Pre-training and multi-task training for federated keyword extraction

Lingqi Guo1,2,3, Haifeng Sun1, Zhizhong Shi2 and Tao Lei2
1State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
2Autohome Inc., Beijing, China
3Corresponding author’s e-mail: guolingqi@ebupt.com

Abstract. The generalization ability of supervised model is relatively weak in keyword extraction technology. In order to effectively improve the robustness and accuracy of the model, the key to overcome this problem is collect more data for training process. However, the text as an privacy information, it is harder and harder to collect. To solve this problem, we apply federal learning to use user privacy data locally to improve the keyword extraction performance. In order to integrate the unlabeled user utterances to the semantic parsing model, we proposed a Pre-training and multi-task based federated keyword extraction model. The user models learn local information via unsupervised setting and the core model learns the supervised setting.

1. Introduction
Keyword Extraction refers to extract most representative words from text, which is a fundamental work in natural language process, such as text retrieval, abstract generation, and text classifications [1,2]. In order to extract keywords more accurately and effectively, the researchers put forward two effective ways. One is the unsupervised method, which extract keywords by their statistical features, such as the frequency of words, the position of the first occurrence of words and so on. The representatives of such methods are TF-IDF [3], YAKE [4]. And some researchers, inspired by PageRank [5] algorithm, build a word graph model to extract keywords, such as TextRank [6]. The other one is a supervised method. After manually labeling keywords, a machine learning based model is trained using the word frequency and location information of the words. The representative of this kind of algorithm is Kea proposed by Ian et al [7].

However, statistical features still have their limitations. With the rapid development and wide application of deep learning technology in recent years, researchers pay more attention to extract keywords by deep learning methods and have achieved good results. Marco et al. [8] proposed to use GloVe word vector to encode text, and then train BiLSTM (Bi-Directional Long Short-Term Memory) model to extract keywords. Previous keyword extraction mainly used models to extract words that already in the original text, but could not cope with the situation that keywords did not in the original text. In order to solve this problem, Rui Meng et al [9] proposed to use Copy mechanism and Seq2Seq model to generate keywords, and this work also achieved good results.

It is found that the accuracy and recall rate of the supervised models are better than the unsupervised ones. The main reason for this situation is that the unsupervised learning method has learned the statistical features of texts which is generally of high migration, but it is difficult to learn
the complex semantic relations between words. However, the supervised learning model can learn the relations between texts, also these features are difficult to migrate in cross-domain situations. In order to combine the high accuracy and recall from supervised methods with the high generalization from unsupervised methods. The intuitive way is to use more language datasets to train the supervised methods. If the keywords extraction algorithm could be continue learned, the performance could be improved. Considering that text data often contains rich user private data, it is difficult to collect large data directly. Thanks to the recently proposed federated learning mechanism [10], which could distribute training models with user private data locally, the keywords extraction could be continue learned.

The federated learning mechanism will send the copy of model for each user. Then user could continue training the model with the data from themselves. At last, parameters of all the model would synchronize to the core center. From the application of keywords extraction perspective, the data from users are unlabeled, which make the supervised model training processing is difficult to process. For this purpose, a two state keywords extraction model should be designed. For the center model, the full model could be trained in supervised setting, and the user copy models are trained in unsupervised setting.

In recent years, more and more pre-training language models have been proposed, and their good performance in natural language processing proves that the Two-stage training method combining pre-training and fine-tuning training is effective. The two-stage training method applies open domain corpus in the pre-training stage, which makes the model more extensive. In the fine-tuning stage, the datasets of specific fields or specific tasks are applied, which makes the model more suitable for downstream tasks. Marco et al. [8] also used two-step training to extract keywords, which proved the effectiveness of this method. In multi-task training, we need use other tasks to auxiliary train the keywords extraction model. The goal of auxiliary task should be similar to that of keyword extraction task, but the difficulty of auxiliary task is lower, so the auxiliary task can help keyword extraction task to remove noise and make results correct in actual training process.

