Enhancing Pre-trained Language Representation for Multi-Task Learning of Scientific Summarization

Ruipeng Jia
1. Institute of Information Engineering
Chinese Academy of Sciences
2. School of Cyber Security
University of Chinese Academy of Sciences
Beijing, China
jiaruipeng@iie.ac.cn

Yannan Cao
Institute of Information Engineering
Chinese Academy of Sciences
Beijing, China
caoyanan@iie.ac.cn

Fang Fang*
Institute of Information Engineering
Chinese Academy of Sciences
Beijing, China
fangfang0703@iie.ac.cn

Jinpeng Li
1. Institute of Information Engineering
Chinese Academy of Sciences
2. School of Cyber Security
University of Chinese Academy of Sciences
Beijing, China
lijinpeng@iie.ac.cn

Yanbing Liu
Institute of Information Engineering
Chinese Academy of Sciences
Beijing, China
liuyanbing@iie.ac.cn

Pengfei Yin
Institute of Information Engineering
Chinese Academy of Sciences
Beijing, China
yinpengfei@iie.ac.cn

***Abstract*—This paper aims to extract summarization and keywords from scientific articles simultaneously, while abstract extraction (AE) and key extraction (KE) are considered as auxiliary tasks to each other. For the data scarcity in scientific AE and KE tasks, we propose a multi-task learning framework which uses huge unlabeled data to learn scientific language representation (pre-training) and uses smaller annotated data to transfer the learned representation to AE and KE (fine-tuning). Although the pre-trained language model performs well in universal natural language tasks, its capacity still has a margin of improvement for specific tasks. Inspired by this intuition, we use another two tasks *keyword masking* and *key sentence prediction* before the fine-tuning phase to enhance the language representation for AE and KE. This language representation enhancing stage uses the same labeled data but different optimization objectives with the fine-tuning phase. In order to evaluate our model, we develop and release a high-quality annotated corpus for scientific papers with keywords and abstract. We conduct comparative experiments on this dataset, and experimental results show that our multi-task learning framework achieves the state-of-the-art performance, proving the effectiveness of the language model enhancing mechanism.**

***Index Terms*—extractive summarization, keywords, pre-train, multi-task**

I. Introduction

The exponential increase of scientific publications in the past decades motivates the development of automatic summarization and keyword extraction for scientific articles. Text summarization aims to generate the shorter coherent version of the source article which retains its salient information, while keyword extraction aims to identify several core words or multi-words of the source article. These tasks can facilitate the understanding and accessing of long scientific articles. Both previous keyword extraction and summarization models are roughly divided into two paradigms, *extractive* model [1], [2] and *generative* one [3], [4]. In this paper, we focus on extractive models, i.e., keyword extraction (KE) and abstract extraction (AE).

*Corresponding author: Fang Fang

. . . A ***biofilm*** that has a high impact in chronic bacterial infection, being known for the damage it causes in lungs of patients with cystic fibrosis, is formed by the Gram-negative bacteria ***Pseudomonas aeruginosa***. This is an opportunistic human pathogen that can cause acute infections in hospitalized people, especially those immunocompromised, such as patients with acquired immunodeficiency syndrome (AIDS), and neutropenic patients due to chemotherapy treatments. P. aeruginosa also cause deleterious infections in individuals with burns, pneumonia in patients receiving artificial ventilation, and keratitis in contact lens wearers. The aim of the present study was to evaluate possible changes in the molecular profile of biofilms from P. aeruginosa in varying stages of ***maturity*** in two distinctsurfaces (glass and polypropylene) using ***MALDI***-TOF MS. In addition, the morphology of such biofilm stages was examined and compared by ***scanning electron microscopy*** and ***atomic force microscopy*** (AFM) . . .

TABLE I
PART OF SCIENTIFIC ARTICLE. WORDS WITH ITALIC FONT ARE KEYWORDS AND SENTENCES WITH UNDERLINE ARE THE CONTENT IN ABSTRACT.

Although deep neural network based works show their effectiveness in KE and AE, most of them just focus on a single task and the improvement has reached a bottleneck. In fact, scientific KE and AE are highly relevant tasks, because the abstract could be viewed as the much shorter source text for KE, while keywords could provide important clues for AE. As shown in Table I, given the source scientific litera-

