UniKeyphrase: A Unified Extraction and Generation Framework for Keyphrase Prediction

Huanqin Wu[1*], **Wei Liu**[1*], **Lei Li**[2], **Dan Nie**[1], **Tao Chen**[1], **Feng Zhang** [1], **Di Wang** [1]

[1]Tencent AI Platform Department, China
[2]Beijing University of Posts and Telecommunications
{huanqinwu,thinkweeliu,kathynie,vitochen,jayzhang,diwang}@tencent.com
leili@bupt.edu.cn

Abstract

Keyphrase Prediction (KP) task aims at predicting several keyphrases that can summarize the main idea of the given document. Mainstream KP methods can be categorized into purely generative approaches and integrated models with extraction and generation. However, these methods either ignore the diversity among keyphrases or only weakly capture the relation across tasks implicitly. In this paper, we propose **UniKeyphrase**, a novel end-to-end learning framework that jointly learns to extract and generate keyphrases. In UniKeyphrase, stacked relation layer and bag-of-words constraint are proposed to fully exploit the latent semantic relation between extraction and generation in the view of model structure and training process, respectively. Experiments on KP benchmarks demonstrate that our joint approach outperforms mainstream methods by a large margin.

1 Introduction

Keyphrases are several phrases that highlight core topics or information of a document. Given a document, the KP task focuses on automatically obtaining a set of keyphrases. As a basic NLP task, keyphrase prediction is useful for numerous downstream NLP tasks such as summarization (Wang and Cardie, 2013; Pasunuru and Bansal, 2018), document clustering (Hulth and Megyesi, 2006), information retrieval (Kim et al., 2013).

Keyphrases of a document fall into two categories: *present keyphrase* that appears continuously in the document, and *absent keyphrase* which does not exist in the document. Figure 1 shows an example of a document and its keyphrases. Traditional KP methods are mainly extractive, which have been extensively researched in past decades (Witten et al., 2005; Nguyen and Kan, 2007; Medelyan et al., 2009; Lopez and Romary, 2010; Zhang et al., 2016; Alzaidy et al., 2019; Sun et al., 2020). These methods aim to select text spans or phrases directly in the document, which show promising results on present keyphrase prediction. However, extractive methods cannot handle the absent keyphrase, which is also significant and requires a comprehensive understanding of document.

Document: On selecting an optimal wavelet for detecting singularities in traffic and vehicular data. …… applications of wavelet transform s (wts) in traffic engineering have been introduced however , …… , second order difference , oblique cumulative curve , and short time fourier transform) . it then mathematically describes wts ability to detect singularities in traffic data . …… , it is shown that selecting a suitable wavelet largely depends on the specific research topic , and that the mexican hat wavelet generally gives a satisfactory performance in detecting singularities in traffic and vehicular data .
Present keyphrases: { wavelet transform, oblique cumulative curve, short time fourier, the mexican hat wavelet }
Absent keyphrases: { singularity detection, traffic data analysis }

Figure 1: An example of an input document and its expected keyphrases. Blue and red denote present and absent keyphrases, respectively.

To mitigate this issue, several generative methods (Meng et al., 2017; Chen et al., 2018; Ye and Wang, 2018; Wang et al., 2019; Chen et al., 2019b; Chan et al., 2019; Zhao and Zhang, 2019; Chen et al., 2020; Yuan et al., 2020) have been proposed. Generative methods mainly adopt the sequence-to-sequence (seq2seq) model with a copy mechanism to predict a target sequence, which is concatenated of present and absent keyphrases. Therefore, the generative approach can predict both kinds of keyphrases. But these methods treat present and absent keyphrases equally, while these two kinds of keyphrase actually have different semantic properties. As illustrated in Figure 1, all the present keyphrases are specific techniques, while the absent keyphrases are tasks or research areas.

Thus several integrated methods (Chen et al., 2019a; Ahmad et al., 2020) try to perform multi-

* Equal contribution.

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 825–835
August 1–6, 2021. ©2021 Association for Computational Linguistics

task learning on present keyphrase extraction (PKE) and absent keyphrase generation (AKG). By treating present and absent keyphrase prediction as different tasks, integrated methods clearly distinguish the semantic properties for these two kinds of keyphrases. But integrated models suffer from two limitations. Firstly, these approaches are not trained in an end-to-end fashion, which causes error accumulation in the pipeline. Secondly, integrated methods just adopt a bottom shared encoder to implicitly capture the latent semantic relation between PKE and AKG, while this relation is essential for the KP task. As illustrated in Figure 1, the ground truth of PKE are specific techniques, which are all used for the "singularity detection" task in the "traffic data analysis" area. Such semantic relation between PKE and AKG can bring benefits for KP. Actually, semantic relations like "technique-task-area" between two tasks are common in the KP task. However, these integrated methods are weak at modeling it.

To address these issues, we propose a novel end-to-end joint model, UniKeyphrase, which adopts a unified pretrained language model as the backbone and is fine-tuned with both PKE and AKG tasks. What's more, UniKeyphrase explicitly captures the mutual relation between these two tasks, which brings benefits for keyphrase prediction: present keyphrases can provide an overall sense about salient parts of the document for AKG, and absent keyphrases viewed as high-level latent topics of the document can also supply PKE with global semantic information.

