

Received January 14, 2020, accepted February 21, 2020, date of publication March 2, 2020, date of current version March 13, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2977508

Keyphrase Generation With CopyNet and Semantic Web

XUN ZHU^{1,2}, CHEN LYU³, AND DONGHONG JI¹

¹Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China

²School of Mathematics and Computer Science, Jiangnan University, Wuhan 430056, China

³Collaborative Innovation Center for Language Research and Services, Guangdong University of Foreign Studies, Guangzhou 510420, China

Corresponding authors: Chen Lyu (lvchen1989@whu.edu.cn) and Donghong Ji (dhji@whu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772378, in part by the Social Science Foundation of Ministry of Education of China under Grant 18JZD015 and Grant 19YJCZH114, in part by the Natural Science Foundation of Hubei Province under Grant 2012FFA088, in part by the Special Innovation Project of Guangdong Education Department under Grant 2018KTSCX059, and in part by the National Key Research and Development Program of China under Grant 2017YFC1200500.


ABSTRACT Keyphrases provide core information for users to understand the document. Most previous works utilize machine learning based methods for keyphrases extraction and achieve promising performance. However, these methods focus on identify keyphrases from the input text, and can not extract keyphrases that do not appear in the text. In this paper, we present an encoder-decoder framework, which incorporating copying mechanism, to generate keyphrases for the given text. This framework (CopyNet) integrates the generation part and copying part. The generation part generates the keyphrase from the predefined vocabulary, and the copy part gets the keyphrases from the source text. Furthermore, we improve the CopyNet by using different probability of the two parts. To incorporate more related information for keyphrase generation, the automatically built keyphrase semantic web is merged into the dataset to participate in the training process of the neural network. Semantic similarity based and word co-occurrence based methods are used for keyphrase semantic web construction. We build a large-scale biomedical keyphrase dataset to evaluate the system performance. Experiments show that our improved CopyNet can achieve better performance with different portions of the generation and copying part, and the incorporation of the semantic web also effectively improves the keyphrase generation.

INDEX TERMS Keyphrase generation, encoder-decoder model, copying mechanism, semantic web.

I. INTRODUCTION

Keyphrases are the basic units for expressing the semantic information of the document. They are usually regarded as phrases that represent the salient concepts of a document [1], and provide users with core information. High quality keyphrases are important for users to better understand the key ideas of the documents, and automatic keyphrase extraction has received much academic interest over the past years [2]–[6]. Furthermore, it is also an important prerequisite task for downstream applications, such as summarization, information retrieval and question-answering.

Many studies have been conducted on automatic keyphrase extraction [7]. On one hand, unsupervised methods, such as TF-IDF [8] based ranking method, achieve comparable performance in this task [9], [10]. On the other hand,

The associate editor coordinating the review of this manuscript and approving it for publication was Bijju Issac .

various supervised learning methods, such as support vector machines (SVMs), are used to identify keyphrases [6], [11].

However, these systems have two main drawbacks. One is that these systems can only identify keyphrases from the input text. As we know, some keyphrases do not occur in the source document. For example, in scientific publications, authors often use the phrases, which can express the semantic meaning of the article, as the keyphrases. They do not care whether these keyphrases are directly used in their article. These keyphrases that are not occur in the source document are referred as absent keyphrases. Discovering these absent keywords is a promising way to improve the performance.

The other is that they need to pay much attention on feature engineering efforts. Various kinds of features have been used for this task, such as frequency features (e.g. TF-IDF) and syntactic features. Recently, deep learning has been widely used in natural language processing (NLP) [12]–[15], and it brings hope to reduce manual feature

engineering in various tasks. Compare with hand-designed features and traditional discrete feature representation, it provides a different way to automatically learn dense feature representation for text, such as words, phrases and sentences.

To overcome the above problems, we present an encoder-decoder framework to generate keyphrases for the given text. The encoder part uses the long short-memory (LSTM) to capture the semantics of the source text, and the decoder part generates keyphrases based on the content representation. The attention mechanism is also introduced to the framework to better represent the context vector in the keyphrases generation process. The encoder-decoder model generates keyphrases from the predefined vocabulary, and we can get absent keyphrases for the given text.

But traditional encoder-decoder framework is confronted with the out-of-vocabulary (OOV) problem, because it only generates keyphrases from the vocabulary. Since some of these OOV keyphrases may occur in the source text, we introduce copying mechanism into this framework to select some important segments as the keyphrases. The copying mechanism is similar to the recurrence operation in human language processing, which directly selects some important words from the source text as the keyphrases or answers for some questions. Thus, the encoder-decoder framework with attention and copying mechanism (CopyNet) generates keyphrases based on the semantic information and the important text information of the source document.

Despite the semantic information and text information of the document, some related information is also important for keyphrases generation. In this paper, we automatically build the keyphrase semantic web, which contains related keyphrases for one keyphrase. The semantic web is built from the large-scale biomedical articles, and each article contain its abstract and corresponding keyphrases. The results of the semantic web are merged into the biomedical dataset to participate in the training process of the neural network, and more related information is considered to generate keyphrases.

Evaluation results on the biomedical keyphrases corpus demonstrate the effectiveness of our proposed method. The contributions of this paper is as following:

- We present the encoder-decoder framework with copying mechanism to generate keyphrases, and this framework has the ability to predict absent keyphrases and OOV keyphrases. Different probabilities for generation part and copying part is used to improve the performance of the CopyNet.
- We build a large-scale biomedical dataset to evaluate the performance. One example in this dataset contains the abstract and its corresponding keyphrases in one biomedical article.
- Keyphrase semantic web is automatically constructed. More related information is introduced into the model to generate better keyphrases.