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

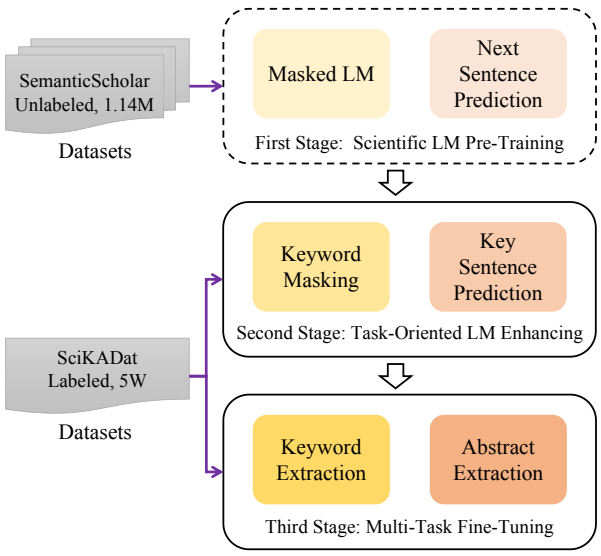

Fig. 1. Overview of SciKABERT. Transfer learning with three stages for KE and AE.

ture, the keywords "biofilm", "pseudomonas aeruginosa", etc. commonly occur in the summarization sentences. It is intuitive that, training both KE model and AE model simultaneously could promote the performance of each task. So, this paper aims to perform a multi-task learning of keyword extraction and summarization.

However, a multi-task model for KE and AE requires both word-level and sentence-level labels, which are not included in existing scientific datasets. Most annotated datasets just apply to single task [4], [5] or they are too small (contains no more than a few thousand pieces of data) to train a robust model [6], [7]. As annotating scientific articles requires domain-specific expert knowledge, it is difficult to construct huge labeled datasets. Fortunately, language models (LM) such as Bidirectional Encoder Representations from Transformers (BERT) are learned unsupervised and could be easily transferred to universal downstream natural language processing (NLP) tasks [8], [9]. Inspired by this background, we aim to use huge unlabeled data for learning scientific language representation and use small annotated data for transferring the learned representation to KE and AE. We note that, previous pre-trained LMs are generally designed for universal NLP tasks, so this paper also considers how to enhance the performance of LM for AE and KE.

In this paper, we propose a new pre-training and fine-tuning framework for multi-task learning of KE and AE. As shown in Figure 1, this framework can be divided into three stages: (i) We pre-train the language model on huge scientific unlabeled data with masked LM task and next sentence prediction task. (ii) We enhance the pre-trained language model SciKABERT (stands for Scientific Keyword Extraction and Abstract Extraction Oriented BERT) using keyword masking task and key sentence prediction task. The SciKABERT is initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data. In keyword masking task, we mask some keywords at random, and then predict those masked tokens. Key sentence prediction task aims to predict which sentence is the key one. These two tasks could boost the performance of the language model for KE and AE respectively. (iii) We apply the learned SciKABERT to the multi-task of KE and AE by just appending one word classifier layer and one sentence classifier layer, and fine-tune parameters using labeled data. Be different from the general frameworks for transfer pre-training LM [8] to universal NLP tasks, which just contains one fine-tuning stage (the 3rd stage in our approach), our framework introduces another phase (the 2nd stage) to enhance the language representation for KE and AE tasks. These two stages use the same annotated data but different optimization objectives.

To overcome data scarcity in scientific key extraction and summarization, we semi-automatically construct an annotated scientific dataset, named SciKADat, which consists of 48419 public accessible articles (with abstract and keywords) in computer science domain. We distill the introduction as the source text, label each word according to the given keywords and label sentences using a greedy algorithm to maximize ROUGE score between the selected sentences and the given abstract [10]. Rule-based preprocessing and manual check are further carried out to improve the quality of SciKADat. We compare the performance of our model on SciKADat with several important baselines. Experimental results show that our model achieves the state-of-the-art results on both KE and AE. Besides, we carry out ablation experiments which demonstrate both the effectiveness of the two-stage fine-tuning mechanism and that of the multi-task learning.

In summary, the main contributions of this paper are as follows:

- We propose a multi-task learning framework for scientific summarization and keywords extraction.
- We propose SciKABERT which is trained by keyword masking task and key sentence prediction task to boost the performance of the language model for keyword extraction and abstract extraction.
- We construct a large scale high-quality scientific dataset SciKADat which contains introduction, abstract and keywords of publications. Experimental results show the effectiveness of the LM enhancing mechanism and that of the multi-task learning framework.

II. Related Work

1) Text Summarization: Text summarization is the task of automatically generating a shorter version while remaining the main information and the task has two paradigms: generative summarization (also known as abstractive summarization) and extractive summarization. Generative summarization requires text rewriting and may contain words not appeared in the original text. Most recent generative models [5] are based on Seq2Seq and attention mechanism. However, the generated summaries are liable to reproduce factual details inaccurately

and tend to repeat themselves. Pointer mechanism [11], GAN model [12], reinforcement learning [13] and bottom-up attention [3] are introduced to promote the performance of the basic Seq2seq model. However, the deep semantic information is still hard to control through these RNN-based generative approaches. For extractive models, hierarchical Transformer [1] and BERT [8] are pre-trained with large unlabeled data with deep unidirectional architectures and achieve state-of-the-art performance on CNN/DM with extractive method. These single task models can not grasp the subjects exactly, which is vital in summarization. In our model, we propose task-oriented LM and multi-task training to further improve the performance.

