Automatic Keyphrase Generation by Incorporating Dual Copy Mechanisms in Sequence-to-Sequence Learning

Siyu Wang[1], Jianhui Jiang[1], Yao Huang[1] and Yin Wang[1,2]
[1]Gusu Laboratory of Materials, Suzhou, China
[2]Hongzhiwei Technology (Shanghai) Co., Ltd, Shanghai, China
{wangsiyu2022, jiangjianhui2021, huangyao2021}@gusulab.ac.cn
wangyin@hzwtech.com

Abstract

The keyphrase generation task is a challenging work that aims to generate a set of keyphrases for a piece of text. Many previous studies based on the sequence-to-sequence model were used to generate keyphrases, and they introduce a copy mechanism to achieve good results. However, we observed that most of the keyphrases are composed of some important words (seed words) in the source text, and if these words can be identified accurately and copied to create more keyphrases, the performance of the model might be improved. To address this challenge, we propose a DualCopyNet model, which introduces an additional sequence labeling layer for identifying seed words, and further copies the words for generating new keyphrases by dual copy mechanisms. Experimental results demonstrate that our model outperforms the baseline models and achieves an obvious performance improvement.

1 Introduction

A keyphrase is a short piece of text that summarizes and abstracts the main semantics of a long text (named "document" or "source text" in this study). High-quality keyphrase promotes readers to efficiently understand, summarize and access documents' content (Meng et al., 2017). Not only that, extracting high-quality keyphrase had been widely applied to many downstream tasks in Natural Language Processing and Data Mining, such as Information Retrieval (Jones and Staveley, 1999), Text Summarization (Zhang et al., 2004), Text Categorization (Hulth and Megyesi, 2006) and Opinion Expression Mining (Berend, 2011). Thus, how to automatically extract high-quality keyphrases has become popular research topics in recent decades (Augenstein et al., 2017; Kim et al., 2010). Due to accessibility of text data, many datasets from scientific articles are used as benchmarks for keyphrase extraction algorithms. Thus, our study also focuses on extracting keyphrases from the scientific articles.

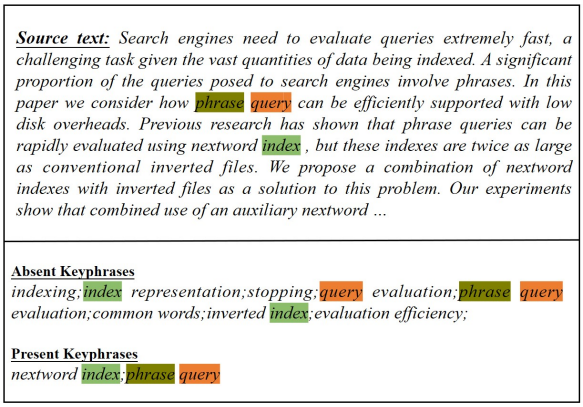

Figure 1: An example of keyphrases. The colored words (seed words) in the source text appear repeatedly in keyphrases.

Generally speaking, keyphrases can be divided into two categories: ***present keyphrase*** and ***absent keyphrase***. Present keyphrases are the explicit words that appear directly in source text, and vice versa for absent keyphrase. Many previous studies have focused on how to extract present keyphrases from documents. These studies (Campos et al., 2020; Hulth, 2003; Bougouin et al., 2014; Boudin, 2018; Bennani-Smires et al., 2018) consider the keyphrases extraction as a ranking task, which extracts a set of candidate phrases from the source text, and then selects keyphrases from the sorted candidates with the higher importance score. In recent years, some studies have also attempted to use deep learning methods for present keyphrase extraction. For example, Alzaidy et al. (2019) considered the present keyphrases extraction as a sequence labeling task by using LSTM-CRF model to label sequence, and obtained a better performance. Sun et al. (2021) used popular BERT (Devlin et al., 2019) model to extract present keyphrases. However, these methods were not expert in extracting absent keyphrases because the

Proceedings of the 29th International Conference on Computational Linguistics, pages 2328–2338
October 12–17, 2022.

source text has no absent keyphrase. To solve this problem, some sequence-to-sequence (Sutskever et al., 2014) based models were used to generate present/absent keyphrases, such as (Yuan et al., 2020; Chen et al., 2018; Meng et al., 2017; Chen et al., 2019; Ye et al., 2021). They first encoded the source text and then dynamically outputted corresponding present/absent keyphrases through a decoder. However, the decoder usually generates high-frequency words and ignores many out-of-vocabulary words, so it is unsatisfied in the task. To address this problem, Meng et al. (2017) incorporated a copy mechanism (Gu et al., 2016) in decoder to successfully predict out-of-vocabulary (OOV) words, namely OOV words copy mechanism. Besides, in many scientific articles, we observe that some words repeatedly appear in many keyphrases. For example, as shown in Figure 1, such as "*index*" appears in keyphrases "*nextword index*, *index representation*, *inverted index*", and "*query*" appears in keyphrases "*query evaluation*, *phrase query*" (these words are called seed words in this study). Therefore, if these seed words from source text can be identified and applied by the decoder to generate keyphrases as much as possible, the performance of the model will be greatly improved.