To this all, how to design such model to meet the federated learning mechanism is the key problem. In this paper, we uses the Encoder of Transformer [11], which performs well in the field of natural language processing, as the main component of the model. In order to improve the generalization ability of the model, a large number of papers from users are used to pre-train the model. Then, in order to better adapt to the field of downstream tasks, the pre-training model is fine-tuned by using datasets in specific fields. In order to further improve the accuracy of the model and keep the consistency with the pre-training tasks, the multi-task training method is adopted in the fine-tuning stage. After the test set is predicted, some rules are added to sort out the predicted results.

To summarize, our main contributions include:

1. We proposed a federated keywords extraction learning based method to further improve the performance. In order to meet the federated learning mechanism and the reality that the user lacks labeled data, we separate the keywords extraction into two part. The supervised setting could be used for the center model, and the unsupervised setting could be used for user copy models.

2. We proposed a novel multi-task based keywords extraction algorithm to best leverage the most data for improving the accuracy and generality of the model. After a large number of ablation experiments, it is proved that the model proposed in this paper has greatly and effectively improved the robustness and accuracy of keyword extraction, compared with the mainstream model.

2. Related work

Keywords refer to words that are more scientific and efficient to reflect the content of the article, and keyword extraction technology is a basic but important technology in the field of natural language processing. In 1972, Jones and others proposed TF-IDF algorithm, which was used by researchers to
extract keywords and achieved good results. Nowadays, the technology of keyword extraction has made remarkable progress.

There are two main ways to solve the problem of keyword extraction. One is the unsupervised learning method, which mainly uses the statistical features of words to get keywords through weight calculation. The typical algorithm in this aspect is YAKE algorithm proposed by Campos et al., which makes use of the statistical characteristics of word frequency, the position of the first occurrence of the word, whether the candidate words are capitalized, and then uses the weight to find the ranking of the candidate list to get keywords. In recent years, some researchers have used graphs to extract keywords. The typical work is TextRank, PositionRank [12] and other algorithms. Influenced by the pre-training model, SUN et al. [13] proposed an unsupervised method to extract keywords by using word vectors and sentence vectors.

Another main way to extract keywords is supervised learning method. This method often requires a lot of manual labels, but its accuracy is usually higher than that of unsupervised methods. KEA algorithm proposed by Ian et al. trains machine learning models such as Naive Bayesian Classifier to extract keywords through statistical features. With the development of deep learning, people pay more attention to how to extract keywords accurately by using deep learning model. In 2018, Marco et al. [8] used BiLSTM to extract keywords. This method uses word vectors as features to train neural networks, which has good robustness. Zhang et al. [14] proposed a new LSTM structure, which uses attention mechanism to extract keywords through Two-stage training. R. Meng et al. took keyword extraction task as generation task, and generated keywords according to text content by using Seq2Seq model and Copy mechanism. The previous unsupervised methods performed poorly in accuracy, while the supervised methods performed well in specific fields but poor in migration. The method proposed in this paper uses a large amount of open domain data for pre-training, which makes the model both accurate and migratory.

3. Methods

The main content of this chapter is the details and training methods of the model proposed in this paper. The model structure is mainly described in section 3.1, and the training method of the model is described in section 3.2.

3.1. Model architecture

The main structure of the keyword extraction model proposed in this paper is the encoder structure in Transformer. [15] The method of pre-training and fine-tuning is used to solve the problem of poor cross-domain performance of supervised keyword extraction method. The overall structure of the model is shown in Figure 1. The model mainly includes text coding layer, encoder, sentence classification layer and keyword extraction layer.

![Figure 1. Model structure.](image)

Assuming that the input text sequence is \( \{w_1, w_2, ..., w_n\} \), the text sequence is coded as vector sequence \( \{v_1, v_2, ..., v_n\} \) through the text coding layer.

