ERU-KG: Efficient Reference-aligned Unsupervised Keyphrase Generation

Anonymous ACL submission

Abstract

Unsupervised keyphrase prediction has gained growing interest in recent years. However, existing methods typically rely on heuristically defined importance scores, which may lead to inaccurate informativeness estimation. In addition, they lack consideration for time efficiency. To solve these problems, we propose ERU-KG, an unsupervised keyphrase generation (UKG) model that consists of a phraseness and an informativeness module. The former generate candidates, while the latter estimate their relevance. The informativeness module innovates by learning to *model informativeness through references* (e.g., queries, citation contexts, and titles) *and at the term-level*, thereby 1) capturing how the key concepts of the document are perceived in different contexts and 2) estimate informativeness of phrases more efficiently by aggregating term informativeness, removing the need for explicit modeling of the candidates. ERU-KG demonstrates its effectiveness on keyphrase generation benchmarks by outperforming unsupervised baselines and achieving on average 89% of the performance of a supervised baseline for top 10 predictions. Additionally, to highlight its practical utility, we evaluate the model on text retrieval tasks and show that keyphrases generated by ERU-KG are effective when employed as query and document expansions. Finally, inference speed tests reveal that ERU-KG is the fastest among baselines of similar model sizes.

1 Introduction

Keyphrases are short sequences of words that describe the core concepts of a document. Automatically predicting keyphrases is a crucial problem, as the outputs can be utilized in various downstream tasks, such as document retrieval (Zhai, 1997; Gutwin et al., 1999; Jones and Staveley, 1999; Witten et al., 2009; Fagan, 2017; Boudin et al., 2020) and document visualization (Chuang et al., 2012). There are two approaches for keyphrase prediction,

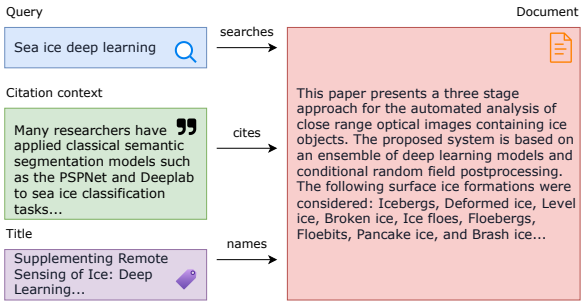

Figure 1: An example of the different type of references.

namely *keyphrase extraction* (KE) and *keyphrase generation* (KG). The two approaches differ in the output space, where keyphrase generation additionally predicts absent keyphrases. Since human tend to use both present and absent keyphrases to describe documents, *keyphrase generation* has received much attention in recent years.

In this work, we focus on *unsupervised keyphrase generation* (UKG). In line with previous work, we target a model that receives a document as input and predicts present and absent keyphrases. The desired UKG model must learn to generate keyphrases without labeled data. Being able to build an UKG model in the unsupervised setting is highly desirable, since labeled data is often expensive and difficult to obtain. In addition, KG models are expected to be used to process large volumes of documents, as evidenced by their potential applications. For example, when utilized for document visualization or retrieval tasks, these models must efficiently handle entire corpora. Therefore, it is desirable for KG models to be *time efficient*, to manage large scale data processing.

There are two challenges of building a keyphrase generation model that meet those requirements. The **first challenge** is ensuring *accurate informativeness estimation*. *Informativeness* refers to how well the phrase illustrates the core concepts of the text. Without labeled keyphrases, it is not straight-

forward to train a model that captures informativeness. Unsupervised approaches, including unsupervised keyphrase extraction (UKE) and generation, rely on heuristically designed importance scores as proxies for estimating informativeness (see Section A). However, since these importance scores are heuristically defined, they may lead to inaccurate estimations.

The **second challenge** is *efficient informativeness estimation*. Existing keyphrase generation methods typically employ a seq2seq approach that directly model the distribution of keyphrases given a document. This could make keyphrase generation slow due to the autoregressive approach taken by most models (Wu et al., 2022b). Existing UKE models, on the other hand, separate candidate phrase generation and informativeness estimation. While candidate generation is typically fast, modern UKE approaches leverage complex importance scoring function that require modeling of a document and all its candidates, potentially slowing down the process. Specifically, embedding-based approaches (Bennani-Smires et al., 2018; Sun et al., 2020) generate embeddings for the given text and all candidates, then measure informativeness via proximity in the embedding space. On the other hand, language model-based approaches (Ding and Luo, 2021; Kong et al., 2023) use pretrained language models (PLMs) to score each document-candidate pair individually.

Our **key idea** for addressing the **first challenge** is learning to *model informativeness through references*. We propose that accurate informativeness estimation can be achieved by capturing *community perception* of a document's key concepts, i.e. the central ideas as recognized by domain experts and readers. This community perception can be learned by analyzing *references* - the different contexts that mention the document. We illustrate this observation in Figure 1, where we consider three types of *references*, including *queries* (how the document is retrieved), *citation contexts* (how the document is cited) and *titles* (how the authors summarize their own work). These references provide insights into of what the community considers the key concepts of the text.

Next, our **key idea** for addressing the **second challenge** is learning to *model informativeness at the term-level* rather than at the phrase-level. Estimating informativeness for each candidate phrase can be computationally expensive and slow down keyphrase generation. Instead, we propose estimating informativeness at the term-level. In particular, we leverage pairs of references and documents to train a term importance predictor, which are used to estimate informativeness of phrases by aggregating informativeness of its constituent terms, removing the need to explicitly model each candidate phrase individually.

We summarize the contributions of our paper. **Firstly**, we propose **ERU-KG**: an **E**fficient, **R**eference-aligned, **U**nsupervised **K**eyphrase **G**eneration model. ERU-KG comprises two components - a *phraseness* and an *informativeness module*. The former generates present and absent keyphrase candidates by extracting noun phrases from the given text and retrieving present keyphrases from textually-similar documents. On the other hand, the informativeness module incorporates our novel key ideas to tackle the identified challenges. **Secondly**, we conduct *groundtruth-based evaluation* and show that ERU-KG outperforms unsupervised baselines and comes very close to CopyRNN (Meng et al., 2017), a supervised model. **Thirdly**, to assess the utility of generated keyphrases, we carry out *retrieval-based evaluation*. Our results show that keyphrases generated by ERU-KG enhance text retrieval performance when employed as query and document expansions. **Finally**, we perform inference speed test to assess the time-efficiency of ERU-KG, showing that our method is faster than existing KE and KG baselines with comparable model sizes.

2 Methodology

Figure 2 presents an overview of ERU-KG. Our proposed model takes as input a document $\boldsymbol{x}$ and outputs sets of present and absent keyphrases $\boldsymbol{Y}_{\boldsymbol{x}}^{\text{present}}$ and $\boldsymbol{Y}_{\boldsymbol{x}}^{\text{absent}}$, each containing k keyphrases. Similar to (Do et al., 2023), ERU-KG consists of two modules, called *phraseness* and *informativeness*. The former is responsible for generating candidates, while the latter decides which best represents the core concepts of the given text.

2.1 Informativeness Module

The informativeness module is responsible for ranking candidate phrases. As mentioned above, it incorporates our key ideas to addressing the challenges of accurate and efficient informativeness estimation: modeling informativeness through references and at the term-level. Specifically, we lever-

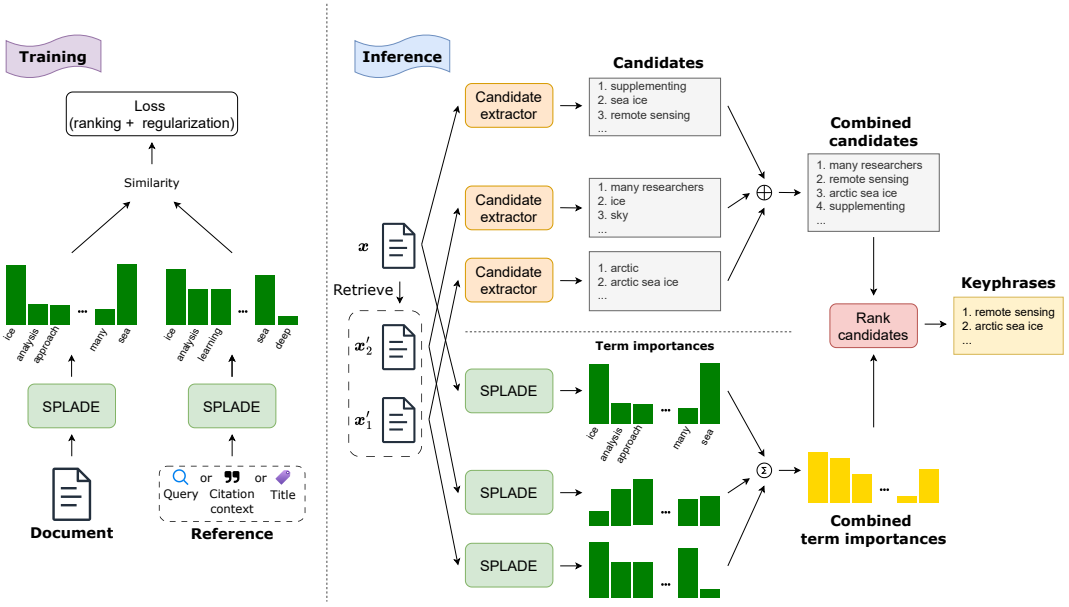

Figure 2: Overview of **ERU-KG**. Further details of the inference process are provided in Algorithm 1

age pairs of references and documents to train a term importance predictor, which is used to estimate informativeness of candidate phrases in the during inference.