To address this challenge, we propose a novel sequence-to-sequence model (named DualCopyNet) to incorporate dual copy mechanisms for generating present/absent keyphrases. Since there is no labeled data, it is very difficult to directly extract the seed words from the source text. Thus, we try to extract present keyphrases as seed words. Specifically, besides a canonical encoder layer in DualCopyNet, we also introduce a sequence labeling layer for extracting present keyphrases (seed words). In addition, in the decoder layer, we introduce two kinds of copy mechanism. **Seed words copy mechanism**, it enables the decoder to generate phrases by selecting appropriate words from seed words. **OOV words copy mechanism**, it is a feasible solution that enables the decoder to predict OOV words by selecting appropriate words from the source text (Meng et al., 2017). Moreover, the decoder of DualCopyNet softly fuses the dual copy probability and generation probability through a gate mechanism to copy words (seed or OOV words) from the source text and generate words from the vocabulary. When training the model, we use a multi-task learning approach to optimize the primary task (generating keyphrases) and the auxiliary task (predicting seed words). Finally, we conduct experiments on four datasets. The results show that DualCopyNet has an obvious performance improvement in predicting present/absent keyphrases. The contributions of our paper are as follows:

- We introduce sequence labeling layer in the sequence-to-sequence architecture for predicting seed words and dynamically copy these words to generate more keyphrases.
- We design a novel decoder that incorporates dual copy mechanisms and uses a multi-task learning approach to optimize the model when generating present and absent keyphrases.
- On the four experimental datasets, our model outperforms most of the baseline models and obtains better results. Meanwhile, we demonstrate the positive effect of the dual copy mechanism by ablation study.

2 Related Work

2.1 Keyphrase Extraction

Many previous works (Hulth, 2003; Boudin, 2018; Witten et al., 1999; Bougouin et al., 2014) have been focusing on the study of keyphrases extraction. Generally, the extraction consists of two main steps: (1) Identifying candidate phrases by special hand-crafted rules (Hulth, 2003; Medelyan et al., 2009). (2) Sorting the candidate phrases to obtain keyphrases. For example, (Boudin, 2018; Bougouin et al., 2014; Campos et al., 2020; Mihalcea and Tarau, 2004) used an unsupervised approach to rank candidates. In recent years, some studies used a supervised approach for ranking, such as (Sun et al., 2021). And they achieved good results by introducing the BERT (Devlin et al., 2019) model. In addition, some studies considered keyphrase extraction as a sequence labeling task (Alzaidy et al., 2019). Although extraction-based methods obtained good results, they lacked an ability to predict absent keyphrases.

2.2 Keyphrase Generation

Due to previous methods' drawbacks for predicting absent keyphrases, Meng et al. (2017) first proposed a CopyRNN model to generate words from vocabulary and copy words from the source

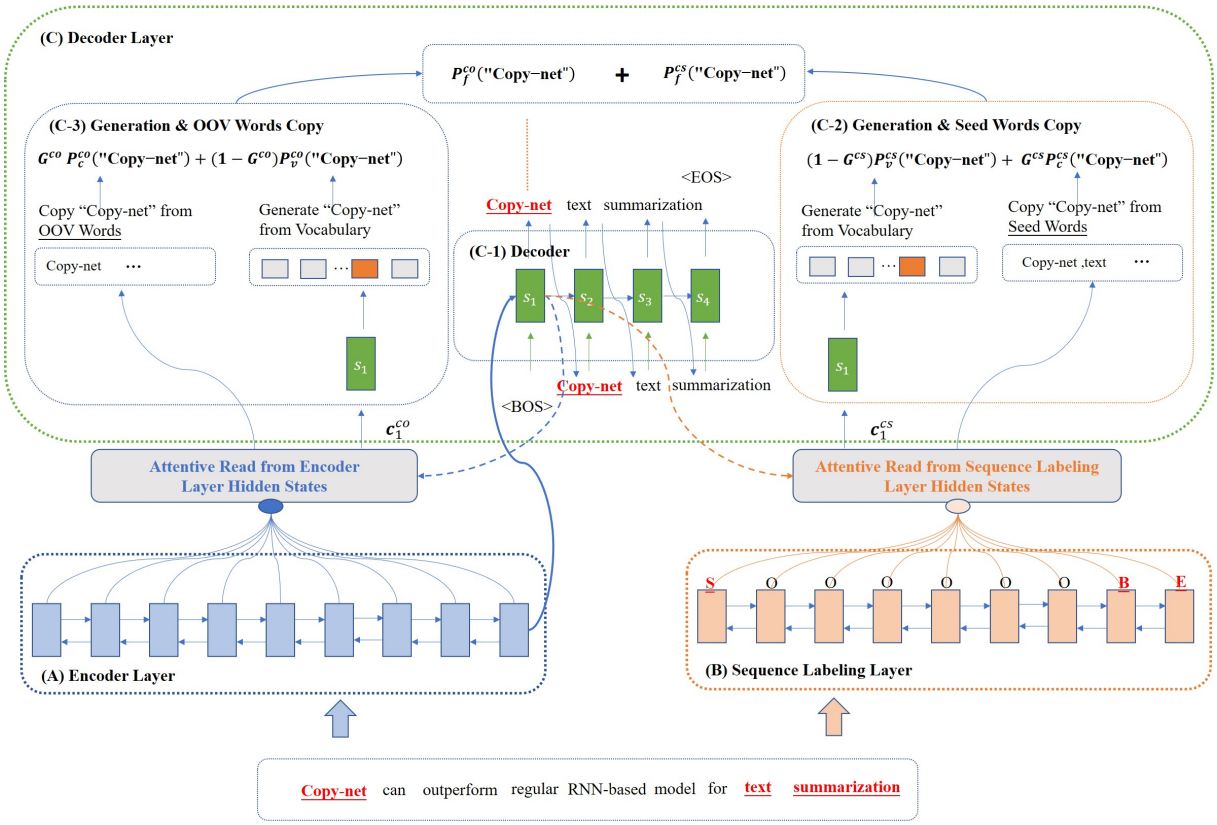

Figure 2: The overall structure of DualCopyNet, which includes three parts: **Encoder Layer**, **Sequence Labeling Layer** and **Decoder Layer**.