According to Vaswani's work [10], in the multi-head attention layer of encoder, three different weight matrices are initialized for each \( head_i \), which are Query vector (\( Q_i \)), Key vector (\( K_i \)) and Value vector (\( V_i \)), and \( d_{K_i} \) represents the dimension of Key vector. Then the calculation formula of self-attention is:
The formula for calculating multi-head attention is as follows, where $W_0$ is the weight matrix and $MultiHead$ is the multi-head attention.

$$MultiHead = Concat(head_1, ..., head_i)W_0$$

The sentence classification layer is softmax layer, which is used to classify the encoder results and calculate the distribution probability. If the input text contains keywords, the label is 1, otherwise it is 0. The purpose of setting this layer is to better judge the prediction result quality of keyword extraction task. The expression of the i-th sample result $y^i_{CLS}$ output by the sentence classification layer is:

$$y^i_{CLS} = \text{Softmax}(W_C H_C + b_C)$$

where $y^i_{CLS}$ represents the correct label of the i-th sample, and $y^i_{CLS}$ represents the posterior probability of the i-th sample predicted by the sentence classification layer.

The keyword extraction layer is also softmax layer, which is used to find the probability distribution of keywords in the sequence. The value $y^i_{KE}$ of the i-th word tag predicted by the keyword extraction layer is expressed as:

$$y^i_{KE} = \text{Softmax}(W_K H_K + b_K)$$

where $y^i_{KE}$ represents the correct label of the i-th word in the text, and $y^i_{KE}$ represents the posterior probability of the i-th word predicted by the keyword extraction layer.

The input and output diagram of the model is shown in Figure 2. The input text is “a method of construction of a nonlinear extrapolation algorithm is proposed.”, which is converted into an index sequence, and then encoded into a vector sequence. The annotated keyword of the input text is “Nonlinear Extrapolation Algorithm”, so the output label at the sentence classification layer is 1. In the output sequence of the keyword extraction layer, the output label at the beginning of the keyword is 1, the output label at the following keyword parts is 2, and the output labels at other positions are 0.

![Figure 2. Model architecture.](image-url)
3.2. Pre-training for user models

In order to enhance the generalization ability of the model and make full use of the data, this paper uses two stages of pre-training and fine-tuning in the training stage. In the pre-training stage, a large number of user local text information are used as unlabeled data. After the text is divided into n sentences, the i-th sample is \((s^i, l^i)\), where \(s^i\) represents the ith sentence, and \(l^i\) represents whether there are keywords in \(s^i\). The value of \(l^i\) is as follows:

\[
l^i = \begin{cases} 
1, & \text{If } s^i \text{ contains keywords} \\
0, & \text{If } s^i \text{ does not contain keywords}
\end{cases}
\]

(7)

The output of the model in the pre-training stage is only the output of the sentence classification layer, so the loss function \(L_{\text{pre-train}}\) in the pre-training stage is:

\[
L_{\text{pre-train}} = L_{\text{CLS}}
\]

(8)

In which \(L_{\text{CLS}}\) stands for the loss calculation function of sentence classification task.

3.3. Fine-tuning for core model

After the pre-training stage, the parameters of the user models are upload to the core model, the core model as the way introduced in [11]. Then the fine-tuning training with labeled data will process for the core model. In order to improve the precision of fine-tuning training, the method proposed in this paper adopts multi-task method to carry out fine-tuning training. Then the i-th data in the training set can be expressed as \((s^i, l^i_k, l^i_c)\), where \(s^i\) represents the text of the i-th data, \(l^i_c\) represents whether the sentence contains keywords or not, and \(l^i_k\) is a label sequence with the same length as \(s^i\), which words in \(s^i\) are keywords. Let the label \(k^p\) of \(l^i_k\) represents whether the p-th word in \(s^i\) is a keyword, then the value of \(k^p\) is as follows:

\[
k^p = \begin{cases} 
0, & \text{If the p-th word is not a keyword} \\
1, & \text{If the p-th word is the starting position of the keyword} \\
2, & \text{If the p-th word is a keyword}
\end{cases}
\]

(9)

The two tasks in the fine-tuning stage are sentence classification and keyword extraction, so the loss function \(L_{\text{fine-tuning}}\) in the fine-tuning stage is:

\[
L_{\text{fine-tuning}} = WL_{\text{CLS}} + L_{\text{KE}}
\]

(10)

where \(W\) represents the weight, \(L_{\text{CLS}}\) represents the loss function of sentence classification task, and \(L_{\text{KE}}\) represents the loss function of keyword extraction task.