2) Keyword Extraction: Keywords are selected to intuitively express the subject of an article, so that readers can determine whether to read it with a glance. Traditional keyword extraction method is based on TF-IDF [14], but the inherent relations between words are not taken into account. Then approaches based on word graphs have been proposed. TextRank [15] is the first graph-based approach for keyword extraction. PositionRank [16] is an unsupervised approach to extract keywords from scholarly documents. However, these graph-based methods are unsupervised and identify keywords from words graph manually, which lack flexibility to cope with different types of documents. End-to-end neural approaches have attracted more attention in recent studies. RNN mechanism [17], and Seq2Seq architecture [4] are proposed to facilitate the keyword generation. While these RNN-based methods ignore the correlation between keywords. Graph-based method [2] employs graph ranking, and retrieval model [18] explores the power of retrieval and extraction, achieving state-of-the-art performance. Different from these methods, our model, which is very good extractive model based on pre-trained science LM, can be used to extract keywords with the collaboration of AE.

3) Pre-trained NLP Models: Transfer learning with language models have demonstrated that rich, unsupervised pre-training is an integral part of many language understanding systems, such as ELMo [19], GPT [20] and BERT [21]. Typically, these methods first pre-train neural networks on large-scale unlabeled text corpora, and then fine-tune the models on downstream tasks. Many tasks should leverage the context in both direction, while the language model objective is unidirectional. With the capability of modeling bidirectional contexts, BERT achieves better performance than other pre-training approaches. Inspired by these, our SciKABERT pre-train task-oriented BERT as the base model and train multi-task with downstream tasks.

III. Methodology

A. Preliminary

Definition 1: Abstract Extraction

Let d denotes an source document, which contains multiple sentences $[s_1, s_2, ..., s_m]$, where s_i is the i-th sentence in d. Abstract extraction can be defined as the task of assigning a label of $y^i_{sen} \in \{0, 1\}$ to each s_i, indicating whether this sentence should be included in the abstract.

Definition 2: Keyword Extraction

Otherwise, document d can also be represented by several words $[w_1, w_2, ..., w_n]$, where w_i is the i-th words in d. Keyword extraction can be defined as the task of assigning a label of $y^i_{key} \in \{0, 1\}$ to each w_i, indicating whether this word should be included in keywords. If continuous words are chosed as keywords, then we process them as a keyphrase.

We aim to train these two extraction tasks simultaneously. Our multi-task learning framework consists of three stages: pre-training a scientific LM with the architecture of BERT, enhancing the scientific LM for AE and KE and fine-tuning the multi-task model.

B. Scientific LM Pre-Training

To pre-train a scientific LM, we employ the same architecture of deep bidirectional Transformers with BERT. In each layer, there is a multi-head self-attention sublayer and a linear affine sub-layer with the residual connection. The self-attention can be described as mapping a query (matrix Q) and a set of key (matrix K) - value (matrix V) pairs to an output. The attention distribution in self-attention layer is shown as below:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (1)$$

where d_k is output dimension, and $\frac{1}{\sqrt{d_k}}$ is used to scale the dot product.

The language model is trained on unlabeled scientific data using two unsupervised prediction tasks, masked LM and next sentence prediction, which is the same as BERT.

C. Task-Oriented LM Enhancing

In the previous stage, we learn a language model which is easily transferred to universal scientific tasks, but maybe not optimal for specific task. In order to enhance the language representation for KE and AE, we fine-tune the learned model using two new tasks, keyword masking and key sentence prediction.

1) Task #1: Keyword Masking: In this task, we simply mask some percentage of the labeled keywords at random, and then predict those masked keywords. This procedure is similar to the masked LM task in BERT and other *Cloze* tasks [1], [22]. We mask 15% WordPiece tokens in each sequence. If the ratio of keywords in a sequence exceeds 15%, we just mask the keywords randomly; otherwise, we mask all keywords and mask some common words at random. During test time, the input text is not masked. To make our model can adapt to documents without masks, we don't always replace the selected word with `[MASK]` token. The same as [21], once a word is selected as one of the 15% masked words, we replace it with one of three methods as follows.

