# Keyphrase Extraction with Incomplete Annotated Training Data

**Yanfei Lei**
Beihang University, China
leiyanfei@buaa.edu.cn

**Chunming Hu**
Beihang University, China
hucm@buaa.edu.cn

**Guanghui Ma**
Beihang University, China
magh@act.buaa.edu.cn

**Richong Zhang** *
Beihang University, China
zhangrc@buaa.edu.cn

## Abstract

Extracting keyphrases that summarize the main points of a document is a fundamental task in natural language processing. Supervised approaches to keyphrase extraction(KPE) are largely developed based on the assumption that the training data is fully annotated. However, due to the difficulty of keyphrase annotating, KPE models severely suffer from incomplete annotated problem in many scenarios. To this end, we propose a more robust training method that learns to mitigate the misguidance brought by unlabeled keyphrases. We introduce negative sampling to adjust training loss, and conduct experiments under different scenarios. Empirical studies on synthetic datasets and open domain dataset show that our model is robust to incomplete annotated problem and surpasses prior baselines. Extensive experiments on five scientific domain datasets of different scales demonstrate that our model is competitive with the state-of-the-art method.[1]

## 1 Introduction

Keyphrase extraction is the task of automatically extracting a set of phrases that provide a concise summary of a document content. An effective keyphrase extraction (KPE) system can benefit a wide range of natural language processing and information retrieval tasks(Hasan and Ng, 2014), such as text categorization(Hulth and Megyesi), opinion mining(Berend, 2011) and recommendation(Pudota et al., 2010). Recent supervised neural methods typically treat keyphrase extraction as a classification problem(Augenstein and Søgaard, 2017; Sun et al., 2020; Xiong et al., 2019), where given phrases are classified as keyphrases or non-keyphrases. Although supervised methods perform well in this task, it requires a large amount of labeled data which is extremely expensive and time-consuming to collect in many application scenarios(Liu et al., 2012).

The latest advances in neural KPE are mainly carried out on data sets in scientific domain datasets.(Augenstein et al., 2017; Meng et al., 2017). Because scientific domain has sufficient and high-quality training data for neural methods: some training data are annotated by authors, and the author is the person who is most familiar with their articles, the keyphrases annotated by author are high-quality. However, due to the requirements for professional knowledge in scientific domain, some keyphrases annotated by readers are incomplete. Table 1 shows an example case with incomplete keyphrase annotation. Obviously, "model validation" is more suitable as a keyphrase than "paper", "model" and "use", but "model validation" isn't annotated as keyphrase.

It has been suggested that keyphrase annotation is highly subjective(Sterckx et al., 2016). In real world scenarios, most potential applications of KPE deal with diverse documents originating from sparse sources that are rather different from scientific papers(Xiong et al., 2019). They often involve various domains whose contents target much wider audiences than scientists and require a large amount of high-quality labeled data which is extremely expensive to collect(Wang et al., 2018).

There are several prior works focused on data quality and insufficient labeled data issues, Sterckx et al.(Sterckx et al., 2016) treat supervised keyphrase extraction as Positive Unlabeled Learning(Ren et al., 2014) by reweighting importance of training samples. Wang et al.(Wang et al., 2018) used the idea of transfer learning and proposed Topic-based Adversarial Neural Network (TANN). They exploited unlabeled data in the target domain and data in the resource-rich source domain to alleviate incomplete annotated problem. However,

*Corresponding Author

[1]Our code and models are available at https://github.com/fredia/NS-KPE

Table 1: An example case from Kp20K[2] with incomplete keyphrase annotations, which "model validation" is more suitable as a keyphrase than "paper", "model" and "use", but "model validation" isn't annotated as keyphrase.

| | |
|---|---|
| **title** | The use of graphical models in model validation |
| **abstract** | The use of graphical models for model specification and in modelling is increasing rapidly. This paper discusses the use of these graphical models in model validation. |
| **annotated keyphrases** | paper,model,use,graphic model |

transfer learning needs to have a strong correlation between the source domain and the target domain.

