Copyright ⓒ 2021 KSII

KI-HABS: Key Information Guided Hierarchical Abstractive Summarization

Mengli Zhang[1], Gang Zhou[1,*], Wanting Yu[1], Wenfen Liu[2]
[1] State Key Laboratory of Mathematical Engineering and Advanced Computing
Henan 450001, China
[e-mail: zml1122y@163.com]
[2] School of Computer Science and Information Security, Guangxi Key Laboratory of Cryptogpraphy and Information Security, Guilin University of Electronic Technology
Guangxi 541004, China
[e-mail: gzhougzhou@126.com]
*Corresponding author: Gang Zhou

Received November 2, 2020; revised January 11, 2021; revised May 9, 2021; revised July 6, 2021; accepted November 24, 2021; published December 31, 2021

Abstract

With the unprecedented growth of textual information on the Internet, an efficient automatic summarization system has become an urgent need. Recently, the neural network models based on the encoder-decoder with an attention mechanism have demonstrated powerful capabilities in the sentence summarization task. However, for paragraphs or longer document summarization, these models fail to mine the core information in the input text, which leads to information loss and repetitions. In this paper, we propose an abstractive document summarization method by applying guidance signals of key sentences to the encoder based on the hierarchical encoder-decoder architecture, denoted as KI-HABS. Specifically, we first train an extractor to extract key sentences in the input document by the hierarchical bidirectional GRU. Then, we encode the key sentences to the key information representation in the sentence level. Finally, we adopt key information representation guided selective encoding strategies to filter source information, which establishes a connection between the key sentences and the document. We use the CNN/Daily Mail and Gigaword datasets to evaluate our model. The experimental results demonstrate that our method generates more informative and concise summaries, achieving better performance than the competitive models.

Keywords: neural network, deep learning, NLP, abstractive summarization, selective encoding

This work was supported in part by the National Natural Science Foundation of China (61862011), Guangxi Science and Technology Foundation (2018GXNSFAA138116, 2019GXNSFGA245004).

1. Introduction

With the explosive growth of textual information on the Internet, an efficient automatic summarization system has become an urgent need. The ultimate goal of document summarization is to generate a concise and readable summary for the document while keeping its gist [1]. Overall, extractive summarization [2-5] and abstractive summarization [6-18] are the two main methods of document summarization. Extractive models directly copy a few significant sentences or keywords from the source text to form summaries, which is actually a simple compression of the source document. Abstractive models can automatically generate new words and linguistic phrases that are not present in the input document. Compared with extractive methods, abstractive summarization is considered much closer to the way human make a summary, but also more challenging [19].

Recently, thanks to the continuous development of the encoder-decoder model [20], abstractive summarization models [10,13,21] are able to generate summaries with high ROUGE scores. However, because the document contains multiple sentences, the relationship between these sentences is complex, and there is a long-distance dependency, which makes it difficult for the traditional sequence-to-sequence (seq2seq) model to capture important information in the document. Therefore, the summary generated by the seq2seq-baseline model will largely obscure the main information of the input document, and even contains duplicate sentences [22]. Researchers found that documents and their summaries essentially have a sentence-word hierarchical structure rather than just a flat sequence of words [23].

Hierarchical neural models have shown strong performance in document-based language models [23] and document classification [24] tasks. In 2015, Li et al. [25] proposed a basic hierarchical abstractive summarization model, and Ref. 3 further expanded their model, summaries generated by their method are significantly better than similar methods in terms of informativity and readability.

Although the document consists of multi-sentences, not all sentences contain gist information or useful information. Usually, a few sentences can express the core information of the document. In 2017, Nallapati et al. [26] summarized the content of the document just by directly extracting key sentences as a summary. Further, Chen et al. [27] rewrote the extracted sentences to construct summary sentences, which further improved the readability of the generated summary. In 2018, Cao et al. [28] utilized template sentences to guide the generation of summary and also achieved good results. In this paper, we further prove that the key sentences in the document can facilitate the generation of the summary. Therefore, we construct a key sentence extractor to extract key sentences and utilize these sentences to guide the encoding process.

The key sentences in the source document contain almost all significant information [27]. Therefore, we believe the key sentences of the document can provide a powerful signal to guide the document summarization process. Based on this, we propose a key sentences guided abstractive document summarization model under hierarchical encoder-decoder architecture. We apply key-sentences-guided selective encoding strategies to filter source information by investigating the interactions between the input document and the key sentences. For training, we first determine the ground-truth key sentences by calculating the ROUGE-L recall score, and use them to train an extractor. Then, the extractor is used as a plug-in, integrated into the hierarchical encoder-decoder model, to select salient sentences from the input. Finally, use the selected key sentences to control the selection gate network, which can select and filter the sentence representations to produce a tailored sentence representation by controlling the information in the sentence level.

Our contributions are as follows:

- Based on the hierarchical encoder-decoder architecture, we propose an abstractive document summarization method guided by the key sentences in the original input document. We propose a co-selective encoding to select information for both the document and the key sentences jointly. Then, using a gate vector to rebuild sentences representation and key sentences representation, respectively.
- A key sentences extractor. We train a key sentences extractor to extract the significant sentences with high informativity in the input document. The extractor consists of a hierarchical bidirectional GRU. The top layer is the classification layer that decides whether or not each sentence belongs to the key sentences.
- We conduct experiments on the *CNN/Daily Mail* and English *Gigaword* datasets, proving that our model significantly performs better than the competitive methods.

2. Related Work

The seq2seq is one of the mainstream frameworks in generating abstractive summaries. Rush et al. [10] proposed a CNN encoder and neural network language model under the seq2seq framework, which was the first application of the seq2seq model to the abstractive summarization task. After that, Zhou et al. [15], Li et al. [21] and Chopra et al. [29] further improved the RNN-based summarization model. In 2016, Gu et al. [30] added a copy mechanism. In 2017, Paulus et al. [31] proposed an intra-decoder neural attention mechanism, See et al. [13] introduced coverage vectors, they extended the seq2seq-baseline model. In recent years, the pointer mechanism has performed better and better. Li et al. [25] first constructed a hierarchical encoder-decoder model, which they used to train an automatic encoder for document summarization. Inspired by them, Cheng et al. [2], Li et al. [21] and Tan et al. [32] solved the long dependency problem based on encoder-decoder hierarchical architecture. In 2018, Li et al. [22] made further improvements on the basis of the hierarchical document structure. Chen et al. [27], Cao et al., [28] and Gehrmann et al. [33] operated at the sentence level of the input document and successfully captured the dependencies between sentences. In our work, we adopt hierarchical encoder-decoder architecture as our basic framework. The basic hierarchical encoder-decoder architecture is shown in **Fig. 1**.