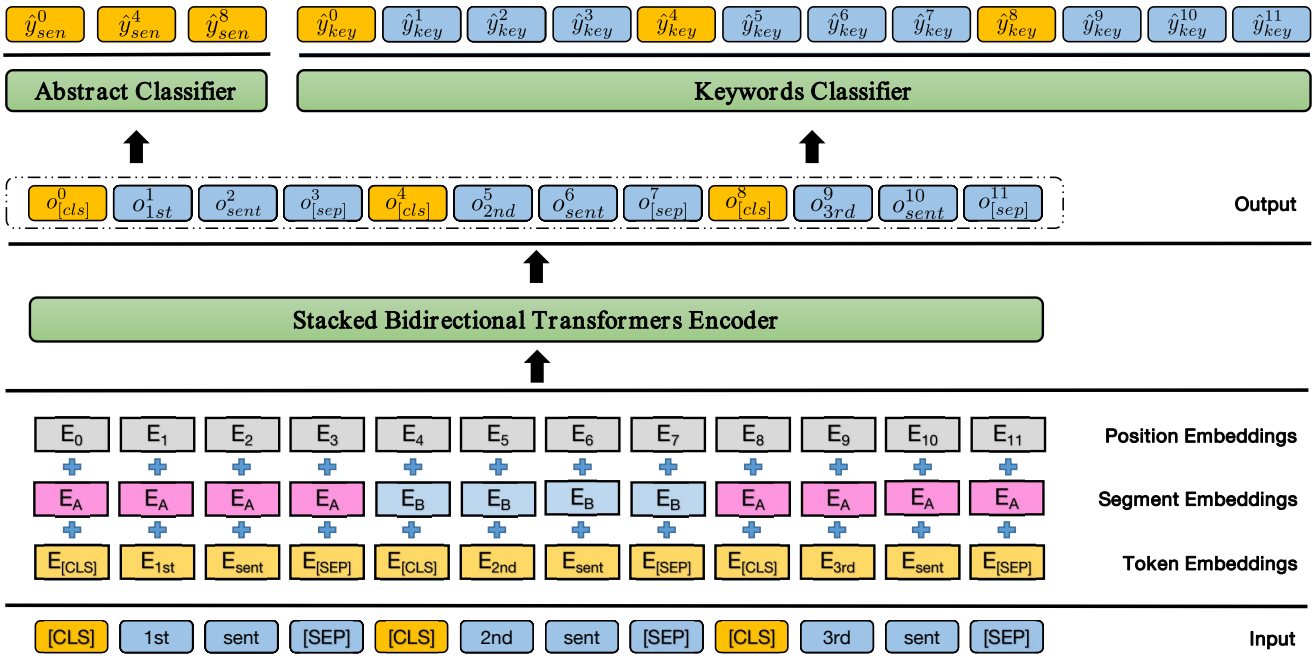

Fig. 2. The architecture of SciKABERT during multi-task fine-tuning. Abstract Classifier only takes care of `[CLS]`, which is the representation of one sentence. Keywords Classifier needs process all tokens. `[SEP]` is a special separator token at the end of sentence, `E` means the input embedding.

- replacing the chosen words with `[MASK]` token 80% of the time.
- replacing the word with a random word 10% of the time.
- keeping the word unchanged 10% of the time.

2) Task #2: Key Sentence Prediction: The next sentence prediction (NSP) task in BERT samples two spans of text from corpus and predicts a binary objective to understand the relationship between these two segments. `[CLS]` is the first token of every sequence, and it is used as the aggregate sequence representation for NSP.

Different from NSP, we sample continuous separated sentences from source article and try to predict which ones are crucial. In our model, the unit of input text is sentence rather than segment. So, we need to assign individual representation to each sentence. Just as BERTSUM [8], we insert a `[CLS]` token before each sentence and a `[SEP]` token after each sentence to highlight the boundary between sentences. In this way, our model can predict key sentence through multiple `[CLS]` tokens, by labeling 1 to crucial ones and 0 to others. In addition, our task pays more attention to the importance of sentence instead of sequential relationship, which has the same goal with AE.

For segment embeddings, BERT only has two labels: E_A, E_B, because there are only two segments in NSP. However, there are multiple sentences in our model, so we assign E_A and E_B alternately to each sentence in our task to distinguish different sentences, as shown in Figure 2.

Keyword masking task and key sentence prediction task are trained together with cross entropy objective function, in the light of MLM and NSP.

D. Multi-Task Fine-Tuning

The Multi-Task model is SciKABERT appending with one KE module and one AE module. The parameters of SciKABERT are first initialized with the parameters learned from the previous stage, and then fine-tuned using labeled data.

1) Keyword Extraction Module: For keyword extraction task, we will get the final predicted score $\hat{y}^i_{key}$ for each word, which is calculated by adding a simple classifier to the SciKABERT output. Then we choose binary classification entropy between $\hat{Y}_{key}$ and gold label Y_{key} as the loss function. Just as [8], there are three types of classifier stacked on the top of BERT output: Linear Classifier, RNN Classifier, Transformer Classifier.