II. RELATED WORK

A. KEYPHRASE EXTRACTION

Previous works on keyphrase extraction usually focus on documents in different domains, including news [16], scientific [4], meeting transcripts [9] and web text [17], [18]. The methods used for keyphrase extraction fall into two lines: supervised learning and unsupervised learning.

In the supervised learning research line, keyphrase extraction is formalized as a classification problem. These works first extract candidate phrases using some heuristic rules, and then train a classification model to predict whether a candidate phrase is a keyphrase or not. Different features have been used for this task [2], [19]–[21], including frequency features (e.g. TF-IDF), structural features, syntactic features and external resource-based features.

In the unsupervised learning research line, it is usually formalized as a ranking problem. The keyphrases are usually ranked based on the TF-IDF [9], [22], [23] and term informativeness [24]. The TF-IDF based ranking has been shown to perform well in this task. Graph-based ranking is also widely used in unsupervised methods [16], [25]. It aims to build a graph and rank its nodes, which represent candidate keyphrases, according to their importance.

Scientific information extraction has attracted much attention in recent years, and SemEval 2017 Task10 provides a benchmark to evaluate the keyphrase extraction performance [6]. Similar to named entity recognition (NER), the keyphrase extraction can be formalized as a sequence labelling problem, and top three systems all used recurrent neural network (RNN)-based methods [26], [27]. All these keyphrase extraction systems can only extract keyphrases from the source text and they are not able to get absent keyphrases for the given text.

B. ENCODER-DECODER MODEL

The encoder-decoder model is first proposed for machine translation [28], [29]. It transforms the input sequence to the output sequence (the source language to the target language). The RNN-based encoder encodes the input sequence into a feature vector and the RNN-based decoder decodes a given feature vector into the output sequence. Both the input and output sequence are variable-length, and can be effectively represented in this model. The encoder-decoder model can be applied to NLP task and achieve promising performance [30]–[32].

LSTM [33], a variant of RNN, can also be used in this model to better capture the context information of the sequence [29]. The attention mechanism [34] is introduced into the model and it allows different importance of the input in the encoder. To leverage some important information from the source text, some works explore methods to copy appropriate parts of the input sequence into the output sequence [35]–[37].

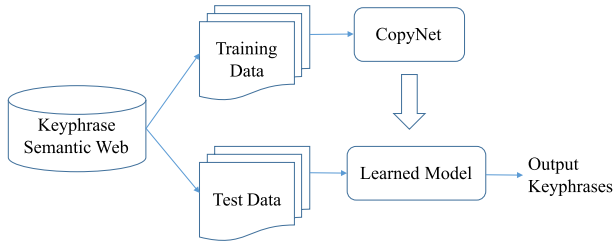


FIGURE 1. The overview of our keyphrase generation system.

III. METHODS

In this section, we describe our proposed model for automatic keyphrase generation in detail. Fig. 1 illustrates the overview of our keyphrase generation system.

We present the CopyNet to generates keyphrases based on the semantic information and the important text information of the source document. Furthermore, keyphrase semantic web is automatically constructed, and it can be merged into the dataset to improve the performance of different variations of the CopyNet.

A. TASK FORMALIZATION

Given a source text t , and it can be formalized as a sequence of tokens $t = \{x_1, x_2, \dots, x_L\}$. The aim of the keyphrase generation task is to generate a sequence of keyphrases k for the source text.

$$k = \{p_1, p_2, \dots, p_N\} \quad (1)$$

where p_i is one of the keyphrases, and it can be represented as a sequence of words:

$$p_i = \{w_1, w_2, \dots, w_{l_i}\} \quad (2)$$

where l_i is the length of the keyphrase p_i .

Considering the keyphrases sequence k , we add the token “1” between the adjacent keyphrases p_i and p_j , while the token “0” is added between the adjacent words w_m and w_n in each keyphrase. Thus, the keyphrases for the source text t can be represented as a sequence of tokens, and this task can be formalized as a generation task.

B. ENCODER-DECODER FRAMEWORK

In this section, we present the encoder-decoder framework for keyphrases generation. Traditional encoder-decoder framework uses two basic RNNs (one is the encoder part, the other is the decoder part) to generate keyphrases. The encoder transforms the input sequence (source text t) to the context semantic vector c , and the decoder decodes the vector c to the output sequence (keyphrases k) with arbitrary length.

Fig. 2 illustrates the basic structure of the encoder-decoder framework. The input sequence of the framework is “ABC” and its output sequence is “WXYZ”. The token “<EOS>” is the end symbol of the input.

Formally, the encoder transforms the input sequence $x = \{x_1, x_2, \dots, x_L\}$ to the context vector c . RNN can be used for

the context vector generation process. The word x_t is the input at time t , and the hidden state h_t is defined as:

$$h_t = f(x_t, h_{t-1}) \quad (3)$$

When the whole input sequence is processed by the encoder, the context vector c is computed by:

$$c = g(h_1, h_2, \dots, h_L) \quad (4)$$

where the function g summarizes the hidden states, such as 1) summing or averaging all the hidden states for the input sequence; 2) directly selecting the last hidden state h_L as the context vector.

The decoder uses another RNN to unfold the context vector c to the output sequence $\{y_1, y_2, \dots, y_{L'}\}$ with arbitrary length. It predicts the output y_t at time t by:

$$s_t = f(s_{t-1}, y_{t-1}, c) \quad (5)$$

$$P(y_t | y_{i < t}, c) = g(y_{t-1}, s_t, c) \quad (6)$$

where s_t is the hidden state, $y_{i < t}$ represents the predicted history and y_t is the predicted output at time t . The predicted output y_t is selected from the vocabulary according to the probability $P(y_t | y_{i < t}, c)$ predicted by the classifier g .