Specifically, UniKeyphrase employs two mechanisms to capture the relation from model structure and training process, respectively. Firstly, stacked relation layer is applied to repeatedly fuse PKE and AKG task representations to explicitly model the relation between the two sub-tasks. In detail, we adopt a co-attention based relation network to model the co-influence. Secondly, a bag-of-words constraint is designed for UniKeyphrase, which aims to provide some auxiliary global information of the whole keyphrases set during training.

Experiments conducted on the widely used public datasets show that our method significantly outperforms mainstream generative and integrative models. The contributions of this paper can be summarized as follows:

- We introduce a novel end-to-end framework UniKeyphrase for unified PKE and AKG.
- We design stacked relation layer (SRL) to explicitly capture the relation between PKE and AKG.
- We propose bag-of-words constraint (BWC) to explicitly feed global information about present and absent keyphrases to the model.

2 Related Works

2.1 Keyphrase Extraction

Most existing extraction approaches can be categorized into two-step extraction methods and sequence labeling approaches. Two-step extraction methods first identify a set of candidate phrases from the document by heuristics, such as essential n-grams or noun phrase (Hulth, 2003). Then, the candidate keyphrases are sorted and ranked to get predicted results. The scores can be learned by either supervised algorithms (Nguyen and Kan, 2007; Medelyan et al., 2009; Lopez and Romary, 2010) or unsupervised graph ranking methods (Mihalcea and Tarau, 2004; Wan and Xiao, 2008). For sequence labeling approaches, documents are fed to an encoder then the model learns to predict the likelihood of each word being a keyphrase (Zhang et al., 2016; Alzaidy et al., 2019; Sun et al., 2020).

2.2 Keyphrase Generation

Keyphrase generation focuses on predicting both present and absent keyphrases. Meng et al. (2017) first propose CopyRNN which is a seq2seq framework with attention and copy mechanism. Then a semi-supervised method for the exploitation of the unlabeled data is investigated by Ye and Wang (2018). Chen et al. (2018) employ a review mechanism to reduce duplicates. Chen et al. (2019b) focus on leveraging the title information to improve keyphrases generation. The latent topics of the document are exploited to enrich features by Wang et al. (2019). Zhao and Zhang (2019) utilize linguistic constraints to prevent model from generating overlapped phrases. Chan et al. (2019) introduce a reinforcement learning approach for keyphrase generation. Chen et al. (2020) propose an exclusive hierarchical decoding framework to explicitly model the hierarchical compositionality of a keyphrase set. Yuan et al. (2020) introduce a new model to generate multiple keyphrases as delimiter-separated sequences.

2.3 Integrated Methods

To explicitly distinguish the present and absent keyphrases, integrated extraction and generation approach have been applied to the KP task. Chen et al. (2019a) aim at improving the performance of the generative model by using an extractive model. Ahmad et al. (2020) propose SEG-Net, a neural keyphrase generation model that is composed of a selector for selecting the salient sentences in a document, and an extractor-generator that extracts and generates keyphrases from the selected sentences. In contrast to these methods, our joint approach can explicitly capture the relation between extraction and generation in an end-to-end framework.

3 Approach

In this section, we describe the architecture of UniKeyphrase. Figure 2 gives an overview of UniKeyphrase, which consists of three components: extractor-generator backbone based on UNILM, a stacked relation layer for capturing the relation between PKE and AKG, and bag-of-words constraint for considering the global view of two tasks in training. In the following sections, the details of UniKeyphrase are given.

3.1 Extractor-Generator Backbone

Given a document $\mathbf{X} = \{x_1, ..., x_m\}$, KP aims at obtaining a keyphrase set $\mathbf{K} = \{k_1, ..., k_{|K|}\}$. Naturally, $\mathbf{K}$ can be divided into present keyphrase set $\mathbf{K}_p = \{k^p_1, ..., k^p_{|K_p|}\}$ and absent keyphrase set $\mathbf{K}_a = \{k^a_1, , ..., k^a_{|K_a|}\}$ by judging whether keyphrases appear exactly in the source document. UniKeyphrase decomposes the KP into PKE and AKG, and jointly learns two tasks in an end-to-end framework.

UniKeyphrase treats PKE as a sequence labeling task and AKG as a text generation task. To jointly learn in an end-to-end framework, UniKeyphrase adopts UNILM (Dong et al., 2019) as the backbone network. UNILM is a pre-trained language model, which can perform sequence-to-sequence prediction by employing a shared transformer network and utilizing specific self-attention masks to control what context the prediction conditions on.

As shown in Figure 2, with a pre-trained UNILM layer, the contextualized representation for the source document can attend to each other from both directions, which is convenient for PKE. While the representation of the target token can only attend to the left context, as well as all the tokens in the source document, which can be easily adapted to AKG.

Specifically, for a document $\mathbf{X}$, all absent keyphrases will be concatenated as a sequence. Then we randomly choose tokens in this sequence, and replace them with the special token [MASK]. The masked sequence is defined as $\mathbf{K}^m_a$. We further concatenate document $\mathbf{X}$ and $\mathbf{K}^m_a$ with [CLS] and [SEP] tokens as the input sequence:

$$\mathbf{I} = \{[\mathrm{CLS}]\, \mathbf{X}\, [\mathrm{SEP}]\, \mathbf{K}^m_a\, [\mathrm{SEP}]\} \tag{1}$$

Afterwards, we feed input sequence into UNILM and obtain output hidden state $\mathbf{H}$:

$$\mathbf{H} = \mathrm{UNILM}(\mathbf{I}) \tag{2}$$

the hidden state $\mathbf{H} = \{h_1, ..., h_T\}$ (T is the number of input tokens in the UNILM) will be used as the input of stacked relation layer for jointly modeling PKE and AKG.

