SEKE: Specialised Experts for Keyword Extraction

Matej Martinc and **Hanh Thi Hong Tran** and **Senja Pollak** and **Boshko Koloski**
Jožef Stefan Institute, Ljubljana, Slovenia
{name}.{surname}@ijs.si

Abstract

Keyword extraction involves identifying the most descriptive words in a document, allowing automatic categorisation and summarisation of large quantities of diverse textual data. Relying on the insight that real-world keyword detection often requires handling of diverse content, we propose a novel supervised keyword extraction approach based on the mixture of experts (MoE) technique. MoE uses a learnable routing sub-network to direct information to specialised experts, allowing them to specialise in distinct regions of the input space. SEKE, a mixture of Specialised Experts for supervised Keyword Extraction, uses DeBERTa as the backbone model and builds on the MoE framework, where experts attend to each token, by integrating it with a bidirectional Long short-term memory (BiLSTM) network, to allow successful extraction even on smaller corpora, where specialisation is harder due to lack of training data. The MoE framework also provides an insight into inner workings of individual experts, enhancing the explainability of the approach. We benchmark SEKE on multiple English datasets, achieving state-of-the-art performance compared to strong supervised and unsupervised baselines. Our analysis reveals that depending on data size and type, experts specialise in distinct syntactic and semantic components, such as punctuation, stopwords, parts-of-speech, or named entities. Code is available at https://github.com/matejMartinc/SEKE_keyword_extraction.

1 Introduction

Keyword extraction, i.e., extraction of words that represent crucial semantic aspects of the text, is crucial for efficiently summarizing, indexing, and retrieving relevant information from large textual corpora, making it one of the most fundamental NLP tasks (Song et al., 2023). While the first automated solutions have been proposed more than two decades ago (Tumey, 1999; Mihalcea and Tarau, 2004; Hulth, 2003), the task is far from solved, and the scientific community is still actively trying to improve the performance of the keyword extractors (see Section 2 for details).

Recently, the so-called foundation models have completely revolutionised the way natural language processing (NLP) is done, and the development of large language models (LLMs) has enabled task-agnostic architectures to solve downstream NLP tasks in a zero-shot fashion without the need for labelled data (Brown et al., 2020). While recent related studies (Martínez-Cruz et al., 2024; Luo et al., 2024) and experiments conducted in this study show that LLMs' keyword extraction performance is still not comparable to specialised supervised approaches, their zero-shot performance nevertheless remains impressive. It was made possible, to a large extent, by the integration of novel training regimes and architectural components, such as Low-Rank Adaptation (LoRA) (Hu et al., 2021) and Mixture of Experts (MoE) (Jacobs et al., 1991), which make models more efficient, while requiring fewer computational resources. While these components have been successfully employed in an unsupervised language modelling setting with abundant data, there are less studies exploring their effectiveness in a supervised setting with much fewer resources. The main reason for this is that these components were developed with a specific objective of improving the scalability of the models (Shazeer et al., 2017) and therefore their effect on the performance is somewhat neglected or measured only indirectly, that is, studies focus on how the use of these components allows for greater scalability, which in turn leads to performance gains.

In contrast, in this study, we focus on the performance improvements obtained from these components without increasing the size of the backbone model. The hypothesis we want to test is whether it is beneficial for keyword extraction to allow specific parts of the network to specialise for specific

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 14191–14205

types of tokens. Since the "keywordiness" of a token is to a large extent determined by its specific contextual role, we explore if introducing a gating network that assigns different parts of the sequence to specialised layers, which would only attend to specific subsequences and tokens according to their semantic and grammatical role, could improve the current state-of-the-art (SOTA) in the field. In addition, we explore what type of specialisation different experts undertake and how much data is required for a successful specialisation. We test our approach on several datasets with different amounts of training data available, to determine the data threshold that still allows the convergence of expert layers. Finally, we examine whether synergy can be achieved between MoE and bidirectional Long short-term memory (BiLSTM) network that would lower this threshold and improve overall performance by amplifying the sequential information lost by the context slicing caused by the MoE strategy. The contributions of the paper are as follows.

- We propose a novel token classification head architecture (see Figure 1) that combines MoE and BiLSTM. The approach outperforms other SOTA keyword extraction approaches on most datasets. As far as we know, this is the first research that studies the usage of MoE in a supervised sequence-labelling scenario.

- We demonstrate that experts specialise in different syntactic and semantic structures, ranging from punctuation characters and part-of-speech tags to specific words and named entities, depending on the domain they are trained on and the size of the training data.

- We show that adding MoE layers during fine-tuning generally has a positive effect on the model's performance no matter the dataset, but is somewhat dependent on the amount of data available. In addition, we show that the data requirements can be lowered by introducing additional BiLSTM layers.

2 Related Work

2.1 Mixture of Experts

The idea for Mixture of experts (MoE) comes from the study by Jacobs et al. (1991). The concept involved a supervised procedure for a system composed of separate networks, each responsible for a

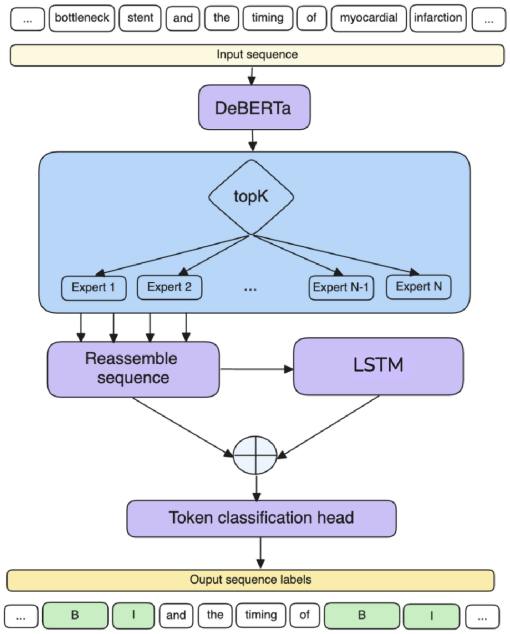

Figure 1: The general architecture given the input.

different subset of training cases. Each network, or expert, specialises in a distinct region of the input space. The choice of expert is managed by a gating network, which determines the weights for each expert. Both the experts and the gating network are trained during the training process.