text. Subsequent studies have made many improvements on the basis of CopyRNN. (1) For problem with few training samples in certain domains, Ye and Wang (2018) proposed semi-supervised keyphrase generation method to leverage labeled data and large-scale unlabeled samples for learning. (2) Many studies ignored relationships among keyphrases, therefore Chen et al. (2018) proposed a new sequence-to-sequence architecture to capture correlation among keyphrases for generating new keyphrases. (3) Another problem is that the current researches ignored the leading role of the title. And Chen et al. (2019) realized the problem, then proposed a novel model named TGNet for keyphrase generation. (4) Different source texts should contain different number of keyphrases. Therefore, Yuan et al. (2020) proposed a recurrent generative model to generate multiple keyphrases by delimiter-separated sequences. (5) Keyphrases are inherently a disordered set rather than an ordered sequence, so Ye et al. (2021) proposed a new training paradigm ONE2SET to concatenate keyphrases without a predefined order.

In recent years, some new technologies have also been applied for keyphrase generation task, such as Reinforcement Learning (Chan et al., 2019) and Generative Adversarial Networks (GANs) (Swaminathan et al., 2020).

3 Methodology

3.1 Problem Definition

Given a keyphrase dataset that contains N data samples, and the i^{th} is denoted as $(x^{(i)}, p^{(i)})$, where $x^{(i)}$ is a source text, $p^{(i)}$ is a set of keyphrases. $p^{(i)}$ contains M_i keyphrases and denotes as $p^{(i)} = (p^{(i,1)}, p^{(i,2)}, \ldots, p^{(i,M_i)})$, where $p^{(i,j)}$ is one of $p^{(i)}$. $x^{(i)}$ and $p^{(i,j)}$ are word sequences:

$$x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots, x_{l_{x^{(i)}}}^{(i)}), \quad (1)$$

$$p^{(i,j)} = (y_1^{(i,j)}, y_2^{(i,j)}, \ldots, y_{l_{p^{(i,j)}}}^{(i,j)}), \quad (2)$$

where $l_x^{(i)}$ and $l_p^{(i,j)}$ is length of $x^{(i)}$ and $p^{(i,j)}$ respectively. The keyphrase generation task is to generate a set of keyphrase $p^{(i)}$ from the source text $x^{(i)}$, namely maximizes the probability $\prod_{i=1}^{N} \prod_{j=1}^{M_i} P(p^{(i,j)}|x^{(i)})$.

3.2 DualCopyNet Architecture

In this section, we will introduce the proposed DualCopyNet in detail. The model is based on

the sequence-to-sequence framework (Sutskever et al., 2014) and the copy mechanism (Gu et al., 2016). The structure shown in Figure 2 includes three parts: **Encoder Layer**, **Sequence Labeling Layer** and **Decoder Layer**.

Specifically, we first feed the source text into an encoder layer and a sequence labeling layer to obtain corresponding contextual representations. Then, we send the representations to a decoder layer to produce keyphrases. For a better generation, we introduce a dual copy mechanisms namely seed words copy mechanism and OOV words copy mechanism in the decoder layer. After that, the decoder dynamically generates words from the vocabulary or copies useful words from the source text.

Encoder Layer. To better generate contextual representations from a text, DualCopyNet adopts bi-directional GRUs (Cho et al., 2014) to encode the source text. And the text is composed of word embedding, which is defined as follows:

$$\boldsymbol{X} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n), \tag{3}$$

where $\boldsymbol{w}_i \in \mathbb{R}^{u_1}$ denotes the word embedding of the i^{th} word in the source text. Let u_1 be the dimension of the word embedding and n be the length of the source text. Then, $\boldsymbol{X} \in \mathbb{R}^{n \times u_1}$ is sent into the encoder layer, we employ bi-directional GRUs (Bi-GRU) to read the text sequence from two directions and output the hidden state of each word as follows:

$$\begin{aligned} \overrightarrow{\boldsymbol{u}_i} &= \overrightarrow{GRU}(\boldsymbol{w}_i, \boldsymbol{u}_{i-1}), \\ \overleftarrow{\boldsymbol{u}_i} &= \overleftarrow{GRU}(\boldsymbol{w}_i, \boldsymbol{u}_{i+1}). \end{aligned} \tag{4}$$

We then concatenate $\overrightarrow{\boldsymbol{u}_i} \in \mathbb{R}^{u_2}$ and $\overleftarrow{\boldsymbol{u}_i} \in \mathbb{R}^{u_2}$ to get the hidden state $\boldsymbol{u}_i$ of the i^{th} word, whose length is $2u_2$ and computed as follows:

$$\boldsymbol{u}_i = [\overrightarrow{\boldsymbol{u}_i}; \overleftarrow{\boldsymbol{u}_i}]. \tag{5}$$