Key sentences have been proved beneficial for extractive document summarization systems. Recently, research on extractive summarization based on neural networks has focused on extracting and sorting complete sentences [2,35,36]. In 2018, Liu et al. [17] proposed an extractive summarization model based on RNN that extracts full sentences. Other recent researches explore alternative approaches to sentences selection [36-38]. But they either extract the entire sentence directly as part of the summary, or only reconstruct the sentence at word level, without considering the impact of the key sentences in the entire input document as a whole. But in fact, the set of key sentences can guide sentence-level encoding as a whole, thus controlling the flow of information between the encoder and the decoder, which helps increase the informativity and readability of the generated summary.

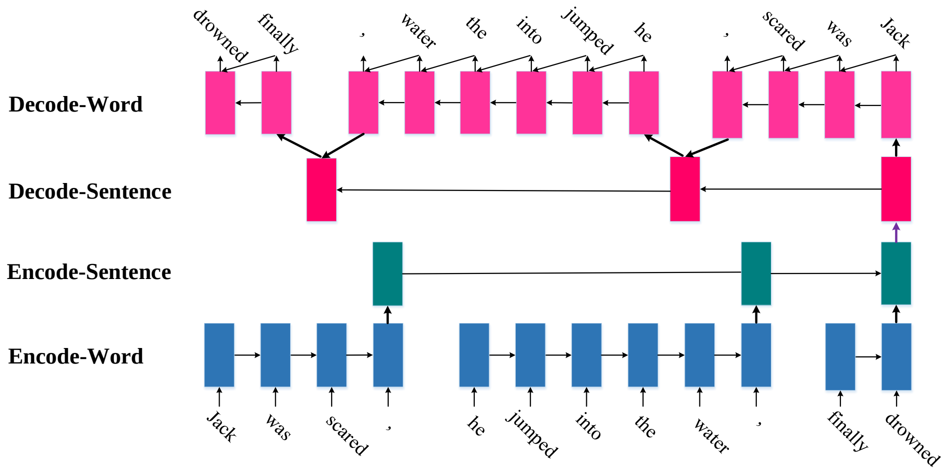


Fig. 1. The basic hierarchical encoder-decoder architecture.

3. Our Model

In this paper, we consider the task of summarizing an input document made up of multiple long sentences into a multi-sentence summary. The hierarchical encoder-decoder architecture can significantly reduce long dependency problem. Our hypothesis is that the key sentences can provide essential clues for the gist of the input document. Based on this, we introduce a key sentences driving mechanism at the sentence level, which can clearly indicate the valuable content of the input document.

Therefore, we construct a Key Information Guided Hierarchical Abstractive Summarization model (KI-HABS), and the framework of our model is shown in **Fig. 2**. The KI-HABS model is an encoder-decoder architecture. In the encoding stage (the left half of **Fig. 2**), we first utilize the trained key sentence extractor to extract key sentences from the input long text, these sentences contain significant information of the original text. And then, we encode these selected sentences together with the original input to get their vector representation. In order to deal with long texts, we adopt a more effective hierarchical architecture. In addition, in order to filter out the redundant information in the extracted key sentences, we designed a gate fusion mechanism to fuse the input context representation and the key sentence context representation. Finally, in the decoding stage (the right half of **Fig. 2**), we utilize the fused context vector to guide the KI-HABS model to generate summaries.

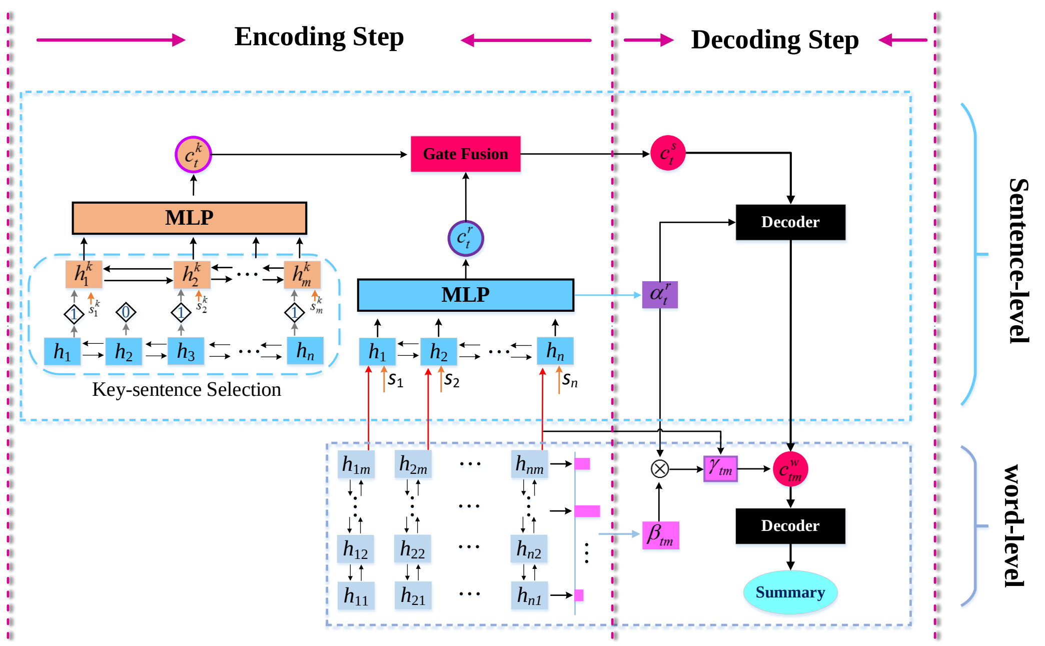


Fig. 2. The overall framework of our model. It is a hierarchical encoder-decoder architecture composed of word-level and sentence-level. In sentence level, we employ a key sentence extractor to select key sentences. We utilize a co-selective encoding to select information for both the document and the key sentences jointly. Then, using a gate vector to rebuild sentences representation and key sentences representation, respectively. Finally, we use the fusion context vector to generate summaries.