C. ATTENTION MECHANISM

In the traditional encoder-decoder framework, the encoder transforms the source sequence into a context vector c , and the decoder predicts the output at each time based on the same context vector c and the predicted sequence. The same context vector c , which does not consider the different importance of the hidden states in the encoder, limits the performance of the framework.

The attention mechanism was first introduced into this framework in the machine translation task [34]. Different from the context vector described in the above traditional framework, the context vector in the attention mechanism aims to capture more information from the input sequence. It is computed by the weighted sum of the hidden states in the encoder, and also changes over the time t in the decoder:

$$c_t = \sum_{j=1}^L \alpha_{tj} h_j \quad (7)$$

$$\alpha_{tj} = \frac{\exp(\psi(s_{t-1}, h_j))}{\sum_{j=1}^L \exp(\psi(s_{t-1}, h_j))} \quad (8)$$

where h_j is the hidden state in the source sequence at time j and L is the length of the source sequence. ψ is a feed-forward neural network, and it represents the attention score between the position t in the target sequence and the position j in the source sequence.

D. CopyNet

The encoder-decoder framework with attention can be used for keyphrases generation. Compared with the traditional statistical-based method, the classical encoder-decoder

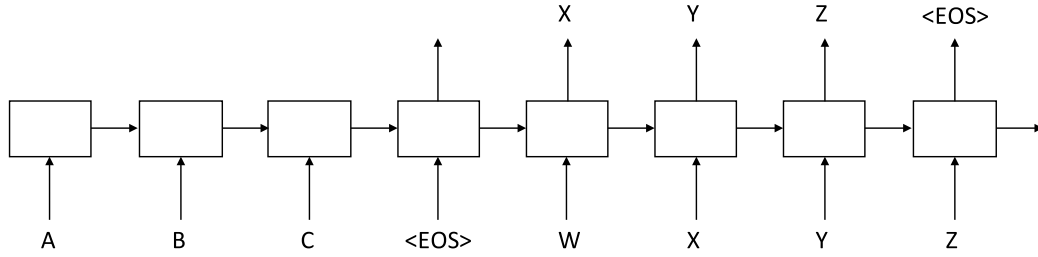


FIGURE 2. The basic structure of the encoder-decoder framework.

framework with attention mechanism has the ability to generate keyphrases that do not occur in the source text. It outputs a word at each time in the decoding process, and these predicted words come from a predefined vocabulary. However, some words in the keyphrases may not appear in the vocabulary. These unknown words cause the common out-of-vocabulary (OOV) problem in the keyphrases generation task and limit the performance of this framework.

In this section, we introduce the copying mechanism into the encoder-decoder framework, and present the CopyNet model to tackle this problem. The copying mechanism selects some important words from the source text as the keyphrases. If some important words in the source text are OOV words in the encoder-decoder framework, the copying mechanism may copy them from the text. Thus, the CopyNet for keyphrase extraction can handle the OOV problem, and has the ability to generate and copy keyphrases. Fig. 3 illustrates the overall structure of the CopyNet.

We can see that the CopyNet also belongs to the encoder-decoder framework and contains the encoder and decoder part. Different from the traditional encoder-decoder framework, the copying mechanism is introduced into the decoder part.

1) ENCODER

The CopyNet encoder is similar to that in the traditional encoder-decoder framework. As described in Fig. 3, it is implemented by bidirectional RNN. The input representation at each time is the word embedding of the corresponding word in the source sequence. h_t is the output representation at the current time t . All the hidden states for the source sequence form the encoder hidden state $M = \{h_1, h_2, \dots, h_L\}$. M contains all the output representation at each time, and L is the length of the source sequence.

At the end of the input sequence, we can get the context vector c as described in the Equation 4. The attention mechanism can be also introduced into the encoder-decoder framework as illustrated in Fig. 3 and the context vector c is computed by the Equation 7.

2) DECODER

Different from the traditional RNN-based decoder, the decoder, which incorporates the copying mechanism, predicts the output sequence by two parts. One is the generation part

$P_g(y_t|y_{i<t}, x)$, it is the probability of generating a token from the vocabulary as described in the Equation 6. The other is the copying part $P_c(y_t|y_{i<t}, x)$, it is the probability of copying a token from the source sequence:

$$P(y_t|y_{i<t}, x) = P_g(y_t|y_{i<t}, x) + P_c(y_t|y_{i<t}, x) \quad (9)$$

where $x = \{x_1, x_2, \dots, x_L\}$ is the source sequence, and $P(y_t|y_{i<t}, x)$ is the probability of generating the token y_t at time t in the target sequence.

Furthermore, different probability of the two parts is applied to improve the CopyNet in this paper. It predicts the output y_t by:

$$P(y_t) = \lambda P_g(y_t) + (1 - \lambda) P_c(y_t) \quad (10)$$

We assume that the vocabulary V is the predefined vocabulary in the generation part, and UNK is the OOV word. Given a source sequence $x = \{x_1, x_2, \dots, x_L\}$, and X is the set that contains all the words in x . Since X may contains some words that are not in the vocabulary V , copying mechanism selects appropriate words from X and enables the CopyNet to output some OOV words. The probability of the generation and copying part are denoted as following:

$$P_g(y_t|y_{i<t}, x) = \begin{cases} \frac{1}{Z} \exp(\phi_g(y_t)) & y_t \in V \\ 0 & y_t \in X \cap \bar{V} \\ \frac{1}{Z} \exp(\phi_g(UNK)) & y_t \notin V \cup X \end{cases} \quad (11)$$

$$P_c(y_t|y_{i<t}, x) = \begin{cases} \frac{1}{Z} \sum_{j:x_j=y_t} \exp(\phi_c(x_j)) & y_t \in X \\ \frac{1}{Z} \exp(\phi_c(UNK)) & other \end{cases} \quad (12)$$

where $\phi_g(y_t)$ and $\phi_c(y_t)$ are score functions for the generation part and copying part. Z is the normalization term. As described in the Equation 9, 11 and 12, all the words are divided into four classes, and their probability is illustrated in Fig. 4.