More recently, the study by Eigen et al. (2013) proposed integrating MoE as components within deep neural networks. This allowed MoE to function as layers in a multilayer network, enabling models to be both large and efficient, which led to the employment of the MoE method in the NLP setting. The study by Shazeer et al. (2017) proposed a 137-billion-parameter LSTM model (the prevalent NLP architecture for language modelling at the time) by introducing sparsity, which allowed for very fast inference even at large scales. This work, which focused on translation, encountered several challenges such as high communication costs and training instabilities. After the success of the Mixtral 8x7B (Jiang et al., 2024), MoE have been used in several LLMs because they enable more efficient pre-training with fewer computing resources, allowing greater scalability in terms of model and data set size. The studies show that a MoE model achieves the same quality as its dense counterpart much faster during pre-training, leading to its usage in training multi-trillion parameter models (Fedus et al., 2022).

Although several studies showed that MoE are an efficient strategy to achieve greater scalability, very few studies (if any) focused on performance

gains that could be obtained by this method. To our knowledge, no study to date explored how MoE can benefit supervised keyword extraction (or in fact any sequence labelling) tasks.

2.2 Keyword Extraction

Contemporary studies on keyword extraction treat it either as a text generation or a sequence labelling task. Yuan et al. (2020) proposed an encoder-decoder RNN architecture featuring two mechanisms, *semantic coverage* and *orthogonal regularisation*, which treated keyword extraction as text generation. These mechanisms ensure that the representation of a generated keyword sequence is semantically aligned with the overall meaning of the source text. In contrast, several recent sequence labelling approaches tackle the task with transformers. Sahrawat et al. (2020) tested several transformer architectures, namely BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and GPT-2, and two distinct sequence labelling heads, a BiLSTM and a BiLSTM head with an additional Conditional random fields layer (BiLSTM-CRF). The TNT-KID approach (Martinc et al., 2022) on the other hand adapted the transformer architecture specifically for keyword extraction tasks by re-parameterizing the attention mechanism to focus more on positional information. Finally, Sun et al. (2021) proposed JointKPE, an open domain keyphrase extraction (KPE) architecture also based on pre-trained transformer language models. It ranks keyphrases by assessing their informativeness across the entire document and is simultaneously trained on a keyphrase chunking task to ensure the phraseness of the keyphrase candidates.

Just recently, Luo et al. (2024) proposed Diff-KPE, a text diffusion approach to generate enhanced keyphrase representations utilizing the supervised Variational Information Bottleneck (VIB). Diff-KPE generates keyphrase embeddings conditioned on the entire document using BERT and then injects the generated keyphrase embeddings into each phrase representation. Another recent approach is HybridMatch (Song et al., 2024), which consists of the representation-focused Siamese encoder component and the interaction-based component that estimates relatedness between candidate phrases and the corresponding document.

Keyword extraction can also be tackled in an unsupervised manner. In general, these approaches can be divided into four main categories, namely statistical (e.g., YAKE (Campos et al., 2020)), graph-based (e.g., TextRank (Mihalcea and Tarau, 2004) and RaKUn (Škrlj et al., 2019)), embeddings-based (e.g., Key2Vec (Mahata et al., 2018)), and language model-based methods. Notably, large language model-based methods tend to be highly effective, driven by the rise of LLMs, such as ChatGPT (Brown et al., 2020). These models have been employed in several recent keyword extraction studies (Martínez-Cruz et al., 2024; Luo et al., 2024; Lee et al., 2023; Peng et al., 2024), demonstrating promising results due to their zero-shot capabilities, which allow them to perform tasks without task-specific fine-tuning (Kojima et al., 2022). However, while significantly outperforming other unsupervised approaches, these models still do not offer performance comparable to the SOTA supervised methods. To some extent, this might be related to the deficiency observed by Zhang et al. (2024), which has highlighted the theoretical drawbacks of autoregressively trained LLM models for semantic classification tasks, attributing these limitations to their fixed token positioning, which restricts inter-sample connections. In contrast, masked language models, such as BERT (Devlin et al., 2019) and DeBERTa (He et al., 2020), leverage flexible token targeting to overcome these issues. Additionally, through fine-tuning they can be easily (and cheaply) adapted to each specific keyword assignment regime, which vary across different domains.

3 Methods

3.1 Architecture

Our approach (visualised in Figure 1) is based on fine-tuning a pre-trained transformer architecture, in our case DeBERTa (He et al., 2020), with a specialised token classification head. The architecture was chosen since it showcased a SOTA performance on several downstream NLP tasks, among them also sequence-labelling tasks such as NER (Shon et al., 2022). Additionally, we hypothesised that the model is especially appropriate for keyword extraction due to its novel disentangled attention mechanism. In the DeBERTa model, each word is represented using two vectors that encode its content and position separately, and the attention weights among words are computed using disentangled matrices on their contents and relative positions, respectively. We assumed that this would allow the model to better distinguish between the positional and semantic/lexical information and therefore make it possible to assign

attention to some tokens purely based on their position in the text, improving the overall performance. This hypothesis is based on a previous study by Martinc et al. (2022) showing that token position is especially important in the keyword identification task due to the disproportionate distribution of keywords in the document, which is skewed towards the beginning of the document (i.e., more keywords appear at the beginning of the document). By considering the importance of positional information, they fed the positional encoding to the attention mechanism directly, subsequently managing to improve the performance.

The main novelty of our approach is a customised token classification head with MoE. It is motivated by the hypothesis that token-based specialization offered by MoE might be especially useful for keyword extraction. The specialization of different attention heads and attention layers in the transformer architecture has been extensively analysed in the past, also in the context of keyword extraction (Martinc et al., 2022). This specialization is mostly embeddings-based, i.e., different parts of the embedding space for each token are specialized. In contrast, the specialization enabled by MoE occurs on the sequence level, since the router decides for each token in the sequence to which expert it should go. Since the "keywordiness" of a specific token (or a sequence of tokens) to a large extent depends on a context, we assumed that this type of specialization might bring benefits.