Transformer model achieved great performance in NLP, so we apply a two-layer Transformer for BERT outputs:

$$\begin{aligned} \tilde{h}^l &= \mathrm{LayerNorm}(h^{l-1} + \mathrm{MultiHeadAtt}(h^{l-1})) \\ h^l &= \mathrm{LayerNorm}(\tilde{h}^l + \mathrm{FeedForwardNN}(\tilde{h}^l)) \end{aligned} \tag{2}$$

where l is the depth of stacked layers, h^0 is the BERT outputs with positional embeddings.

The final output layer is also a linear classifier:

$$\hat{y}^i_{key} = \sigma(h^L_i W_h + b^i_h) \tag{3}$$

where W_h and b_h are parameters of linear layer in Transformer classifier, h^L_i is the top layer vector for i-th word of Transformer outputs and L is 2 in our model.

2) Abstract Extraction Module: Abstract extraction is similar to keyword extraction except for that we only care about the

representation of [CLS]. So we employ another Transformer classifier:

$$\hat{y}^i_{sen} = \sigma(h_i^{'} W_h^{'} + b_h^{'i}) \tag{4}$$

where $W_h^{'}$ and $b_h^{'}$ are the parameters of linear layer in Transformer classifier, $h_i^{'}$ is the sentence-level output of top Transformer layer.

E. Multi-Task Learning Objective Functions

Our model consists of two parts, abstract and keyword extraction. The loss function of each task is shown as follows:

$$\begin{aligned} J_{key} &= \\ &- \frac{1}{N}\sum_{i=1}^{N}[y^i_{key}\log(\hat{y}^i_{key}) + (1-y^i_{key})\log(1-\hat{y}^i_{key})] \\ J_{abs} &= \\ &- \frac{1}{M}\sum_{i=1}^{M}[y^i_{sen}\log(\hat{y}^i_{sen}) + (1-y^i_{sen})\log(1-\hat{y}^i_{sen})] \end{aligned} \tag{5}$$

where M is the number of sentences in article, N is the number of tokens in article, y^i_{key} and y^i_{sen} are the ground-truth labels.

For the purpose of improving shared deep stacked bidirectional Transformers encoder, we train KE and AE model simultaneously. The joint multi-task objective is minimized by:

$$J = \lambda J_{abs} + (1-\lambda) J_{key} \tag{6}$$

where λ is hyper-parameters that determines weights of two objectives, and it is tuned on the validation set.

IV. Experiment

A. Dataset

There has been several publicly-available datasets with scientific papers which are designed for single-task. *Krapivin* [7] provides 2,304 papers with full-text and author-assigned keywords. *KP20k* [4] consists of 567,830 papers without the full-text or introduction. *SemEval-2010* [6] contains 288 articles collected from the ACM Digital Library. However, these datasets don't involve all elements (abstract, keywords and article), or the number of papers is too small. So we construct a large labeled dataset for KE and AE named SciKADat.

We first download English scientific papers with introduction, abstract and keywords from various online digital libraries, which is open access to everyone, including ScienceDirect, WWW and KDD, etc. Note that, we use introduction as the source article because it involves main information of the whole paper and is easy to handle. Then, we preprocess the dataset using heuristic rules such as removing papers of which the abstracts are copied from introduction directly, removing sentences which has formula or special symbols. Considering the extraction task, we only preserve the present keywords, which can cover most of original ones. Finally, we ask several graduate students who mastering in English to review each abstract with the readability and coherence, then we remove bad ones.

Dataset	Train	Validation	Test
Num of Articles	40916	3717	3786
Introduction Sentences	25.53	25.45	25.55
Introduction Words	636.22	633.40	637.25
Abstract Sentences	4.28	4.25	4.27
Abstract Words	97.63	96.76	97.43
Keywords	5.16	5.15	5.09

TABLE II
SciKADat Statistics: number of articles and average number of others

The words are easy to label according to the keywords provided in the original paper. To label the sentences according to the abstract in the original paper, we use a greedy algorithm to generate an oracle abstract for each introduction by greedily selecting sentences that can maximize ROUGE scores [10]. We assigned label 1 to sentences in oracle abstract and 0 to others.

After above automatically preprocessing and labeling, SciKADat contains 48,419 high-quality articles with labeled keywords and key sentences. We use 40,916 articles as the training set, 3,717 ones as the validation set and 3,786 ones as the test dataset. The details are shown in Table II:

B. Implementation Details

Our first stage is Science LM Training, which is similar to the pre-training of BERT. Fortunately, SciBERT [23] is trained on papers from the unlabeled corpus of SemanticScholar [1], with corpus size of 1.14M and 3.1B tokens. So we reuse SciBERT as our first stage.

In the second stage, we implement our model with the pre-trained parameters from previous stage, "scibert-scivocab-uncased" of SciBERT [2]. Our model was keeping training by following the original method [3] on a single machine with 4 Nvidia Tesla V100 GPUs.

