other ATS models are just a keyword sequence. It shows that our model also has an advantage in syntactic structure.

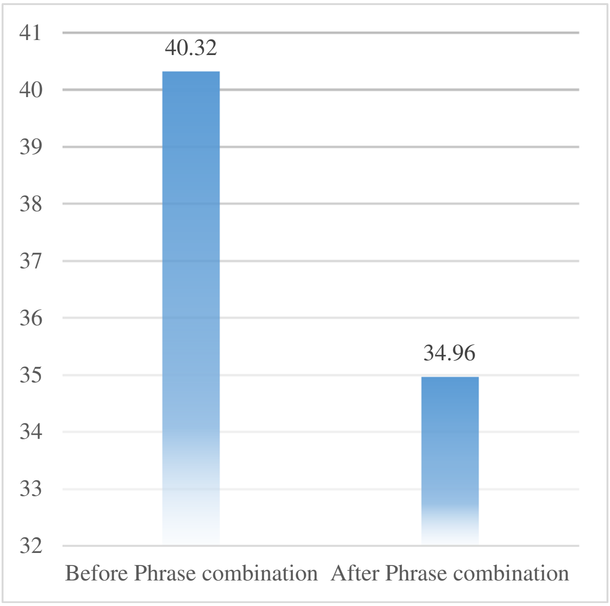


Fig. 10 Average number of different phrases in one text

3.5 Some further observations

In the future, there are many points in ATS models that deserve our attention. (i) CoreNLP is a very useful tool because CoreNLP makes it very easy to apply linguistic analysis tools to process text content. The accuracy of word segmentation using CoreNLP is very high, but text pre-processing is time consuming. Accordingly, we hope that we can use or develop a more convenient text pre-processing tool to help us complete this task. (ii) Determining semantic similarity between phrases is difficult. We use WordNet to complete this task. However, WordNet cannot give the similarity between phrases directly, so we may be able to provide potential improvements to solve this problem in future. (iv) Training deep learning models are a very time-consuming task and determining the threshold δ in the decoder of the model requires manual tuning. In the future, we can analyze the relationship between the threshold δ and some influencing factors like text length and then give a more precise threshold definition. (v) Using the Rouge evaluation method to determine the quality of a generated summary is very limited. The Rouge evaluation method only calculates the repeatability of N-gram and cannot analyze the quality of a generated summary from

Table 3 Experimental results

Model	ROUGE-1(%)	ROUGE-2(%)
LEAD	28.1	14.1
Moses	27.8	14.1
ABS	28.1	12.4
CoRe	30.5	16.2
ATSDL	**34.9**	**17.8**

The highest result in each column is marked in bold-face a more comprehensive perspective. Therefore, we may use more novel summarization evaluation methods to evaluate our LSTM-CNN based ATSDL model in future.

4 Related work

In the past few years, a vast majority of research in text summarization has been extractive. This research consists of identifying key sentences or passages in the source document and reproducing them as summaries [3, 5, 7, 8, 14, 18, 19, 21–23].

Traditional approaches to extractive summarization rely heavily on human-engineered features. Therefore, Jianpeng Cheng [5] proposes a data-driven approach based on neural networks and continuous sentence features. They develop a general framework for single-document summarization composed of a hierarchical document encoder and an attention-based extractor. This architecture allows them to develop different classes of summarization models that can extract sentences or words. In another paper, Marina Litvak [14] introduces and compares between two novel approaches, supervised and unsupervised, for identifying the keywords to be used in extractive summarization of text documents. Both of these approaches are extensions of the graph-based syntactic representation of text and web documents, which enhances the traditional vector-space model by taking into account some structural document features. Many techniques have been developed for summarizing English text(s). However, very few attempts have been made for Bengali text summarization. Kamal Sarkar [21] presents a method for Bengali text summarization that extracts important sentences from a Bengali document to produce a summary. Kam-Fai Wong [22] proposes a learning-based approach to combine various sentence features. They are categorized as surface, content, relevance and event features. Surface features are related to extrinsic aspects of a sentence. Content features measure a sentence based on content-conveying words. Event features represent sentences by events they contained. Finally, relevance features evaluate a sentence from its relatedness with other sentences. Mahmood Yousefi-Azar and Len Hamey [23] present methods of extractive query-oriented single-document summarization using a deep auto-encoder to compute a feature space from the term-frequency input.