There exists multiple term importance predictors in the area of text retrieval. One option is DeepCT (Dai and Callan, 2019), which predicts importances for all terms appearing in a given document. However, since DeepCT is not designed to model importances of absent terms, it is not suitable for evaluating absent candidate phrases. In another line of work, EPIC (MacAvaney et al., 2020), SparTerm (Bai et al., 2020) and SPLADE (Formal et al., 2021b,a) predict importances for all terms in a vocabulary, making them more suitable for evaluation of both present and absent candidates.

Among these models, SPLADE is the most suitable for predicting keyphrases. Different from EPIC and SparTerm, SPLADE employs explicit sparsity regularization mechanisms, which encourages assigning non-zero importance for only the most relevant terms. In the next sections, we discuss the term importance predictor: SPLADE (§2.1.1), training data (§2.1.2) and informativeness estimation (§2.1.3).

2.1.1 Term Importance Predictor: SPLADE

SPLADE predicts term importances based on the logits produced by the Masked Language Modeling (MLM) layer. In particular, $w_{ij}^{\boldsymbol{x}}$ denotes the importance, predicted by MLM layer, of the term i in the input document $\boldsymbol{x}$ and the term j in BERT vocabulary. Then, the importance of j given $\boldsymbol{x}$ is computed by max pooling

$$w_j^{\boldsymbol{x}} = \max_{i \in \boldsymbol{x}} \log(1 + \text{ReLU}(w_{ij}^{\boldsymbol{x}})) \quad (1)$$

Model training. SPLADE is trained by optimizing a ranking loss and two regularization losses

$$\mathcal{L} = \mathcal{L}_{rank-IBN} + \lambda_q \mathcal{L}_{reg}^q + \lambda_d \mathcal{L}_{reg}^d \quad (2)$$

where $\mathcal{L}_{reg}$ is the sparse regularizer introduced in (Paria et al., 2020). Given a training batch, containing the query q_i, the positive (referenced) document d_i^+ and the negative document d_i^-, the ranking loss $\mathcal{L}_{rank-IBN}$ is a contrastive loss that maximizes the relevance of d_i^+, while lowering the relevance of d_i^-. Relevance is measured by dot product between q and d representations from Eq. 1. For further details, we refer readers to the original papers (Formal et al., 2021b,a).

2.1.2 Training Dataset

To train SPLADE, we build a training set $\mathcal{T} = \{(r_i, d_i^+, d_i^-)\}_{i=1}^{|\mathcal{T}|}$, containing triplets, where r_i is a reference, while d_i^+ and d_i^- denote positive (referenced) and negative documents, respectively. We note that references r_i are used in place of queries q_i. In this work, we focus on scientific text, as all three types of references (queries, citation contexts, and titles) are readily accessible in this domain.

Query. Our work leverages training data from the Search task within SciRepEval [1] (Singh et al.,

[1] https://huggingface.co/datasets/allenai/scirepeval/viewer/search

2023), which contains 478,000 queries from real users on Semantic Scholar. Each query accompanies a list of candidates and their relevance score. We build triplets from this dataset by regarding query as reference r_i. We concatenate the title and abstract of each candidate as d_i^+ for those with a relevance score > 0, and as d_i^- for those with a relevance score = 0.

Citation context. We utilize the permissively licensed subset of unarXive[2] (Saier et al., 2023), which contains over 165,000 full-text documents. For each document, we extract citing sentences as references r_i. We employ only citation contexts that cite one paper, or collectively cite several paper as a single group, to ensure focus on the concepts of the referenced document. The concatenated titles and abstracts of cited articles are chosen to be positive documents d_i^+. Negative documents d_i^- are similarly constructed by concatenating titles and abstracts but are selected from research articles cited in different sections of the same paper, distinct from the section containing the citing sentence.

Title. We continue to utilize unarXive dataset (Saier et al., 2023). More specifically, we designate titles as r_i and corresponding abstracts as d_i^+ for research articles. For negative documents d_i^-, we select abstracts of research articles cited within the paper.

2.1.3 Estimating Informativeness of Phrases

In this section, we explain how to measure informativeness of phrases based on the term importance predictor described above. A simple approach is to aggregate the importance of the component terms. More formally, the probability that a candidate phrase $\boldsymbol{c}$ is informative given the document $\boldsymbol{x}$ is defined as

$$P_{\text{in}}(\boldsymbol{c}|\boldsymbol{x}) \propto f(\boldsymbol{c}, \boldsymbol{x}) = \frac{1}{|\boldsymbol{c}| - \gamma} \sum_{c_i \in \boldsymbol{c}} w_{c_i}^{\boldsymbol{x}} \quad (3)$$

where $w_{c_i}^{\boldsymbol{x}}$ is the predicted importance of term $c_i \in \boldsymbol{c}$. Next, γ is the length penalty, which is used to control the preference towards longer candidates. A negative value of γ leads to larger value of $f(\boldsymbol{c}, \boldsymbol{x})$ for longer candidates, and vice versa.

Although SPLADE can evaluate importance of absent terms, the scores for these terms are often underestimated. On a set of 20k documents sampled from SciRepEval Search, only 25% of terms with non-zero importances are absent terms. This could lead to inaccurate ranking of absent candidates. To mitigate this problem, our approach is inspired by pseudo-relevance feedback (Cao et al., 2008), which is to incorporate additional context from related documents. In particular, the importance of each candidate is determined by its importance in the given document $\boldsymbol{x}$ and its related documents $\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})$. Consequently, the informativeness probability is redefined as follows

$$P_{\text{in}}(\boldsymbol{c}|\boldsymbol{x}) \propto \hat{f}(\boldsymbol{c}, \boldsymbol{x}) = \frac{1}{|\boldsymbol{c}| - \gamma} \sum_{c_i \in \boldsymbol{c}} \hat{w}_{c_i}^{\boldsymbol{x}} \quad (4)$$

$$\hat{w}_{c_i}^{\boldsymbol{x}} = \alpha\, w_{c_i}^{\boldsymbol{x}} + (1 - \alpha) \sum_{\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})} \tilde{s}_{\boldsymbol{x}', \boldsymbol{x}}\, w_{c_i}^{\boldsymbol{x}'} \quad (5)$$

Here, $\mathcal{N}(\boldsymbol{x})$ is retrieved using BM25 from a document collection $\mathcal{D}$. The hyperparameter α controls the relative contribution of the given document and its related documents. $\tilde{s}_{\boldsymbol{x}', \boldsymbol{x}} = \frac{s_{\boldsymbol{x}', \boldsymbol{x}}}{\sum_{\boldsymbol{x}'' \in \mathcal{N}(\boldsymbol{x})} s_{\boldsymbol{x}'', \boldsymbol{x}}}$ is the normalized similarity between two documents, where $s_{\boldsymbol{x}', \boldsymbol{x}}$ denotes the BM25 similarity score. It is worth noting that the term importances of the documents in $\mathcal{D}$ are precomputed and therefore no additional computations are required.

2.2 Phraseness Module

The phraseness module is responsible for generating keyphrase candidates, including present and absent ones. A discussion in (Do et al., 2023) mentions that most keyphrases are noun phrases (Chuang et al., 2012) and absent keyphrases can be found in other documents (Ye et al., 2021). Based on this idea, we employ a candidate generation procedure that extract noun phrases from 1) the given document $\boldsymbol{x}$ and 2) its related documents $\mathcal{N}(\boldsymbol{x})$. More formally, given a document, its candidate set $\hat{\boldsymbol{C}}_{\boldsymbol{x}} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ...\}$ containing keyphrase candidates, is obtained as follows

$$\hat{\boldsymbol{C}}_{\boldsymbol{x}} = \boldsymbol{C}_{\boldsymbol{x}} \cup \boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})} = \boldsymbol{C}_{\boldsymbol{x}} \cup \bigcup_{\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})} \boldsymbol{C}_{\boldsymbol{x}'} \quad (6)$$

where $\boldsymbol{C}_{\boldsymbol{x}}$ denotes the set of noun phrases extracted from $\boldsymbol{x}$. $\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}$ denotes the set of noun phrases extracted from $\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})$. To assign a phraseness probability of each candidate $\boldsymbol{c} \in \hat{\boldsymbol{C}}_{\boldsymbol{x}}$, we compute the likelihood of drawing it from the candidate set of the given document ($\boldsymbol{C}_{\boldsymbol{x}}$) or from its related documents ($\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}$)

[2] https://zenodo.org/records/7752615

$$\mathrm{P}_{\text{pn}}(\boldsymbol{c}|\boldsymbol{x}) = \beta\, P(\boldsymbol{c}|\boldsymbol{C_x}) + (1-\beta) \sum_{\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})} \tilde{s}_{\boldsymbol{x}',\boldsymbol{x}}\, P(\boldsymbol{c}|\boldsymbol{C}_{\boldsymbol{x}'}) \tag{7}$$

$$P(\boldsymbol{c}|\boldsymbol{C}) = \begin{cases} \frac{1}{|\boldsymbol{C}|}, & \boldsymbol{c} \in \boldsymbol{C} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

The parameter β controls the contribution from the given document and its related documents.