3.2 Stacked Relation Layer

Based on the UNILM, we can obtain the output hidden $\mathbf{H}$. Instead of directly using the UNILM hidden for PKE and AKG, we use the SRL to explicitly model the relation between these two tasks. Actually, modeling the cross-impact and interaction between different tasks in joint model is a common problem (Qin et al., 2020a,b, 2019).

Specifically, SRL takes the initial shared representations $\mathbf{P}^0 = \mathbf{A}^0 = \{h_1, ..., h_T\}$ as input and aims to obtain the finally task representations $\mathbf{P}^L$ and $\mathbf{A}^L$ (L is the number of stacked layers), which consider the cross-impact between PKE and AKG. Besides, SRL can be stacked to repeatedly fuse PKE and AKG task representations for better capturing mutual relation.

Formally, given the lth layer inputs $\mathbf{P}^l = \{p^l_1, ..., p^l_T\}$ and $\mathbf{A}^l = \{a^l_1, ..., a^l_T\}$, stacked relation layer first apply two linear transformations with a ReLU activation over the input to make them more task-specific, which can be written as follow:

$$\mathbf{P}^{l'} = \mathrm{LN}(\mathbf{P}^l + \max(0, \mathbf{W}^l_P \mathbf{P}^l + \mathbf{b}^l_P)) \tag{3}$$

$$\mathbf{A}^{l'} = \mathrm{LN}(\mathbf{A}^l + \max(0, \mathbf{W}^l_A \mathbf{A}^l + \mathbf{b}^l_A)) \tag{4}$$

where LN represent the layer normalization function (Ba et al., 2016).

Then the relation between the two tasks will be integrated base on task-specific representations. In this paper, we adopt co-attention relation networks.

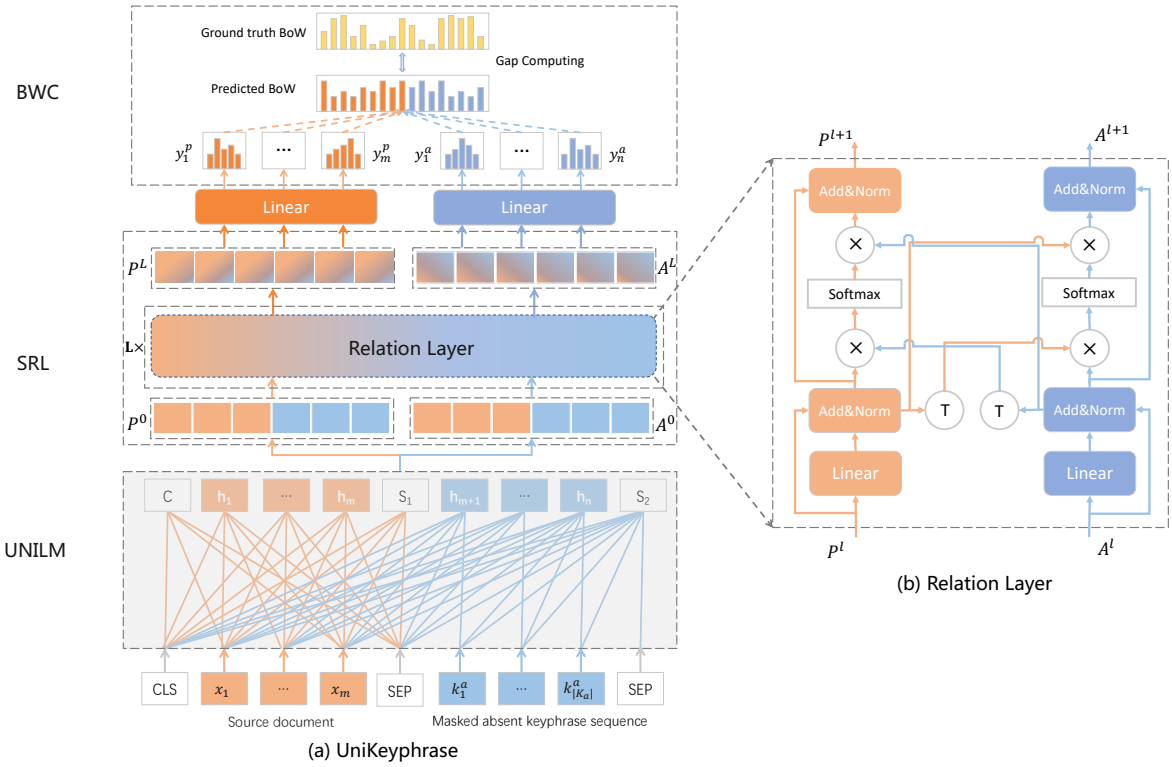

Figure 2: The architecture of our model

Co-Attention is an effective approach to model the important information of correlated tasks. We extend the basic co-attention mechanism from token level to task representations level. It can produce the PKE and AKG task representations considering each other. Therefore, we can transfer useful mutual information between two tasks. The process can be formulated as follows:

$$\mathbf{P}^{l+1} = \mathrm{LN}(\mathbf{P}^{l'} + \mathrm{softmax}(\mathbf{P}^{l'}(\mathbf{A}^{l'})^{\top})\mathbf{A}^{l'}) \quad (5)$$

$$\mathbf{A}^{l+1} = \mathrm{LN}(\mathbf{A}^{l'} + \mathrm{softmax}(\mathbf{A}^{l'}(\mathbf{P}^{l'})^{\top})\mathbf{P}^{l'}) \quad (6)$$

where $\mathbf{P}^{l+1} = \{p_1^{l+1}, ..., p_T^{l+1}\}$ and $\mathbf{A}^{l+1} = \{a_1^{l+1}, ..., a_T^{l+1}\}$ are the lth layer updated representations.

After stacked relation layer, we can obtain the outputs $\mathbf{P}^L = \{p_1^L, ..., p_m^L\}$ and $\mathbf{A}^L = \{a_1^L, ..., a_n^L\}$. We then adopt separate decoders to perform PKE and AKG by using the task representations of corresponding position , which can be denoted as follows:

$$\mathbf{y}_i^p = \mathrm{softmax}(\mathbf{W}^p p_i^L + \mathbf{b}^p) \quad (7)$$

$$\mathbf{y}_j^a = \mathrm{softmax}(\mathbf{W}^a a_j^L + \mathbf{b}^a) \quad (8)$$

where $\mathbf{y}_i^p$ and $\mathbf{y}_j^a$ are the predicted distribution for present keyphrase and absent keyphrase respectively; $\mathbf{W}^p$ and $\mathbf{W}^a$ are transformation matrices; $\mathbf{b}_p$ and $\mathbf{b}_a$ are bias vectors.

3.3 Bag-of-Words Constraint

UniKeyphrase divides the KP task into two subtasks, PKE and AKG. These two sub-tasks are optimized separately, which lacks the awareness of global information about the total keyphrase set. Such global information can be the amount of all keyphrases or the common words between present and absent keyphrases. Bag of words (BoW) is a suitable medium for describing this information. In this paper, we feed global information to UniKeyphrase by constructing constraints based on the BoW of keyphrases. The word count in BoW can provide guidance about task relation for PKE and AKG training in a global view.

Specifically, we calculate the gap between the model predicted keyphrase BoW and ground truth keyphrase BoW, then add it into the loss. Hence UniKeyphrase can get a global view of keyphrases allocation and adjust two tasks during training.

We first collect present and absent keyphrase BoW from model. For present keyphrases, since PKE is a sequence labeling task, we collect all words that labeled as keyphrases, and construct present predicted BoW V^p. We use the sum of

corresponding label probabilities as the count of word w in V^p:

$$V^p(w) = \sum_{i \in \mathcal{I}_w} max(\mathbf{y}_i^p) \tag{9}$$

where y_i^p denotes all predicted label probabilities at time step i. $\mathcal{I}_w$ is all position of word w in document. Maximum operation is used for selecting the probability of predicted label. For absent keyphrase, the generation probability of all steps are accumulated as predicted absent BoW $V^a(w)$.

$$V^a(w) = \sum_{j=1}^{N} \mathbf{y}_j^a(w) \tag{10}$$

After acquiring the predicted present and absent keyphrase BoW, we concatenate these two parts as the total predicted BoW V, then calculate the error compared with ground truth BoW $\hat{V}$. To reserve the word count information, we use Mean Square Error (MSE) function:

$$\mathcal{L}_{BoW} = \frac{1}{|\mathcal{V}|} \sum_{w \in \mathcal{V}} (V(w) - \hat{V}(w))^2 \tag{11}$$

It is worth noting that $\mathcal{V}$ is the collection of words that make up the ground truth keyphrases and predicted keyphrases. So the BWC only affects a small subset of the whole vocabulary for each sample. This can help reduce the noise and stabilize the training process.

In practice we increase the weight of BWC logarithmically from zero to a defined maximum value w_m, the weight of BWC on t step can be denoted as follows:

$$w_{BoW}(t) = log(\frac{e^{w_m} - 1}{t_{total}} t + 1) \tag{12}$$

where t_{total} is the total step of training. The reason to adjust the weight is the same as Ma et al. (2018). The BWC should take effect when predicted results are good enough. Therefore we first assign a small weight to BWC at the initial time, and gradually increase it when training.

3.4 Training

For the PKE task, objection is formulated as:

$$\mathcal{L}_{PKE} = -\sum_{i=1}^{M} \sum_{c=1}^{C} w_c \hat{\mathbf{y}}_i^{(c,p)} \log \left(\mathbf{y}_i^{(c,p)} \right) \tag{13}$$

where M refers to the length of document, C refers to the number of label, w_c is the loss weight for the positive label. $\hat{\mathbf{y}}_i^p$ refers the gold label.

For the AKG task, training objection is to maximize the likelihood of masked tokens, which is formulated as:

$$\mathcal{L}_{AKG} = -\sum_{i=1}^{N} \sum_{j=1}^{V_s} \hat{\mathbf{y}}_i^{(j,a)} \log \left(\mathbf{y}_i^{(j,a)} \right) \tag{14}$$

where N refers to the number of masked tokens, V_s refers to the size of vocabulary. $\hat{\mathbf{y}}_i^a$ refers the ground-truth word.