For the third stage, classifiers for abstract and keyword extraction are jointly fine-tuned simultaneously with a unified loss function. Following the recommended settings in the BERT code, we set a maximum sentence length of 128 tokens and total length of 512 tokens. We train the model for 100,000 steps on same machine with batch size of 20. Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$ is used for fine-tuning. The hyper-parameters λ for J_{abs} and J_{key} is 0.6. Learning rate schedule is following with warming-up on first 10,000 steps, using the strategies in Transformer [24]. The dropout rate in all layers are 0.1.

C. Evaluation Metric

The standard metrics for keyword extraction performance are *precision*, *recall* and *F-measure* (F1). Precision is defined

[1]https://www.semanticscholar.org/
[2]https://github.com/allenai/scibert
[3]https://github.com/google-research/bert

as the number of correctly predicted keywords over the number of all predicted keywords. Recall is defined by the number of correctly predicted keywords over the total ground-truth keywords. F1 is the weighted average of precision and recall. For summarization experiments, we use the common metrics, F1 of ROUGE-1, ROUGE-2 and ROUGE-L[4], which are computed based on overlapping lexical units between generated summaries and golden ones.

V. Result and Analysis

In this section, we firstly evaluate KE and AE on several extractive and generative baselines respectively, shown in Table III and Table IV. Then we carry out some ablation studies about the performance of different classifier layers, the efficiency of different stages and the contribution of each components in SciKABERT, shown in Table V and Table VI. We also analyze the score trend of KE and AE with different factors in Figure 3.

A. Evaluation on Abstract Extraction

In order to evaluate the effectiveness of our model on AE, we select four extractive methods and six generative ones as baselines, which achieve the best performance on CNN/DN dataset.

- *Generative Methods.* Attention-Based Seq2Seq [5], [25] puts forward a generative method for summarization with neural network. Pointer-Generator [11], DeepRL [13] and GAN [12] use sequence to sequence framework, respectively plusing copy mechanism and coverage modeling, reinforcement learning and generative adversarial learning. Unified Model [26] is the combine of extraction and generation model. Bottom-Up [3] generates summarization by combining word prediction model with bottom-up attention model.
- *Extractive Methods.* Lead-3 and TextRank [15] are classical unsupervised and extractive models for summarization. SummaRuNNer [10] is RNN-based extractive model. The most recent work BERTSUM [8] is a pretraining language model for extractive summarization, and outperforms HIBERT [1] obviously on CNN/DM.

As shown in Table III, the extractive methods generally perform better than generative ones. On our dataset, SummaRuNNer significantly outperforms many other generative or extractive approaches, and TextRank is better than almost all Seq2Seq-based models. The reason may be that generative methods will generate many duplicate words and irrelevant expressions. Bottom-Up outperforms other generative models because there is more explicit sentence compression with bottom-up attention. It is worth noting that BERT-based models achieve prominent improvement on AE. BERTSUM gets 2.51, 4.37, 2.46 promotion on Rouge-1, Rouge-2, Rouge-L, compared with SummaRuNNer. SciKABERT achieves the state-of-the-art performance with the task-oriented LM for summarization task.

[4]https://github.com/andersjo/pyrouge

Models (%)	Rouge-1	Rouge-2	Rouge-L
Seq2Seq	41.60	19.96	35.93
Pointer-Generator	42.33	20.25	36.57
Unified-Model	43.19	20.67	36.68
DeepRL	42.76	18.82	36.90
GAN	41.92	19.65	35.71
Bottom-Up	43.92	21.65	37.21
Lead-3	31.61	11.70	26.54
TextRank	43.20	25.56	40.42
SummaRuNNer	45.66	26.11	42.77
BERTSUM	48.17	30.48	45.23
SciKABERT	**49.34**	**31.85**	**46.55**

TABLE III
THE PERFORMANCE OF DIFFERENT MODELS ON AE.

Models (%)	$\mathbf{F_1@3}$	$\mathbf{F_1@5}$	$\mathbf{F_1@10}$
CNN	18.42	15.18	11.03
CopyCNN	31.74	26.38	20.27
RNN	19.53	15.58	12.18
CopyRNN	32.37	28.72	20.92
TF-IDF	13.21	11.63	9.12
TextRank	15.58	13.77	8.50
PositionRank	30.44	27.74	22.05
YAKE	19.73	16.51	12.73
SeqPointer	35.40	31.88	24.55
GraphPointer	36.18	32.42	26.38
BERT	38.66	34.32	27.57
SciKABERT	**39.51**	**35.47**	**29.20**

TABLE IV
THE PERFORMANCE OF DIFFERENT MODELS ON KE.

B. Evaluation on Keyword Extraction

To evaluate the efficiency of our model on KE, we select four generative methods and seven extractive ones as baselines.