However, as the size of $\mathcal{N}(\boldsymbol{x})$ grows, the size of $\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}$ (and therefore $\hat{\boldsymbol{C}}_{\boldsymbol{x}}$) may grow significantly. The large number of candidates slows down keyphrase generation process, regardless of how fast informativeness estimation is. To limit the number of candidates for speeding up the KG process, we employ two strategies for pruning $\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}$.

Strategy 1: *Pruning low informativeness and low reliability candidates from each* $\boldsymbol{C}_{\boldsymbol{x}'}$. The informativeness of a candidate given the input document $\boldsymbol{x}$ depends not only on $\boldsymbol{x}$ but also on how important that candidate is to the related documents. Specifically, we can see from Eq. 4 and 5 that unimportant candidates given the related documents are likely to have low informativeness and hence unlikely to be chosen as keyphrases. Based on this idea, we prune $\boldsymbol{C}_{\boldsymbol{x}'}$ by keeping only the top 10 $\boldsymbol{c} \in \boldsymbol{C}_{\boldsymbol{x}'}$ with the highest value of $f(\boldsymbol{c}, \boldsymbol{x}')$ see (Eq. 3).

Next, we further prune $\boldsymbol{C}_{\boldsymbol{x}'}$ based on their *reliability*. Inspired by (Boudin and Aizawa, 2024), we estimate phrase reliability by using the number of documents in which they appear as one of the most informative candidates. Specifically, we employ $\boldsymbol{G}_{\mathcal{D}}$, which is a glossary formed by retaining noun phrases that appear in the top 10 most informative candidates for at least three documents $\boldsymbol{x}' \in \mathcal{D}$. Applying the first strategy, the pruned candidate set from related documents $\boldsymbol{x}'$, denoted as $\tilde{\boldsymbol{C}}_{\boldsymbol{x}'}$, is defined as follows

$$\tilde{\boldsymbol{C}}_{\boldsymbol{x}'} = \text{Top}_{10}(\boldsymbol{C}_{\boldsymbol{x}'}, f) \cap \boldsymbol{G}_{\mathcal{D}} \tag{9}$$

We note that $\tilde{\boldsymbol{C}}_{\boldsymbol{x}'}$ is precomputed for every document in $\mathcal{D}$ and therefore no additional computations are required in the inference phase. The pruned candidate sets $\tilde{\boldsymbol{C}}_{\boldsymbol{x}'}$ are used in place of $\boldsymbol{C}_{\boldsymbol{x}'}$ in Eq. 6 and 7.

Strategy 2: *Pruning low phraseness candidates from* $\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}$. As will be discussed in §2.3, candidates chosen as keyphrases need to exhibit high informativeness, but also phraseness probability. Therefore, candidates with low phraseness are unlikely to be chosen as keyphrases. Based on this idea, we prune $\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}$ by retaining only the top 100 with the highest value of $P_{\text{pn}}(\boldsymbol{c}|\boldsymbol{x})$. Applying the second strategy, the final candidate set is redefined as follows

$$\hat{\boldsymbol{C}}_{\boldsymbol{x}} = \boldsymbol{C}_{\boldsymbol{x}} \cup \text{Top}_{100}(\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}, P_{\text{pn}}) \tag{10}$$

2.3 Combining Phraseness and Informativeness

To generate keyphrases, we combine the two modules. Specifically, given an input text, we first apply the phraseness module to generate keyphrase candidates $\hat{\boldsymbol{C}}_{\boldsymbol{x}}$. Next, we evaluate the informativeness of each candidate. The candidates are ranked based on a composite ranking score, which is computed as the product-of-experts (Hinton, 2002) of the phraseness and informativeness probabilities

$$P_{\text{kp}}(\boldsymbol{c}|\boldsymbol{x}) \propto P_{\text{pn}}(\boldsymbol{c}|\boldsymbol{x})^{\lambda} \times P_{\text{in}}(\boldsymbol{c}|\boldsymbol{x}) \tag{11}$$

where λ is a hyperparameter that controls the importance of phraseness in the ranking score.

Position penalty. Previous work have shown that position information is useful for predicting present keyphrases (i.e. keyphrase extraction) (Florescu and Caragea, 2017; Boudin, 2018; Gallina et al., 2020). Therefore, we include this feature into measuring informativeness of phrases. In particular, we adopt the position penalty defined in (Do et al., 2023). The final ranking score is defined as follows

$$\boldsymbol{s}_{\boldsymbol{x}}(\boldsymbol{c}) = \omega_{\boldsymbol{x}}(\boldsymbol{c}) P_{\text{kp}}(\boldsymbol{c}|\boldsymbol{x}) \tag{12}$$

where $\omega_{\boldsymbol{x}}(\boldsymbol{c}) = 1 + \frac{1}{\log_2[\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{c})+1]}$ is the position penalty. The position $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{c})$ is the number of words preceding the phrase $\boldsymbol{c}$ in $\boldsymbol{x}$. This penalty prioritizes phrases appearing earlier in the text. For absent phrases, we define $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{c}) \to \infty$ and therefore $\omega_{\boldsymbol{x}}(\boldsymbol{c}) \to 1$. Finally, top ranked (present or absent) candidates are chosen as (present or absent) keyphrases.

Switching between generation and extraction. Our proposed framework can flexibly switch between generation and extraction. This is achieved by setting both interpolation hyperparameters, α and β (Eq. 5 and 7, respectively) to 1. Setting these two parameters to 1 disables the use of $\mathcal{N}(\boldsymbol{x})$ and therefore is equivalent to not retrieving any related documents, i.e. $|\mathcal{N}(\boldsymbol{x})| = 0$.

3 Experiments

In this work, we assess the effectiveness of ERU-KG using two evaluation methods: **Ground truth-based** and **Retrieval-based** evaluation. The former measures the alignment between predicted keyphrases and human-annotated keyphrases, while the latter assesses the usefulness of predicted keyphrases when applied to text retrieval tasks. More specifically, retrieval-based evaluation aim to determine if keyphrases effectively serve as query and document expansion to enhance text retrieval performance. The datasets, baselines & evaluation metrics, and experiment results are respectively presented in §3.1, §3.2 and §3.3.

One of the core contributions of this work is that keyphrase generation can be made more *time-efficient* by leveraging term-based representations of documents. To validate this, we conduct **Inference speed** evaluation (§3.4).

3.1 Datasets

We present the statistics for the evaluation datasets in Table 4.

Ground truth-based evaluation. We utilize 5 datasets, namely *SemEval* (Kim et al., 2010), *Inspec* (Hulth, 2003), *NUS* (Nguyen and Kan, 2007), *Krapivin* (Krapivin et al., 2009) and *KP20K* (Meng et al., 2017) for the ground truth-based evaluation of our model. We follow previous work and form the testing document by concatenating the title and abstract of each testing example.

Retrieval-based evaluation. We utilize 6 scientific retrieval datasets. Four of these datasets - *TREC-COVID* (Voorhees et al., 2021), *SCIDOCS* (Cohan et al., 2020), *SciFact* (Wadden et al., 2020) and *NFCorpus* (Boteva et al., 2016) - are sourced from the BEIR benchmark (Thakur et al., 2021). The other two datasets are *DORIS-MAE* (Wang et al., 2024) and *ACM-CR* (Boudin, 2021).

3.2 Baselines & Evaluation Metrics

3.2.1 Baselines

Ground truth-based evaluation. We evaluate our proposed model by comparing against four unsupervised keyphrase extraction algorithms: TextRank (Mihalcea and Tarau, 2004), MultiPartiteRank (Boudin, 2018), EmbedRank (Bennani-Smires et al., 2018), and PromptRank (Kong et al., 2023).

Additionally, we compare our model with three unsupervised keyphrase generation methods: AutoKeyGen (Shen et al., 2022), UOKG (Do et al., 2023) and TPG (zero-shot setting) (Kang and Shin, 2024). Finally, we include CopyRNN (Meng et al., 2017) as a supervised baseline.