The output logits from the DeBERTa backbone are first fed to the gate network or router that determines which tokens in the input sequence are sent to which expert. More specifically, in the setting with *n* experts, the gate network (G) decides which experts (E) receive a part of the input:

$$y = \sum_{i=1}^{n} G(x)_i E_i(x)$$

We model the routing as a weighted multiplication, where hypothetically all experts could contribute to processing each part of the input. Nevertheless, the additional *top_k* parameter ensures that only top *k* experts contribute to the respective part of the input, and other experts are excluded from the computational graph, hence saving computational resources. Following Shazeer et al. (2017), we employ the so-called Noisy Top-k Gating, which introduces some (tunable) noise and then keeps the top k values. Noise is introduced for load balancing, i.e., to prevent the gating network from converging in a way that the same few experts are activated for most of the tokens, which would make training inefficient. The final weight distribution is obtained by feeding the input (x) through a dense layer (W_g) and adding some random noise in the following way:

$$H(x)_i = (x \cdot W_g)_i + \text{RND} \cdot \text{Softplus}((x \cdot W_{noise})_i)$$

Here, RND represents a weight vector of random numbers from a normal distribution with mean 0 and variance 1. After that, we filter out only the top k experts out of v:

$$KeepTopK(v,k)_i = \{v_i \text{ if } v_i \text{ in top k, else } -\inf\}$$

We finally apply the softmax to filtered experts:

$$G(x) = \text{Softmax}(\text{KeepTopK}(H(x),k))$$

Note that the router is composed of learnable parameters and is fine-tuned at the same time as the rest of the network. On the other hand, each expert is represented as a neural network consisting of three sequential dense layers W_g with additional activation and dropout layers. More specifically, the expert is represented as:

$$E(x) = \text{Dropout}(\text{ReLU}(x \cdot W_g) \cdot W_g) \cdot W_g$$

The outputs of different experts, which attend to specific tokens assigned to them by the routing network, are first reassembled into a single sequence representation. Then, a two-layer randomly initialised encoder, consisting of dropout and two BiLSTM layers, is added (with element-wise summation) to this sequence. The employment of the BiLSTM module is there to amplify the sequential information lost by the context slicing caused by the MoE strategy, i.e., the weakening of the positional information due to overspecialization. This is especially problematic in case of multi-word keyphrases, where different parts of the keyphrase are attended by different experts. This adaptation is also motivated by the findings from the related work (Sahrawat et al., 2020), where it was shown that putting a BiLSTM classification head on top of different transformer models improved the keyword extraction capabilities of the model.

The output sequence of the MoE and BiLSTM encoders is finally fed to the feed-forward classification layer, which returns the output matrix of size SL $*$ NC, where SL stands for sequence length and NC stands for the number of classes. In our case, the NC is 3, since we model the keyword extraction task as sequence labelling and employ the so-called BIO labelling approach (same as, for example, in Sahrawat et al. (2020)). Using the BIO annotation regime, each word in the document is assigned one of the three possible labels: k_b denotes that the word is the first word in a keyphrase, k_i means that the word is inside a keyphrase, and k_o indicates that the word is not part of a keyphrase.

The identified keyphrases are extracted from the labelled sequence produced by the model, and we conduct additional filtering, where we remove keyphrases containing punctuation (except dashes and apostrophes), and deduplication, where we remove duplicate keyphrases. The final output list contains a maximum of 10 keyphrases per input document.

4 Experimental Setting

4.1 Datasets

We conduct experiments on selected benchmark datasets used in the SOTA papers to which we compare our approach. More specifically, we experiment on 6 publicly available datasets with MIT license from the related work (Meng et al., 2017; Martinc et al., 2022; Xiong et al., 2019), covering three domains, news, scientific articles, and web pages.

We present the datasets in Table 1, which contains information about the domain, the number of train, validation, and test documents, as well as average number of keywords per document. There is notable variability in a number of keywords per document, with some datasets requiring extraction of twice (or even three times) as many keywords as others.

4.2 Experiments

We test different settings to determine the effect that the proposed architectural additions have on the performance of the model. We are interested in the specific contributions of the MoE and BiLSTM layers to the performance of the model (either separately or together); therefore, we compare the settings containing these components to the baseline DeBERTa model with just a (usual) feed-forward token classification head.

Dataset	Domain	Train	Valid #Docs	Test #Docs	Train #KW per doc	Valid #KW per doc	Test #KW per doc
KP20k	**CS**	530,000	20,000	20,000	3.30	3.31	3.32
Inspec	**CS**	/	1,500	500	/	7.21	7.51
Krapivin	**CS**	/	1,844	460	/	2.87	3.22
KPTimes	**News**	259,923	10,000	10,000	2.34	2.31	2.35
JPTimes	**News**	-	-	10,000	/	/	3.84
openKP	**Web**	134,894	6,616	6,614	2.00	1.90	1.92

Table 1: Statistics for keyword extraction datasets. This table presents the number of documents in the train, validation, and test sets for each dataset. The **Domain** column indicates the source of the datasets (computer science (CS), news, or web), while **#KW per doc** shows the average number of keywords per document.

We employ the DeBERTa model[1] and fine-tune it using Low-Rank Adaptation (LoRA) (Hu et al., 2021). We freeze all layers in the backbone model and train only low-rank perturbations to query and value weight matrices in the model. Additionally, we train all the matrices in the custom token classification head (i.e., MoE, BiLSTM, and dense classification layers are randomly initialised and fine-tuned). See Section A for details about the hyperparameter setting used.

In order to determine how the size of the training dataset affects our approach, we employ two distinct training scenarios. In the **validation set training** scenario, the DeBERTa model was fine-tuned just on each dataset's validation sets, which were randomly split into 80 percent of documents used for fine-tuning and 20 percent of documents used for hyperparameter optimisation and test set model selection. We fine-tune the models for up to 20 epochs and employ early stopping. Note that the *JPTimes* dataset with no available predefined validation-test splits is only used for testing. Here, the model was fine-tuned on the *KPTimes-valid* dataset.

In the **full training** scenario, we fine-tune the models on all available data, same as in related work (Meng et al., 2017; Martinc et al., 2022). More specifically, before fine-tuning the model on three relatively small validation datasets containing scientific papers (*KP20k-valid*, *Inspec-valid*, and *Krapivin-valid*), we first "pre-train" the model (using the same token classification objective as in the validation dataset fine-tuning stage) for 20 epochs on the large *KP20k-train* dataset also containing scientific articles. We do the same for

[1]More specifically, the "deberta-v3-base" version available at https://huggingface.co/microsoft/deberta-v3-base under the MIT license.

news datasets, i.e., before fine-tuning the model on the *KPTimes-valid* dataset, we "pre-train" it on the large *KPTimes-train* dataset. For the *JPTimes* dataset with no available predefined validation-test splits, the keyword detection model was "pre-trained" on *KPTimes-train* and fine-tuned on the *KPTimes-valid* dataset. For the openKP dataset, the model is "pre-trained" on the *openKP-train* dataset.