Sequence Labeling Layer. As mentioned above, we observe that some seed words repeatedly appear in many keyphrases. Therefore, we expect to identify these seed words first and then dynamically copy them into the output of the decoder. Since there is no labeled data, it is very difficult to directly extract the seed words from the source text. Therefore, in this study, we consider identified present keyphrases as seed words and introduce a sequence labeling layer based on LSTM-CRF (Huang et al., 2015) in DualCopyNet to extract these seed words. Specifically, we first input the word embedding $\boldsymbol{x}_i \in \mathbb{R}^{u_1}$ at time step i into the bidirectional LSTM (Hochreiter and Schmidhuber, 1997) to obtain the hidden state $\boldsymbol{v}_i = [\overrightarrow{\boldsymbol{v}_i}; \overleftarrow{\boldsymbol{v}_i}]$ and derive the emission potential from $\boldsymbol{v}_i$. Meanwhile, an additional Conditional Random Field (CRF) (Lafferty et al., 2001) layer is employed to calculate the most probable tag for the each word. Then we use the BIESO scheme (*Begin, Intermediate, End, Single, Other*) to identify the seed words in the texts. For given a source text x, the conditional probability of a target tag^* is computed by:

$$P(tag^*|x) = \frac{\exp^{score(x, tag^*)}}{\sum_{\widetilde{tag}} \exp^{score(x, \widetilde{tag})}}, \tag{6}$$

where the function $score$ is defined as:

$$\begin{aligned} score(x, tag) = \sum_i log\phi_{emit}(i, tag_i) + \\ log\phi_{trans}(tag_{i-1} \rightarrow tag_i), \end{aligned} \tag{7}$$

where $\phi_{trans}(tag_{i-1} \rightarrow tag_i)$ is the transition score from tag_{i-1} to tag_i. $\phi_{emit}(i, tag_i)$ is the score of the tag_i for the i^{th} input word and comes from the hidden state of the Bi-LSTM at timestep i.

Finally, the loss function of the sequence labeling layer is defined as:

$$\mathcal{L}^s = -log(P(tag^*|x)). \tag{8}$$

Decoder Layer. The source text has been encoded into two kinds of contextual representations through an encoder layer and a sequence labeling layer, respectively. Further, we adopt the decoder based on the attention mechanism to fuse the dual copy mechanisms to generate keyphrases. The decoder is created with one-way GRU. For each time step t, the GRU fuses the hidden state $\boldsymbol{s}_{t-1}$ and the word embedding $\boldsymbol{e}_{t-1}$ of the output word y_{t-1}, which is computed by:

$$\boldsymbol{s}_t = GRU(\boldsymbol{e}_{t-1}, \boldsymbol{s}_{t-1}), \tag{9}$$

where $t-1$ denotes previous time step, $\boldsymbol{e}_0$ is the embedding of the start token '<BOS>'.

Generation and Seed Words Copy. As mentioned above, the sequence labeling layer is adopted to predict seed words. Then we expect to dynamically copy these words into the output sequence for generating the keyphrases. Specifically,

we first use the decoder vector $\widetilde{\boldsymbol{s}_t^{cs}}$ to generate the next word, where $\widetilde{\boldsymbol{s}_t^{cs}} = [\boldsymbol{c}_t^{cs}; \boldsymbol{s}_t]$ is derived from concatenating $\boldsymbol{c}_t^{cs}$ and $\boldsymbol{s}_t$, and the context vector $\boldsymbol{c}_t^{cs}$ is computed as a weighted sum of hidden representations by the concatenate attention mechanism (Luong et al., 2015):

$$\boldsymbol{c}_t^{cs} = attn(\boldsymbol{s}_t, [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots \boldsymbol{v}_n], \boldsymbol{W}_{att}^{cs}). \quad (10)$$

Then, the probability of generating the word $P_v^{cs}(y_t)$ from vocabulary is computed by:

$$P_v^{cs}(y_t|y_{t-1}, x) = softmax(\boldsymbol{W}_{v2}(\boldsymbol{W}_{v1}\widetilde{\boldsymbol{s}_t^{cs}} + b_{v1}) + b_{v2}). \quad (11)$$

Furthermore, a copy mechanism (Gu et al., 2016) is adopted to efficiently extract the in-text information and strengthen the generation capability of our model. We first calculate a soft gate G_t^{cs} to dynamical select the way of output, that is whether generating from the vocabulary or copying from the seed words at time step t, which is defined as:

$$G_t^{cs} = sigmoid(\boldsymbol{W}_g^{cs}\boldsymbol{m}_t^{cs} + b_g^{cs}), \quad (12)$$

where $\boldsymbol{m}_t^{cs} = [\boldsymbol{e}_{t-1}; \widetilde{\boldsymbol{s}_t^{cs}}]$ is the concatenation of the embedding of previous output word $y_{(t-1)}$ and $\widetilde{\boldsymbol{s}_t^{cs}}$. Then, the probability of predicting a word y_t by copying from seed words or generating from vocabulary is defined as:

$$P_f^{cs}(y_t) = G_t^{cs}P_c^{cs}(y_t) + (1 - G_t^{cs})P_v^{cs}(y_t), \quad (13)$$

where $P_c^{cs}(y_t) = \sum_{i:x_i=y_t} a_{t,i}^{cs}$ is a probability of seed word copy for y_t . And $\sum_{i:x_i=y_t} a_{t,i}^{cs}$ is the normalized attention weight between $\boldsymbol{s}_t$ and sequence labeling layer hidden state $\boldsymbol{v}_i$.