4. Experiments

This chapter mainly describes the details of the experiment, and proves the effectiveness of the method proposed in this paper through comparative experiments. Section 4.1 mainly talks about the source of experimental data, experimental operation steps and model evaluation methods; Section 4.2 talks about the comparison experiment of model details settings; Section 4.3 talks about the comparison between the model proposed in this paper and other mainstream models.

4.1. Experimental data

The experimental data of this paper are mainly abstract parts of academic papers of various disciplines. In the pre-training stage, 100,000 pieces of training set data, 10,000 pieces of dev set data and 10,000 pieces of test set data were obtained from abstracts of academic papers in various fields, such as physics, chemistry and machinery. As shown in Table 1, in the pre-training task, the ratio of positive and negative samples in training set, dev set and test set is close to 1:1. Firstly, the rules are used to judge whether the sentence contains keywords, and then the sentence samples are labeled with or without keywords according to equation (6). There are three public datasets used in the fine-tuning
experiment: two keyword extraction datasets in the field of computer science and one keyword dataset in the open field. The details of the three datasets used are as follows:

1. **Krapivin** [16]: This dataset contains 2304 papers published by ACM and keywords marked by the author. In this paper, the first 400 papers of this dataset are selected as test sets, 50 as dev sets, and the rest as training sets for fine-tuning training.

2. **NUS** [17]: This dataset contains 211 articles, with keywords marked by volunteers and article authors. There is no official standard for dividing training sets and test sets. In this paper, 150 articles are used as training sets, 20 as dev sets, and the rest as test sets.

3. **Semeval 2010** [18]: This dataset contains 284 academic papers in various fields collected by ACM, which is the official dataset of Semeval2010 Task5 Competition. Among them, 100 articles are used as test sets, and the rest are used as training sets and dev sets.

The specific experimental operation steps of the model proposed in this paper are as follows: 

- a) Remove non-English texts and texts with a long length; 
- b) Use NLTK [19] tool to divide the article into sentences, and then split each sentence into word sequences; 
- c) Use rules to generate tags of sentence classification tasks and keyword extraction tasks, and cleaning the datasets used in pre-training for the second time, so as to keep the integrity of the text and the accuracy of the tags as much as possible; 
- d) Conduct pre-training, save model parameters after pre-training, and then carry out fine-tuning training of the pre-trained model on the text dataset in the computer field; 
- e) After the fine-tuning, predict the results of the test set; 
- f) After getting the prediction results, evaluate the performance of the model and calculate the F1@5 score of the test set.

In the stage of model evaluation, in order to be consistent with the mainstream evaluation methods, the text of the model prediction test set is used to list the top five results of the model prediction probability from high to low and the correct label to calculate the F1@5 score.

| Table 1. Experimental data analysis of pre-training. |
|-----------------------------------------------|
| Train set | 100000 | 56386 | 43614 |
| Dev set   | 10000  | 4728  | 5272  |
| Test set  | 10000  | 5645  | 4355  |

4.2. **Ablation experiment**

In order to ensure the validity of the experimental results, the deep learning framework used in this paper is Tensorflow [20]. Adam optimizer is used in both pre-training and fine-tuning training, and the initial learning rate is set to 0.0001. In the fine-tuning stage, if the loss decreases less than 0.01 after five rounds of training, the learning rate becomes 0.2 times of the original one. The maximum sequence length is set to 256, the encoder layer number is set to 6, and the 8-head attention mechanism is adopted. Therefore, the following comparative experiments are set up in this paper:

1. The Transformer model is used to extract keywords, without pre-training stage;
2. The encoder model of Transformer is used to train the keyword extraction task directly, without adding sentence classification task and pre-training;
3. The encoder model is trained separately in each dataset, and two tasks of sentence classification and keyword extraction are added in the training stage;
4. After pre-training the model, only the keyword extraction task is added in the fine-tuning training stage of the model.
5. After pre-training the model, two tasks of sentence classification and keyword extraction are added in the fine-tuning stage of the model.