3.1 Hierarchical Encoder

The encoder can encode the input document into a vector representation in the hidden layer. Formally, given an input document ***D***, consisting of multiple sentences: $D=\{s_1,s_2,\mathrm{L}\ ,s_n\}$, n represents the number of sentences in the document. Each sentence can be represented by the words that make it up: $s_i=\{w_{i1},w_{i2},\mathrm{L}\ ,w_{im}\}$. The encoder at the word and sentence levels encode the words and the sentences into a vector representation, respectively.

In our framework, we use the bidirectional GRU encoder $\{\mathrm{GRU}^{\mathrm{e\_fwd}},\mathrm{GRU}^{\mathrm{e\_bwd}}\}$ at each level, which encodes input text forwardly and backwardly to generate two sequences of the hidden states. After receiving word w_{ij}, the word-level encoder generates its bidirectional hidden representation $(\vec{h}_{i1},\vec{h}_{i2},\mathrm{L}\ ,\vec{h}_{im})$ and $(\overleftarrow{h}_{i1},\overleftarrow{h}_{i2},\mathrm{L}\ ,\overleftarrow{h}_{im})$:

$$\vec{h}_{ij}=\mathrm{GRU}(e_{ij},\vec{h}_{ij-1}), \tag{1}$$

$$\overleftarrow{h}_{ij}=\mathrm{GRU}(e_{ij},\overleftarrow{h}_{ij+1}), \tag{2}$$

where $\vec{h}_{ij}$ and $\overleftarrow{h}_{ij}$ denote the forward and backward hidden state, respectively. e_{ij} denote the embedding of w_{ij}. The final word-level hidden representation $h_{ij}=[\vec{h}_{ij};\overleftarrow{h}_{ij}]$ is the concatenation of $\vec{h}_{ij}$ and $\overleftarrow{h}_{ij}$. In particular, the entire sentence representation is the last hidden vector h_{im}, the final vector representation for *i*-th sentence is $e_i=h_{im}$ that is the input of the encoder in sentence level.

After receiving vector representation e_i for i-th sentence, the encoder in sentence level updates hidden state $h_i = \text{BiGRU}(e_i, h_{i-1})$. Particularly, for the encoders, we believe they can benefit from sharing parameters to promote the capacity of capturing the gist of the input text. So, we use a shared encoder to generate hidden state sequences for both the original sentences and the key sentences. In the following description, h_i and h_i^k denote the hidden representations for the input sentences and the key sentences, respectively.

3.2 Key-sentenc0e Selection

To support our model, we proposed a novel method to select salient sentences. Our research proved that the key sentences in the input document provide significant clues for valuable content, and humans tend to remember them when summarizing. In this work, we first train a key sentences extractor to extract the significant sentences with high informativity in the document.

In 2017, Nallapati et al. [26] trained a sentence classifier in the extractive summarization task. Our extractor is different from theirs, we need to label sentences to indicate whether they are key sentences ("1" means is the key sentence, "0" is not a key sentence). Our extractor is shown in **Fig. 3**, which consists of a hierarchical bidirectional GRU. The top layer is the classification layer that decides whether or not each sentence belongs to the key sentences.

During training, cross entropy is used as the loss function. We minimize the negative log-likelihood of the labels observed during training, as follows:

$$L_{ext} = -\frac{1}{N}\sum_{1}^{N}(t_n \log p\{t_n = 1\} + (1 - t_n)\log p\{t_n = 0\}), \tag{3}$$

where $t_n \in \{0,1\}$ is the label for the sentence s_n, indicating whether it is the key sentence($t_n = 1$ as the key sentence). N is the number of sentences in document $\boldsymbol{D}$.

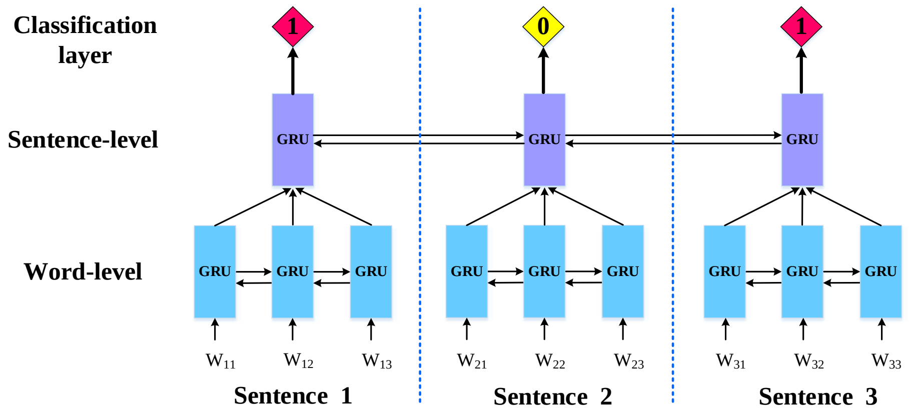


Fig. 3. The architecture of the extractor

3.3 Ground-truth Sentences

In fact, the standard *CNN/Daily Mail* dataset does not provide the key sentences of the input document. Therefore, in order to train our key sentences extractor and key-sentence-guided summarizer, we need to build ground-truth sentence sets for each document.

In our model, extractor are used to extract highly informative sentences from documents, which means the key sentences should contain most of the important information of the original document. ROUGE-L recall score can reflect the generalization of a sentence to the original text. Therefore, we calculate the ROUGE-L recall score between sentence s_i and the reference summary to obtain its informativity score, as follows:

$$Score(s_i) = \mathrm{ROUGE}(s_i, s^*), \tag{4}$$

where s_i and s^* represent *i*-th sentence and the reference summary, respectively. After that, we rank sentences according to their scores from highest to lowest informativity. Then, pop up the first-ranked sentences as a candidate key sentence. If the candidate sentence popped up can increase the informativity of the existing key-sentence list, we push it into the list. In particular, in order to ensure that the selected sentence can increase the informativity of the existing key-sentence list, we compare it with the previously selected sentences. If there are more than three words in it also appear in the previously selected sentences, we think it contains too much redundant information, then give it up. Repeat this until the candidate sentence does not meet the requirements. Finally, we obtain a list of all the ground-truth key sentences and train the extractor by minimizing Eq. (3).