Generation and OOV Words Copy. The probability of generating the word $P_v^{co}(y_t)$ for current step is computed by:

$$P_v^{co}(y_t|y_{t-1}, x) = softmax(\boldsymbol{W}_{v4}(\boldsymbol{W}_{v3}\widetilde{\boldsymbol{s}_t^{co}} + b_{v3}) + b_{v4}), \quad (14)$$

where $\widetilde{\boldsymbol{s}_t^{co}} = [\boldsymbol{c}_t^{co}; \boldsymbol{s}_t]$ and the context vector $\boldsymbol{c}_t^{co}$ is computed by:

$$\boldsymbol{c}_t^{co} = attn(\boldsymbol{s}_t, [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_n], \boldsymbol{W}_{att}^{co}). \quad (15)$$

Then, the second soft gate G_t^{co} is computed by:

$$G_t^{co} = sigmoid(\boldsymbol{W}_g^{co}\boldsymbol{m}_t^{co} + b_g^{co}), \quad (16)$$

where $\boldsymbol{m}_t^{co} = [\boldsymbol{e}_{t-1}; \widetilde{\boldsymbol{s}_t^{co}}]$. Eventually, the probability of predicting a word y_t by copying from OOV words or generating from vocabulary is computed by:

$$P_f^{co}(y_t) = G_t^{co}P_c^{co}(y_t) + (1 - G_t^{co})P_v^{co}(y_t), \quad (17)$$

where $P_c^{co}(y_t) = \sum_{i:x_i=y_t} a_{t,i}^{co}$ is a probability of OOV word copy for y_t. And $\sum_{i:x_i=y_t} a_{t,i}^{co}$ is the normalized attention score between $\boldsymbol{s}_t$ and encoder hidden state $\boldsymbol{u}_i$.

In the end, the final probability distribution of predicting a word y_t is computed by summation of $P_f^{cs}(y_t)$ and $P_f^{co}(y_t)$:

$$P_f(y_t) = P_f^{cs}(y_t) + P_f^{co}(y_t). \quad (18)$$

3.3 Training Loss

DualCopyNet is based on sequence-to-sequence structure and involves two tasks: keyphrase generation and sequence labeling. Thus, the loss function contains two parts: the sequence labeling layer introduces an additional CRF loss (equation 8), and the decoder layer adopts the negative log likelihood (NLL) loss, which is defined as:

$$\mathcal{L}^g = -\sum_{t=1}^{L_y} logP_f(y_t|y_{t-1}, x, \theta). \quad (19)$$

We define the overall loss function with the CRF loss and the NLL loss:

$$\mathcal{L} = \mathcal{L}^s + \mathcal{L}^g. \quad (20)$$

The loss is calculated as the average over mini batch. Finally, we use Adam (Kingma and Ba, 2014) to optimize the model.

4 Experimental Settings

In this section, we will describe the training and testing sets used for the experiments, then introduce the baseline models and evaluation metrics.

Dataset	|Samples|	#KP	%A-KP
Inspec	500	9.79	26.42
NUS	211	10.81	45.36
SemEval	100	14.43	55.61
KP20k	20000	5.26	37.23

Table 1: Statistics for the four testing datasets. |Samples|: the number of samples on the dataset, #KP: the avg number of keyphrases, %A-KP: the proportion of absent keyphrase.

4.1 Dataset

We choose the largest public dataset KP20k (Meng et al., 2017) for training models, it contains a large number of high-quality academic papers mainly from the computer science field. The dataset has 527,830 articles for training and 20000 articles for validation.

Furthermore, we evaluate our model on four testing datasets widely adopted in previous works (Chen et al., 2018; Yuan et al., 2020; Chen et al., 2019; Ye et al., 2021; Swaminathan et al., 2020; Chen et al., 2020), including Inspect (Hulth, 2003), NUS (Nguyen and Kan, 2007), KP20k (Meng et al., 2017) and SemEval (Kim et al., 2010). Table 1 summarizes the statistics of each testing dataset.

4.2 Baselines

In our experiments, we choose the following keyphrase generation models as the baselines:

- catSeq (Yuan et al., 2020): An RNN-based attentional encoder-decoder model with copy mechanism.
- catSeqD (Yuan et al., 2020): catSeq augmented with orthogonal regularization and semantic coverage mechanism.
- catSeqCorr (Chen et al., 2018): a new sequence-to-sequence architecture for keyphrases generation, which captures correlation among multiple keyphrases in two ways.
- catSeqTG (Chen et al., 2019): a new sequence-to-sequence architecture for keyphrases generation, which explicitly considers the leading role of the title to the overall document main body.
- catSeq-RF (Chan et al., 2019): An extension of catSeq with RL-based finetuning, which introduces an adaptive reward function and encourages the model to generate both sufficient and accurate keyphrases.
- GAN-mr (Swaminathan et al., 2020): A novel model for keyphrase generation approach using Generative Adversarial Networks (GANs).
- ExHiRD-h (Chen et al., 2020): An exclusive hierarchical decoding model with a hard exclusion mechanism.