To assess the performance of our models and compare the results of our keyword extraction approach to other SOTA methods, we adopt the same evaluation methodology as in Meng et al. (2017); Martinc et al. (2022); Luo et al. (2024). We use the same test splits for the sake of comparability between our approach and others, and measure F1@k with $k \in \{1, 3, 5\}$ on the *openKP* dataset and $k \in \{5, 10\}$ on other datasets[2]. To determine the exact match, we first lowercase the candidate keywords and the gold standard keywords, and then apply Porter Stemmer (Porter, 1980) to both.

5 Results and Analysis

In Table 2, we present the results of our approach and compare them to several baselines (see Section 2.2 for details). TF-IDF, TextRank, RaKUN and YAKE algorithms are unsupervised and do not require any training. Same applies to the LLM-based approaches, namely ChatGPT[3], ChatGLM2-6B[4], GPT4o-mini (Achiam et al., 2023) and Gemini-1.5-flash-8B (Team et al., 2023)[5].

For the supervised baselines, CopyRNN, CatSeqD, GPT-2, GPT-2+BiLSTM-CRF, and TNT-KID, we list results reported by Martinc et al. (2022), for Diff-KPE, we list results reported by Luo et al. (2024), for JointKPE, we list results from Sun et al. (2021), and for HybridMatch, we list results from Song et al. (2024). Note that GPT-2, GPT-2 + BiLSTM-CRF and TNT-KID are sequence labelling approaches, same as ours. Since they return a non-predetermined number of keywords (i.e., the approaches learn during training how many keywords they should return), they require adaptation to each specific keyword labelling regime for optimal performance and were therefore fine-tuned on the validation datasets. In contrast, JointKPE, DIffKPE nad HybridMatch baselines return a predetermined number of best ranked candidates (3, 5 or 10, depending on the evaluation criteria) and do not require validation set fine-tuning for adaptation. Same applies to the two generation baselines, CopyRNN and CatSeqD, which tend to generate a large set of predicted keywords, from which a fixed number of top predicted phrases (3, 5 or 10) is taken to compare with the ground truth. Finally, besides testing our custom sequence labelling heads (containing MoE, BiLSTM, or both), we also report results for the unmodified pre-trained DeBERTa model with a standard feed-forward token classification head.

In the **validation set training scenario**, by adding the MoE on top of the DeBERTa model (see configuration DeBERTa MoE), we improve the performance of the vanilla model (see configuration DeBERTa) with a feed-forward token classification head on four out of six datasets according to all evaluation criteria. The improvement is not observed for the two smallest datasets, *Krapivin* and *Inspec*. On the other hand, adding the additional BiLSTM head to the vanilla DeBERTa model improves the performance of the base model on all datasets (see configuration DeBERTa BiLSTM) and according to almost all criteria (the only exception being the F1@1 criterion on the *openKP* dataset). The results therefore indicate a synergy between the MoE and BiLSTM layers, since the model with both MoE and BiLSTM layers manages to substantially improve the base model and also outperforms the DeBERTa MoE and DeBERTa BiLSTM models on four out of six datasets according to all criteria.

In the **full training scenario**, the DeBERTa MoE model outperforms the vanilla DeBERTa on three out of six datasets according to all evaluation criteria. Interestingly, the DeBERTa MoE model fails on all computer science datasets, which suggests that expert specialisation might be harder on this domain. On the other hand, the DeBERTa MoE BiLSTM model outperforms the vanilla one on all but one dataset and criterion, the F1@1 criterion on the *openKP* dataset.

In this scenario, we also manage to substantially improve the SOTA on the *Inspec*, *KPTimes*, and *JPTimes* datasets according to both criteria. While we outperform the best-performing model *JoinKPE* in F1@10 on the *KP20k* and the *Krapivin* datasets,

[2]Note that the values of k differ between *openKP* and other datasets. This is due to the variability of keyword assignment regimes across different domains, i.e., news and scientific articles on average have more keywords than web pages (see Table 1 for details). We report the same F1@k for each dataset as in related work to facilitate easy comparison.

[3]We report results for the gpt-3.5-turbo version tested in Luo et al. (2024).

[4]We list results from Luo et al. (2024).

[5]see Section D for details about the LLM experiments.