On the other hand, many scholars attempt to paraphrase the original text in their own words. As such, text summaries are abstractive in nature and seldom consist of the reproduction of original sentences from the document. With the emergence of deep learning as a viable alternative for many NLP tasks, scholars have started considering this framework as an attractive, fully data-driven alternative to abstractive text summarization [7, 12, 19, 20, 24]. In a very recent work, Konstantin Lopyrev [19] describes an application of an encoder-decoder recurrent neural network with LSTM units and attention to generate headlines from the text of news articles. In another very recent work, Alexander M. Rush [24] proposes a fully data-driven approach to abstractive sentence summarization. His method utilizes a local attention-based model that generates each word of the summary conditioned on the input sentence. Sumit Chopra [7] introduces a conditional recurrent neural network (RNN) which generates a summary of an input sentence. The conditioning is provided by a novel convolutional attention-based encoder, which ensures that the decoder focuses on the appropriate input words at each step of generation. In another paper that is closely related to our work, Ramesh Nallapati [20] models abstractive-based text summarization using Attentional Encoder-Decoder Recurrent Neural Networks~~,~~ and show that, these models achieve state-of-the-art performance on two different corpora [10]. Our work starts with the same framework as Jiatao

Gu [12], where he incorporates copying into a neural network based on Seq2Seq learning and proposes a new model called COPYNET with encoder-decoder structure. COPYNET can nicely integrate the typical method of word generation in the decoder with the new copying mechanism that can choose subsequences in the input sequence and put them at proper places in the output sequence [4]. Zhang [24] proposes a simple yet powerful purely convolutional framework for learning sentence representations. SummaRuNNer [17], a Recurrent Neural Network (RNN) based sequence model for extractive summarization of documents [25–27], achieves a better performance than or comparable to state-of-the-art.

5 Conclusion

In this paper, we develop a novel LSTM-CNN based ATSDL model that overcomes several key problems in the field of TS. The present ETS models are concerned with syntactic structure, while present ATS models are concerned with semantics. Our model draws on the strengths of both of summarization models. The new ATSDL model firstly uses phrase extraction method called MOSP to extract key phrases from the original text, and then learns the collocation of phrases. After training, the model will generate a phrase sequence that meets the requirement of syntactic structure. In addition, we use phrase location information to solve the rare words problem that almost all ATS models would encounter. Finally, we conduct extensive experiments on two different datasets and the result shows that our model outperforms the state-of-the-art approaches in terms of both semantics and syntactic structure.

References

1. Angeli G, Tibshirani J, Wu J et al (2014) Combining distant and partial supervision for relation extraction[C]. EMNLP, pp 1556–1567
2. Bing L, Li P, Liao Y et al (2015) Abstractive multi-document summarization via phrase selection and merging[J]. arXiv preprint arXiv:1506.01597
3. Cao Z, Li W, Li S et al (2016) Attsum: joint learning of focusing and summarization with neural attention[J]. arXiv preprint arXiv:1604.00125
4. Chen M, Weinberger KQ, Sha F (2013) An alternative text representation to TF-IDF and Bag-of-Words[J]. arXiv preprint arXiv:1301.6770
5. Cheng J, Lapata M (2016) Neural summarization by extracting sentences and words[J]. arXiv preprint arXiv:1603.07252
6. Chopra S, Auli M, Rush AM (2016) Abstractive sentence summarization with attentive recurrent neural networks[C]. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp 93–98
7. Colmenares CA, Litvak M, Mantrach A et al (2015) HEADS: headline generation as sequence prediction using an abstract feature-rich space[C]. HLT-NAACL, pp 133–142
8. Erkan G, Radev DR (2004) Lexrank: graph-based lexical centrality as salience in text summarization[J]. J Artif Intell Res 22:457–479
9. Filippova K, Altun Y (2013) Overcoming the lack of parallel data in sentence compression[C]. EMNLP, pp 1481–1491
10. Gu J, Lu Z, Li H et al (2016) Incorporating copying mechanism in sequence-to-sequence learning[J]. arXiv preprint arXiv:1603.06393
11. Hu B, Chen Q, Zhu F (2015) Lcsts: a large scale chinese short text summarization dataset[J]. arXiv preprint arXiv:1506.05865
12. Li J, Luong MT, Jurafsky D (2015) A hierarchical neural autoencoder for paragraphs and documents[J]. arXiv preprint arXiv:1506.01057