Considering the BWC, the overall loss of UniKeyphrase is formulated as:

$$\mathcal{L} = \mathcal{L}_{PKE} + \mathcal{L}_{AKG} + w_{BoW} \mathcal{L}_{BoW} \tag{15}$$

4 Experiments

4.1 Datasets and Evaluation

We follow the widely used setup of the deep KP task: train, validation and test on the KP20K (Meng et al., 2017) dataset, and give evaluation on three more benchmark datasets: NUS (Nguyen and Kan, 2007), INSPEC (Hulth, 2003) and SEMEVAL (Kim et al., 2010). We follow the preprocess, post-process, and evaluation setting of Meng et al. (2017, 2019); Yuan et al. (2020)[1]. Specifically, we use the partition of present and absent provided by Meng et al. (2017) and calculate F_1@5 and F_1@M (use all predicted keyphrases for F_1 calculation) after stemming and removing duplicates.

4.2 Experimental Setup

Setting: We reuse most hyper-parameters from pre-trained UNILM[2]. The layer number of SRL is set to 2. We use $w_m = 1.0$ when adjusting the weight of BWC. PKE loss weights w_c for the positive label is set to 5.0. we set batch size to 256, and maximum length to 384. During decoding, we use beam search for AKG, and beam size is set as 5. We train our model on the training set for 100 epochs. It takes about 40 minutes per epoch to train UniKeyphrase on 8 Nvidia Tesla V100 GPU cards with mixed-precision training. More details are provided in Appendix B.

[1]we follow the official GitHub repository to prepare datasets and evaluation scripts which are available on https://github.com/memray/OpenNMT-kpg-release.

[2]we use the official provided pre-trained model, which is available on https://unilm.blob.core.windows.net/ckpt/unilm1-base-cased.bin.

Type	Model	KP20k		NUS		SemEval		Inspec	
		F_1@5	F_1@M	F_1@5	F_1@M	F_1@5	F_1@M	F_1@5	F_1@M
Generative	CatSeq	29.1	36.7	32.3	39.7	24.2	28.3	22.5	26.2
	CatSeqTG	29.2	36.6	32.5	39.3	24.6	29.0	22.9	27.0
	CatSeq(TRM)	29.1	36.8	32.8	40.5	24.5	28.8	22.5	26.4
	CatSeqD	28.5	36.3	32.1	39.4	23.3	27.4	21.9	26.3
	ExHiRD-h	31.1	37.4	–	–	28.4	33.5	25.3	29.1
Integrated	KG-KE-KR-M	31.7	–	28.9	–	20.2	–	25.7	–
	SEG-NET	32.3	38.1	40.1	**45.9**	29.8	34.1	24.6	30.1
Joint	UniKeyphrase	**40.8**	**42.8**	**43.4**	43.5	**41.6**	**40.9**	**29.0**	**31.1**

Table 1: Results on present keyphrase prediction.

Baselines: We compare two kinds of strong baselines (generative, integrated) to give a comprehensive evaluation on the performance of UniKeyphrase.

- **Generative**: Generative models can predict both present and absent keyphrases under the seq2seq framework. CatSeq (Yuan et al., 2020) is the classic setting of keyphrase seq2seq model. We report the performance of CatSeq and various improved models on it, including CatSeqTG (Chen et al., 2019b), CatSeq (TRM) (Ahmad et al., 2020) and CatSeqD (Yuan et al., 2020). A recently released model is also included for comparing, which is ExHiRD-h (Chen et al., 2020).

- **Integrated**: Integrated model often combine multiple modules to perform extractive and abstractive tasks. But they are not end-to-end. Two latest integrated models are recorded for comparison. including KG-KE-KR-M (Chen et al., 2019a) and SEG-NET (Ahmad et al., 2020)

4.3 Main Results

In this section, we show the experimental results of the baseline methods and our model on present keyphrase extraction and absent keyphrase generation. Besides, we also study the average number of unique predicted keyphrases per document to further show the advantages of our model.

4.3.1 Present and Absent Keyphrase Prediction

The present and absent keyphrase prediction performance of all methods are shown in Table 1 and Table 2. From the results, we can find that our joint framework outperforms most state-of-the-art generative baseline by a significant margin, which demonstrates the effectiveness of our UniKeyphrase. We notice that the UniKeyphrase does not outperform the SEG-NET on F_1@M for present keyphrase extraction on `NUS` dataset. One potential reason is that the source document length and sentence number of `NUS` are much larger than the KP20k training set. SEG-NET employs an additional sentence selector for filtering sentences without keyphrase, which is more adaptable for processing long documents like `NUS`.

4.3.2 Number of Predicted Keyphrases

The number of predicted keyphrases indicates the model's understanding of input documents. From the previous work (Chen et al., 2020), we find the average number of unique predicted keyphrases per document is much lower than the gold average keyphrase number in most datasets. The number of unique keyphrases predicted by UniKeyphrase and baselines is compared in Table 3. We can find that UniKeyphrase predicts more (especially in absent keyphrases) than baseline methods, which is closer to ground truth. Meanwhile, we find UniKeyphrase leads to predict more keyphrases than the ground-truth (especially on KP20k). We leave solving the over prediction keyphrases problem as our future work.