To overcome the impaction of incomplete annotated problem in keyphrase extraction, we introduced negative sampling to allow unlabeled keyphrases to have opportunities to be considered as keyphrase to participate in the training. Most previous model treated unlabeled keyphrases as negative samples, which will mislead the biased results of model training. In this study, we treat keyphrase extraction task as classification problem. First, we apply pre-trained model RoBERTa(Liu et al., 2019) to get word embeddings, based on word representations, use sliding CNN to extract candidate gram features, and then utilize a classifier layer to divide the candidate grams into **keyphrase** and **non-keyphrase** categories. At the same time, in order to explore the impact of incomplete annotated problem in different scenarios, we constructed different synthetic datasets by randomly masking some keyphrases, and conducted experiments on scientific domain datasets, synthetic datasets and open domain dataset. Extensive experiments show that our model well handles unlabeled keyphrases and surpasses prior baselines, and our model can obviously reduce the misguidance brought by unlabeled keyphrases in training when the incomplete problem is serious.

## 2 Related Work

In this section, we briefly review two classes of closely related works: keyphrase extraction approaches and learning with noisy data.

### 2.1 Keyphrase Extraction Approaches

In most traditional existing literature, keyphrase extraction has been formulated as a two-step process(Yuan et al., 2020). First, lexical features such as part-of-speech tags are used to determine a list of phrase candidates. Second, a ranking algorithm, such as TextRank(Mihalcea and Tarau, 2004), Multi-Layer perceptron and Support Vector Machine(Lopez and Romary, 2010), is adopted to rank the candidate list and the top ranked candidates are selected as keyphrases.

Because of the similarity with Named Entity Recognition(NER) task, keyphrase extraction is treated as a sequence labeling problem by using IOB tagging scheme(Alzaidy et al., 2019). Each word in the sentence is labeled as B-tag if it is the beginning of a keyphrase, I-tag if it's inside but not the first one within the keyphrase, or O-tag otherwise. Similarly, span-based models which are popular in NER task, also are utilized in keyphrase extraction task(Mu et al., 2020; Sun et al., 2020). They treat the spans, instead of single words, as the basic units for labeling.

Sometimes keyphrases are absent from the source text, the simple extraction methods mentioned above can only extract present keyphrase. Meng et al.(Meng et al., 2017) first proposed the CopyRNN, a neural generative model that both generates words from vocabulary and points to words from the source text with an encoder-decoder framework.

The supervised methods mentioned above have an assumption that the labeled data is completely credible, while the noise in annotated data is ignored. Zhu et al.(ZHU and WU, 2004) suggested that noise in labels tends to be more harmful than noise in features. Learning with noisy data will be introduced in next subsection.

### 2.2 Learning with Noisy Data

A number of approaches have been proposed to train DNNs with noisy data. One line of approaches formulate explicit or implicit noise layers to characterize the distribution of noisy and true labels(Hedderich and Klakow, 2018). Another line of approaches use correction methods to reduce the influence of noisy data. Sterckx et al.(Sterckx et al., 2016) reduce the influence of noisy data by reweighting the importance of unlabeled candidate phrases in a two-stage Positive Unlabeled Learning setting. Recently, a few other methods have also

[2]This example case is the 278404th sample in Kp20k-train dataset, which is available at OpenNMT-kpg-release

been proposed that use noise tolerant loss functions to achieve robust learning, Li et al.(Li et al., 2021) use negative sampling to adjust train loss, avoid training NER models with unlabeled entities. In our work, we treat KPE as classification problem, and introduce negative sampling to alleviate incomplete annotated problem.

## 3 Problem Formulation

### 3.1 Keyphrase Extraction

According to the setting of Meng et al.(Meng et al., 2017), we denote phrases that do not match any contiguous subsequence of a document as **absent** keyphrases, and the ones that fully match a part of the document as **present** keyphrases. In this work, we only focus on present keyphrase extraction and treat it as a classification problem.