13. Lin CY, Hovy E (2003) Automatic evaluation of summaries using n-gram co-occurrence statistics[C]. Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1. Association for Computational Linguistics, pp 71–78
14. Litvak M, Last M (2008) Graph-based keyword extraction for single-document summarization[C]. Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization. Association for Computational Linguistics, pp 17–24
15. Lopyrev K (2015) Generating news headlines with recurrent neural networks[J]. arXiv preprint arXiv: 1512.01712
16. Nallapati R, Zhou B, Gulcehre C et al (2016) Abstractive text summarization using sequence-to-sequence rnns and beyond[J]. arXiv preprint arXiv:1602.06023
17. Nallapati R, Zhai F, Zhou B (2017) Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In: Proc. Thirty-First AAAI Conference on Artificial Intelligence, pp 3075–3081
18. Ribeiro R, Marujo L, Martins de Matos D et al (2013) Self reinforcement for important passage retrieval[C]. Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval. ACM, pp 845–848
19. Riedhammer K, Favre B, Hakkani-Tür D (2010) Long story short–global unsupervised models for keyphrase based meeting summarization[J]. Speech Comm 52(10):801–815
20. Rush AM, Chopra S, Weston J (2015) A neural attention model for abstractive sentence summarization[J]. arXiv preprint arXiv:1509.00685
21. Sarkar K (2012) Bengali text summarization by sentence extraction[J]. arXiv preprint arXiv:1201.2240
22. Wong KF, Wu M, Li W (2008) Extractive summarization using supervised and semi-supervised learning[C]. Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. Association for Computational Linguistics, pp 985–992
23. Yousefi-Azar M, Text HL (2017) summarization using unsupervised deep learning[J]. Expert Syst Appl 68: 93–105
24. Zhang Y, Shen D, Wang G et al (2017) Deconvolutional paragraph representation learning[C]. Advances in Neural Information Processing Systems, pp 4172–4182
25. Zhou Q (2016) Research on heterogeneous data integration model of group enterprise based on cluster computing[J]. Clust Comput 19(3):1275–1282. https://doi.org/10.1007/s10586-016-0580-y
26. Zhou Q (2018) Multi-layer affective computing model based on emotional psychology[J]. Electron Commer Res 18(1):109–124. https://doi.org/10.1007/s10660-017-9265-8
27. Zhou Q, Liu R (2016) Strategy optimization of resource scheduling based on cluster rendering[J]. Clust Comput 19(4):2109–2117. https://doi.org/10.1007/s10586-016-0655-9

Shengli Song is currently an associate professor with Xidian University, China. He received his Ph.D. degree in Computer Science and Technology from Xidian University, Xi'an, China, in 2011. His current research interests include semantic computing, text analytics and intelligent system.

Haitao Huang is currently a master student majoring in Software Engineering in Xidian University. His research interest is mobile data analysis.

Tongxiao Ruan is currently a master student majoring in Computer Science and Technology in Xidian University. His research interest is machine learning.