- *Generative Methods.* Comparative generative methods include CNN, CopyCNN [27] RNN and CopyRNN [4]. These approaches could identify keyphrases that do not appear in the text and copy mechanism is vital for keywords generation.
- *Extractive Methods.* Comparative extractive methods include TF-IDF [14], TextRank [15], PositionRank [16] , YAKE [28], SeqPointer and GraphPointer [2]. The former four approaches are unsupervised method. TF-IDF and YAKE utilize text statistical features to select the most important keywords. TextRank and PositionRank are graph-based method and identify keywords from words graph. While end-to-end neural approaches have attracted more attention in recent studies, and SeqPointer and GraphPointer are the most promising ones among them. Besides, BERT achieves ground-breaking performance on multiple NLP tasks and we compare our model with it in experiments.

Table IV shows the results of comparative keyword extraction methods including generative methods in the top block and extractive ones in the bottom block. On our dataset, PositionRank outperforms other unsupervised models, for that it is designed for scholarly documents. The CopyCNN and

CopyRNN with copy mechanism significantly improved the performance base on RNN model, and SeqPointer with pointer network gets further promotion. For that copy or pointer mechanism can extract keywords exactly. GraphPointer combines the graph and pointer mechanism and outperforms other supervised approaches. All BERT-based models outperform previous models by a substantial margin, especially for our SciKABERT. It demonstrates that our task-oriented LM enhancing and multi-task method are essential for downstream tasks.

C. Ablation Studies

1) Classifier Layer: We experiment with three kinds of classifier layer in our model: Linear Layer, RNN Layer and Transformer Layer. We evaluate these classifiers using F-measures **F@3**, **F@5** for KE and Rouge-1, Rouge-L for AE. As illustrated in Table V, SciKABERT with Transformer classifier achieved the best performance on four metrics, and Linear is the worst one.

SciKABERT (%)	$\mathbf{F_1@3}$	$\mathbf{F_1@5}$	R-1	R-L
+Linear	38.93	34.82	48.28	45.38
+RNN	39.28	35.36	48.97	46.27
+Transformer	**39.51**	**35.47**	**49.34**	**46.55**

TABLE V
RESULTS OF DIFFERENT CLASSIFIERS FOR EXTRACTION TASK

2) Training Mechanism: This section shows the contribution of different components of SciKABERT, including keyword masking (KM), key sentence prediction (KSP), scientific LM, task-oriented LM and multi-task. The results are shown in Table VI. For keyword extraction, task-oriented LM is most important and the second one is KM, for the reason that keywords masking task improved the representations of keywords. For abstract extraction, multi-task learning plays an important role and task-oriented LM is the second one, for the reason that KE help AE to grasp subjects exactly. We can also learn about that KSP is most useless in our model.

D. Effect of Training Steps Number

In this section, we analyze the influence of different factors for the performance. Figure 3 gives the score curves of three models: the BERT, SciKABERT without task-oriented LM and SciKABERT. As shown in the figure, we evaluate the model every 1,000 steps during training. Keyword extraction gets stable after 25,000 steps, while it is 10,000 steps for abstract extraction, which is because that sentence-level objectives are information-rich to distinguish. The SciKABERT achieves best performance all the time except for the beginning, for the reason that our multi-task unified objective will spend some time to get the right state.

Models (%)	$\mathbf{F_1@3}$	$\mathbf{F_1@5}$	R-1	R-L
SciKABERT	**39.51**	**35.47**	**49.34**	**46.55**
w/o Multi-Task	39.23	35.03	48.32	45.43
w/o Task LM	39.07	34.58	48.59	45.69
w/o Scientific LM	39.38	35.26	49.03	46.25
w/o KM	39.10	34.94	49.17	46.24
w/o KSP	39.45	35.31	49.29	46.44

TABLE VI
RESULTS OF ABLATION STUDIES FOR SCIKABERT

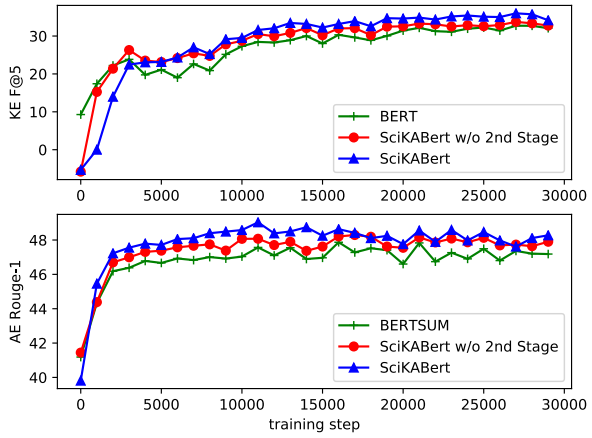

Fig. 3. Trends of $\mathbf{F_1@5}$ score for KE and Rouge-1 score for AE on the validate set with different training step