Our key sentences selection task is to perform at sentence level, the output layer is a classifier over the hidden representation h_i for each sentence in the document. The classifier predicts one of the following two labels: ‘1’ for the key sentence, and ‘0’ for the not key sentence.

3.4 Key-sentences-guided Selective Encoding

In the hierarchical encoder architecture, beyond encoding for the sentence in the sentence level, we believe each key sentence contributes differently to the document summarization task. And thus, we propose a co-selective encoding to select information for both the document and the key sentences jointly. Then, using a gate vector to rebuild sentences representation h_i' and key sentences representation $h_i^{k'}$, respectively.

Specifically, a co-selective gate vector for each h_i and h_i^k computed as follows:

$$coGate_i = \mathrm{sigmoid}(W_q h_i + U_q a^k), \tag{5}$$

$$coGate_i^k = \mathrm{sigmoid}(W_q h_i^k + U_q a^r), \tag{6}$$

where $a^k = [h_1^k; h_2^k; \cdots; h_n^k]$ is the key sentences sequence representation, $a^r = [h_1; h_2; \cdots; h_n]$ is the document representation. Then, h_i' and $h_i^{k'}$ are computed as follows:

$$h_i' = h_i \odot coGate_i, \tag{7}$$

$$h_i^{k'} = h_i^k \odot coGate_i^k, \tag{8}$$

where $\odot$ is element-wise multiplication.

Using a hierarchical encoder-decoder architecture, we solve the long sentence dependency problem. Furthermore, by extracting key sentences and co-selective encoding, our model can better capture the relationship between the sentences and the hidden important clues provided by them.

3.5 Hierarchical Decoder

The decoder utilizes the vector representation of the input text passed by the encoder to generate summaries. In the hierarchical architecture, first the sentence-level decoder uses the vector representation of the document sentence passed by the encoder to generate the vector representation of the summary sentence. Then the word-level decoder decodes each sentence to get the final words. In the decoding phase, our model adopts a single-layer unidirectional LSTM as the decoder both in the sentence level and word level.

3.5.1 Sentence-level decoding

At the sentence level, we apply a dual-attention mechanism to generate the context vector based on attention over both the source sentence and the extracted key sentences. Furthermore, we also use an intra-temporal attention function [39]. Intra-temporal attention allows the attention mechanism to fully consider the decision of the previous decoding step when making a new decision, which can effectively avoid repetition.

More specifically, at each decoding step *t*, the attention score of the hidden state h_i' is calculated as follows:

$$e_{ti}^{r} = v_a^T \tanh(W_a h_i' + U_a s_t + b_{attn}) \,, \tag{9}$$

where b_{attn} , W_a , W_s and v are learnable parameters, s_t is current decoder state (*t*-th summary sentence). We normalize the attention weights with the following temporal attention function, penalizing input sentences that have obtained high attention scores in past decoding steps:

$$e_{ti}^{r'} = \begin{cases} \exp(e_{tm}^{r}) & \text{if } t{=}1 \\ \dfrac{\exp(e_{ti}^{r})}{\sum_{j=1}^{t-1}\exp(e_{ji}^{r})} & \text{otherwise} \end{cases} . \tag{10}$$

Then, we compute the normalized attention scores α_{ti}^{r} , and the sentence-level context vector c_t^r using α_{ti}^{r} :

$$\alpha_{ti}^{r} = \frac{e_{ti}^{r'}}{\sum_{l=1}^{n} e_{tl}^{r'}} \,, \tag{11}$$

$$c_t^r = \sum_{i=1}^{n} \alpha_{ti}^{r} h_i' \,. \tag{12}$$

Similar to input sentences, the key sentences attention α_{ti}^{k} and key sentences context vector c_t^k can be calculated using $h_i^{k'}$ and s_t. Next, we adopt a gated fusion mechanism to incorporate the influence of key sentences into the decoding process. We first compute a fusion gate vector using two context vectors and then combine context vectors by the gate, as follows:

$$g_t = \mathrm{sigmod}(W_g c_t^r + U_g c_t^k) \,, \tag{13}$$

$$c_t^s = g_t \,\mathrm{e}\, c_t^r + (1 - g_t) \,\mathrm{e}\, c_t^k \,. \tag{14}$$

And, in the sentence-level decoding step, c_t^s can be used.

3.5.2 Word-level Decoding

We use word-level attention on the word level. In each word generation step, our model can realize the summary sentence word by word by locating the relevant words in the source sentence. Firstly, we define e_{tm}^{ij} as attention score of the hidden input state h_{ij} at decoding time step *t*:

$$e_{tm}^{ij} = v_b^T \tanh(W_b h_{ij} + U_b s_{tm} + b'_{attn}) , \tag{15}$$

where s_{tm} denotes the hidden state (while generate *m*-th word in *t*-th summary sentence).

$$\beta_{tm}^{ij} = \frac{\exp(e_{tm}^{ij})}{\sum_l \exp(e_{tm}^{il})} . \tag{16}$$

Specially, β_{tm}^{ij} denotes the contribution of the word w_{ij} in the source sentence s_i in *t*-th decoder timestep.

Since the above word-level attention exists in every source sentence, we normalize it to achieve a word-level global attention distribution as:

$$\gamma_{tm}^{i} = \beta_{tm}^{i} \alpha_{ti}^{r} . \tag{17}$$

Then the word-level context vector can be computed, as follows:

$$\boldsymbol{c}_{tm}^{w} = \sum_i \sum_j \gamma_{tm}^{ij} h_{ij} . \tag{18}$$

Finally, using the global attention distribution $\boldsymbol{c}_{tm}^{w}$, we can calculate the probability distribution P_{gen} for the final words, as follows:

$$P_{gen}(w'_{tm}) = \text{softmax}(W'(W[s_{tm}, c_{tm}^{w}] + b) + b') , \tag{19}$$

where W' , W , b , b' are learnable parameters. Furthermore, we also introduce the copy mechanism [13] in the model, which can copy words in the original input documents and solve the problem of out-of-vocabulary (OOV) words.