Retrieval-based evaluation. We compare ERU-KG with keyphrase generation methods mentioned above. For all keyphrase generation models, we generate keyphrases for each document (or query). We employ the top 10 present keyphrases and top 10 absent keyphrases (20 total) as query and document expansions. In the case of TPG, we evaluate its performance solely on query expansion, due to its slow inference speed.

In addition, we compare our model with well-established methods for document and query expansion, specifically DocT5Query (Nogueira et al., 2019b,a) for document expansion and RM3 (Abdul-Jaleel et al., 2004) for query expansion.

3.2.2 Evaluation Metrics

Ground truth-based evaluation. In line with previous work, we utilize the macro-average F1-score and Recall for evaluation of present and absent keyphrases. For both, we conduct evaluations at top 5 and 10 predictions. Before evaluation, both the predicted and ground truth keyphrases are processed using Porter Stemmer (Porter, 1980), after which duplicates are removed. Our implementation of F1-score is similar to that of (Chan et al., 2019). Specifically, for F1@k we add wrong keyphrases until the number of predictions reaches k if a model predicts fewer than k keyphrases. The purpose of this processing step is to eliminate the favor towards models that produce fewer keyphrases.

Retrieval-based evaluation. We utilize recall at top 1000 (**R@1000**) as the primary evaluation metric, with the aim to assess the effectiveness of generated keyphrases in enhancing the recall of First-stage Retrieval.

3.3 Results

3.3.1 Ground truth-based Evaluation

Table 1 presents the performance of our proposed method and the baselines on the five benchmark datasets. In addition, we report the average performances.

Present keyphrase generation. For generating present keyphrases, our proposed method achieves the best or second-best performance across all datasets except Inspec. While our model does not outperform the baselines on every dataset, it achieves the highest average results overall. Notably, compared to CopyRNN, a supervised base-

Present keyphrase generation												
	SemEval		Inspec		NUS		Krapivin		KP20K		Avg	
	F@5	F@10	F@5	F@10	F@5	F@10	F@5	F@10	F@5	F@10	F@5	F@10
TextRank	16	20.3	29.3	36.2	11.6	16.6	10.1	13.6	9.1	11.6	15.2	19.7
MultiPartiteRank	22.3	22.5	26.3	30.3	23.7	22.2	17.9	15.9	18.4	15.9	21.7	21.4
EmbedRank	23.5	25.2	27.9	33.4	23.8	22.3	18.6	17.7	19.5	16.8	22.7	23.1
EmbedRank (SBERT)	25.4	27.1	**35.1**	**39.8**	22.5	24.1	20.7	19.3	18.3	17.1	24.4	25.5
PromptRank	16.1	19.9	33.4	37.5	18.5	19.8	15.9	15.5	16.3	15.6	20	21.7
AutoKeyGen	22.1	24.4	23.1	23.7	26.1	**27.1**	20.6	18.6	20.4	19	22.5	22.6
UOKG	21.5	22.1	23.9	22.9	27.8	26.2	**21.5**	17.9	21	17.6	23.1	21.3
TPG	24.7	22.2	34	33.3	25	21.3	20.3	16.3	18.7	14.2	24.5	21.5
ERU-KG-small	27.4*	30.1*	28.4	35.7	**28.1***	26.9	20.7	**19.6**	21.6*	19.2	25.2	26.3*
ERU-KG-base	**27.6***	**30.6***	29	36	27.8	27	21.3	19.5*	**22***	**19.4***	**25.5**	**26.5***
Supervised - CopyRNN	29.6	29.7	22.6	23.7	37.2	34.3	30.1	24.5	30.6	25.7	30	27.6
Absent keyphrase generation												
	SemEval		Inspec		NUS		Krapivin		KP20K		Avg	
	R@5	R@10	R@5	R@10	R@5	R@10	R@5	R@10	R@5	R@10	R@5	R@10
AutoKeyGen	0.7	1.1	1.8	2.6	2.3	3.2	2.5	3.7	2.2	3.6	1.9	2.8
UOKG	1.4	2.3	1.9	2.9	2.5	3.6	4.6	**6.9**	2.6	4.5	2.6	4
TPG	0.4	0.8	1.5	2.4	1.7	2.4	1	1.2	1.2	1.9	1.2	1.7
ERU-KG-small	2.1*	**3.1**	**5.4***	**6.5***	**3.7***	**5.9***	**5**	6.2	**6***	8*	**4.4***	**5.9***
ERU-KG-base	**2.3***	3*	5.3*	**6.5***	3.4*	5.5*	4.9	6.2	**6***	**8.1***	**4.4***	5.8*
Supervised - CopyRNN	2.3	2.8	3.5	4.9	5.9	7.8	7.9	10.8	7.1	9.3	5.3	7.1

Table 1: Keyphrase generation performances on five benchmark datasets. The best results are bolded, while the second-best are underlined. Experiments for AutoKeyGen, UOKG, TPG, CopyRNN, and our method are conducted three times, with the mean reported. Both F1 and Recall are presented as percentages. * indicates significance over AutoKeyGen, UOKG and TPG with $p < 0.05$.

line, our model demonstrates competitive results. Specifically, CopyRNN outperforms ERU-KG by only 1.1 percentage point in the overall F1@10 score. This illustrates the effectiveness of our approach, particularly since it is independent of human-labeled keyphrases.

Absent keyphrase generation. For generating absent keyphrases, our model achieves the best performance across all benchmark datasets, leading to the highest average results overall. Furthermore, our approach continues to demonstrate competitive performance in comparison to the supervised baseline.

3.3.2 Retrieval-based Evaluation

Table 2 displays the performance of our model and the baselines on six text retrieval evaluation datasets. In addition, we report average performance across datasets. For KG models, we investigate their effectiveness in three settings: 1) when employed as query expansion (*Query*); 2) when employed as document expansion (*Doc*) and 3) when employed as both query and document expansion (*Both*).

Comparison with KG methods. In the *Query* and *Both* setting, ERU-KG consistently achieve the best performance among existing KG models across datasets, with one exception being the *ACM-CR* dataset, where ERU-KG is second best after CopyRNN. In the *Doc* setting, the performance gain is less consistent. In particular, although our proposed method achieves performance that matches or exceeds the baselines on the majority of datasets, it is outperformed by all baselines on *TREC-COVID* and *DORIS-MAE*.

In addition, it is worth noting that when employed as query and document expansion in conjunction (i.e. *Both* setting), ERU-KG on average results in superior performance comparing to *Query* and *Doc* setting, where query and document expansion are employed individually. This effect is not evident in other KG models.

Comparison with existing expansion methods. ERU-KG achieves performance on par with RM3 in the *Query* setting, DocT5Query in the *Doc* setting, and DocT5Query + RM3 in the *Both* setting. While it does not demonstrate a clear performance advantage over existing expansion methods, it offers a distinct benefit in terms of visualizability. Specifically, the keyphrases generated by ERU-KG are more structured and concise, making them easier to visualize compared to the term-based expansions of RM3 and the synthetic queries produced by DocT5Query.

3.4 Inference Speed Evaluation

We evaluate the inference speed of our method to measure its time efficiency. Throughput (**TP**), defined as the number of documents processed

Type	Model	SCIDOCS	SciFact	TREC-COVID	NFCorpus	DORIS-MAE	ACM-CR	Avg
-	BM25	56.4	97.7	39.6	37	70.1	71.5	62.1
	+ RM3	59	98	44.5	**56.5**	59.6	**74.4**	65.3
	+ AutoKeyGen	52.3	97	33.4	48.7	70.4	69.2	61.8
	+ UOKG	54.2	98	35.4	48.6	69	70.2	62.6
Query	+ TPG	54.1	98.3	34.5	48.1	**73.9**	71	63.3
	+ CopyRNN	53.6	97.7	35.8	48	72.8	73.8	63.6
	+ ERU-KG-small	58.5	99.3	43.7	56.3	**73.9**	72.1	67.3
	+ ERU-KG-base	58.7	99	43.2	54.8	73.4	72.6	67
	+ docT5query	57	98	43.2	37	-	-	-
	+ AutoKeyGen	57	97.3	40.5	37.3	69.8	71.3	62.2
	+ UOKG	57.7	97.7	40.9	37.5	70.1	72.4	62.7
Document	+ CopyRNN	57	97.3	40.8	37.2	69.7	71.6	62.3
	+ ERU-KG-small	59.9	98.3	38.5	39	68.9	73	62.9
	+ ERU-KG-base	60	98.3	39.6	38.7	68	72.7	62.9
	+ docT5query + RM3	59.7	98.3	**47.7**	**56.5**	-	-	-
	+ AutoKeyGen	52.8	97	33.5	48.3	69.3	68	61.5
	+ UOKG	54.8	98.3	36.1	49.2	69.1	69.4	62.8
Both	+ CopyRNN	54.7	97.5	32.4	48.1	72	73.8	63.1
	+ ERU-KG-small	62.4	**100**	46.2	56.2	73.6	72.8	**68.5**
	+ ERU-KG-base	**62.9**	99.7	46.7	55.6	71.7	73.5	68.4

Table 2: Retrieval-based evaluation (**R@1000**) on four benchmark datasets. For each dataset, we **bold** the best overall results and underline the best results in each type (query expansion, document expansion and both).

per second, serves as the primary metric for this assessment. ERU-KG is tested in two scenarios: *keyphrase extraction* and *keyphrase generation*. In the *keyphrase extraction* scenario, we compare ERU-KG (α and β set to 1, as described in §2.3) against EmbedRank and PromptRank, using SBERT in place of Sent2vec for EmbedRank to ensure a fair comparison. For the *keyphrase generation* scenario, we benchmark ERU-KG against the previously mentioned KG baselines, along with an additional baseline, PromptKP (Wu et al., 2022b) — a non-autoregressive supervised keyphrase generation model. Furthermore, we evaluate two configurations of ERU-KG by varying the size of $\mathcal{N}(\boldsymbol{x})$, setting it to 100 (default), 50 and 10. For fair comparison, we run all experiments with batch size of 1, on the same hardware (see §B.4), using a dataset composed of SemEval, Inspec, NUS and Krapivin.