VI. CONCLUSION

This paper proposes a multi-task learning framework to extract keywords and abstract of scientific articles simultaneously. This framework consists of three stages: pre-training a scientific LM, enhancing the LM for specific tasks and fine-tuning the multi-task model. Beyond the traditional pre-training and fine-tuning phases, in order to enhance the language representation for AE and KE, we use two tasks *keyword masking* and *key sentence prediction* before the fine-tuning phase. Besides, to overcome data scarcity problem, we develop and release a high-quality annotated corpus for scientific papers with keywords and abstract. We conduct comparative experiments on this dataset, and experimental results show that our multi-task learning framework achieves the state-of-the-art performance. Although the LM enhancing mechanism is able to promote the performance of our multi-task model, there are many restrictions to the input text and it is hard to balance the separated loss functions in the unified objective. In the future work, we will pay more attention to these problems.

VII. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (NO.Y850371101). We thank all authors for their contributions and all anonymous reviewers for their constructive comments.

REFERENCES

[1] X. Zhang, F. Wei, and M. Zhou, "HIBERT: document level pre-training of hierarchical bidirectional transformers for document summarization," *CoRR*, vol. abs/1905.06566, 2019.

[2] Z. Sun, J. Tang, P. Du, Z.-H. Deng, and J.-Y. Nie, "Divgraphpointer: A graph pointer network for extracting diverse keyphrases," *arXiv preprint arXiv:1905.07689*, 2019.

[3] S. Gehrmann, Y. Deng, and A. M. Rush, "Bottom-up abstractive summarization," *arXiv preprint arXiv:1808.10792*, 2018.

[4] R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, and Y. Chi, "Deep keyphrase generation," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 582–592. [Online]. Available: https://www.aclweb.org/anthology/P17-1054

[5] R. Nallapati, B. Zhou, C. Gulcehre, B. Xiang *et al.*, "Abstractive text summarization using sequence-to-sequence rnns and beyond," *arXiv preprint arXiv:1602.06023*, 2016.

[6] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, 2010, pp. 21–26.

[7] M. Krapivin, A. Autaeu, and M. Marchese, "Large dataset for keyphrases extraction," University of Trento, Tech. Rep., 2009.

[8] Y. Liu, "Fine-tune bert for extractive summarization," *arXiv preprint arXiv:1903.10318*, 2019.

[9] H. Zhang, Y. Gong, Y. Yan, N. Duan, J. Xu, J. Wang, M. Gong, and M. Zhou, "Pretraining-based natural language generation for text summarization," *arXiv preprint arXiv:1902.09243*, 2019.

[10] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in *AAAI*, 2017.

[11] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," *CoRR*, vol. abs/1704.04368, 2017. [Online]. Available: http://arxiv.org/abs/1704.04368

[12] L. Liu, Y. Lu, M. Yang, Q. Qu, J. Zhu, and H. Li, "Generative adversarial network for abstractive text summarization," in *AAAI*, 2018.

[13] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=HkACIQgA-

[14] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, 1972.

[15] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004, pp. 404–411.

[16] C. Florescu and C. Caragea, "Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1105–1115.

[17] Q. Zhang, Y. Wang, Y. Gong, and X. Huang, "Keyphrase extraction using deep recurrent neural networks on twitter," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 836–845.

[18] W. Chen, H. P. Chan, P. Li, L. Bing, and I. King, "An integrated approach for keyphrase generation via exploring the power of retrieval and extraction," *arXiv preprint arXiv:1904.03454*, 2019.

[19] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[20] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *NAACL*, 2019.

[22] X. Wu, T. Zhang, L. Zang, J. Han, and S. Hu, "Mask and infill: Applying masked language model to sentiment transfer," *arXiv preprint arXiv:1908.08039*, 2019.

[23] I. Beltagy, A. Cohan, and K. Lo, "Scibert: Pretrained contextualized embeddings for scientific text," *arXiv preprint arXiv:1903.10676*, 2019.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[25] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," *CoRR*, vol. abs/1509.00685, 2015. [Online]. Available: http://arxiv.org/abs/1509.00685

[26] W.-T. Hsu, C.-K. Lin, M.-Y. Lee, K. Min, J. Tang, and M. Sun, "A unified model for extractive and abstractive summarization using inconsistency loss," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018, pp. 132–141. [Online]. Available: http://aclweb.org/anthology/P18-1013

[27] Y. Zhang, Y. Fang, and X. Weidong, "Deep keyphrase generation with a convolutional sequence to sequence model," in *2017 4th International Conference on Systems and Informatics (ICSAI)*. IEEE, 2017, pp. 1477–1485.

[28] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt, "Yake! collection-independent automatic keyword extractor," in *European Conference on Information Retrieval*. Springer, 2018, pp. 806–810.