The generation score function $\phi_g(v_i)$ is the same as traditional RNN-based encoder-decoder [34], and it is denoted as:

$$\phi_g(v_i) = v_i^\top W_g s_t, \quad v_i \in V \cup UNK \quad (13)$$

where W_g is the parameter matrix, and v_i^\top is the one-hot indicator vector for v_i .

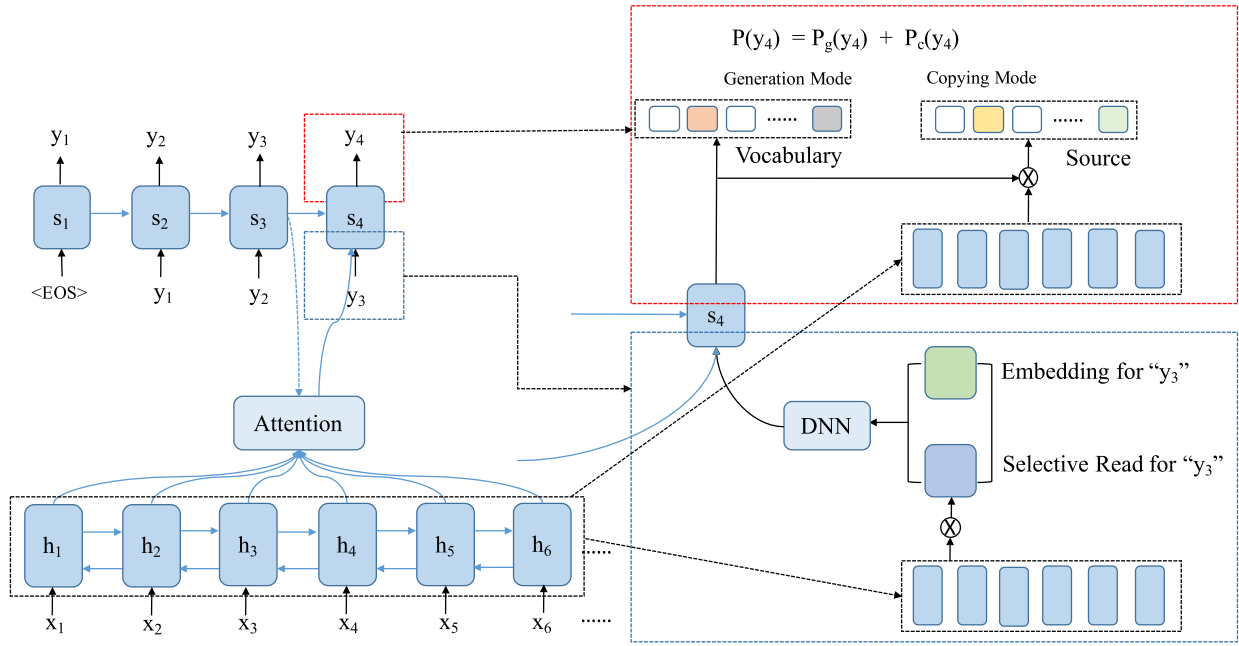


FIGURE 3. The overall structure of the CopyNet.

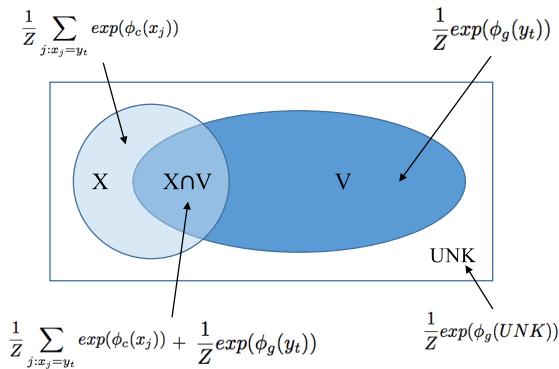


FIGURE 4. Probability for different classes.

The copying score function $\phi_c(x_j)$ for selecting x_j as the keyphrase word is denoted as:

$$\phi_c(x_j) = \sigma(h_j^T W_c) s_t, \quad x_j \in X \quad (14)$$

where h_j is the hidden state for x_j , W_c is the parameter matrix, and σ is the non-linear function. More details are described in (Gu et al., 2016) [36].

E. SEMANTIC WEB CONSTRUCTION

In this section, keyphrase semantic web is constructed based on the semantic relations between different keyphrases. The results of the semantic web can also be merged into the original corpus to participate in the training process of the neural network. For the training data, the semantic-related keyphrases in the semantic web are found for the annotated keyphrases. The source text and these semantic-related keyphrases form the new source text of this training example.

For the test data, since the annotated keyphrases can not be used in the system, we find keyphrases in the dictionary that appear in the source text, and semantic-related keyphrases are got for these keyphrases from the semantic web. The source text and these semantic-related keyphrases form the new source text of this testing example.