	KP20k		Inspec		Krapivin		KPTimes		JPTimes		openKP		
	F1@5	**F1@10**	**F1@5**	**F1@10**	**F1@5**	**F1@10**	**F1@5**	**F1@10**	**F1@5**	**F1@10**	**F1@1**	**F1@3**	**F1@5**
Unsupervised baselines													
TF-IDF	7.2	9.4	16.0	24.4	6.7	9.3	17.9	15.1	26.6	22.9	19.6	22.3	19.6
TextRank	18.0	15.0	28.6	33.9	18.5	16.0	2.2	3.0	1.2	2.6	5.4	7.6	7.9
RaKUN	17.7	16.0	10.1	10.8	12.7	10.6	16.8	13.9	22.5	18.5	8.81	10.12	6.45
YAKE	14.1	14.6	20.4	22.3	21.5	19.6	10.5	11.8	10.9	13.5	3.73	4.24	2.94
Large Language Model baselines													
ChatGLM2-6b	/	/	25.1	30.1	/	/	/	/	/	/	16.0	11.0	8.6
ChatGPT	23.2	10.8	35.2	33.9	12.8	11.7	27.9	/	/	/	20.8	20.4	16.6
GPT4o-mini	22.7	19.8	34.4	42.1	22.1	20.5	19.5	16.6	29.3	25.2	34.0	26.8	26.8
Gemini-1.5-flash-8B	26.5	23.6	35.9	42.1	26.6	25.9	20.4	26.9	31.8	27.5	33.1	25.5	25.6
Supervised baselines													
CopyRNN	31.7	27.3	24.4	28.9	30.5	26.6	40.6	39.3	25.6	24.6	21.7	23.7	21.0
CatSeqD	34.8	29.8	27.6	33.3	32.5	28.5	42.4	42.4	23.8	23.8	/	/	/
GPT-2	27.5	27.8	41.3	46.9	25.3	25.3	42.1	42.3	33.1	33.6	/	/	/
GPT-2+BiLSTM-CRF	35.5	36.0	46.2	52.4	28.7	28.8	47.8	47.9	38.6	38.9	/	/	/
TNT-KID	33.6	33.8	46.0	53.6	31.0	32.0	48.5	48.5	35.9	36.1	/	/	/
JointKPE	**41.7**	34.4	35.2	35.0	**36.0**	29.2	/	/	/	/	36.4	39.1	33.8
Diff-KPE	**41.7**	34.3	32.3	35.0	35.0	31.4	/	/	/	/	**37.8**	38.5	32.7
HybridMatch	/	/	/	/	/	/	/	/	/	/	37.3	39.4	34.1
Our approaches - validation set training													
DeBERTa	29.2±1.3	29.4±1.3	44.8±0.6	53.9±1.1	26.8±1.1	27.5±1.0	41.9±0.7	41.9±0.7	35.2±0.9	35.4±0.9	27.6±1.5	36.5±0.9	37.2±0.8
DeBERTa MoE	31.1±1.0	31.5±1.1	44.1±0.7	53.7±0.5	26.9±0.3	27.4±0.5	42.5±0.4	42.5±0.4	36.4±2.4	36.8±2.6	27.7±1.0	38.0±0.7	38.9±0.6
DeBERTa BiLSTM	31.2±1.0	31.5±1.0	47.7±0.7	56.6±0.8	29.2±1.5	29.6±1.5	44.1±0.7	44.2±0.7	36.2±1.2	36.6±1.3	26.8±1.1	37.2±0.5	38.1±0.4
DeBERTa MoE BiLSTM	32.7±1.1	33.0±1.1	47.2±1.1	56.5±1.0	29.5±1.1	30.1±1.0	44.9±0.3	45.1±0.3	37.4±0.5	37.8±0.6	27.1±0.9	38.9±0.5	40.1±0.4
Our approaches - full training													
DeBERTa	34.7±0.6	35.1±0.6	48.1±0.8	57.4±0.8	31.8±1.9	32.1±1.9	54.5±0.6	54.5±0.6	37.5±1.4	37.7±1.4	33.7±0.9	41.4±1.2	41.7±1.2
DeBERTa MoE	33.5±0.7	33.9±0.6	47.1±0.5	56.3±1.1	30.0±0.8	30.2±0.8	**55.0±0.4**	**55.1±0.4**	38.7±0.9	39.1±1.0	34.2±0.4	42.4±0.4	42.8±0.4
DeBERTa BiLSTM	34.4±0.6	34.7±0.6	47.6±0.6	57.5±0.8	32.3±1.5	32.6±1.4	53.5±0.1	53.5±0.1	38.1±0.9	38.2±1.0	31.9±2.3	39.8±1.0	40.2±0.8
DeBERTa MoE BiLSTM	36.0±0.3	**36.5±0.3**	**49.4±1.0**	**58.8±0.4**	34.3±0.9	**34.7±0.9**	54.8±1.0	55.0±0.9	**39.0±1.7**	**39.3±1.8**	33.3±1.5	**42.5±0.6**	**42.9±0.5**

Table 2: Empirical evaluation of the keyword extractors. We report the average over five runs and the standard deviation. Bold represents the best score according to a specific criterion on a specific dataset.

our best-performing configuration (DeBERTa MoE BiLSTM) on the other hand achieves uncompetitive performance in terms of F1@5 on these two datasets. On the *openKP*, our best approach (DeBERTa MoE BiLSTM) fails to beat several approaches (among others, also the two LLM-based baselines, GPT4o-mini and Gemini-1.5-flash-8B) according to the F1@1 criterion. The performance comparison is nevertheless favourable according to the other two criteria.

The results also show that the unsupervised models in general achieve much worse performance than the supervised models. While the LLM-based approaches outperform other unsupervised baselines, their performance lags behind supervised approaches. Out of the four tested LLMs, the two proprietary models, GPT4o-mini and Gemini-1.5-flash-8B, perform the best, with Gemini-1.5-flash-8B being the best model on most datasets.

5.1 Examining the Specialisation of Experts

Next, we assess how individual experts correlate with the tokens they operate on to determine their specific types of **specialisation**. During inference on the test sets, we extracted the expert with the highest weight (i.e., the expert contributing the most out of the top k) for each token in the input sequence. We then correlated the selected expert per token with the token's grammatical and semantic attributes. On the grammatical/lexical level, we consider: whether a token is a specific word, part-of-speech tag (**POS**), punctuation character or not (**Punct**), or stopword or not (**Stop.**). On the semantic level, we examine the predicted labels for each token (**Labels**), whether a given token is a named entity (**bin. NE**), and if so, the specific type of named entity (**NE**). This way, we try to identify different types of **specialisation**, i.e., whether specialisation is rooted in syntax or semantics.

We annotated the tokens with POS tags and NE labels using the SpaCy library (Honnibal et al., 2020). In total, we obtain six different categories of specialisation, which we annotate on the dataset level (i.e., we obtain annotations for the concatenation of documents belonging to a specific dataset). Since we work with categorical values, we measure the correlation between the highest-weighted experts and each category by employing *Cramér's V* statistic (Cramér, 1999) with the bias correction proposed in Bergsma (2013). We consider the two training regimes, validation set and full training, separately, to examine the impact of data size. We present the results in Table 3.

In the validation set training scenario, a strong

Experts to	Inspec	Krapivin	KP20k	KPTimes	JPTimes	openKP
Validation set training						
Words	0.609	0.456	0.592	0.465	0.525	0.475
Punct.	0.701	0.391	0.688	0.076	0.063	0.000
Stop.	0.614	0.403	0.555	0.160	0.096	0.339
POS	0.571	0.376	0.528	0.260	0.323	0.305
Labels	0.257	0.184	0.135	0.093	0.107	0.153
bin. NE	0.085	0.025	0.056	0.271	0.198	0.148
NE	0.070	0.045	0.046	0.246	0.286	0.162
Full training						
Words	0.575	0.468	0.459	0.465	0.367	0.521
Punct.	0.128	0.081	0.048	0.082	0.040	0.009
Stop.	0.257	0.105	0.115	0.168	0.092	0.224
POS	0.415	0.265	0.262	0.267	0.165	0.255
Labels	0.228	0.120	0.341	0.184	0.089	0.269
bin. NE	0.039	0.029	0.043	0.238	0.131	0.171
NE	0.048	0.028	0.052	0.237	0.139	0.167

Table 3: Results of the Cramer's V statistic for correlation between the most weighted expert for a specific token and the specialisation categories examined, across **Validation set training** and **Full training** regimes. Strongest, second strongest, and third strongest correlations per dataset and regime are marked.

correlation between specific words and specific experts is observed on all datasets. This is not surprising, since MoE works on the token level. In contrast, the correlation between experts and labels (I, O, and B) tends to be moderate at best (i.e., between 0.1 and 0.3) on most datasets. One can also observe a moderate (and on some datasets strong, i.e., more than 0.5) correlation between specific experts and POS tags, suggesting that specialisation is somewhat influenced by the syntax of the text.