4.3 Implementation Details

The models catSeq, catSeqD, catSeqCorr, catSeqTG and catSeq-RF are implemented by (Chan et al., 2019). Followed by (Yuan et al., 2020; Chen et al., 2020; Swaminathan et al., 2020; Chan et al., 2019), when training our model, the ground-truth keyphrase sequence is the concatenation of present and absent keyphrases. Then the present keyphrases are sorted according to the initial orders arose in the document, and the absent keyphrases keep their original orders. Furthermore, we replace all digits with the symbol <digit> and define the vocabulary V with the most frequent words numbered 50,000.

The embedding size and hidden size of GRU, LSTM are set to 150; The batch size is 64 and learning rate is 0.0001; The gradient clipping is set to 1 and dropout is set to 0.1. The hyper-parameters are tuned on validation set. Early stopping is applied when the validation loss stops dropping three continuous evaluations. During testing, we set the maximum depth of the beam search as 6 and the beam size as 200. While on KP20k dataset, due to the large amount of test data, we set the beam size as 20. We implement the model using Pytorch (Paszke et al., 2019) and train the model using NVIDIA 3090TI and Ubuntu System.

4.4 Evaluation Metrics

Same as previous work (Chen et al., 2019; Swaminathan et al., 2020; Chan et al., 2019; Ye et al., 2021; Chen et al., 2020), we adopt the macro-averaged F1@5 and F1@M as the evaluation metrics. F1@M compares all keyphrases predicted by the model with the ground-truth to compute the F1 score. And F1@5 compares top 5 keyphrases predicted by the model with the ground-truth. Specifically, when the number of predictions is less than five, F1@5 will be the same as F1@M, so we must randomly append incorrect keyphrases to fill five predictions instead of directly using the original predictions. Furthermore, we also apply Porter Stemmer for preprocessing before comparisons.

5 Results and Analysis

5.1 Keyphrases Prediction

In this section, we will evaluate the performance of the model in predicting present keyphrase and absent keyphrase separately.

The performances of predicting present keyphrase are shown in Table 2. As we can see

Model	Inspec		NUS		SemEval		KP20k	
	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5
catSeq	0.262	0.225	0.397	0.323	0.283	0.242	0.367	0.291
catSeqD	0.263	0.219	0.394	0.321	0.274	0.233	0.363	0.285
catSeqCorr	0.269	0.227	0.390	0.319	0.290	0.246	0.365	0.289
catSeqTG	0.270	0.229	0.393	0.325	0.290	0.246	0.366	0.292
catSeq-RF	0.300	0.250	**0.426**	0.364	0.327	0.285	**0.383**	0.310
GAN-mr	0.299	0.258	0.417	0.348	\	\	0.378	0.303
ExHiRD-h	0.291	0.253	\	\	0.335	0.284	0.374	0.311
DualCopyNet	**0.342**	**0.284**	0.395	**0.379**	**0.339**	**0.315**	0.337	**0.312**

Table 2: F1 of present keyphrases prediction on four datasets. The best/second results in each column are highlighted with bold/underline.

Model	Inspec		NUS		SemEval		KP20k	
	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5
catSeq	0.008	0.004	0.028	0.016	0.028	0.020	0.032	0.015
catSeqD	0.011	0.007	0.024	0.014	0.024	0.016	0.031	0.015
catSeqCorr	0.009	0.005	0.024	0.014	0.026	0.018	0.032	0.015
catSeqTG	0.011	0.005	0.018	0.011	0.027	0.019	0.032	0.015
catSeq-RF	0.017	0.009	0.031	0.019	0.027	0.018	**0.047**	0.024
GAN-mr	0.019	**0.013**	0.038	0.026	\	\	0.045	**0.032**
ExHiRD-h	**0.022**	0.011	\	\	0.025	0.017	0.032	0.016
DualCopyNet	0.014	0.012	**0.055**	**0.038**	**0.029**	**0.023**	0.042	0.025

Table 3: F1 of absent keyphrases prediction on four datasets. The best/second results in each column are highlighted with bold/underline.

that DualCopyNet greatly outperforms the whole baseline models on F1@5, especially on Inspec, NUS and SemEval datasets, but there is only a slight improvement on the KP20k dataset. For F1@M, DualCopyNet also achieves the best results on the Inspec and SemEval datasets, outperforms all models, which demonstrates the effectiveness of our method.

Predicting absent keyphrases is a challenging task. As shown in Table 3, we can see that all models are poor in predicting absent keyphrases comparing to predicting present keyphrases. In this task, DualCopyNet achieves better performance on the NUS and SemEval datasets and outperforms all baseline models. While, the performance is slightly lower than some baseline models on Inspect and KP20k datasets.

Overall, the advantage of our model is more obvious on datasets containing more target keyphrases, such as NUS, SemEval and Inspec. Because the more keyphrases, the greater the number of seed words included, and the model can achieve a greater performance improvement. In contrast, as can be seen from Table 1, The average number of keyphrases on KP20k is much lower than the other three datasets. Therefore, our model does not achieve the best results on KP20k dataset.