We focus on the keyphrases provided by the authors in the biomedical literatures, and discover the semantic relatedness between keyphrases. The example for one keyphrase is illustrated in Fig. 5. Semantic similarity based and word co-occurrence based methods are used to build the keyphrase semantic web, respectively.

1) SEMANTIC SIMILARITY BASED METHOD

Word2vec is an efficient and effective algorithm [38], [39], which learns a continuous feature vector to represent the word. The training data is a large-scale raw corpus, and the learned word representation can capture semantic information of the word. Following this way, we learn the keyphrases representation from the biomedical corpus, and build the semantic web using the following steps:

- **Step1:** The keyphrases dictionary, which contains all the unique keyphrases in the biomedical articles, is constructed.
- **Step2:** We preprocess the biomedical articles, and only keyphrase segments are retained in the corpus. These keyphrases are used as the training corpus of the word2vec method.
- **Step3:** Set the parameters for the training process and train the word2vec model.
- **Step4:** For each keyphrase in the dictionary, we select top-N keyphrases based on the similarity calculated by

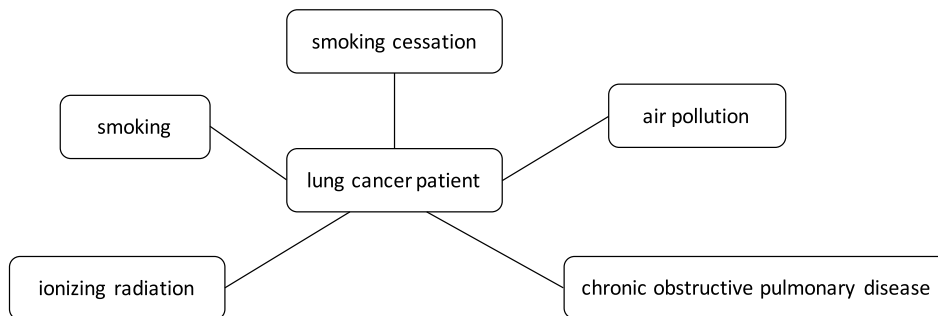


FIGURE 5. The semantic web for the keyphrase “lung cancer patient”.

the word2vec model. These top-N keyphrases forms the semantic web for the keyphrase.

2) WORD CO-OCCURRENCE BASED METHOD

In a large raw corpus, if two words are often occur in a fixed size window, we assume that these two words tend to have semantic relatedness. The word co-occurrence matrix is constructed based on the co-occurrence counts between the word and its context. The size of the matrix is $|V| * |V|$, and $|V|$ is the size of the keyphrase dictionary. V_{ij} represents the co-occurrence counts between the keyphrase p_i and p_j .

We construct the semantic web based on the word co-occurrence matrix using the following steps:

- **Step1:** Data preprocessing. We only focus on the co-occurrence count between different keyphrases, and the document is considered as the context window.
- **Step2:** Co-occurrence matrix construction. We calculate the co-occurrence count for all keyphrases in the dictionary. If two different keyphrases occur in one document, their co-occurrence count increases by 1.
- **Step3:** Semantic web construction. We select top-N keyphrases based on the co-occurrence count. These top-N keyphrases forms the semantic web for the keyphrase.

IV. EXPERIMENTS

A. EXPERIMENTAL SETTINGS

1) DATA AND EVALUATION

We crawled large amounts of biomedical articles from PubMed¹ for the keyphrase generation system evaluation and semantic web construction. The articles, that consist of title, abstract and keyphrases provided by the authors, were used for the experiments.

To ensure the performance of the automatic constructed keyphrase semantic web, we build specific semantic web for different diseases or topics. In this paper, cancer-related biomedical articles are selected to build the keyphrase semantic web. Based on these articles, we also generate the <title+abstract, keyphrases> pairs for system evaluation, and keyphrases provided by the authors are used as the golden standard keyphrases for the source text. This dataset is

¹<https://www.ncbi.nlm.nih.gov/pubmed/>

TABLE 1. Statistics of the biomedical keyphrase dataset.

Item	Value
number of articles	10282
average character length of the abstract	187
max character length of the abstract	345
average character length of the keyphrase	11
max character length of the keyphrase	31

available at <http://dx.doi.org/10.21227/xd0p-wd52>. It is split into two parts, namely 90% for training and 10% for testing in the experiments. Table 1 shows the statistics of the biomedical keyphrase generation corpus used in the experiment.

Precision (P), recall (R) and F1-score are used for the evaluation of the keyphrase generation system. We use extract match, when determining whether the keyphrase is correctly predicted.

Since there is no standard result for our large-scale keyphrase semantic web in previous work, we can not automatically evaluate the quality of the semantic web by standard evaluation. Manual evaluation is performed for the keyphrase semantic web built in our experiment. We randomly select 10% keyphrase from the dataset and manually build the semantic web for them by reviewing related literatures. The manually built keyphrase semantic web is used as the golden standard.

For each keyphrase, top-5 semantic-related keyphrases are selected by different methods to form the semantic web. The overlapping ratio between the automatically built semantic web and the golden standard is used for measuring the quality of the semantic web.

2) HYPER-PARAMETER SETTINGS

The hyper-parameters mainly include two parts. One is the structure definition of the neural network, including the size of different embeddings and the size of each hidden layer. The other part includes the hyper-parameters used in the training process.

We use GLOVE word embeddings [40] for our word embeddings initialization, and the dimension of word embeddings is 100. To ensure the training efficiency of the model, we set the max character length of the keyphrase to 30 according to the statistics of the dataset. Table 2 lists the details of these hyper-parameters.

TABLE 2. Hyper-parameter settings.