3.6 Learning

We first use the constructed ground-truth key sentences pre-training extractor by minimizing L_{ext} in Eq. 3. Furthermore, we utilize the trained extractor as a plug-in of our model to extract key sentences for input documents. The final distribution is a weighted sum of the generation distribution. In training, the purpose of the model is to maximize the probability of generating a summary. So, we set the negative log-likelihood loss function as follows:

$$J(\theta) = -\frac{1}{|\boldsymbol{T}|} \sum_{(x,y)\in T} \log p(y \mid x; \theta) , \tag{20}$$

where $\boldsymbol{T}$ denotes a set of document summary pairs and θ is the model parameter. We use Adagrad [40] with learning rate 0.001 to optimize the model parameters θ .

4. Experiments and Evaluation

4.1 Dataset

We utilize the *CNN/Daily Mail* dataset [13] and annotated English Gigaword dataset [41] to evaluate the effect of the model. These datasets have been widely used in abstractive summarization tasks. For the *CNN/Daily Mail* dataset, we keep the named entities in the text and operate directly on the original dataset. We think this is necessary, because a good summarization model needs to handle named entities when facing real-time tasks. During training and testing, we limit the input documents to 800 tokens. For summaries, the length is limited to 100 tokens during training and 120 during testing. The final dataset is a non-anonymized version. The statistics of the two datasets are presented in **Table 1**.

Table 1. The statistics of *CNN/Daily Mail* and *Gigaword* datasets.

Dataset		Count	AvgSourceLen	AvgTargetLen
CNN/Daily Mail	Train	287,226	781	56
	Dev.	13,368	762	55
	Test	11,490	764	57
Gigaword	Train	3.8M	31.4	8.3
	Dev.	189K	31.7	8.3
	Test	1951	29.7	8.8

4.2 Experimental Setup

For all experiments, we set the size of the encoder and decoder hidden state to 256 in word level. For the encoder and decoder at the sentence level, we utilize 512-dimensional hidden states. The dimension of word embedding is 128. During the encoding and decoding process, we maintain a vocabulary of 50,000 words. In the test phase, when generating the summaries, we set the beam size to 4 and 8 in sentence level and word level, respectively.

4.3 Comparative Methods

We compare our model with some neural summarization approaches, including both abstractive models and extractive models. Among them, Lead-3 and SummaRuNNer are extractive models, and the rest is the abstractive model, as follows:

- **Lead-3** [26]. A widely used extractive baseline model, selecting the first three sentences from the original text as the summary.
- **SummaRuNNer** [26]. An extractive summarization model based on RNN, converting the extractive summarization problem into a sequence classification problem: make a binary classification of each input sentence.
- **Point-Gen** [13]. An extension of the seq2seq-baseline model that copies words from the original text through the pointer network, and penalizes the words in the input that were paid too much attention during the previous decoding, which solves the problem of sentence duplication.
- **GPG (**Generalized Pointer Generator**)** [42]. A pointer generation model with stronger generalization ability. It enables the pointer network to “edit” its copied words rather than simply hard copying.
- **SAGCopy (Self-Attention Guided Copy Mechanism)** [43]: A Transformer-based abstractive summarization model that utilizes the centrality of each source word to guide the copy process explicitly.
- **Hierarchical-baseline** [25]. A basic hierarchical encoder-decoder model, which encodes the input documents at the sentence level and the word level and introduces an attention mechanism at the sentence level when decoding.
- **Hierarchical stru-Reg** [1]. A hierarchical encoder-decoder model. It captures the structural features of the input documents by modelling the attention mechanism at the sentence level, thereby improving the informativity and readability of the summaries.

4.4 Result

We use the standard ROUGE metric to evaluate our model and report the F1 scores of ROUGE-1, ROUGE-2 and ROUGE-l with the Porter stemmer option. In order to show the advantages of our method more visually, we further count the duplicates in the generated summaries. We also randomly select 50 examples from *CNN/Daily Mail* dataset and use them for human evaluation to ensure that our increase in ROUGE scores is also followed by an increase in human readability and quality. Finally, we conduct several ablation experiments to verify the effectiveness of key-sentences-guided selective encoding.

4.4.1 ROUGE Metric

Table 2 shows the rouge evaluation results. Among them, KI-HABS-Transformer indicates that the transformer is used as the encoder, and KI-HABS-GRU indicates that the GRU is used as the encoder. It can be seen from Table 2 that the KI-HABS-GRU model performs best compared with similar models, but it is not as good as the Transformer-based model SAGCopy. After replacing the encoder with Transformer, the performance of the model (KI-HABS-Transformer) has been improved, and the result is no weaker than SAGCopy, which proves the effectiveness and scalability of the key information-guided framework we proposed. Specifically, compared with the hierarchical baseline model, our method KI-HABS significantly improves the ROUGE score. Moreover, the KI-HABS model is superior to the previous state-of-the-art hierarchical method Hierarchical Stru-Reg. The summary generated by our model contains almost all the significant information in the original text, which shows that key sentences in the document contain more core information. In particular, Lead-3 performs well on the *CNN/Daily Mail* dataset, which to a certain extent shows that the sentence-level contains more document information in the multi-sentences summarization.

Table 2. ROUGE for two datasets. Results with * mark are taken from the corresponding papers

Method	CNN/Daily Mail			Gigaword		
	R-1	R-2	R-L	R-1	R-2	R-L
	Extractive Results					
Lead-3	40.34	17.70	36.57	30.12	13.36	27.85
SummaRuNNer*	39.6	16.2	35.3	-	-	-
	Abstractive Results					
Point-Gen*	39.53	17.28	36.38	36.23	17.42	36.15
Seq2seq-baseline*	36.64	15.66	33.42	34.04	15.95	31.68
Hierarchical-baseline	34.95	14.79	32.68	31.11	13.97	28.33
Hierarchical Stru-Reg*	40.30	18.02	37.36	-	-	-
GPG*	40.95	18.01	37.46	37.23	19.02	34.66
SAGCopy*	42.53	19.92	39.44	38.86	19.91	36.06
KI-HABS-GRU	**41.07**	**18.51**	**38.14**	**37.64**	**19.34**	**35.22**
KI-HABS-Transformer	**42.14**	**19.72**	**40.58**	**38.62**	**20.16**	**35.98**

4.4.2 Duplicates Comparison

Specially, in order to show the advantages of our method more visually, we further count the duplicates in the generated summaries. The results are shown in Fig. 4. From the figure, we can know that the summaries generated by our model contain less repetitions compared with the Seq2seq-baseline, Point-Gen, Hierarchical-baseline and GPG. Compared with Hierarchical Stru-Reg, our model also has a certain improvement. Therefore, our model not only digs into the salient information in the documents and reduces redundancy.