We present the results in Table 3. ERU-KG achieves the best throughput in both scenarios. Results in the keyphrase generation scenario requires further explanations. In the default setting, i.e. $|\mathcal{N}(\boldsymbol{x})| = 100$, our proposed method fails to achieve a clear advantage over all baselines. However, when setting $|\mathcal{N}(\boldsymbol{x})|$ to smaller sizes, e.g. 50 or 10, ERU-KG becomes significantly faster. This shows that the retrieval of related documents is the bottleneck and create a trade-off between effectiveness and efficiency, as will be illustrated in §C.2, retrieving fewer related documents cause the performance to drop.

Scenario	Model name	Note	Model size	TP (doc/s)
Keyphrase extraction	EmbedRank (SBERT)	-	33M	43.5
	PromptRank	-	60M	1.4
	ERU-KG-base	$\alpha = 1, \beta = 1$	66M	**72.9***
Keyphrase generation	AutoKeyGen	-	37M	9.7
	UOKG	-	37M	4.8
	TPG	-	139M	0.8
	CopyRNN	-	37M	11
	PromptKP	-	110M	10.4
	ERU-KG-base	$\lvert\mathcal{N}(\boldsymbol{x})\rvert = 100$	66M	10.9
	ERU-KG-base	$\lvert\mathcal{N}(\boldsymbol{x})\rvert = 50$	66M	12.1*
	ERU-KG-base	$\lvert\mathcal{N}(\boldsymbol{x})\rvert = 10$	66M	**15.5***

Table 3: Throughput (**TP**) of ERU-KG and baselines. We **bold** and underline the highest and second-highest throughput in each scenario. * denotes significance over the second-best baselines with $p < 0.05$, respectively. Statistical significance tests are conducted separately for each scenario.

4 Conclusion

In this paper, we propose ERU-KG, an unsupervised keyphrase generation model that 1) captures how the community perceives key concepts and 2) estimate informativeness of phrases efficiently. Experiments on keyphrase generation benchmarks demonstrate the effectiveness of ERU-KG. We further validate its performance through evaluations from text retrieval perspective. Notably, the inference speed assessment highlights the model's time efficiency, significantly enhancing its potential for real-world applications.

Limitations

In this section, we discuss the limitations of our work. Firstly, we conducted experiments only in the scientific domain, and therefore it is unclear how ERU-KG would perform in other domains.

Secondly, we limited our analysis to only three types of references, which may not encompass all possible types (e.g. Tweets referencing a research article). Including additional type of references could improve the performance of our proposed model. Lastly, the design of our phraseness module does not allow customization for absent keyphrase generation. Specifically, since our phraseness module source (absent) keyphrase candidates from other documents, it lacks the flexibility to adapt to the specific context of the given document.

References

Nasreen Abdul-Jaleel, James Allan, W Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard. *Computer Science Department Faculty Publication Series*, page 189.

Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. 2020. Sparterm: Learning term-based sparse representation for fast text retrieval. *arXiv preprint arXiv:2010.00768*.

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

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38*, pages 716–722. Springer.

Florian Boudin. 2016. pke: an open source python-based keyphrase extraction toolkit. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 69–73, Osaka, Japan. The COLING 2016 Organizing Committee.

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

Florian Boudin. 2021. Acm-cr: A manually annotated test collection for citation recommendation. In *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 280–281. IEEE.

Florian Boudin and Akiko Aizawa. 2024. Unsupervised domain adaptation for keyphrase generation using citation contexts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 598–614, Miami, Florida, USA. Association for Computational Linguistics.

Florian Boudin, Ygor Gallina, and Akiko Aizawa. 2020. Keyphrase generation for scientific document retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1126, Online. Association for Computational Linguistics.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing (IJCNLP)*, pages 543–551.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. 2018. Yake! collection-independent automatic keyword extractor. In *Advances in Information Retrieval: 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings 40*, pages 806–810. Springer.

Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. 2008. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 243–250.

Cornelia Caragea, Florin Adrian Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. 2014. Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1435–1446, Doha, Qatar. Association for Computational Linguistics.

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.

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

Jason Chuang, Christopher D Manning, and Jeffrey Heer. 2012. "without the clutter of unimportant words" descriptive keyphrases for text visualization. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 19(3):1–29.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Zhuyun Dai and Jamie Callan. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval. *arXiv preprint arXiv:1910.10687*.

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

Haoran Ding and Xiao Luo. 2021. AttentionRank: Unsupervised keyphrase extraction using self and cross attentions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1919–1928, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lam Do, Pritom Saha Akash, and Kevin Chen-Chuan Chang. 2023. Unsupervised open-domain keyphrase generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10614–10627, Toronto, Canada. Association for Computational Linguistics.

Joel L Fagan. 2017. Automatic p hrase indexing for document retrieval: an examination of syntactic and non-syntactic methods. In *ACM SIGIR Forum*, volume 51, pages 51–61. ACM New York, NY, USA.

Corina Florescu and Cornelia Caragea. 2017. 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)*, pages 1105–1115, Vancouver, Canada. Association for Computational Linguistics.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021a. Splade v2: Sparse lexical and expansion model for information retrieval. *arXiv preprint*.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021b. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 2288–2292, New York, NY, USA. Association for Computing Machinery.

Ygor Gallina, Florian Boudin, and Béatrice Daille. 2020. Large-scale evaluation of keyphrase extraction models. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, pages 271–278.

Krishna Garg, Jishnu Ray Chowdhury, and Cornelia Caragea. 2022. Keyphrase generation beyond the boundaries of title and abstract. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5809–5821, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sujatha Das Gollapalli and Cornelia Caragea. 2014. Extracting keyphrases from research papers using citation networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28.

Carl Gutwin, Gordon Paynter, Ian Witten, Craig Nevill-Manning, and Eibe Frank. 1999. Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems*, 27(1-2):81–104.

Geoffrey E Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800.

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.

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

Byungha Kang and Youhyun Shin. 2024. Improving low-resource keyphrase generation through unsupervised title phrase generation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8853–8865, Torino, Italia. ELRA and ICCL.

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, Uppsala, Sweden. Association for Computational Linguistics.

Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, and Xiaoyan Bai. 2023. PromptRank: Unsupervised keyphrase extraction using prompt. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9788–9801, Toronto, Canada. Association for Computational Linguistics.

Mikalai Krapivin, Aliaksandr Autaeu, Maurizio Marchese, et al. 2009. Large dataset for keyphrases extraction.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2356–2362.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Expansion via prediction of importance with contextualization. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 1573–1576.

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, Vancouver, Canada. Association for Computational Linguistics.

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, Barcelona, Spain. Association for Computational Linguistics.

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

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019a. From doc2query to doctttttquery. *Online preprint*, 6(2).

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019b. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, Louisiana. Association for Computational Linguistics.

Biswajit Paria, Chih-Kuan Yeh, Ian EH Yen, Ning Xu, Pradeep Ravikumar, and Barnabás Póczos. 2020. Minimizing flops to learn efficient sparse representations. *arXiv preprint arXiv:2004.05665*.

Martin F Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Tarek Saier, Johan Krause, and Michael Färber. 2023. unarXive 2022: All arXiv Publications Pre-Processed for NLP, Including Structured Full-Text and Citation Network. In *2023 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 66–70, Los Alamitos, CA, USA. IEEE Computer Society.

V Sanh. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Xianjie Shen, Yinghan Wang, Rui Meng, and Jingbo Shang. 2022. Unsupervised deep keyphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11303–11311.

Amanpreet Singh, Mike D'Arcy, Arman Cohan, Doug Downey, and Sergey Feldman. 2023. SciRepEval: A multi-format benchmark for scientific document representations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5548–5566, Singapore. Association for Computational Linguistics.