Type	Hyper-parameter
Dim(emb(word))	100
Hidden layer size	100
Initial learning rate	0.001
number of training iterations	40

TABLE 3. Results of different variations of the CopyNet.

Model	P(%)	R(%)	F1(%)
Only Generation Mode	16.07	16.28	16.17
Only Copying Mode	12.26	12.14	12.20
Probabilistic Mixture Mode ($\lambda = 0.3$)	17.09	17.01	17.05
Probabilistic Mixture Mode ($\lambda = 0.5$)	17.31	17.19	17.25
Probabilistic Mixture Mode ($\lambda = 0.7$)	18.07	18.01	18.04

3) BASELINES

Gu et al., 2016 [36] introduced the copying mechanism into the encoder-decoder framework. Besides the generation mode of the framework, it enables the model to select some important sequences from the input text. As described in Equation 9, 11 and 12, it predicts the output y_t at time t by $P(y_t) = P_g(y_t) + P_c(y_t)$. To evaluate the effect of different modes, we expand the CopyNet model into the following variations in the experiment:

- **Only Generation Mode:** Traditional statistical-based method can only identify keyphrases that occur in the source text. The generation model can overcome this problem and generate absent keyphrases. It predicts the output y_t by $P(y_t) = P_g(y_t)$.
- **Only Copying Mode:** The copying mechanism selects some important words from the source text as the keyphrases. It predicts the output y_t by $P(y_t) = P_c(y_t)$.
- **Probabilistic Mixture Mode:** Different probability for generation mode and copying mode is applied to improve the CopyNet. It predicts the output y_t by $P(y_t) = \lambda P_g(y_t) + (1 - \lambda) P_c(y_t)$.

B. RESULTS AND DISCUSSION

1) COMPARISON BETWEEN DIFFERENT VARIATIONS OF THE CopyNet

Table 3 shows the keyphrase generation performance of different variations of the CopyNet. Three different probabilities (λ) are used in the experiment. We can see that the probabilistic mixture mode with $\lambda = 0.7$ performs best among these baselines.

The generation part generates keyphrases from the predefined vocabulary. Compared with the system with only copying model, the system with only generation mode improves the F1-score with the gain of 3.97%. Furthermore, we can tune the probability of the generation part and the copying part in the CopyNet. With the generation probability λ increases, the keyphrase generation system performs better. Compared with the probabilistic mixture mode with $\lambda = 0.3$, the system with $\lambda = 0.7$ improves the F1-score from 17.05% to 18.04%.

TABLE 4. Results of different keyphrase semantic webs.

Semantic Web	Overlapping Rate
semantic similarity based semantic web	13%
word co-occurrence based semantic web	56%

TABLE 5. Effects of the keyphrase semantic web.

Model	+web(%)	-web(%)
Only Generation Mode	18.01	16.17
Only Copying Mode	14.15	12.20
Probabilistic Mixture Mode ($\lambda = 0.3$)	19.12	17.05
Probabilistic Mixture Mode ($\lambda = 0.5$)	19.53	17.25
Probabilistic Mixture Mode ($\lambda = 0.7$)	20.05	18.04

2) KEYPHRASE SEMANTIC WEB CONSTRUCTION

Table 4 lists the overlapping rate of different keyphrase semantic webs. We can see that the word co-occurrence based semantic web is better than the semantic similarity based semantic web.

In the semantic similarity based method, word2vec algorithm is used to train the feature representation for the keyphrases. Different from the previous research, we only use keyphrase list as the training corpus, instead of the complete sentences. While in the word co-occurrence based method, the related keyphrases for one specific keyphrase is the keyphrases with top co-occurrence counts in the raw corpus. In our experiment, the word co-occurrence based method can identify the semantic relatedness between keyphrases more accurately.

3) EFFECTS OF THE KEYPHRASE SEMANTIC WEB

In this section, we conduct the experiment to evaluate the effects of the keyphrase semantic web. The results of the semantic web can be merged into the dataset to participate in the training process.

Table 5 shows the F1-score of different systems with and without the keyphrase semantic web. “+web” represents that the semantic web is merged into the training and testing data, while “-web” means not. We can see that the keyphrase semantic web improves the performance of different models. When incorporating the semantic web into the training and test data, the F1-score of the probabilistic mixture mode with $\lambda = 0.7$ improves the F1-score from 18.04% to 20.05%.

4) CASE STUDY

As shown in Table 6, incorporating the keyphrase semantic web into the neural network framework improves the keyphrase generation performance of the CopyNet. The results of the CopyNet is better than the generation mode, while the generation mode is better than the copying mode.

The model with only copying mode performs worst among these models. The likely reason is that it can not copy all parts of the keyphrase. For example, the copying mode only selects “Centrifuga” for the keyphrase “Centrifugal proteomic reactor”. Although it can copy some important part from the source text, its ability to identify the boundary of the keyphrase needs improvement. The generation model generates words from a predefined vocabulary and

TABLE 6. An example with its predicted keyphrases by different systems. The upper part is the source text and the annotated keyphrases of this example. The lower part is the keyphrase predicted by different systems. “+web” represents that the semantic web is merged into the training and testing data, while “-web” means not.