The summaries generated by the Point-Gen, Seq2seq-baseline, GPG and Hierarchical-baseline model contain a lot of repeated sentences and phrases. The Point-Gen and GPG models even make fake facts. Compared with other models, our model generates more complete summaries with more salient information of input text, which shows that key sentences can provide more core information about the original text and guide the summary generation process.

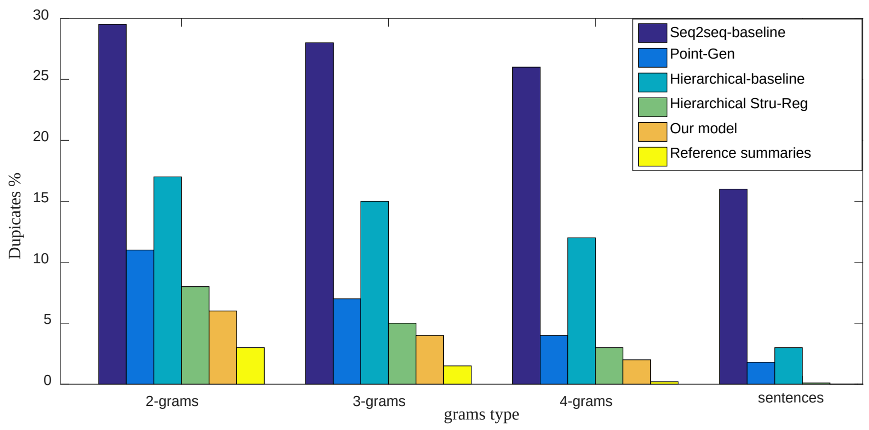


Fig. 4. Duplicates of different model

4.4.3 Human Evaluation

Furthermore, we randomly select 50 examples from *CNN/Daily Mail* dataset and use them for human evaluation to ensure that our increase in ROUGE scores is also followed by an increase in human readability and quality. The experiment is conducted under blackbox conditions, which means the human evaluator does not know which summaries come from which model or which one is the reference. All participants score from two aspects, readability (it measures whether the summary conforms to human language habits) and relevance (it measures whether the summary contains all the main information of the original text), 1 is the lowest score, and 5 is the highest.

The results are shown in **Table 3**. Our method with key-sentences guidance achieves higher scores than other abstractive methods except for the reference.

Table 3. Human evaluation results

Method	Readability	Relevance
Point-Gen	2.96	3.12
Seq2seq-baseline	2.33	2.56
Hierarchical-baseline	3.04	3.33
Hierarchical Stru-Reg	3.27	3.54
GPG	3.11	3.43
SAGCopy	3.75	4.11
KI-HABS-GRU	**3.57**	**3.98**
KI-HABS-Transformer	**3.86**	**4.25**
Reference	4.58	4.79

4.4.4 Ablation Experiments

Finally, in order to verify the impact of key sentences on the model, we conduct several ablation experiments. After extracting the key sentences, we randomly select 1/4, 1/2, 2/3 and all the key sentences to guide the selective encoding. Specifically, based on the Hierarchical-baseline, we add different numbers of key sentences in turn: +1/*4KS* indicates adding a quarter of the key sentences, +1/2*KS* indicates adding one-half of the key sentences, +2/3*KS* indicates adding two thirds of the key sentences, +*full-KS* indicates adding all the key sentences.

Table 4. ROUGE for adding different number of key sentences

Method	Rouge-1	Rouge-2	Rouge--L
Hierarchical-baseline	34.95	14.79	32.68
+1/4-*KS*	36.67	15.98	34.16
+1/2-*KS*	38.45	17.05	36.02
+2/3-*KS*	39.21	17.48	37.51
+full-KS (Our model)	**41.07**	**18.51**	**38.34**

The results are shown in **Table 4**. Our method achieves higher scores than other compared models, which verifies the effectiveness of key-sentences-guided selective encoding. Moreover, as the proportion of key sentences increases, the generated summaries achieve higher ROUGE scores. **Fig. 5** shows this effect more intuitively. Therefore, we can conclude that applying guidance signals of key sentences to the encoder based on the hierarchical encoder-decoder architecture have significant contributions to the summarization performance.