4.4 Ablation Study

In this section, we check the improvement brought by SRL and BWC. Several ablation experiments are conducted to analyze the effect of different components. The ablation experiment on `SemEval` is shown in Table 4. The results show the effectiveness of different components of our method to the final performance.

Effectiveness of stacked relation layer: In this setting, we conduct experiments on the multi-task framework where PKE and AKG promote each other only by the hidden state of UniLM, From

[3]Reports from Yuan et al. (2020), which do not report absent metrics for this model. The original paper also does not give detailed numbers.

Type	Model	KP20k		NUS		SemEval		Inspec	
		F_1@5	F_1@M	F_1@5	F_1@M	F_1@5	F_1@M	F_1@5	F_1@M
Generative	CatSeq	1.5	3.2	1.6	2.8	2.0	2.8	0.4	0.8
	CatSeqTG	1.5	3.2	1.1	1.8	1.9	2.7	0.5	1.1
	CatSeq(TRM)	1.5	3.1	1.1	1.8	1.9	2.7	0.5	0.9
	CatSeqD	1.5	3.1	1.5	2.4	1.6	2.4	0.6	1.1
	ExHiRD-h	1.6	3.2	–	–	1.7	2.5	1.1	2.2
Integrated	KG-KE-KR-M[3]	–	–	–	–	–	–	–	–
	SEG-NET	2.0	3.8	1.4	2.4	2.1	3.1	0.9	1.4
Joint	UniKeyphrase	**4.7**	**4.7**	**3.6**	**3.7**	**3.0**	**3.2**	**2.9**	**2.9**

Table 2: Results on absent keyphrase prediction.

Model	**Inspec**		**SemEval**		**KP20k**	
	#PK	#AK	#PK	#AK	#PK	#AK
Ground Truth	7.64	2.10	6.28	8.12	3.32	1.93
Transformer	3.17	0.70	3.24	0.67	3.44	0.58
catSeq	3.33	0.58	3.45	0.64	3.70	0.51
catSeqD	3.33	0.58	3.47	0.63	3.74	0.50
catSeqCorr	3.07	0.53	3.15	0.62	**3.36**	0.50
ExHiRD-h	4.00	**1.50**	3.65	0.99	3.97	0.81
SEG-NET	-	-	-	-	3.79	1.14
UniKeyphrase	**5.19**	2.74	**8.15**	**3.04**	6.29	**2.72**

Table 3: Results of average numbers of predicted unique keyphrases. "#PK" and "#AK" are the number of present and absent keyphrases respectively. **Bold** denotes the prediction closest to the ground truth.

Model	Present		Absent	
	F_1@5	F_1@M	F_1@5	F_1@M
UniKeyphrase	41.6	40.9	3.0	3.2
w/o SRL	38.5	37.6	2.9	3.1
w/o BWC	40.0	39.5	2.8	2.8

Table 4: Ablation study on SemEval dataset

the result, we can see that the performance drops both in present keyphrase and absent keyphrase without stacked relation layer. This demonstrates that explicitly modeling the relation between PKE and AKG with stacked relation layer can benefit them effectively.

Effectiveness of bag-of-words constraint: In this setting, we remove our bag-of-words constraint and there is no global constraint for two tasks. The results show a drop in KP performance, indicating that capturing the global constraint of the result by BWC is effective and important for our method.

4.5 Analysis

4.5.1 SRL Analysis

To better understand the SRL module, we analyze the impact of stacked layers and give a visualization of the inner state of SRL.

Analysis of SRL Layer Number: We explore the impact of the stack number of relation network. The comparison of total keyphrase prediction result, which regardless of the present or absent of keyphrases, are shown in Table 5. We can find that setting deeper layers could generally result in better performance when the number of stacked layers is less than three, which proves the effectiveness of stacked layers. It is worth noting that when the number of stacked layers is larger than two, the KP performance drops. We suppose that when the relation network becomes deeper, the over-interaction will lose the diversity of two task representations.

Model	Total F_1@M
UNILM based keyphrase generation	23.7
UniKeyphrase with 0 layer SRL	23.1
UniKeyphrase with 1 layer SRL	25.2
UniKeyphrase with 2 layer SRL	**25.3**
UniKeyphrase with 3 layer SRL	24.5

Table 5: Total keyphrase prediction on SemEval dataset by different setting. UniKeyphrase with 0 layer SRL means UniKeyphrase without SRL module.

Visualization Analysis for SRL: To better understand what the SRL network has learned, we compare the distance between the PKE representation and AKG representation in different settings. In detail, we randomly sample 2000 pairs of PKE representation vector and AKG representation vector on different positions from test data and compute euclidean metric in each pair. As shown in Figure 3, the blue points mean the Euclidean metric between PKE and AKG representation vector without SRL layer, while the yellow points mean the Euclidean metric with SRL layer.