Given a document $\mathrm{d} = [w_1, w_2, \cdots, w_n]$, and the **present** annotated keyphrase set $\mathrm{y} = \{y_1, y_2, \cdots, y_m\}$, where $n$ is the length of document and $m$ is the amount of **present** keyphrases. For each keyphease $y_i$ in the keyphrase set, $y_i = \{w_{pos_i}, \cdots, w_{pos_i+len_i-1}\}$, which $pos_i$ is the start position of $i_{th}$ keyphrase, and $len_i$ is the number of words in $i_{th}$ keyphrase. As shown in Fig 1, since the length of most keyphrases are less than or equal to 5, we extract all N-grams$(1 <= N <= 5)$ as candidates firstly, which 1-gram $= \{\{w_1\}, \{w_2\}, \cdots, \{w_n\}\}$, 2-gram $= \{\{w_1, w_2\}, \{w_2, w_3\}, \cdots, \{w_{n-1}, w_n\}\}$, grams of other lengths are similar, let $c_i^N$ present the $i_{th}$ N-gram, and then apply a classifier to classify grams into keyphrases or non-keyphrases.

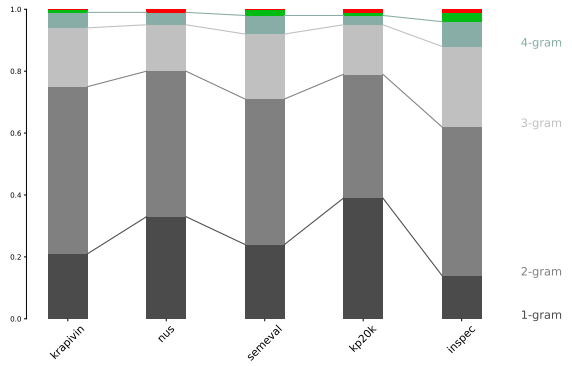



Figure 1: The distribution of the number of words in keyphrase. Different colors represent the proportion of different $N$-grams in the dataset. Green indicates the proportion of 5-gram, and red represents keyphrases with more than 5 words.

### 3.2 Incomplete Annotated problem

As mentioned above, keyphrase annotation is highly subjective(Sterckx et al., 2016) and requires the professional knowledge of the annotator, it is difficult to annotate keyphrases. Due to the complexity of keyphrase annotation or the heedlessness of human annotators, some ground truth keyphrase $\hat{y}$ of document $d$ are not covered by annotated keyphrase set y, $\hat{y} \notin \mathrm{y}$.

## 4 Methodology

Inspired by the effectiveness of negative sampling in the field of NER(Li et al., 2021), we introduce negative sampling to keyphrase extraction field to solve incomplete annotated problem. CNN-based keyphrase extraction model will be presented firstly, and then we will introduce how to use negative sampling to improve the training process to make the model more robust.

### 4.1 CNN-based keyphrase extraction model

As shown in fig 2, our model mainly includes two components:(1) A feature extractor for candidate grams via sliding CNN. (2) A binary classifier which can verify whether a gram is a keyphrase.

#### 4.1.1 Gram feature Extractor via sliding CNN

Inspired by textCNN proposed by Kim(Kim, 2014), we apply five filters with different kernel size to extract gram features. Let $h_i \in \mathbb{R}^k$ be the k-dimensional word embedding corresponding to the $i_{th}$ token in the document. In general, let $h_{i:i+j}$ refer to the concatenation of word vectors $h_i, h_{i+1}, \cdots, h_{i+j}$. A convolution operation involves a filter $w_i \in \mathbb{R}^{Nk}$, which is applied to a window of $N$ words to produce a new gram feature. The representation of the $i_{th}$ N-gram $c_i^N$ is calculated as:

$$g_i^N = CNN(h_{i:i+N-1}) \quad (1)$$

Here $h_{i:i+N-1}$ is the concatenation of $i_{th}$ N-gram $c_i^N$ word embeddings. Five filters are applied to each possible window of words to produce $n+(n-1)+(n-2)+(n-3)+(n-4)$ candidate grams.