Source Text	Proteomic analysis of minute amount of colonic biopsies by enteroscopy sampling. Colorectal cancer (CRC) is one of the most common types of malignant tumor worldwide. Currently, although many researchers have been devoting themselves in CRC studies, the process of locating biomarkers for CRC early diagnosis and prognostic is still very slow. Using a centrifugal proteomic reactor-based proteomic analysis of minute amount of colonic biopsies by enteroscopy sampling, 2620 protein groups were quantified between cancer mucosa and adjacent normal colorectal mucosa. Of which, 403 protein groups were differentially expressed with statistic significance between cancer and normal tissues, including 195 up-regulated and 208 down-regulated proteins in cancer tissues. Three proteins (SOD3, PRELP and NGAL) were selected for further Western blot validation. And the resulting Western blot experimental results were consistent with the quantitative proteomic data. SOD3 and PRELP are down-regulated in CRC mucosa comparing to adjacent normal tissue, while NGAL is up-regulated in CRC mucosa. In conclusion, the centrifugal proteomic reactor-based label-free quantitative proteomic approach provides a highly sensitive and powerful tool for analyzing minute protein sample from tiny colorectal biopsies, which may facilitate CRC biomarkers discovery for diagnoses and prognoses.	
Annotated Keyphrases	1. Centrifugal proteomic reactor; 2. Colorectal cancer; 3. Endoscopy; 4. Label-free proteomics; 5. Minute amount	
	+web(%)	-web(%)
Only Generation Mode	Mortality complication MDV 3100 colorectal cancer Endoscopy	Mortality complication quality of life colorectal cancer Exposure
Only Copying Mode	Colorectal cancer Centrifugal Excitability	Colorectal cancer Minute
Probabilistic Mixture Mode ($\lambda = 0.5$)	colorectal cancer Minute amount endoscopy Colonoscopy Default mode network	colorectal cancer Genetic testing MDV 3100 Centrifugal proteomic Carotid body tumors

TABLE 7. Semantic web for the keyphrase “Centrifugal proteomic reactor”.

Keyphrase	Semantic Web
Centrifugal proteomic reactor	Minute amount Endoscopy Label-free proteomics Colorectal cancer MDV 3100

lots of boundary indicators are included in the training data. Thus, the generation mode can identify the boundaries more accurately.

Table 7 shows the word co-occurrence based keyphrase semantic web for the keyphrase “Centrifugal proteomic reactor”. Incorporating the semantic web into the dataset provides more useful information for keyphrase generation.

5) LIMITATIONS OF OUR MODEL

We have conducted detailed analysis on the experimental results and there are still some limitations of our model.

1) Golden Standard. Keyphrases annotated by the authors are used as the golden standard for the source text in our experiment. In the annotation process, different annotators may give different keyphrases for the same document. This subjective factor causes that the performance of the keyphrase generation system does not seems very high.

2) Keyphrase Boundary. Additional tokens are used to identify the boundary of the keyphrase. We add the token “1” between the adjacent keyphrases, while the token “0” is added between the adjacent words in the keyphrase. Although keyphrase list can be got from the output sequence in this simple way, there are still wrong boundaries that directly generate wrong keyphrases.

V. CONCLUSION

In this paper, we presented the encoder-decoder framework with copying mechanism to generate keyphrases for the given text. Copying mechanism was introduced into this framework to handle the OOV problem, and some important words were selected from the source text as the keyphrases. The automatically constructed keyphrase semantic web could be merged into the dataset to participate in the training process. Experiments on the biomedical dataset demonstrated the effectiveness of our models. Tuning the probability of the generation part and the copying part in the CopyNet achieved better performance, and incorporating the keyphrase semantic web improved the performance of different variations of the CopyNet.

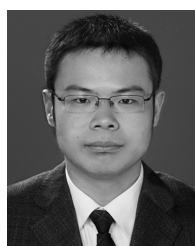
REFERENCES

- [1] P. D. Turney, “Learning algorithms for keyphrase extraction,” *Inf. Retr.*, vol. 2, no. 4, pp. 303–336, May 2000.
- [2] A. Hulth, “Improved automatic keyword extraction given more linguistic knowledge,” in *Proc. Conf. Empirical Methods Natural Lang. Process.* Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 216–223.
- [3] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, “KEA: Practical automated keyphrase extraction,” in *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific.* Harrisburg, PA, USA: IGI Global, 2005, pp. 129–152.
- [4] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, “Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles,” in *Proc. 5th Int. Workshop Semantic Eval.*, 2010, pp. 21–26.
- [5] G. Berend, “Exploiting extra-textual and linguistic information in keyphrase extraction,” *Natural Lang. Eng.*, vol. 22, no. 1, pp. 73–95, 2016.
- [6] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, “SemEval 2017 task 10: ScienceIE—extracting keyphrases and relations from scientific publications,” in *Proc. 11th Int. Workshop Semantic Eval. (SemEval)*, 2017, pp. 546–555.