By looking at the other three types of correlation, namely the correlation between experts and NEs, between experts and stopword tokens, and between experts and punctuation tokens, one can see a very distinct difference between the computer science and other datasets. While there is a strong tendency for experts to specialise for stopword and punctuation tokens on all the computer science datasets (*Inspec*, *Krapivin*, and *KP20k*), this tendency is much weaker on other (news and web page) datasets, where one can observe almost no correlation. Here one can observe a specialisation of experts for NEs, which is almost non-existent on the computer science datasets. A correlation is almost the same if we binarize the NE sequence (i.e., to NE/non-NE tokens) or if we look for correlations between experts and 23 labels returned by the SpaCy NE recognition pipeline. The specialisation for NEs can be perhaps explained by the fact that news and web datasets contain much more NEs than scientific papers and that many of them are in fact labelled as keywords. The strong correlation with simple syntactic patterns on the computer science datasets can be on the other hand explained by the lack of obvious semantic clues, such as NEs, in these datasets.

When comparing the specialisation in the validation set training scenario and the full training scenario, one can see that correlation to labels, words, NEs, and binary NEs is mostly the same. On the other hand, one can see a decrease in correlation to POS tags, stopwords and punctuation on the computer science datasets, when the models are trained on more data. The assumption is that this is the consequence of expert specialisation becoming more complex when the amount of data allows it.

5.2 Error Analysis of MoE Performance by Keyword Length

To test the hypothesis that the MoE component struggles with longer keywords due to context slicing, we have conducted an additional error analysis. This ablation study compares the performance of DeBERTa MoE BiLSTM and DeBERTa MoE model on keywords of varying lengths.

Specifically, for each test dataset, we created subsets of documents containing keywords of a specific length, ranging from one to ten words. We then evaluated the performance in terms of F1@1 score for each model on these subsets, considering only the keywords of that specific length (we use F1@1 due to small set of keywords of specific length per document).

Table 4 presents the differences in F1@1 scores (calculated as DeBERTa MoE BiLSTM F1@1 - DeBERTa MoE F1@1). Positive values indicate that DeBERTa MoE BiLSTM outperformed DeBERTa MoE on keywords of specific length, while negative values indicate the opposite. The results show that the standard MoE model is indeed less effective at extracting long, multi-word keywords compared to the model with an BiLSTM encoder (i.e., the difference is generally bigger on longer keywords). This analysis supports the hypothesis that context slicing in the MoE architecture is a key deficiency.

6 Conclusion

In summary, we investigated the potential of MoE for keyword extraction tasks. We introduced SEKE, a novel architecture that combines MoE and BiLSTM built upon the DeBERTa model. This approach achieved superior performance compared to existing SOTA models on standard datasets for key-

Dataset/Keyword Length	1	2	3	4	5	6	7	8	9	10
Inspec	**0.0018 (n=291)**	-0.0042 (n=472)	**0.0136 (n=396)**	**0.0303 (n=209)**	**0.0353 (n=85)**	**0.2610 (n=23)**	**0.1004 (n=10)**	**0.2381 (n=3)**	0.0000 (n=1)	0.0000 (n=1)
JPTimes	**0.0175 (n=9241)**	-0.0386 (n=6642)	-0.0362 (n=1576)	**0.0036 (n=274)**	**0.0417 (n=48)**	**0.1333 (n=15)**	**0.2500 (n=4)**	0.0000 (n=4)	—	0.0000 (n=1)
KP20k	-0.0196 (n=9479)	**0.0035 (n=14905)**	**0.0202 (n=7943)**	**0.0675 (n=2040)**	**0.0527 (n=450)**	**0.0103 (n=97)**	0.0000 (n=27)	0.0000 (n=6)	0.0000 (n=5)	—
KPTimes	**0.0222 (n=7184)**	**0.0814 (n=5027)**	**0.1121 (n=1766)**	**0.1569 (n=465)**	**0.1192 (n=151)**	**0.0909 (n=88)**	**0.3824 (n=34)**	**0.5556 (n=9)**	0.0000 (n=1)	0.0000 (n=2)
Krapivin	**0.0002 (n=206)**	**0.0139 (n=381)**	**0.0273 (n=174)**	**0.0063 (n=50)**	0.0000 (n=8)	0.0000 (n=6)	—	—	—	—
openKP	**0.0180 (n=3125)**	**0.0311 (n=4118)**	**0.0211 (n=1956)**	**0.0159 (n=599)**	**0.0072 (n=139)**	**0.0385 (n=26)**	0.0000 (n=5)	0.0000 (n=4)	0.0000 (n=1)	—

Table 4: Difference in F1@1 scores between DeBERTa MoE BiLSTM and DeBERTa MoE (DeBERTa MoE BiLSTM F1@1 - DeBERTa MoE F1@1) on keywords of different word length across different datasets.

word extraction. The study revealed that adding BiLSTM can help mitigate the data limitations, showing the feasibility of the approach even on small training datasets. In the future, we will try to pinpoint the exact dataset size thresholds, down to which the proposed approach is still feasible. Additionally, we will expand the evaluation to more benchmarks to further strengthen the analysis.

In this work, our primary goal was to evaluate the general effectiveness of our proposed framework using a consistent and reasonable configuration across all experiments. To maintain a fair comparison and avoid overfitting to specific datasets, we used commonly adopted default values (e.g., 4 experts, top-k = 2) without extensive hyperparameter tuning. In the future, a more exhaustive hyperparameter search will be conducted, which could yield additional performance gains and further insights into the model behaviour.