#### 4.1.2 Binary Classifier

After obtaining the candidate gram representations, we employ a non-linear classify layer to predict

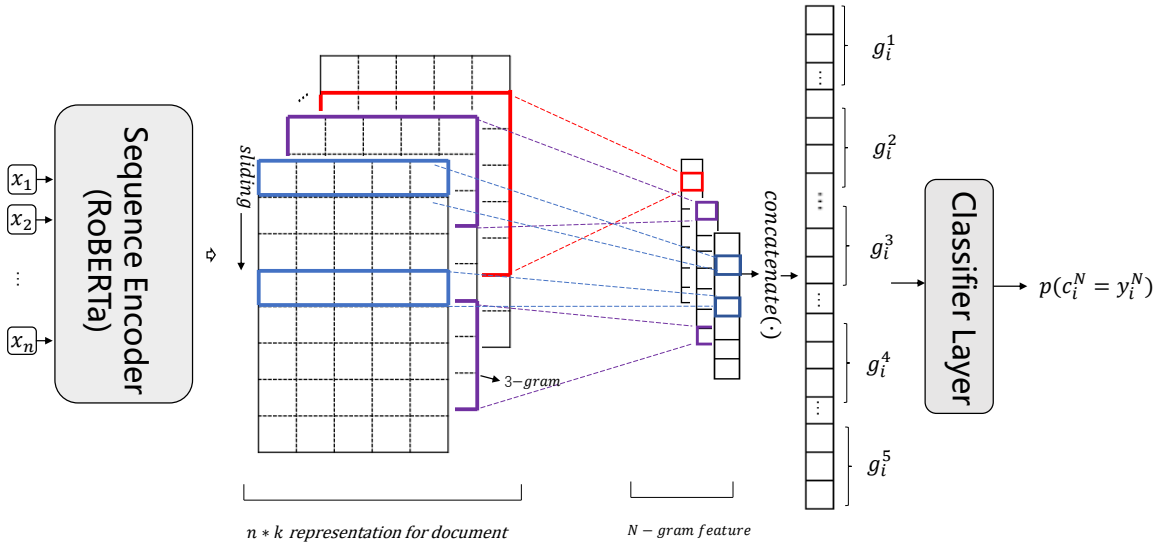



Figure 2: Illustration of our CNN-based model for keyphrase extraction.

whether a gram is a keyphrase based on it's representation.

$$p(c_i^N = y_i^N) = softmax(tanh(W_h * g_i^N + b)) \tag{2}$$

where $g_i^N$ is the representation of $i_{th}$ N-gram, $y_i^N$ is the label of whether the gram $c_i^N(w_{i:i+n-1})$ is a keyphrase or not, $W_h$ and $b$ are parameters to be learned.

### 4.2 Training via Negative Sampling

In this work, keyphrase extraction is viewed as a classification problem. Existing models optimize this classification problem by direct cross-entropy function while ignore incomplete annotated problem. Meanwhile, according to statistics(table 2), we can see that the positive and negative samples are extremely unbalanced, one document only has 3 keyphrases on average, but there are close 500 unlabeled grams as non-keyphrases. We introduce negative sampling to adjust training loss. On the one hand, negative sampling lets unlabeled keyphrases have opportunities to be considered as keyphrase to participate in the training. On the other hand, it makes positive and negative samples more balanced.

Specifically, we randomly sample a small subset of unlabeled grams as the negative samples. Let $\mathbb{C}$ be the candidate grams set, the negative candidate grams set $\tilde{\mathbb{C}}$ can be denoted as:

$$\tilde{\mathbb{C}} = \{c_i^N | c_i^N \notin \mathbf{y}, 1 \leq N \leq 5, 0 \leq i \leq n - N\}, \tag{3}$$

based on $\tilde{\mathbb{C}}$, we sample a subset $\hat{\mathbb{C}}$ from the whole negative candidate grams set. The number of grams in set $\hat{\mathbb{C}}$ is $\lceil \alpha * |\tilde{\mathbb{C}}| \rceil$, the sampling rate $\{\alpha | 0 < \alpha < 1\}$ is optional in practice, where $\lceil \rceil$ is the ceiling function, we guarantee that at least one negative candidate gram is sampled. Then we compute the cross-entropy loss with positive labeled keyphrases and the sampled negative candidate grams as flow:

$$\begin{aligned} \mathcal{L} = (\sum_{c_i^N \in \mathbf{y}} -log(P(c_i^N = 1))) + \\ (\sum_{c_i^N \in \hat{\mathbb{C}}} -log(P(c_i^N = 0))) \end{aligned} \tag{4}$$

Negative sampling can reduce the risk of training a KPE model with incomplete annotated problem by incorporating some randomness into the training loss.