Mingyang Song, Huafeng Liu, Yi Feng, and Liping Jing. 2023. Improving embedding-based unsupervised keyphrase extraction by incorporating structural information. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1041–1048, Toronto, Canada. Association for Computational Linguistics.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Yi Sun, Hangping Qiu, Yu Zheng, Zhongwei Wang, and Chaoran Zhang. 2020. Sifrank: a new baseline for unsupervised keyphrase extraction based on pre-trained language model. *IEEE Access*, 8:10896–10906.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*, volume 54, pages 1–12. ACM New York, NY, USA.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.

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

Jianyou Andre Wang, Kaicheng Wang, Xiaoyue Wang, Prudhviraj Naidu, Leon Bergen, and Ramamohan Paturi. 2024. Scientific document retrieval using multi-level aspect-based queries. *Advances in Neural Information Processing Systems*, 36.

Ian H Witten, David Bainbridge, and David M Nichols. 2009. *How to build a digital library*. Morgan Kaufmann.

Di Wu, Wasi Uddin Ahmad, and Kai-Wei Chang. 2022a. Pre-trained language models for keyphrase generation: A thorough empirical study. *arXiv preprint arXiv:2212.10233*.

Huanqin Wu, Baijiaxin Ma, Wei Liu, Tao Chen, and Dan Nie. 2022b. Fast and constrained absent keyphrase generation by prompt-based learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11495–11503.

Jiacheng Ye, Ruijian Cai, Tao Gui, and Qi Zhang. 2021. Heterogeneous graph neural networks for keyphrase generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2705–2715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chengxiang Zhai. 1997. Fast statistical parsing of noun phrases for document indexing. *arXiv preprint cmp-lg/9702009*.

Linhan Zhang, Qian Chen, Wen Wang, Chong Deng, ShiLiang Zhang, Bing Li, Wei Wang, and Xin Cao. 2022. MDERank: A masked document embedding rank approach for unsupervised keyphrase extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 396–409, Dublin, Ireland. Association for Computational Linguistics.

Groundtruth-based evaluation			
Dataset name	#doc	#kps/doc	%absent
SemEval	100	15.2	59.7
Inspec	500	9.8	22
NUS	211	11.6	49.3
Krapivin	460	5.7	51.2
KP20K	19,987	5.3	44.7
Retrieval-based evaluation			
Dataset name	#Query	#Corpus	Avg D /Q
SCIDOCS	1,000	25,657	4.9
SciFact	300	5,183	1.1
TREC-COVID	50	171,332	493.5
NFCorpus	323	3,633	38.2
DORIS-MAE	100	363,133	109.3
ACM-CR	552	114,882	1.8

Table 4: Statistics of test splits of evaluation datasets.

A Related Work

Unsupervised keyphrase extraction (UKE). UKE focuses on identifying keyphrases within the given text. Previous work typically employ a two-stage procedure: 1) *candidate generation* via ngram or noun phrase extraction; 2) *candidate ranking*, where candidates are ranked based on their informativeness and the top-ranked are selected as keyphrases.

Existing methods in UKE can be classified into four categories, namely *statistics-based*, *graph-based*, *embedding-based* and *language model-based*. These categories are distinguished by the importance scoring functions that are used to estimate informativeness, i.e. how candidates are ranked. *Statistics-based* methods (Sparck Jones, 1972; Campos et al., 2018) utilizes features like word frequency, word position, context diversity, etc. *Graph-based* method (Mihalcea and Tarau, 2004; Wan and Xiao, 2008; Bougouin et al., 2013; Gollapalli and Caragea, 2014; Florescu and Caragea, 2017; Boudin, 2018) rank candidates based on different graph-theoretic measures. Embedding-based methods (Bennani-Smires et al., 2018; Sun et al., 2020; Zhang et al., 2022) select candidates that are closest to the given document in the embedding space. *Language model-based* methods utilize Pretrained Language Models (PLMs) to evaluate the informativeness of phrases. (Ding and Luo, 2021) evaluate local and global importance of a candidate by leveraging self and cross attention, (Kong et al., 2023) estimate informativeness by computing the likelihood of generating the candidate given the input text and a pre-specified prompt.

Unsupervised keyphrase generation (UKG). Different from UKE, UKG focuses on generating both present and absent keyphrases. Similar to UKE methods, UKG models typically rely on importance scores, but these scores are utilized in two distinct ways: 1) to extract silver-labeled data for training seq2seq models or 2) to guide the generation of noun phrases towards those that represent the core concepts.

The first approach is exemplified by AutoKeyGen (Shen et al., 2022) and Title Phrase Generation (TPG) (Kang and Shin, 2024). AutoKeyGen trains a seq2seq model on silver-labeled data, where present keyphrases are sourced directly from the text, and absent keyphrases are synthesized by combining present terms. To select present and

absent keyphrases, AutoKeyGen employ an importance score that combine semantic and lexical similarity between keyphrase candidates and the document. TPG proposes extracting phrases from titles as silver-labeled keyphrases to train a seq2seq model.

The second approach is demonstrated by UOKG (Do et al., 2023). UOKG comprises two modules, named *phraseness* and *informativeness*. The former, a seq2seq model trained to generate noun phrases, generate phrases while the latter, an embedding-based importance scoring function, guide this generation towards phrases that are key. Our proposed method, ERU-KG, follows this second approach.

Generation/Extraction of keyphrases using references. The use of references, particularly citation contexts and titles, has been explored in prior work on keyphrase extraction and generation. CiteTextRank (Gollapalli and Caragea, 2014) proposes a graph-based approach that incorporates citation contexts. (Caragea et al., 2014) employ occurrences of candidates in citation contexts as a feature for supervised keyphrase extraction. (Garg et al., 2022) investigate the use of citation contexts as additional information for supervised keyphrase generation. More recently, (Boudin and Aizawa, 2024) proposes a framework that extracts silver-labeled keyphrases from citation contexts for domain adaptation. TG-Net (Chen et al., 2019) leverages titles to enhance input text encodings for supervised keyphrase generation. Recently, (Kang and Shin, 2024) propose TPG as an unsupervised pretraining objective, where the resulting pretrained model can be viewed as an UKG model.

Our proposed approach differs from the existing work. Specifically, our approach leverage references to learn document representations, which are used to generate keyphrases that aligned with the key concepts as recognized by the community. In contrast, existing work typically use references 1) for mining silver-labeled keyphrases or 2) as additional information to enhance the keyphrase extraction/generation process.

Time-efficiency in keyphrase extraction and generation. Efficient processing of large document collections is critical for the practicality of keyphrase extraction and generation models. Despite this, time-efficiency has been underdiscussed in the design of modern keyphrase extraction and generation methods. One notable exception is the work by (Wu et al., 2022b), which employs a non-autoregressive decoding strategy to significantly enhance the speed of keyphrase generation compared to autoregressive approaches. Additionally, (Wu et al., 2022a) shows that prioritizing model depth over width and using deep encoders with shallow decoders has been shown to improve inference latency while maintaining accuracy.

B Implementation Details

B.1 ERU-KG

Informativeness module. We employ SPLADE as our term-importance predictor, as mentioned above. We initialized the models with DistilBERT-base[3] (66M parameters) (Sanh, 2019) for ERU-KG-base and a BERT_L-6_H-512_A-8[4] (33M parameters), which is a BERT (Devlin et al., 2019) with 6 layers, model dimensionality of 512 and 8 attention heads, for ERU-KG-small. Models are trained with the ADAM optimizer, with a learning rate of $2e^{-5}$, a warmup of 20000 steps and a batch size of 32. The models are trained for 100k steps. For FLOPS regularization, we set $\lambda_q = 0.05$ and $\lambda_d = 0.03$. We set the length penalty parameter, as mentioned in Eq. 3 and 4, $\gamma = -0.25$.

Unless specified otherwise, the two interpolation weights α, β (Eq. 4 and 7 respectively), are both set to 0.8. In addition, the balancing parameter λ in Eq. 11, is set to 1.5.

Phraseness module. We employ NLTK's (Bird and Loper, 2004) RegexpParser and extract noun phrases from document with the following grammar

$$(< NN.* | JJ.* > + < NN.* | CD >) | < NN.* >$$

For finding the set of neighbor documents $\mathcal{N}(\boldsymbol{x})$ of the input text $\boldsymbol{x}$, we build BM25 retrievers using the document collection $\mathcal{D}$. In particular, $\mathcal{D}$ is the 630,749 documents from the evaluation and validation split of SciRepEval-Search[5] dataset, alongside with their top 10 present keyphrases and predicted term-importances. We build our retrievers using Pyserini (Lin et al., 2021). In the inference phase, we set $|\mathcal{N}(\boldsymbol{x})| = 100$, unless specified otherwise.