From the Figure 3, we can find that the blue points are under the yellow points, which means the PKE and AKG representation vector without SRL is more similar. In other words, SRL has learned the task-specific representation. Also, the blue points are denser than the yellow points, which

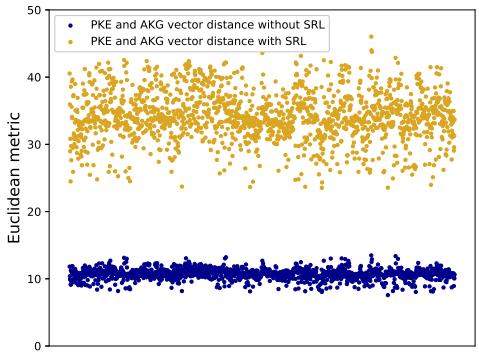

Figure 3: Distance between PKE representation and AKG representation on different settings.

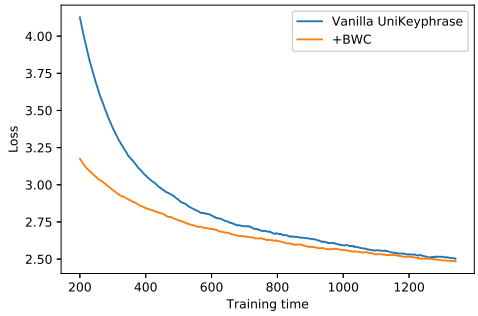

Figure 4: BWC's influence on total training loss (sequence labeling + text generation).

means the PKE and AKG representation with SRL is more diverse than the one without SRL on different samples.

4.5.2 BWC Analysis

Loss Compare: From Figure 4 we can see that the original total loss (labeling and generation) drops more with the help of BWC compared to the vanilla model. BWC actually is an enhancement on the original supervised signal from a global view. It guides the model to learn how many to predict and how to allocate present and absent keyphrases, while original loss only teaches what to predict in each position.

Bag-of-words Error: We also calculate the bag-of-words Error between ground truth and model predicted keyphrases, which is how many tokens are incorrectly predicted. As shown in Figure 5, UniKeyphrase with BWC achieves lower BoW Error compared with the vanilla model. It proves that BWC successfully guides the model to learn a better BoW allocation.

4.5.3 Joint Framework Analysis

In our UniKeyphrase model, we adopt pre-trained model UNILM for KP. So it is necessary to check

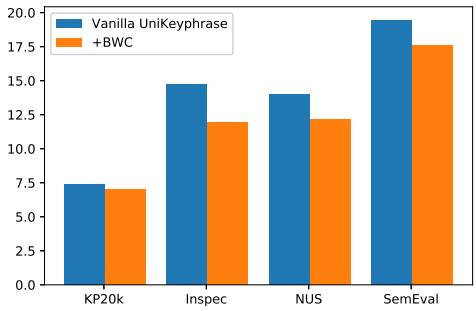

Figure 5: Bag-of-words Error comparison between vanilla and BWC.

that the gain on metrics of our proposed joint framework is not just come from the pre-trained model. In this section, we compare UniKeyphrase with directly using the pre-trained UNILM to perform generative KP.

Specifically, we train a sequence to sequence model for KP based on UNILM. Results are shown in Table 5. From the results, we find that all of the joint models with SRL can further outperform the generative method based on UNILM, demonstrating that the improvement of KP mainly come from our joint framework instead of pre-trained UNILM. We notice that the UniKeyphrase without SRL does not outperform the generative method based on UNILM, which show the significance of modeling the relation between the two sub-tasks in our joint framework.

5 Conclusion and Future Work

This paper focuses on explicitly establishing an end-to-end unified model for PKE and AKG. Specifically, we propose UniKeyphrase, which contains stacked relation layer to model the interaction and relation between the two sub-tasks. In addition, we design a novel bag-of-words constraint for jointly training these two tasks. Experiments on benchmarks show the effectiveness of the proposed model, and more extensive analysis further confirms the correlation between two tasks and reveals that modeling the relation explicitly can boost their performance.

Our UniKeyphrase can be formalized as a unified framework of NLU and NLG tasks. It is easy to transfer it to other extraction-generation NLP tasks. In the future, we will explore to adopt our framework to more scenarios.

Acknowledgments

Lei Li were supported by Beijing Municipal Commission of Science and Technology [grant number Z181100001018035]; Engineering Research Center of Information Networks, Ministry of Education; BUPT Jinan Institute; Beijing BUPT Information Networks Industry Institute Company Limited.

References

Wasi Uddin Ahmad, Xiao Bai, Soomin Lee, and Kai-Wei Chang. 2020. Select, extract and generate: Neural keyphrase generation with syntactic guidance. *arXiv preprint arXiv:2008.01739*.

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.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

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

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

Wang Chen, Hou Pong Chan, Piji Li, Lidong Bing, and Irwin King. 2019a. An integrated approach for keyphrase generation via exploring the power of retrieval and extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2846–2856.

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

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

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13063–13075.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223.

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

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.

Youngsam Kim, Munhyong Kim, Andrew Cattle, Julia Otmakhova, Suzi Park, and Hyopil Shin. 2013. Applying graph-based keyword extraction to document retrieval. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 864–868.

Patrice Lopez and Laurent Romary. 2010. Humb: Automatic key term extraction from scientific articles in grobid. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 248–251.

Shuming Ma, Xu Sun, Yizhong Wang, and Junyang Lin. 2018. Bag-of-words as target for neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 332–338.

Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 1318–1327.

Rui Meng, Xingdi Yuan, Tong Wang, Peter Brusilovsky, Adam Trischler, and Daqing He. 2019. Does order matter? an empirical study on generating multiple keyphrases as a sequence. *arXiv preprint arXiv:1909.03590*.