5.2 Ablation Study

We conduct an ablation study to further analyze dual copy mechanisms and multi-task learning. First, we introduce three variants of DualCopyNet:

- SeqLabelingNet: We remove the decoder layer and only keep the sequence labeling layer.
- OOVCopyNet: We remove the seed words copy mechanism in the decoder layer, and only keep the OOV words copy mechanism.
- DualCopyNet$^{\text{nll}}$: This model is same as DualCopyNet. But we only use the negative log-likelihood loss (Equation 19) to optimize the model when training phase.

The present keyphrases prediction results of the ablation study are shown in Table 4. After adding the seed words copy mechanism, the performance of the models (DualCopyNet and DualCopyNet)

Model	Inspec			NUS			SemEval			KP20k		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SeqLabelingNet	0.275	0.075	0.118	0.446	0.169	0.245	0.389	0.142	0.208	0.349	0.222	0.271
OOVCopyNet	0.407	0.273	0.327	0.353	0.402	0.375	0.285	0.353	0.315	0.258	0.473	0.334
DualCopyNet^{nll}	0.419	0.276	0.332	0.351	0.435	0.388	0.280	0.370	0.319	0.259	0.485	**0.338**
DualCopyNet	0.403	0.298	**0.342**	0.348	0.458	**0.395**	0.303	0.384	**0.339**	0.258	0.485	0.337

Table 4: Ablation study on the four datasets for present keyphrases predication. The P denotes Precision@M, the R denote Recall@M and the F1 denote F1@M. The best/second results in each column are highlighted with bold/underline.

Model	Inspec			NUS			SemEval			KP20k		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OOVCopyNet	0.008	0.012	0.010	0.041	0.032	0.036	0.045	0.013	0.020	0.038	0.036	0.037
DualCopyNet^{nll}	0.013	0.019	**0.015**	0.048	0.039	0.043	0.049	0.025	**0.033**	0.040	0.044	**0.042**
DualCopyNet	0.012	0.015	0.014	0.063	0.048	**0.055**	0.062	0.019	0.029	0.040	0.044	**0.042**

Table 5: Ablation study on the four datasets for absent keyphrase predication. The best/second results in each column are highlighted with bold/underline.

on four datasets has been obviously improved, especially in Recall, which means the model can recall more keyphrases through the seed words copy mechanism, and this is also consistent with our expectations. Due to the KP20k dataset only contains few keyphrases, so there is only a slight improvement on F1. In addition, the precision of SeqLabelingNet achieves the best results on most of datasets, so it is reasonable and effective to identify present keyphrases as seed words and copy them into the decoder.

Since SeqLabelingNet cannot predict the absent keyphrase, there only remains three models. The experimental results are shown in Table 5. In the prediction of absent keyphrases, the models (DualCopyNet^{nll} and DualCopyNet) have obviously improved in Recall, Precision and F1 after adding the seed words copy mechanism. Furthermore, the models DualCopyNet and DualCopyNet^{nll} have the same structure, but DualCopyNet employs multi-task learning when training phase. As can be seen from Table 4 and Table 5, F1 of DualCopyNet outperforms DualCopyNet^{nll} on most of datasets. It proves that multi-task learning can effectively improve the performance of the model.

5.3 Ability to Generate Diverse Keyphrases

To investigate the model's ability of generating diverse keyphrases, we adopt NDCG (Wang et al., 2013) to evaluate models. NDCG is used to evaluate the diversity of generated text, which is widely

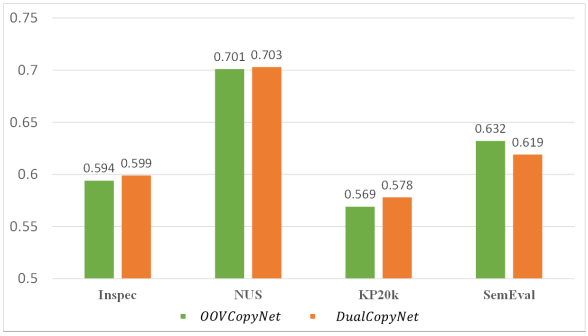

Figure 3: NDCG@10 metrics on four datasets.

used in text generation tasks (Habibi and Popescu-Belis, 2013) and information retrieval (Santos et al., 2013) tasks. The higher NDCG means the more diverse content that the model can generate. It works by penalizing redundant keyphrases and rewarding new keyphrases.

The results are summarized in Figure 3. Compared to OOVCopyNet, DualCopyNet achieves better NDCG@10 on three of the four datasets. Especially, there is an obvious improvement on the KP20k and Inspec datasets. It proves that the introduction of the seed words copy mechanism not only improves the performance of the model, but also generates more diverse phrases.