[3] https://huggingface.co/distilbert/distilbert-base-uncased

[4] https://huggingface.co/google/bert_uncased_L-6_H-512_A-8

[5] https://huggingface.co/datasets/allenai/scirepeval/viewer/search

B.2 Keyphrase Generation/Extraction Baselines

For TextRank and MultiPartiteRank, we use the pke package (Boudin, 2016). EmbedRank is implemented following the description in (Bennani-Smires et al., 2018), with the exception that we employ the same noun phrase extractor described in B.1. For EmbedRank, we employ both Sent2Vec (sent2vec_wiki_unigrams[6]) (Pagliardini et al., 2018), as in the original paper, and SBERT (all-MiniLM-L12-v2[7]) (Reimers and Gurevych, 2019). For PromptRank (Kong et al., 2023), we adopt the official implementation[8].

For AutoKeyGen, UOKG, and CopyRNN, we use the implementations and checkpoints provided by the authors of (Do et al., 2023). Finally, for TPG[9] (Kang and Shin, 2024) and PromptKP[10] (Wu et al., 2022b), we utilize the official implementation.

B.3 RM3 and DocT5Query

For DocT5Query, we utilized the pre-generated queries provided for the datasets within the BEIR benchmark. For RM3, we leveraged Pyserini's (Lin et al., 2021) implementation[11] and utilize the default hyperparameters.

B.4 Computing Infrastructure

We run all our experiments on a server with two AMD EPYC 7302 3GHz CPUs, three NVIDIA Ampere A40 GPUs (300W, 48GB VRAM each), and 256 GB of RAM.

C Ablation Studies

We conduct two ablation studies to understand 1) how different of references (queries, citation contexts and titles) contribute to ERU-KG performance and 2) how retrieving fewer related documents affect our proposed model's performance. In this section, we conduct the experiments on ERU-KG-base, i.e. the version of ERU-KG with informativeness module initialized from DistilBERT-base.

[6] https://github.com/epfml/sent2vec

[7] https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2

[8] https://github.com/NKU-HLT/PromptRank

[9] https://github.com/kangnlp/low-resource-kpgen-through-TPG

[10] https://github.com/m1594730237/FastAndConstrainedKeyphrase

[11] https://github.com/castorini/pyserini

$\lvert\mathcal{N}(\boldsymbol{x})\rvert$	KG-present (F1@10)	KG-absent (R@10)	TR (R@1000)
100	26.5	5.8	68.4
50	26.5	5.5↓	67
10	26.4	4.5↓	63.1↓

Table 5: The performance change when adjusting the size of related documents set $\mathcal{N}(\boldsymbol{x})$, ↓ denotes performance drop larger than 5% in comparison to default setting ($|\mathcal{N}(\boldsymbol{x})| = 100$).

C.1 Contribution of Each Type of References

We study the contribution of each type of references by excluding one type at a time to train variations of ERU-KG. We evaluate the performance change in *keyphrase generation* tasks (F1@10 and R@10 for present and absent keyphrases respectively), *text retrieval* tasks (Recall@1k). We evaluate *text retrieval* in the *Both* setting, where generated keyphrases are used as both query and document expansion. We average the evaluate results across all datasets for each task to measure performance changes. We present the results in Figure 3.

For present keyphrase generation (keyphrase extraction), removing title from the training dataset effect performance the most. This suggest that title is a great source of information for enhancing keyphrase extraction, aligning with previous work (Chen et al., 2019; Song et al., 2023). Regarding absent keyphrase generation, performance decreases when any reference type is removed, suggesting that this task is benefitted by understanding how the given document would be mentioned in different contexts. The same comment can be made for text retrieval, where removing any reference type hurt performance.

C.2 Effect of Retrieving Fewer Related Documents

We study ERU-KG's performance change as it retrieve fewer related documents $\mathcal{N}(\boldsymbol{x})$. Table 5 presents the results.

It can be seen that retrieving fewer related documents only affect absent keyphrase generation and text retrieval. Next, we can see that performance gradually decrease as the fewer related documents are retrieved. Notably, when $|\mathcal{N}(\boldsymbol{x})| = 10$ the performance drop exceeds 5% for both absent keyphrase generation and text retrieval. Combining the results with Table 3, $|\mathcal{N}(\boldsymbol{x})| = 50$ appears to strike a good balance between efficiency and

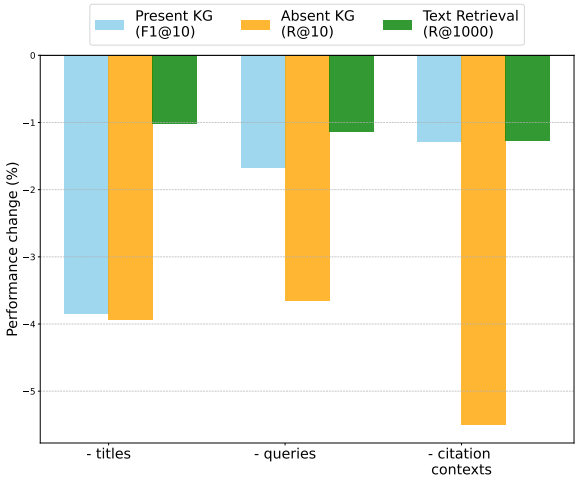

Figure 3: The performance change (in percentage) when excluding one reference type. **-[type]** indicates the omision of **[type]**

effectiveness.

D Case Study

To gain further insights into ERU-KG's effectiveness, we display the keyphrases generated by ERU-KG and the baselines on two types of text, namely *document* and *query*, in Table 6 and Table 7, respectively. For *document*, we use the same example document as in Figure 1. For query, we provide two examples, a long multi-aspected query from DORIS-MAE dataset and a short query from SCIDOCS.

Document. Upon initial examination, there appear to be no significant differences in the predicted present keyphrases across methods, as they all reflect concepts used in reference to the given document. However, considering absent keyphrases, ERU-KG produces keyphrases that are more relevant. Specifically, ERU-KG is able to predict "sea ice classification", "sea ice concentration" and "sea ice detection", which are not only used later in the main body of the given paper, but also used in a citation context ("sea ice classification" is used in the second citation context in Table 6)

Query. It can be seen that keyphrases generated by ERU-KG might be more beneficial as additional information. In the first example, ERU-KG is the only model that can produce the name of alternative GAN techniques (e.g. "ac gan", "am gan", "net gan" and "conditional gan"). Moreover, the introduction of phrases such as "image generation" and "synthetic data" is also suitable for the objective of the user.

In the second example, ERU-KG is the only KG model that manages to generate "brain computer interface" - the full-form version of "BCI". In addition, other absent phrases predicted by ERU-KG, e.g. "domain adaptation", "meta learning", are also highly relevant. On the other hand, it can be seen that absent keyphrases generated by other KG methods do not offer as much valuable additional information. In particular, AutoKeyGen and UOKG produces absent keyphrases that are oftenly reorderings of present terms, while CopyRNN introduces irrelevant keyphrases, such as "world wide web".

E Algorithm Descriptions of ERU-KG

We provide an algorithm description of the inference process of ERU-KG in Algorithm 1.

Algorithm 1: ERU-KG inference

Input: Document $\boldsymbol{x}$, number of output keyphrases k

Output: Sets of present and absent keyphrases $\boldsymbol{Y}_{\boldsymbol{x}}^{\text{present}}$ and $\boldsymbol{Y}_{\boldsymbol{x}}^{\text{absent}}$, each containing k keyphrases

Phraseness module

$\mathcal{N}(\boldsymbol{x}), \{\tilde{s}_{\boldsymbol{x},\boldsymbol{x}'} \mid \boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})\} \leftarrow \text{BM25Retrieve}(\text{query} = \boldsymbol{x}, \text{numdocs} = 100)$ `// Retrieve similar documents and the similarity scores`

$\boldsymbol{C}_{\boldsymbol{x}} \leftarrow \text{NounphraseExtract}(\boldsymbol{x})$

$\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})} \leftarrow \{\}$

foreach $\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})$ **do**

  $\tilde{\boldsymbol{C}}_{\boldsymbol{x}'} \leftarrow \text{GetPrecomputedCandidate}(\boldsymbol{x}')$

  $\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})} \leftarrow \boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})} \cup \tilde{\boldsymbol{C}}_{\boldsymbol{x}'}$

$\hat{\boldsymbol{C}}_{\boldsymbol{x}} \leftarrow \boldsymbol{C}_{\boldsymbol{x}} \cup \text{Top}_{100}(\boldsymbol{C}_{\mathcal{N}(\boldsymbol{x})}, P_{\text{pn}})$