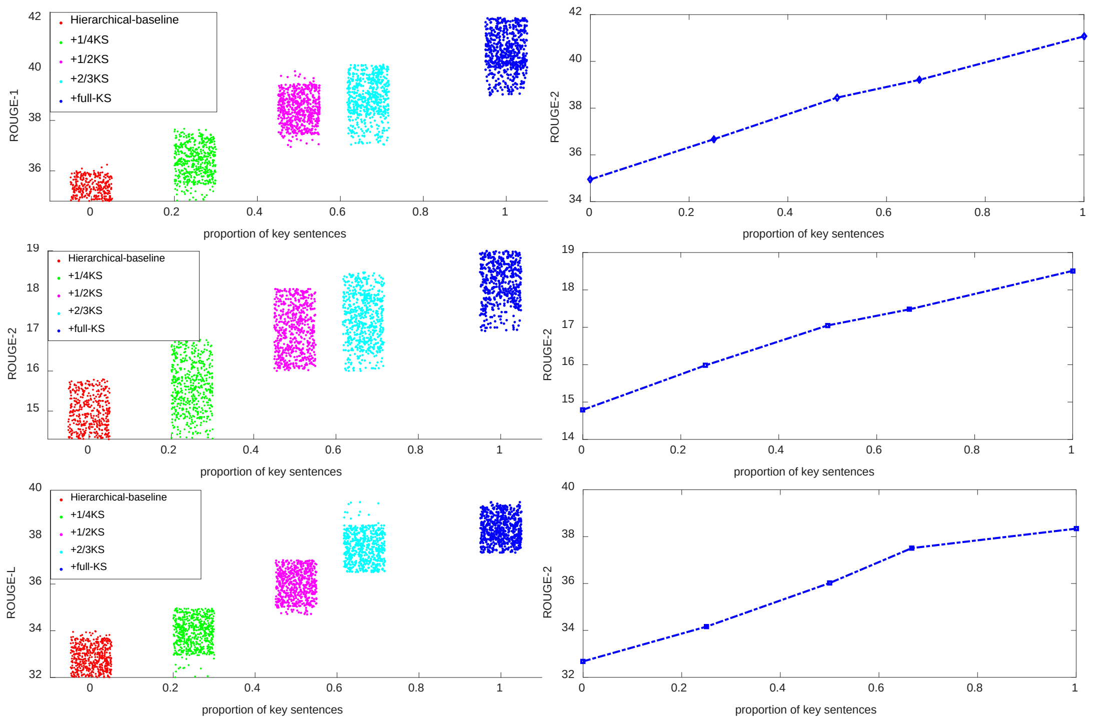


Fig. 5. Comparison of the impact of different proportions of key sentences on model performance. The subgraphs on the left show the ROUGE scores of the random samples from *CNN/Daily Mail* datasets. The subgraphs on the right describe the trend of the influence of different proportions of key sentences on the average rouge scores.

5. Conclusion and Future Work

In our paper, we focus on the abstractive document summarization model. We propose an abstractive document summarization method by applying guidance signals of key sentences to the encoder in our hierarchical encoder-decoder architecture. We use extractive methods to train a key sentences extractor, which can extract the salient sentences in the input document. Then, we apply key-sentences-guided selective encoding strategies to filter source information between the input document and the key sentences. Our model digs into the salient information in the documents and uses them to guide the generation of summaries. To solve this problem, we conduct a lot of experiments. Comparison results of different models show that our model works best.

In the next step of our work, we will try to explore more efficient sentence selection methods and extend our framework to pre-trained models. Moreover, more external information can be added at the sentence level to guide the summary generation, such as fact descriptions of the input document.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (61862011), Guangxi Science and Technology Foundation (2018GXNSFAA138116, 2019GXNSFGA245004).

References

[1] Li W, Xiao X, Lyu Y, et al, "Improving neural abstractive document summarization with structural regularization," in *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4078-4087, 2018. Article (CrossRef Link).

[2] Cheng J, Lapata M, "Neural summarization by extracting sentences and words," in *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 484-494, 2016. Article (CrossRef Link).

[3] Jadhav A, Rajan V, "Extractive summarization with swap-net: sentences and words from alternating pointer networks," in *Proc. of the 56th annual meeting of the association for computational linguistics*, pp. 142-151, 2018. Article (CrossRef Link).

[4] Dong Y, Shen Y, Crawford E, et al., "BanditSum: extractive summarization as a contextual bandit," in *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3739-3748, 2018. Article (CrossRef Link).

[5] Gao Y, Wang Y, Liu L, et al., "Neural abstractive summarization fusing by global generative topics," *Neural Computing and Applications*, 32, 5049–5058, 2020. Article (CrossRef Link).

[6] Zheng J, Zhao Z and Song Z et al., "Abstractive meeting summarization by hierarchical adaptive segmental network learning with multiple revising steps," *Neurocomputing*, 378, 79-188.

[7] Li S, Xu, "A two-step abstractive summarization model with asynchronous and enriched-information decoding," *Neural Computing and Applications*, 33, 1159–1170, 2021. Article (CrossRef Link).

[8] Liang Z, Du J, Li C, "Abstractive Social Media Text Summarization using Selective Reinforced Seq2Seq Attention Model," *Neurocomputing*, 410, 432-440, Oct. 2020. Article (CrossRef Link).

[9] Deng Z, Ma F and Lan R et al., "A Two-stage Chinese text summarization algorithm using keyword information and adversarial learning," *Neurocomputing*, vol. 425, pp. 117-126, 2021. Article (CrossRef Link).

[10] Rush A M, Chopra S, Weston J, "A neural attention model for abstractive sentence summarization," in *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 379–389, 2015. Article (CrossRef Link).

[11] Takase S, Suzuki J, Okazaki N, et al., "Neural headline generation on abstract meaning representation," in *Proc. of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1054-1059, 2016. Article (CrossRef Link).

[12] Chen Q, Zhu X D, Ling Z H, et al., "Distraction-based neural networks for modeling documents," in *Proc. of 2016 International Joint Conference on Artificial Intelligence*, pp. 2754–2760, 2016. Article (CrossRef Link).