5.4 Case Study

In Figure 4, we show examples of keyphrases generated by SeqLabelingNet, OOVCopyNet and DualCopyNet respectively. After inputting the title and the abstract, we can see that DualCopyNet

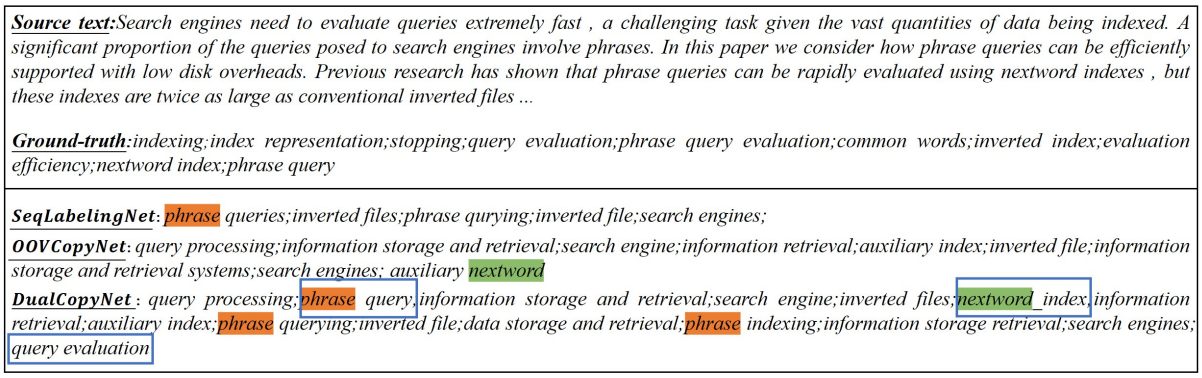

Figure 4: Case study. The keyphrases in the rectangle are truth keyphrases. Words remarked with green indicate copied from OOV words. Words remarked with orange indicate copied from seed words.

can generate more diverse keyphrases, the quantity even exceeds the ground-truth. Not only that, these generated keyphrases are generally reasonable and typical. Next, we can find that DualCopyNet copies the seed words "***phrase***" predicted by the sequence labeling layer to generate some new keyphrases, such as "***phrase indexing***". But the keyphrases generated by OOVCopyNet contain none of the above words, which proves the effectiveness of the seed words copy mechanism. On the other hand, DualCopyNet also generates keyphrases copying from OOV word "***nextword***". Finally, through the case study, it is proved that our model can well integrate the two copy mechanisms and effectively improve the performance of generating keyphrases.

6 Conclusions

In this paper, we propose a novel DualCopyNet for keyphrases generation. Based on the phenomenon of that many keyphrases are composed of seed words in the source text, we design dual copy mechanisms to precisely copy seed words and OOV words from the source text. Furthermore, aiming to obtain seed words, we introduce an additional sequence labeling layer and train the model with a multi-task learning. Finally, the experiments conducted on multiple datasets show our model's achievements are higher than most of baselines. Meanwhile, ablation experiments show a positive effect of the seed word copy mechanism and multi-task learning.

References

Rabah Alzaidy, Cornelia Caragea, and C Lee Giles. 2019. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In *The world wide web conference*, pages 2551–2557.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555.

Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 221–229.

Gábor Berend. 2011. Opinion expression mining by exploiting keyphrase extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1162–1170.

Florian Boudin. 2018. Unsupervised keyphrase extraction with multipartite graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 667–672.

A Bougouin, F Boudin, and B Daille. 2014. Topicrank: Topic ranking for automatic keyphrase extraction. *Revue Traitement Automatique des Langues*, 55(1):4569.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Hou Pong Chan, Wang Chen, Lu Wang, and Irwin King. 2019. Neural keyphrase generation via reinforcement learning with adaptive rewards. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2163–2174.

Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase generation with correlation constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4057–4066.

Wang Chen, Hou Pong Chan, Piji Li, and Irwin King. 2020. Exclusive hierarchical decoding for deep keyphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1095–1105.

Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R Lyu. 2019. Title-guided encoding for keyphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6268–6275.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640.

Maryam Habibi and Andrei Popescu-Belis. 2013. Diverse keyword extraction from conversations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 651–657.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 216–223.

Anette Hulth and Beáta Megyesi. 2006. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 537–544.

Steve Jones and Mark S Staveley. 1999. Phrasier: a system for interactive document retrieval using keyphrases. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–167.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1318–1327.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *International conference on Asian digital libraries*, pages 317–326.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Rodrygo LT Santos, Pablo Castells, Ismail Sengor Altingovde, and Fazli Can. 2013. Diversity and novelty in information retrieval. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 1130–1130.

Si Sun, Zhenghao Liu, Chenyan Xiong, Zhiyuan Liu, and Jie Bao. 2021. Capturing global informativeness in open domain keyphrase extraction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 275–287.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference*

on Neural Information Processing Systems - Volume 2, page 3104–3112.

Avinash Swaminathan, Haimin Zhang, Debanjan Mahata, Rakesh Gosangi, Rajiv Shah, and Amanda Stent. 2020. A preliminary exploration of gans for keyphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8021–8030.

Yining Wang, Liwei Wang, Yuanzhi Li, Di He, and Tie-Yan Liu. 2013. A theoretical analysis of ndcg type ranking measures. In *Conference on learning theory*, pages 25–54.

Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 1999. Kea: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*, pages 254–255.

Hai Ye and Lu Wang. 2018. Semi-supervised learning for neural keyphrase generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4142–4153.

Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021. One2set: Generating diverse keyphrases as a set. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4598–4608.

Xingdi Yuan, Tong Wang, Rui Meng, Khushboo Thaker, Peter Brusilovsky, Daqing He, and Adam Trischler. 2020. One size does not fit all: Generating and evaluating variable number of keyphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7961–7975.

Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. 2004. World wide web site summarization. *Web intelligence and agent systems: an international journal*, 2(1):39–53.