## 5 Experiments

### 5.1 Datasets

We choose scientific domain datasets as well-annotated datasets, use the dataset Kp20k(Meng et al., 2017) for training [3], which contains a large amount of high-quality scientific metadata in the computer science domain from various online digital libraries(Meng et al., 2016). Each of which contains a title and an abstract of a scientific publication as source text, and author-assigned keywords as target keyphrases, we follow the original work's partition of training, development and testing sets. We further test the model trained with

[3] https://github.com/memray/OpenNMT-kpg-release

Table 2: Statistics of the dataset. #doc, #keyphrase and #length denote the number of papers, the average number of keyphrases and the number of word in keyphrase.

| Dataset | | #doc | #keyphrase | #length |
|---|---|---|---|---|
| Test | Inspec | 500 | 7.75 | 2.41 |
| | Krapivin | 460 | 3.98 | 2.11 |
| | Nus | 211 | 5.61 | 1.92 |
| | Semeval | 100 | 6.76 | 2.15 |
| | Kp20k | 19032 | 3.48 | 1.86 |
| Kp20k Validation | | 19067 | 3.47 | 1.87 |
| Kp20k Training | | 488549 | 3.48 | 1.86 |
| OpenKP | | 148124 | 1.79 | 2.02 |

Kp20k on four widely-adopted keyphrase extraction data sets including Inspec(Hulth and Megyesi), NUS(Nguyen and Kan, 2007), SemEval-2010(Kim et al., 2010) and Krapivin(Krapivin et al., 2009). In this paper, we focus on keyphrase extraction, therefore, only the document of at least one **present** keyphrase are used for training and evaluation. The statistics on the number of documents, the average number of keyphrases and the average number of word in keyphrase for each benchmark are summarized in Table 2.

Furthermore, in order to explore the performance of our model in real world scenario, we also choose OpenKP(Xiong et al., 2019) as an open domain dataset, OpenKP includes 134K open domain web pages of various topics from search engine Bing. And we randomly remove some keyphrases of training dataset Kp20k with different masking rate $r_{mask}$ to construct synthetic datasets, to simulate poorly-annotated datasets with different degrees of incomplete annotated problem.

### 5.2 Baselines and Evaluation Metrics

Because keyphrase generation(KPG) can also generate present keyphrase, we compare our model with both extraction and generation approaches. For extraction approaches, we choose two well-known unsupervised algorithms for keyphrase extraction including Tf-Idf(Jones, 1972), TextRank(Mihalcea and Tarau, 2004), and also choose SKE-Large-Cls(Mu et al., 2020), RoBERTa-Chunk(Sun et al., 2020) which published recently. For generation approaches, we compare our models with four supervised algorithms: CopyRNN(Meng et al., 2017), TG-net(Chen et al., 2019), CatSeq(Yuan et al., 2020), and SGG(Zhao et al., 2021).

Note that, the comparison methods mentioned above are mostly methods for scientific domain. For open domain dataset OpenKP(Xiong et al., 2019), we choose CopyRNN(Meng et al., 2017), BLING-KPE(Xiong et al., 2019) and RoBERTa-Chunk(Sun et al., 2020) as comparison models.

Following CopyRNN(Meng et al., 2017) and RoBERTa-Chunk(Sun et al., 2020), we adopt top-K macro-averaged precision, recall and F1 measures as our evaluation metrics for the present keyphrases respectively, $K = 5, 10$ when evaluating scientific domain datasets, and $K = 1, 3, 5$ when evaluating open domain datasets. Here, precision is defined as the number of correctly predicted keyphrases over the number of all predicted keyphrases, we apply Porter Stemmer(Porter, 1980) to both target keyphrases and predicted keyphrases when determining whether the predictions are correct; recall is computed as the number of correctly predicted keyphrases over the total number of data records, and F1 is the harmonic average of precision and recall.

### 5.3 Implementation Details

We set maximal length of source sequence as 510 and maximum N-gram length as five (N = 5). The dimension of word embedding is 768, which obtained by fine-tuning RoBERTa(Liu et al., 2019). Five sliding CNNs with kernel size between 1 and 5 are used to extract the representation of grams. The dimension of CNN outputs and hidden state in classifier are 512. The sampling rate $\alpha$ is set to $\alpha = 0.1$ when training Kp20k, and $\alpha = 0.05$ when training OpenKP. Our model are optimized using Adam with 5e-5 learning rate, 10% warm-up proportion, 24 batch size. We implement our model using PyTorch on a Linux machine with a GPU device Tesla V100 32GB. Code and models are available at https://github.com/fredia/NS-KPE.