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

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

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

Ramakanth Pasunuru and Mohit Bansal. 2018. Multi-reward reinforced summarization with saliency and entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 646–653.

Libo Qin, Wanxiang Che, Yangming Li, Mingheng Ni, and Ting Liu. 2020a. Dcr-net: A deep co-interactive relation network for joint dialog act recognition and sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8665–8672.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087, Hong Kong, China. Association for Computational Linguistics.

Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. 2020b. AGIF: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1807–1816, Online. Association for Computational Linguistics.

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*.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *AAAI*, volume 8, pages 855–860.

Lu Wang and Claire Cardie. 2013. Domain-independent abstract generation for focused meeting summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1395–1405.

Yue Wang, Jing Li, Hou Pong Chan, Irwin King, Michael R Lyu, and Shuming Shi. 2019. Topic-aware neural keyphrase generation for social media language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2516–2526.

Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 2005. Kea: Practical automated keyphrase extraction. In *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*, pages 129–152. IGI global.

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

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

Qi Zhang, Yang Wang, Yeyun Gong, and Xuan-Jing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on twitter. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 836–845.

Jing Zhao and Yuxiang Zhang. 2019. Incorporating linguistic constraints into keyphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5224–5233.

A Dataset Statistics

Type	Dataset	#Examples	Max/Avg #Tokens	Max/Avg #Sentences
Test	Inspec	500	387.0 / 138.4	27.0 / 6.7
	NUS	211	384.0 / 185.6	16.0 / 8.4
	SemEval	100	415.0 / 208.0	18 / 8.8
	KP20k	20000	1116.0 / 178.9	70.0 / 8.1
Validation	KP20k	20000	1862.0 / 179.2	120 / 8.2
Train	KP20k	514154	2924 / 177.9	284 / 8.2

Table 6: Summary of the dataset used in experiments. "#Examples" means the number of sample. "#Tokens" means the number of token. "#Sentences" means the number of sentence.

Relevant statistics about the dataset used in this paper is shown in Table 6.

B Experimental Details

The BWC does not bring extra parameters, hence the trainable parameters of UniKeyphrase come from UNILM and SRL. We use the base version of UNILM, which contains about 110M parameters. Follow UNILM, our model is implemented using PyTorch. The learning rate is 1e-5 and the proportion of warmup steps is 0.1. The masking probability of absent keyphrase sequence is 0.7. For the SRL module, dropout is applied to the output of each layer for regularization, the dropout rate is 0.5. In this paper, we try to set the number of layer by 2,3,4 and choose the best based on validation. For all experiments in this paper, we choose the model that performs best on the KP20k validation dataset.

C Preprocess

The input of UniKeyphrase is the same as BERT, which applies wordpiece tokenizer on raw sentences. So we use the "BIXO" labeling method, where B and I stand for Beginning and Inside of a word in keyphrases, and O denotes any token that Outside of any keyphrase. For any sub-word token in keyphrases(which starts with '##' in processed input) we use X to label it. For example, "voip conferencing system" will be tokenized into "v ##oi ##p con ##fer ##encing system" and be labeled as "B X X I X X I". We concatenate all the tokenized absent keyphrases into one sequence using a special delimiter " ; ". An example of absent keyphrase sequence will like "peer to peer ; content delivery ; t ##f ##rc ; ran ##su ##b".

Document: improving reliability of a shared supplier with competition and spillover s . study spillover effect on competing manufacturers incentives to improve a suppliers reliability . develop a two stage model with supplier improvement , random supply and demand , and competition . characterize manufacturers equilibrium inventory decision . characterize sufficient conditions of existence of equilibrium of manufacturers improvement efforts . explore impact of market characteristics on manufacturers improvement efforts .

Present Ground Truth: {spillover, reliability, supplier improvement}

UNILM(Generative): **1.** spillover **2.** competition **3.** supplier improvement

UniKeyphrase: **1.** supplier improvement **2.** inventory **3.** random supply and demand **4.** competition **5.** reliability **6.** spillover **7.** equilibrium **8.** suppliers

Absent Ground Truth: {supply chain management , game theory}

UNILM(Generative): **1.** supply chain management

UniKeyphrase: **1.** supply chain reliability **2.** supply chain management **3.** game theory

Figure 6: Case study.

D Case Study

We give a case on the KP20k testset in Figure 6. For fairness, we compare with keyphrase generation based on UNILM since our joint models are based on its implementation. Blue and red denote correct present and absent keyphrases, respectively. As shown in Figure 6, UniKeyphrase successfully catches the deep semantic relation similar to the case in the introduction, but here are "methods" for the absent and "application" for the present. Having caught this relation, UniKeyphrase gives more accurate results(predicts absent keyphrases "game theory").

E Evaluation Details

We use F_1@5 and F_1@M as evaluation metric. For calculating F_1@5, since there is no explicit rank score for each predicted keyphrase, we calculate the rank score as follows:

Present: we calculate the average predicted label probabilities of all tokens in a keyphrase as the score. We tried several other scoring strategies like max, min, or the first token's probability as the score. The results show no significant difference(less than 0.1%).

Absent: following previous works, we pick up the top 5 keyphrases in sequence order. The 5 leftmost keyphrases in the predicted sequence are selected as the result.