The analysis revealed that specialisation is domain-specific. Current results indicate that in text with higher complexity (i.e., scientific papers) experts to a larger extent rely on syntactical patterns than in the less complex text, where experts to a larger extent specialise for processing of semantic clues. This hypothesis will be further tested in the future by applying the approach in new domains and languages other than English.

Finally, we can observe that despite the impressive improvements in the development of foundational LLMs, keyword extraction task can still not be sufficiently solved by employing these models, which lag behind supervised approaches on most datasets. Since the biggest advantage of supervised approaches is their ability to adapt to the specifics of the syntax, semantics, content, and keyword tagging regime of the specific corpus (Meng et al., 2017), in the future we will adapt LLMs to the specific corpora and keyword extraction regimes with fine-tuning and in-context learning.

7 Limitations

First, the empirical evaluation was conducted exclusively on English-language datasets from three specific domains (computer science, news, and the web). Consequently, the generalizability of our findings to other languages and more diverse or specialized domains remains to be thoroughly tested.

Furthermore, the model's internal mechanisms may be susceptible to certain weaknesses. One potential issue is an over-reliance on specific Part-of-Speech (POS) patterns for keyword identification. The model might learn heuristic shortcuts, associating keywords primarily with common grammatical structures (e.g., noun-noun compounds or adjective-noun phrases). This could limit its ability to identify valid but unconventional keywords and may lead to false positives. A related concern, specific to the MoE architecture, is the risk of the expert routing mechanism overfitting to superficial cues. The router might learn to dispatch tokens based on shallow features like capitalization or token position rather than deep semantic content, which would prevent true expert specialization and harm the model's robustness on out-of-distribution data.

From a computational and environmental standpoint, the introduction of the BiLSTM encoder in the custom token classification head increases the model's complexity. This makes both inference and fine-tuning slightly slower and more computationally demanding than the vanilla DeBERTa model.

While the performance difference was negligible in our experiments, it could become a significant factor when scaling to much larger datasets. More broadly, the training of large models like DeBERTa is an energy-intensive process with an associated environmental cost, a factor that is critical to consider in the development of ever-larger architectures.

Methodologically, our specialization analysis relied on spaCy for Named Entity Recognition. As a general-domain NLP tool, its performance can be suboptimal on specialized texts like scientific articles. This reliance might have affected the reliability of the analysis by yielding fewer or less accurate named entities, potentially skewing the correlation results.

Finally, it is crucial to consider the broader implications and potential for misuse. A model effective at automated keyword extraction could be exploited for malicious purposes, such as "black-hat" Search Engine Optimization (SEO). Such a system could be used to generate spammy keywords to artificially boost the ranking of low-quality content, thereby degrading the integrity of information retrieval systems.

8 Ethics Statement

This work does not pose any ethical issues. All the data and tools used in this paper are publicly available under the MIT license. No private data or non-public information is used in this work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Wicher Bergsma. 2013. A bias-correction for cramér's v and tschuprow's t. *Journal of the Korean Statistical Society*, 42(3):323–328.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in NIPS*, 33:1877–1901.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Harald Cramér. 1999. *Mathematical methods of statistics*, volume 26. Princeton university press.

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.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2013. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, pages 216–223, Stroudsburg, PA, USA. Association for Computational Linguistics.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Wanhae Lee, Minki Chun, Hyeonhak Jeong, and Hyunggu Jung. 2023. Toward keyword generation through large language models. In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces*, pages 37–40.

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

Yuanzhen Luo, Qingyu Zhou, and Feng Zhou. 2024. Enhancing phrase representation by information bottleneck guided text diffusion process for keyphrase extraction. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 6036–6047, Torino, Italia. ELRA and ICCL.

Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. 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 634–639, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Matej Martinc, Blaž Škrlj, and Senja Pollak. 2022. Tnt-kid: Transformer-based neural tagger for keyword identification. *Natural Language Engineering*, 28(4):409–448.

Roberto Martínez-Cruz, Alvaro J López-López, and José Portela. 2024. Chatgpt vs state-of-the-art models: a benchmarking study in keyphrase generation task. *Applied Intelligence*, 55(50).

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.

Cheng Peng, XI Yang, Kaleb E Smith, Zehao Yu, Aokun Chen, Jiang Bian, and Yonghui Wu. 2024. Model tuning or prompt tuning? a study of large language models for clinical concept and relation extraction. *Journal of biomedical informatics*, 153:104630.

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

Dhruva Sahrawat, Debanjan Mahata, Mayank Kulkarni, Haimin Zhang, Rakesh Gosangi, Amanda Stent, Agniv Sharma, Yaman Kumar, Rajiv Ratn Shah, and Roger Zimmermann. 2020. Keyphrase extraction from scholarly articles as sequence labeling using contextualized embeddings. In *Proceedings of European Conference on Information Retrieval (ECIR 2020)*, pages 328–335, Lisbon, Portugal. Springer.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Suwon Shon, Ankita Pasad, Felix Wu, Pablo Brusco, Yoav Artzi, Karen Livescu, and Kyu J Han. 2022. Slue: New benchmark tasks for spoken language understanding evaluation on natural speech. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7927–7931. IEEE.

Blaž Škrlj, Andraž Repar, and Senja Pollak. 2019. RaKUn: Rank-based keyword extraction via unsupervised learning and meta vertex aggregation. In *International Conference on Statistical Language and Speech Processing*, pages 311–323, Ljubljana, Slovenia. Springer.

Mingyang Song, Yi Feng, and Liping Jing. 2023. A survey on recent advances in keyphrase extraction from pre-trained language models. *Findings of the Association for Computational Linguistics: EACL 2023*, pages 2153–2164.

Mingyang Song, Liping Jing, and Yi Feng. 2024. Match more, extract better! hybrid matching model for open domain web keyphrase extraction. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 17–27.

Si Sun, Zhenghao Liu, Chenyan Xiong, Zhiyuan Liu, and Jie Bao. 2021. Capturing global informativeness in open domain keyphrase extraction. In *Natural Language Processing and Chinese Computing: 10th CCF International Conference, NLPCC 2021, Qingdao, China, October 13–17, 2021, Proceedings, Part II 10*, pages 275–287. Springer.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Peter D Tumey. 1999. Learning to extract keyphrases from text. *NRC Technical Report ERB-l 057. National Research Council, Canada*, pages 1–43.