As shown in Table 2, pre-training was added in Experiment 4 and Experiment 5, and the results were greatly improved compared with Experiment 1 and Experiment 3. After adding multi-tasks, it is found that the performance of the training model in Experiment 5 is obviously better than that in Experiment 4, and the improvement is close to 3%. It is concluded that the performance of the model has been significantly improved after multi-task training.
In the multi-task training stage, this paper selects the sentence classification task which is similar to the keyword extraction task, and the sentence classification task is simpler than the keyword extraction task, so it can help the model to extract keywords more accurately in the learning stage.

Through the above comparative experiments, it is proved that the model proposed in this paper has been greatly improved after adding the pre-training task. At the same time, after adding multi-task training, it has a better performance in keyword extraction task. To conclude, it proves that the model training method used in this paper is effective.

4.3. Comparative experiment with mainstream model

In order to verify the effectiveness of the proposed method, in this section, the model proposed in this paper is compared with other mainstream models in the field of keyword extraction.

In this paper, four mainstream and well-performing keyword extraction models are selected: TF-IDF and YAKE algorithms are selected in unsupervised algorithms; KEA and CopyRNN are selected in supervised algorithm. YAKE algorithm and TF-IDF algorithm belong to unsupervised keyword extraction methods based on statistical features, KEA belongs to supervised keyword extraction method based on statistical features, and CopyRNN model belongs to supervised keyword generation model.

Compared with the mainstream model, the experimental results are shown in Table 3, and the model proposed in this paper has the best results in the current dataset. Compared with TF-IDF, YAKE and KEA models, the method proposed in this paper has been greatly improved. But compared with CopyRNN, the model proposed in this paper cannot deal well with the situation that keywords are not in the text. In addition, when analyzing the experimental results, we find that CopyRNN’s generalization ability is slightly weak, and it is difficult to accurately generate keywords in the open domain. In the computer domain dataset, the model proposed in this paper makes smaller improvement, while in the open domain dataset, the model proposed in this paper makes greater improvement.

### Table 2. The results of control experiment.

| Experiment | Pre-train | Sentence classification | Keywords Extraction | Krapivin | NUS | Semeval |
|------------|-----------|-------------------------|---------------------|----------|-----|---------|
| 1          |           |                         |                     | 0.211    | 0.214 | 0.208   |
| 2          |           |                         |                     | 0.208    | 0.215 | 0.204   |
| 3          |           | √                       |                     | 0.230    | 0.240 | 0.233   |
| 4          | √         |                         |                     | 0.283    | 0.325 | 0.275   |
| 5          | √         | √                       |                     | 0.312    | 0.354 | 0.310   |

### Table 3. Results on Krapivin, NUS and Semeval.

| Model     | Type       | Krapivin | NUS    | Semeval |
|-----------|------------|----------|--------|---------|
| TF-IDF    | Unsupervised | 0.113    | 0.139  | 0.120   |
| YAKE      | Unsupervised | 0.215    | 0.159  | 0.151   |
| KEA       | Supervised  | 0.096    | 0.068  | 0.027   |
| CopyRNN   | Supervised  | 0.305    | 0.342  | 0.291   |
| The proposed model | Supervised | **0.312** | **0.354** | **0.310** |

Through the above experimental analysis of mainstream models, it can be concluded that although the model proposed in this paper is a supervised training method, it has better generalization ability than mainstream supervised models, and can accurately predict keywords in cross-domain datasets or open-domain datasets.
5. Conclusion and future work
The existing keyword extraction technology cannot consider both the generalization and accuracy of the model. In this paper, a two-step training method is proposed to extract keywords. First, a large number of open domain text data are used to pre-train the model, and then specific domain datasets are used to fine-tune the model. After experimental analysis, after adding pre-training and multi-task, the accuracy and generalization ability of the model can be significantly improved, so that the model can extract keywords more accurately and have better migration. In the future work, we will continue to study the keyword extraction algorithm, so that the model can accurately extract keywords with less data.