### 5.4 Results And Analysis

To reduce the performance randomness, the performance of our model is the average of 3 random runs, we set sampling rate $\alpha = 0.1$ when training on Kp20k, and set sampling rate $\alpha = 0.05$ when training on OpenKP. The best results are highlighted in **bold**.

[4]Datasets have been slightly updated after (Yuan et al., 2020). The result of CopyRNN is taken from (Yuan et al., 2020)

Table 3: Performance comparison of different models on well-annotated scientific domain benchmark datasets. Compared baselines include generation approaches[4](the first part of table), and models for extraction approaches. The result of our model are average from 3 runs of experiments with sampling rate $\alpha = 0.1$.

| | Kp20k | | Inspec | | NUS | | Krapivin | | Semeval | |
|---|---|---|---|---|---|---|---|---|---|---|
| | F1@5 | F1@10 | F1@5 | F1@10 | F1@5 | F1@10 | F1@5 | F1@10 | F1@5 | F1@10 |
| CopyRNN | 0.328 | 0.255 | 0.292 | 0.336 | 0.342 | 0.317 | 0.302 | 0.252 | 0.291 | 0.296 |
| TG-net | 0.372 | 0.315 | 0.315 | 0.381 | 0.406 | 0.370 | 0.349 | 0.295 | 0.318 | 0.322 |
| CatSeqD | 0.348 | 0.298 | 0.276 | 0.333 | 0.374 | 0.366 | 0.325 | 0.285 | 0.327 | 0.352 |
| SGG | - | - | 0.306 | 0.359 | 0.363 | 0.358 | 0.288 | 0.253 | 0.338 | 0.336 |
| Tf-Idf | 0.105 | 0.130 | 0.223 | 0.304 | 0.139 | 0.181 | 0.113 | 0.143 | 0.120 | 0.184 |
| TextRank | 0.180 | 0.150 | 0.229 | 0.275 | 0.195 | 0.190 | 0.172 | 0.147 | 0.171 | 0.181 |
| SKE-Large-Cls | 0.392 | 0.33 | 0.294 | 0.334 | 0.403 | 0.364 | 0.309 | 0.252 | 0.361 | 0.358 |
| RoBERTa-Chunk | **0.406** | 0.336 | **0.357** | **0.401** | 0.465 | 0.431 | 0.372 | 0.314 | **0.380** | 0.385 |
| **Our Model** | 0.406 | **0.338** | 0.348 | 0.388 | **0.472** | **0.432** | **0.375** | **0.320** | 0.376 | **0.388** |
| std | 0.0006 | 0.0006 | 0.0026 | 0.0015 | 0.001 | 0.0024 | 0.0022 | 0.0025 | 0.0021 | 0.0044 |

#### 5.4.1 Results on Scientific Domain Datasets

Because most training data of scientific domain datasets are annotated by authors, and the author is the person who is most familiar with their articles, the issue of incomplete keyphrase annotation is not serious, we treat scientific domain datasets as well-annotated datasets. Table 3 show the F1@5 and F1@10 performance of our model and the other baseline methods across multiple scientific domain datasets, the results of Tf-Idf, TextRank and CopyRNN are taken from (Meng et al., 2017), and other reported results are from their corresponding original paper. We can see that for most cases (expect Tf-Idf and TextRank), the extraction approaches outperform all the generation approaches, since extraction is often easier than generation(Wiseman et al., 2017). As shown in Table 3, the F1@5 and F1@10 scores of our model are very close to current best extraction results, our model even upperforms RoBERTa-Chunk by even 1% in NUS, these results indicate that our model is still effective when applied to high-quality data.

Table 4: Performance comparison of different models on open domain dataset(OpenKP).