- [7] K. S. Hasan and V. Ng, "Automatic keyphrase extraction: A survey of the state of the art," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics (Long Papers)*, vol. 1, 2014, pp. 1262–1273.
- [8] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, Jan. 1988.
- [9] F. Liu, D. Pennell, F. Liu, and Y. Liu, "Unsupervised approaches for automatic keyword extraction using meeting transcripts," in *Proc. Hum. Lang. Technol., Annu. Conf. North Amer. Chapter Assoc. for Comput. Linguistics (NAACL)*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 620–628.
- [10] S. R. El-Beltagy and A. Rafea, "KP-miner: Participation in semeval-2," in *Proc. 5th Int. Workshop Semantic Eval.*, 2010, pp. 190–193.
- [11] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Automatic keyphrase extraction from scientific articles," *Lang. Resour. Eval.*, vol. 47, no. 3, pp. 723–742, 2013.
- [12] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12 pp. 2493–2537, Aug. 2011.
- [13] Y. Ren, Y. Zhang, M. Zhang, and D. Ji, "Context-sensitive twitter sentiment classification using neural network," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 215–221.
- [14] C. Lyu, B. Chen, Y. Ren, and D. Ji, "Long short-term memory RNN for biomedical named entity recognition," *BMC Bioinf.*, vol. 18, no. 1, p. 462, Oct. 2017.
- [15] P. Ding, X. Zhou, X. Zhang, J. Wang, and Z. Lei, "An attentive neural sequence labeling model for adverse drug reactions mentions extraction," *IEEE Access*, vol. 6, pp. 73305–73315, 2018.
- [16] X. Wan and J. Xiao, "Single document keyphrase extraction using neighborhood knowledge," in *Proc. AAAI*, vol. 8, 2008, pp. 855–860.
- [17] M. Grineva, M. Grinev, and D. Lizorkin, "Extracting key terms from noisy and multitheme documents," in *Proc. 18th Int. Conf. World Wide Web (WWW)*, New York, NY, USA: ACM, 2009, pp. 661–670.
- [18] Q. Zhang, Y. Wang, Y. Gong, and X. Huang, "Keyphrase extraction using deep recurrent neural networks on Twitter," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 836–845.
- [19] W.-T. Yih, J. Goodman, and V. R. Carvalho, "Finding advertising keywords on Web pages," in *Proc. 15th Int. Conf. World Wide Web (WWW)*, New York, NY, USA: ACM, 2006, pp. 213–222.
- [20] T. D. Nguyen and M.-Y. Kan, "Keyphrase extraction in scientific publications," in *Proc. Int. Conf. Asian Digit. Libraries*, Berlin, Germany: Springer, 2007, pp. 317–326.
- [21] O. Medelyan, E. Frank, and I. H. Witten, "Human-competitive tagging using automatic keyphrase extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Stroudsburg, PA, USA: Association for Computational Linguistics, vol. 3, 2009, pp. 1318–1327.
- [22] Y. Zhang, E. Milios, and N. Zincir-Heywood, "A comparative study on key phrase extraction methods in automatic Web site summarization," *J. Digit. Inf. Manage.*, vol. 5, no. 5, pp. 323–332, 2007.
- [23] K. S. Hasan and V. Ng, "Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art," in *Proc. 23rd Int. Conf. Comput. Linguistics, Posters*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 365–373.
- [24] Z. Wu and C. L. Giles, "Measuring term informativeness in context," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2013, pp. 259–269.
- [25] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2004, pp. 404–411.
- [26] W. Ammar, M. Peters, C. Bhagavatula, and R. Power, "The A12 system at SemEval-2017 task 10 (ScienceIE): Semi-supervised end-to-end entity and relation extraction," in *Proc. 11th Int. Workshop Semantic Eval. (SemEval)*, 2017, pp. 592–596.
- [27] T. Tsujimura, M. Miwa, and Y. Sasaki, "TTI-COIN at SemEval-2017 task 10: Investigating embeddings for end-to-end relation extraction from scientific papers," in *Proc. 11th Int. Workshop Semantic Eval. (SemEval)*, 2017, pp. 985–989.
- [28] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: <http://arxiv.org/abs/1406.1078>
- [29] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. NIPS*, 2014, pp. 3104–3112.
- [30] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," 2015, *arXiv:1509.00685*. [Online]. Available: <http://arxiv.org/abs/1509.00685>
- [31] Z. Li, J. Cai, S. He, and H. Zhao, "Seq2seq dependency parsing," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 3203–3214.
- [32] A. Daza and A. Frank, "Translate and label! An encoder-decoder approach for cross-lingual semantic role labeling," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 603–615.
- [33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [34] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: <http://arxiv.org/abs/1409.0473>
- [35] M. Allamanis, H. Peng, and C. Sutton, "A convolutional attention network for extreme summarization of source code," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2091–2100.
- [36] J. Gu, Z. Lu, H. Li, and V. O. K. Li, "Incorporating copying mechanism in Sequence-to-Sequence learning," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics (Long Papers)*, vol. 1, 2016, pp. 1631–1640.
- [37] W. Zeng, W. Luo, S. Fidler, and R. Urtasun, "Efficient summarization with read-again and copy mechanism," 2016, *arXiv:1611.03382*. [Online]. Available: <http://arxiv.org/abs/1611.03382>
- [38] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: <http://arxiv.org/abs/1301.3781>
- [39] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
- [40] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.



XUN ZHU received the B.S. and M.S. degrees from the School of Computer, Central China Normal University, Wuhan, China, in 1999 and 2004, respectively. She is currently pursuing the Ph.D. degree at the School of Cyber Science and Engineering, Wuhan University. Her research interests include natural language processing, machine learning, and data mining.



CHEN LYU received the B.S. degree in mathematics and applied mathematics from Sun Yat-sen University, Guangzhou, in 2010, and the Ph.D. degree in computer software and theory from Wuhan University, Wuhan, in 2017. He was a Visiting Student with the Singapore University of Technology and Design, from 2014 to 2016. He is currently an Assistant Professor at the Guangdong University of Foreign Studies. His research interests include natural language processing, machine learning, and bioinformatics.



DONGHONG JI received the B.S., M.S., and Ph.D. degrees from the Computer School, Wuhan University, China, in 1988, 1991, and 1995, respectively. He was a Postdoctoral Research Fellow with Tsinghua University, from 1995 to 1998. From 1998 to 2008, he was a Research Scientist with the Institute for Infocomm Research, Singapore. He is currently a Professor and a Ph.D. Supervisor with the School of Cyber Science and Engineering, Wuhan University. His research interests include natural language processing, machine learning, and data mining.