[13] See A, Liu P J, Manning C D, "Get to the point: summarization with pointer-generator networks," in *Proc. of the 2017 Annual Meeting of the Association for Computational Linguistics*, pp. 1073–1083, 2017. Article (CrossRef Link).
[14] Tan J, Wan X, Xiao J, "Abstractive document summarization with a graph-based attentional neural model," in *Proc. of the 2017 Annual Meeting of the Association for Computational Linguistics*, pp. 1171–1181, 2017. Article (CrossRef Link).
[15] Zhou Q, Yang N, Wei F, et al., "Selective encoding for abstractive sentence summarization," in *Proc. of the 2017 Annual Meeting of the Association for Computational Linguistics*, pp. 1095–1104, 2017. Article (CrossRef Link).
[16] Narayan S, Cohen S B, Lapata M, "Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization," in *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797-1807, 2018. Article (CrossRef Link).
[17] Lebanoff L, Song K, Liu F, "Adapting the neural encoder-decoder framework from single to multi-document summarization," in *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4131-4141, 2018. Article (CrossRef Link).
[18] Zhu J, Wang Q, Wang Y, et al., "NCLS: neural cross-lingual summarization," in *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3055, 2019. Article (CrossRef Link).
[19] Li H, Zhu J, Zhang J, et al., "Keywords-guided abstractive sentence summarization," in *Proc. of the AAAI Conference on Artificial Intelligence*, Vol. 34, No. 5, pp. 8196-8203, 2020. Article (CrossRef Link).
[20] Sutskever I, Vinyals O, Le Q V, "Sequence to sequence learning with neural networks," in *Proc. of the 27th International Conference on Neural Information Processing Systems*, pp. 3104–3112, 2014. Article (CrossRef Link).
[21] Nallapati R, Zhou B, Santos C N D, et al., "Abstractive text summarization using sequence-to-sequence rnns and beyond," in *Proc. of the 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 280–290, 2014. Article (CrossRef Link).
[22] Li W, Xiao X, Lyu Y, et al., "Improving neural abstractive document summarization with structural regularization," in *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4078-4087, 2018. Article (CrossRef Link).
[23] Lin R, Liu S, Yang M, et al., "Hierarchical recurrent neural network for document modeling," in *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 899-907, 2015. Article (CrossRef Link).
[24] Yang Z, Yang D, Dyer C, et al., "Hierarchical attention networks for document classification," in *Proc. of the 2016 conference of the North American chapter of the association for computational linguistics*, pp. 1480-1489, 2016. Article (CrossRef Link).
[25] Li J, Luong M T, Jurafsky D, "A hierarchical neural autoencoder for paragraphs and documents," in *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pp. 1106-1115, 2015. Article (CrossRef Link).
[26] Nallapati R, Zhai F, Zhou B, "SummaRuNNer: a recurrent neural network based sequence model for extractive summarization of documents," in *Proc. of the 31th AAAI Conference on Artificial Intelligence*, pp. 3075-3081, 2016. Article (CrossRef Link).
[27] Chen Y C, Bansal M, "Fast abstractive summarization with reinforce selected sentence rewriting," in *Proc. of the 2018 Annual Meeting of the Association for Computational Linguistics*, pp. 675-686, 2018. Article (CrossRef Link).
[28] Cao Z, Li W, Li S, et al., "Retrieve, rerank and rewrite: soft template based neural summarization," in *Proc. of the 2018 Annual Meeting of the Association for Computational Linguistics*, pp. 152-161, 2018. Article (CrossRef Link).

[29] Chopra S, Auli M, Rush A M, "Abstractive sentence summarization with attentive recurrent neural networks," in *Proc. of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 93-98, 2016. Article (CrossRef Link).

[30] Gu J, Lu Z, Li H, et al., "Incorporating copying mechanism in sequence-to-sequence learning," in *Proc. of the 2016 Annual Meeting of the Association for Computational Linguistics*, pp. 1631-1640, 2016. Article (CrossRef Link).

[31] Paulus R, Xiong C, Socher R., "A deep reinforced model for abstractive summarization," in *Proc. of the 2018 International Conference on Learning Representations*, 2017. Article (CrossRef Link).

[32] Tan J, Wan X, Xiao J, "From neural sentence summarization to headline generation: a coarse-to-fine approach," in *Proc. of the 2017 International Joint Conference on Artificial Intelligence*, pp. 4109-4115, 2017. Article (CrossRef Link).

[33] Gehrmann S, Deng Y, Rush A M, "Bottom-up abstractive summarization," in *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4098-4109, 2018. Article (CrossRef Link).

[34] Dlikman A, Last M, "Using machine learning methods and linguistic features in single-document extractive summarization," in *Proc. of DMNLP@ PKDD/ECML*, pp. 1–8, 2016. Article (CrossRef Link).

[35] R. Nallapati, B. Zhou, M. Ma, "Classify or select: neural architectures for extractive document summarization," *arXiv preprint, arXiv: 1611.04244*, 2016. Article (CrossRef Link).

[36] Cohan A, Dernoncourt F, Kim D S, et al., "A discourse-aware attention model for abstractive summarization of long documents," in *Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 615-621, 2018. Article (CrossRef Link).

[37] Li C, Xu W, Li S, et al., "Guiding generation for abstractive text summarization based on key information guide network," in *Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 55-60, 2018. Article (CrossRef Link).

[38] Pasunuru R, Bansal M, "Multi-reward reinforced summarization with saliency and entailment," in *Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 646-653, 2018. Article (CrossRef Link).

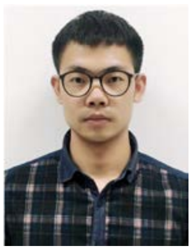
[39] Sankaran B, Mi H, Onaizan Y A, et al., "Temporal attention model for neural machine translation," *Preprint, arXiv:1608.02927*, 2016. Article (CrossRef Link),

[40] Duchi J, Hazan E, Singer Y, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of machine learning research*, vol. 12, pp. 2121–2159, 2011. Article (CrossRef Link).

[41] Napoles C, Gormley M, Durme B V, "Annotated Gigaword," in *Proc. of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pp. 95-100, 2012. Article (CrossRef Link).

[42] Shen X, Zhao Y, Su H, et al., "Improving latent alignment in text summarization by generalizing the pointer generator," in *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3753-3764, 2019. Article (CrossRef Link).

[43] Xu S, Li H, Yuan P, et al., "Self-Attention Guided Copy Mechanism for Abstractive Summarization," in *Proc. of the 58th Annual Meeting of the Association for Computational Linguistics*, pp.1355-1362, 2020. Article (CrossRef Link).



MENGLI ZHANG is currently pursuing PhD with the School of Computer Science and Technology, Information Engineering University, Zhengzhou, China. His research interests include NLP, Automatic text summarization.

GANG ZHOU received his B. E and MA. Eng degrees from Information Engineering University in 1996 and 1999, Ph.D. degree from Beijing University of Aeronautics and Astronautics in 2007. He was with the State key of Laboratory of Mathematical Engineering And Advanced Computing, Information Engineering University, as a research fellow.
Since 2017, he has been the professor of Information Engineering University. His research interests are big data and data mining.

WANTING YU is currently pursuing PhD with the School of Information system engineering, Information Engineering University, Zhengzhou, China. His research interests include intelligent information processing, deep learning, and information security.

WENFEN LIU received the PhD degree in mathematics from Institute of Information Engineering, Zhengzhou, China, in 1998. She is a full professor in Guangxi Key Laboratory of Cryptography and Information Security, School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin, China, and serves as the head of probability statistics. Her research interests include probability statistics, network communications, and information security.