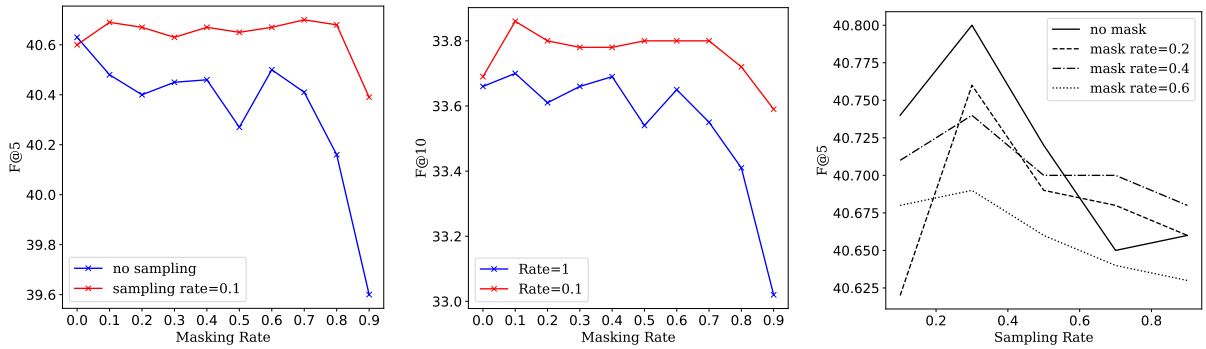
| Method | OpenKP | | |
|---|---|---|---|
| | F1@1 | F1@3 | F1@5 |
| CopyRNN | 0.217 | 0.237 | 0.210 |
| BLING-KPE | 0.267 | 0.292 | 0.209 |
| RoBERTa-Chunk | 0.355 | 0.373 | 0.324 |
| **Our Model** | **0.380** | **0.383** | **0.327** |

#### 5.4.2 Results on Open Domain Dataset(OpenKP)

Different from scientific papers, open domain datasets regularly contain diverse documents originating from sparse sources, and annotators are not as familiar with the original text as authors, the incomplete annotated problem is serious in these datasets. Table 4 show the F1@1, F1@3 and F1@5 score on an open domain dataset(OpenKP(Xiong et al., 2019)), the results of CopyRNN, BLING-KPE and RoBERTa-Chunk are token from (Sun et al., 2020). We can observe that our model has outperformed prior baselines. Compared with RoBERTa-Chunk, we achieve the improvements of 2.5% on F1@1 and 1% on F1@3, and our results is very close to RoBERTa-Chunk on F1@5, this is not surprising since the average number of keyphrases in OpenKP is 1.79. Results on open domain datasets confirm the effectiveness of our model.

#### 5.4.3 Results on Synthetic Datasets

Results on synthetic datasets are shown in Fig 3 (a)(b), we compare the performance of our model and RoBERTa-Chunk under different masking rates. As shown in Fig 3(a)(b),the F1@5 and F1@10 score with sampling rate $\alpha = 0.1$ are always greater than no sampling, and sampling rate $\alpha = 0.1$ drops more slowly than no sampling when masking rate is increasing. The comparison result show that our model can reduce the misguidance brought by unlabeled keyphrases in training, especially in high masking rates.




(a) F1@5 score. (b) F1@10 score. (c) Rate in negative sampling.

Figure 3: The performance of our model with different sampling rate on synthetic datasets. In sub-figure (a) and (b), $x$ axis represents the value of masking rate $r_{mask}$, the larger $r_{mask}$, the more serious incomplete annotated problem is. And $y$ axis represents the F1@5 score and F1@10 score. The red line denote the results of sampling rate $\alpha = 0.1$ and the blue line denote no sampling. In sub-figure (c), $x$ axis represents the value of sampling rate $\alpha$ and $y$ axis represents the F1@5 score on synthetic datasets, different styles of line denote different synthetic datasets with different masking rate.

#### 5.4.4 Sampling Rate $\alpha$ in Negative Sampling

Fig 3 (c) shows the experiments on synthetic datasets with the different sampling rate $\alpha$. The sampling rate $\alpha$ should not as small as possible, too small sampling rate will reduce the number of negative samples, leading to underfitting.