Informativeness module

$w^{\boldsymbol{x}} = \{w_j^{\boldsymbol{x}}\}_{j \in V} \leftarrow \text{SPLADE}(\boldsymbol{x})$ `// Term importances given` $\boldsymbol{x}$`.` V `denotes BERT's vocabulary`

foreach $\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})$ **do**

  $w^{\boldsymbol{x}'} = \{w_j^{\boldsymbol{x}'}\}_{j \in V} \leftarrow \text{SPLADE}(\boldsymbol{x}')$ `// Precomputed`

foreach $j \in V$ **do**

  $\hat{w}_j^{\boldsymbol{x}} \leftarrow \alpha\, w_j^{\boldsymbol{x}} + (1-\alpha) \sum_{\boldsymbol{x}' \in \mathcal{N}(\boldsymbol{x})} \tilde{s}_{\boldsymbol{x},\boldsymbol{x}'}\, w_j^{\boldsymbol{x}'}$

foreach $\boldsymbol{c} \in \hat{\boldsymbol{C}}_{\boldsymbol{x}}$ **do**

  $\hat{f}(\boldsymbol{c}, \boldsymbol{x}) \leftarrow \frac{1}{|\boldsymbol{c}|^{-\gamma}} \sum_{i=1}^{|\boldsymbol{c}|} \hat{w}_{\boldsymbol{x}}(c_i)$

Combining phraseness and informativeness

foreach $\boldsymbol{c} \in \hat{\boldsymbol{C}}_{\boldsymbol{x}}$ **do**

  $P_{\text{in}}(\boldsymbol{c}|\boldsymbol{x}) \leftarrow \hat{f}(\boldsymbol{c}, \boldsymbol{x}) / \sum_{\boldsymbol{c}' \in \hat{\boldsymbol{C}}_{\boldsymbol{x}}} \hat{f}(\boldsymbol{c}', \boldsymbol{x})$ `// Since the final score is only used for ranking, we skip this normalization step in practice and directly set` $P_{\text{in}}(\boldsymbol{c}|\boldsymbol{x}) \leftarrow f_{\boldsymbol{x}}^{\text{in}}(\boldsymbol{c})$

  $P_{\text{kp}}(\boldsymbol{c}|\boldsymbol{x}) \leftarrow P_{\text{pn}}(\boldsymbol{c}|\boldsymbol{x})^{\lambda} \times P_{\text{in}}(\boldsymbol{c}|\boldsymbol{x})$ `// Keyphrase distribution given` $\boldsymbol{x}$`.` $P_{\text{kp}}(\boldsymbol{c}|\boldsymbol{x})$ `is also not normalized since we only use it for ranking`

  $\boldsymbol{s}_{\boldsymbol{x}}(\boldsymbol{c}) \leftarrow \omega_{\boldsymbol{x}}(\boldsymbol{c}) P_{\text{kp}}(\boldsymbol{c}|\boldsymbol{x})$ `// Apply position penalty`

$\boldsymbol{Y} \leftarrow \texttt{sorted}(\hat{\boldsymbol{C}}_{\boldsymbol{x}}, \texttt{sortby} = \boldsymbol{s}_{\boldsymbol{x}}(\boldsymbol{c}), \texttt{descending=True})$

$\boldsymbol{Y}_{\boldsymbol{x}}^{\text{present}} = \{\boldsymbol{y} \in \boldsymbol{Y} \mid \boldsymbol{y} \in \boldsymbol{x}\}[:k]$

$\boldsymbol{Y}_{\boldsymbol{x}}^{\text{absent}} = \{\boldsymbol{y} \in \boldsymbol{Y} \mid \boldsymbol{y} \notin \boldsymbol{x}\}[:k]$

Document	
[DOI: 10.1109/JSEN.2021.3084556] Supplementing Remote Sensing of Ice: Deep Learning-Based Image Segmentation System for Automatic Detection and Localization of Sea-ice Formations From Close-Range Optical Images This paper presents a three-stage approach for the automated analysis of close-range optical images containing ice objects. The proposed system is based on an ensemble of deep learning models and conditional random field postprocessing. The following surface ice formations were considered: Icebergs, Deformed ice, Level ice, Broken ice, Ice floes, Floebergs, Floebits, Pancake ice, and Brash ice. Additionally, five non-surface ice categories were considered: Sky, Open water, Shore, Underwater ice, and Melt ponds. To find input parameters for the approach, the performance of 12 different neural network architectures was explored and evaluated using a 5-fold cross-validation scheme...	
Query	sea ice deep learning
Citation context	1) ...some literatures have utilized real-time ice monitoring using aerial images captured by cameras onboard icebreakers... 2) Many researchers have applied classical semantic segmentation models such as the PSPNet and Deeplab to sea ice classification tasks...
Title	Supplementing Remote Sensing of Ice: Deep Learning-Based Image Segmentation System for Automatic Detection and Localization of Sea-ice Formations From Close-Range Optical Images
ERU-KG	**present**: ice, sea ice, sky, remote sensing, ice floes, underwater ice, sea ice formations, close-range optical images, level ice, brash ice **absent**: sea ice detection, sea ice classification, sea ice concentration, arctic sea ice, antarctic ice sheet, ice sheet, sea ice extent, sea ice image classification, arctic ocean, greenland ice sheet
AutoKeyGen	**present**: ice, ice formations, optical sensors, image segmentation system, image segmentation, approach, optical images, ice floe, floe, deformed ice **absent**: image segmentation approach, image segmentation process, neural network approach, neural networks models, neural network analysis, neural network parameters, convolutional neural networks, neural networks model, segmentation approach, image segmentation techniques
UOKG	**present**: ice, ice formations, neural network architectures, ice objects, deformed ice, surface ice, ice floes, water ice, neural networks, brash ice **absent**: ice field, ice flow, optical flow, ice surface, input data, ice melt, ice sheet, ice shelf, automated approach, satellite images
CopyRNN	**present**: conditional random field, neural networks, random field, image segmentation, remote sensing, neural network, deep learning, ice, ice formations, pancake ice **absent**: deep neural networks, deep neural network, convolutional neural networks, random field neural networks, convolutional neural network, optical ice, random field neural network, ensemble learning, conditional random fields, underwater optical ice
Indexed terms	image segmentation approach, image segmentation process, neural network approach, neural networks models, neural network analysis, neural network parameters, convolutional neural networks, neural networks model, segmentation approach, image segmentation technique

Table 6: Generated keyphrases for an example document, by our proposed model and the baselines. We illustrate the top 10 present and absent keyphrases. In addition, we provide the paper's indexed terms, as well as references of each type (i.e. query, citation context and title) that mentions the given paper.

Query	
[Source: DORIS-MAE] I am seeking alternatives to Generative Adversarial Networks (GANs) that can be applied to image datasets, such as CIFAR-10. The alternative should be capable of generating new data points based on the original data distribution and should perform comparably to GANs across various metrics. Could you provide information on the standard metrics typically used to evaluate the performance of GANs? I anticipate that this alternative method would initially estimate and model the original data distribution, possibly using a neural network, and then generate diverse data points that adhere to the same distribution through an intelligent sampling technique. However, I am open to learning about other promising approaches as well.	
ERU-KG	**present**: alternatives, gans, gan, new data points, alternative, cifar-10, diverse data points, intelligent sampling technique, various metrics **absent**: generation, image generation, gan training, ac gan, data augmentation, synthetic data, am gan, text generation, conditional gan, net gan
AutoKeyGen	**present**: data distribution, original data distribution, original data, data points, data, image data, gans, new data, neural network, standard metrics **absent**: alternative metrics, original data points, standard data, distribution data, data distribution networks, other data points, data networks, various data, neural data, network data
UOKG	**present**: data points, data distribution, image datasets, diverse data points, original data, new data, alternatives, neural network, standard metrics, data **absent**: diverse data sources, multiple data sources, original data set, different data sources, data sampling, various data sources, other data sources, time-series data, open datasets, neural networks
CopyRNN	**present**: neural network, image data, gans, data distribution, sampling, data, image, cifar-10, sampling technique, metrics **absent**: neural networks, data mining, image data mining, generative model, generative neural networks, intelligent image data, artificial neural networks, adversarial neural networks, open neural networks, open data
[Source: SCIDOCS] Real World BCI: Cross-Domain Learning and Practical Applications	
ERU-KG	**present**: real world bci, cross-domain learning, practical applications, bci, domain, rl **absent**: domain adaptation, source domain, target domain, cross domain recommendation, bcis, eeg, cross domain, brain computer interface, cross domain transfer, domain shift
AutoKeyGen	**present**: practical applications, cross-domain learning, real world, real world bci, bci, practical application, learning, world bci, applications, practical **absent**: practical learning, real world applications, learning applications, bci applications, learning models, learning system, learning method, learning model, learning methods, practical systems
UOKG	**present**: real world bci, real world, world bci, practical applications, cross-domain learning, world, real, bci, applications, learning **absent**: real world applications, bci applications, practical learning, learning applications, real applications, real world practical applications, practical real world applications, practical learning applications, real world learning applications, practical world bci applications
CopyRNN	**present**: cross-domain learning, bci, learning, applications, practical, cross-domain, real world, real **absent**: world wide web, cross-domain world wide web, real world wide web, learning world wide web, learning applications, cross-domain applications, support vector machines, real time, finite element method

Table 7: Generated keyphrases for two example queries, by our proposed model and the baselines. We illustrate the top 10 present and absent keyphrases.