Acknowledgements
This work was supported by State Grid Corporation of China Project (SGTJDK00DWJS1900242) Research and application of key technologies of knowledge management and cognitive reasoning in operation inspection field for ubiquitous electric internet of things.

References
[1] Zhang K, Xu H, Tang J, et al. 2006 Keyword extraction using support vector machine[C]//International conference on web-age information management. Springer, Berlin, Heidelberg, 85-96.
[2] Campos R, Mangaravite V, Pasquali A, et al. 2020 YAKE! Keyword extraction from single documents using multiple local features[J]. Information Sciences, 509: 257-289.
[3] Jones K S. 1972 A statistical interpretation of term specificity and its application in retrieval[J]. Journal of documentation.
[4] Campos R, Mangaravite V, Pasquali A, et al. 2018 YAKE! collection-independent automatic keyword extractor[C]//European Conference on Information Retrieval. Springer, Cham, 806-810.
[5] Page L, Brin S, Motwani R, et al. 1999 The pagerank citation ranking: Bringing order to the web[R]. Stanford InfoLab
[6] Mihalcea R, Tarau P. 2004 Textrank: Bringing order into text[C]//Proceedings of the 2004 conference on empirical methods in natural language processing, 404-411.
[7] Witten I H, Paynter G W, Frank E, et al. 2005 Kea: Practical automated keyphrase extraction[M]//Design and Usability of Digital Libraries: Case Studies in the Asia Pacific. IGI global, 129-152.
[8] Basaldella M, Antolli E, Serra G, et al. 2018 Bidirectional lstm recurrent neural network for keyphrase extraction[C]//Italian Research Conference on Digital Libraries. Springer, Cham, 180-187.
[9] Meng R, Zhao S, Han S, et al. 2017 Deep keyphrase generation[C]// Proceedings of the 2017-55th Annual Meeting of the Association for Computational Linguistics, 1: 582-592.
[10] Vaswani A, Shazeer N, Parmar N, et al. 2017 Attention is all you need[C]//Advances in neural information processing systems, 5998-6008.
[11] Andrew H, Kanishka R, Rajiv M, et al. 2019 Federated learning for mobile keyboard prediction[J]. arXiv preprint arXiv, 1811.03604.
[12] Florescu C and Caragea C. 2017 Positionranks: An unsupervised approach to keyphrase extraction from scholarly documents[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 1105-1115.
[13] Sun Y, Qiu H, Zheng Y, et al. 2020 SIFRank: A New Baseline for Unsupervised Keyphrase Extraction Based on Pre-Trained Language Model[J]. IEEE Access, 8: 10896-10906.
[14] Zhang Y, Tuo M, Yin Q, et al. 2020 Keywords extraction with deep neural network model[J]. Neurocomputing, 383: 113-121.
[15] Devlin J, Chang M W, Lee K, et al. 2018 Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805.
[16] Krapivin M, Autaeu A and Marchese M. 2009 Large dataset for keyphrases extraction[R]. University of Trento.

[17] Nguyen T D and Kan M Y. 2007 Keyphrase extraction in scientific publications[C]//International conference on Asian digital libraries. Springer, Berlin, Heidelberg, 317-326.

[18] Kim S N, Medelyan O, Kan M Y, et al. 2010 Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles[C]//Proceedings of the 5th International Workshop on Semantic Evaluation, 21-26.

[19] Loper E and Bird S. 2002 NLTK: the natural language toolkit[J]. arXiv preprint cs/0205028.

[20] Abadi M, Barham P, Chen J, et al. 2016 Tensorflow: A system for large-scale machine learning[C]//12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), 265-283.