## 6 Conclusion and Future Work

In this work, we introduce negative sampling to keyphrase extraction model, to alleviate the misleading of the incomplete annotated problem. According to the length characteristics of the keyphrase, we use sliding CNN to extract the representation of the candidate gram, turn the keyphrase extraction problem into binary classification. In different scenarios and different scales datasets, we have confirmed the effectiveness of our model.

Our work can be extended in many directions. To begin with, currently binary classifier in our model treats each candidate gram individually, but there is usually a relationship between different grams. We could leverage attention mechanism to model the relationship between grams. Moreover, we would like to further explore utilizing smart gating units or designing a network layer to filter unlabeled keyphrases.

## Acknowledgements

This work is supported partly by the National Natural Science Foundation of China (No. 61772059), by the Fundamental Research Funds for the Central Universities, by the Beijing S&T Committee(No. Z191100008619007) and by the State Key Laboratory of Software Development Environment(No. SKLSDE-2020ZX-14).

## 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.

Isabelle Augenstein and Anders Søgaard. 2017. Multi-task learning of keyphrase boundary classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 341–346.

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

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

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the*

*Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273. ACL.

Michael A Hedderich and Dietrich Klakow. 2018. Training a neural network in a low-resource setting on automatically annotated noisy data. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 12–18.

Anette Hulth and Beáta B. Megyesi. 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. ACL.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*.

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. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

M. Krapivin, A. Autaeu, and M. Marchese. 2009. Large dataset for keyphrases extraction.

Yangming Li, lemao liu, and Shuming Shi. 2021. Empirical analysis of unlabeled entity problem in named entity recognition. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zhiyuan Liu, Chen Liang, and Maosong Sun. 2012. Topical word trigger model for keyphrase extraction. In *Proceedings of COLING 2012*, pages 1715–1730.

Patrice Lopez and Laurent Romary. 2010. Humb: Automatic key term extraction from scientific articles in grobid. In *SemEval 2010 Workshop*, pages 4–p.

Rui Meng, Shuguang Han, Yun Huang, Daqing He, and Peter Brusilovsky. 2016. Knowledge-based content linking for online textbooks. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 18–25.

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.

Funan Mu, Zhenting Yu, LiFeng Wang, Yequan Wang, Qingyu Yin, Yibo Sun, Liqun Liu, Teng Ma, Jing Tang, and Xing Zhou. 2020. Keyphrase extraction with span-based feature representations. *arXiv preprint arXiv:2002.05407*.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. ICADL'07, page 317–326, Berlin, Heidelberg. Springer-Verlag.

Martin F Porter. 1980. An algorithm for suffix stripping. *Program*.

Nirmala Pudota, Antonina Dattolo, Andrea Baruzzo, Felice Ferrara, and Carlo Tasso. 2010. Automatic keyphrase extraction and ontology mining for content-based tag recommendation. *International Journal of Intelligent Systems*, 25(12):1158–1186.

Yafeng Ren, Donghong Ji, and Hongbin Zhang. 2014. Positive unlabeled learning for deceptive reviews detection. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 488–498.

Lucas Sterckx, Cornelia Caragea, Thomas Demeester, and Chris Develder. 2016. Supervised keyphrase extraction as positive unlabeled learning. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1924–1929. ACL.

Si Sun, Chenyan Xiong, Zhenghao Liu, Zhiyuan Liu, and Jie Bao. 2020. Joint keyphrase chunking and salience ranking with bert. *arXiv preprint arXiv:2004.13639*.

Yanan Wang, Qi Liu, Chuan Qin, Tong Xu, Yijun Wang, Enhong Chen, and Hui Xiong. 2018. Exploiting topic-based adversarial neural network for cross-domain keyphrase extraction. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 597–606. IEEE.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263. Association for Computational Linguistics.

Lee Xiong, Chuan Hu, Chenyan Xiong, Daniel Campos, and Arnold Overwijk. 2019. Open domain web keyphrase extraction beyond language modeling. In *EMNLP-IJCNLP*.

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*.

Jing Zhao, Junwei Bao, Yifan Wang, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2021. SGG: Learning to select, guide, and generate for keyphrase generation. In *NAACL*.

XINGQUAN ZHU and XINDONG WU. 2004. Class noise vs. attribute noise: A quantitative study of their impacts. *Artificial Intelligence Review*, 22:177–210.