Lee Xiong, Chuan Hu, Chenyan Xiong, Daniel Campos, and Arnold Overwijk. 2019. Open domain web keyphrase extraction beyond language modeling. *arXiv preprint arXiv:1911.02671*.

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, Online. Association for Computational Linguistics.

Qi Zhang, Tianqi Du, Haotian Huang, Yifei Wang, and Yisen Wang. 2024. Look ahead or look around? A theoretical comparison between autoregressive and masked pretraining. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 58819–58839. PMLR.

A Hyperparameters Used

We use the same hyperparameter setting across all datasets, namely, the following values were used during fine-tuning: learning rate of 2e-4, LoRA intrinsic rank of 16, LoRA alpha of 16, LoRA dropout of 0.1 sequence length of 256, 4 experts, and the top k value of 2.

B Model Size and Budget

The final model with MoE and BiLSTM layers, which uses the "deberta-v3-base" backbone (available at `https://huggingface.co/microsoft/deberta-v3-base` under the MIT license) has 274M parameters, out of them 45M are trainable during LoRA fine-tuning.

We used two NVIDIA A100 80GB GPUs in our experiments. The total computational budget for all experiments, i.e., validation set training and full training with five random seeds and four different experimental settings (feed-forward classification head, just MoE layer, just BiLSTM layer, both MoE and BiLSTM layers), was 1400 GPU hours.

C Examples of Keyword Identification

This section presents examples of extracted keywords on randomly selected instances from all datasets.

1.) Inspec:
The Bagsik Oscillator without complex numbers. We argue that the analysis of the so-called Bagsik Oscillator, recently published by Piotrowski and Sladkowski (2001), is erroneous due to: (1) the incorrect banking data used and (2) the application of statistical mechanism apparatus to processes that are totally deterministic.

Predicted keywords: incorrect banking data, statistical mechanism apparatus, bagsik oscillator

True keywords: data, statistical mechanism apparatus, complex numbers, bagsik oscillator

2.) Krapivin:
Solving Equations in the Relational Algebra. Enumerating all solutions of a relational algebra equation is a natural and powerful operation which, when added as a query language primitive to the nested relational algebra, yields a query language for nested relational databases, equivalent to the well-known powerset algebra. We study sparse equations, which are equations with at most polynomially many solutions. We look at their complexity and compare their expressive power with that of similar notions in the powerset algebra.

True keywords: relational algebra, nested relation, equation

Predicted keywords: equations, relational algebra, powerset

3.) KP20k:
An algebraic approach to guarantee harmonic balance method using grobner base. Harmonic balance (HB) method is well known principle for analyzing periodic oscillations on nonlinear networks and systems. Because the HB method has a truncation error, approximated solutions have been guaranteed by error bounds. However, its numerical computation is very time-consuming compared with solving the HB equation. This paper proposes an algebraic representation of the error bound using Grobner base. The algebraic representation enables to decrease the computational cost of the error bound considerably. Moreover, using singular points of the algebraic representation, we can obtain accurate break points of the error bound by collisions.

True keywords: algebraic representation, singular point, error bound, grobner base, harmonic balance method

Predicted keywords: grobner base, bound, harmonic balance, harmonic balance method

4.) KPTimes:
Afghan Police Chief Is Killed as He Tries to Turn Tide Against Taliban. A hard-charging Afghan police chief with deep experience in Afghanistan's long conflict with the Taliban was killed in a blast on Sunday in the country's eastern Nangarhar Province, which has been under threat from the Taliban and affiliates of the Islamic State. The

police chief, Gen. Zarawar Zahid, was visiting an outpost in the Hisarak district when explosives placed near the outpost detonated, according to Attaullah Khogyani, a spokesman for the governor of Nangarhar. One of General Zahid's bodyguards was wounded, Mr. Khogyani said. More Than 14 Years After U.S. Invasion, the Taliban Control Large Parts of Afghanistan At least one-fifth of the country is controlled or contested by the Taliban. The Taliban claimed responsibility for the killing, according to a statement by the insurgency's spokesman, Zabiullah Mujahid. The attack came a week after twin bombings outside Afghanistan's Ministry of Defense killed at least 40 people , including several senior security officials. Nangarhar, which borders Pakistan, has faced mounting security perils over the past couple of years, with new Islamic State affiliates complicating the threat from the Taliban. Zabihullah Zmarai, a member of the provincial council, said the Islamic State posed a danger in five districts, despite repeated operations by the Afghan Army. Out of the 22 districts, only six are secure, he said. The Taliban's presence across nearly a dozen districts varies, Mr. Zmarai said. But the Hisarak district faced a collapse in recent weeks. That drew the attention of General Zahid, who had gone there to supervise a counterattack. Over the past decade, he rose from a bodyguard to a well-regarded police chief of several volatile provinces. His postings included two stints as the police chief of southeastern Ghazni Province, and one term each in Zabul and Paktika Provinces. General Zahid was seen as a hands-on commander, often arriving at the front lines unannounced. When a major cultural event drew world leaders to the ancient city of Ghazni, the general was photographed riding around the city on the back of a motorcycle to check on security measures. He had been wounded twice and had lost two brothers during the decades of war in Afghanistan. In June, he took part in clashes with Pakistani forces that erupted on the border. In a Facebook video that he posted, he appeared beside two mortars and shouted to his men, "Strike hard enough to blow up Nawaz Sharif's home," referring to the prime minister of Pakistan. Sediq Sediqqi, the spokesman for Afghanistan's Ministry of Interior Affairs, called General Zahid "one of the bravest commanders of Afghan police.cHe lost his life on the front line of duty in the fight against terrorism," Mr. Sediqqi said.

True keywords: afghanistan, taliban, zarawar zahid, bombs, nangarhar province,terrorism

Predicted keywords: afghanistan, zarawar zahid, taliban

5.) JPTimes:
Photo report: FOODEX Japan 2013. FoodEx is the largest trade exhibition for food and drinks in Asia, with about 70,000 visitors checking out the products presented by hundreds of participating companies. I was lucky to enter as press; otherwise, visitors must be affiliated with the food industry and pay to enter. The FoodEx menu is global, including everything from cherry beer from Germany and premium Mexican tequila to top-class French and Chinese dumplings. The event was a rare chance to try out both well-known and exotic foods and even see professionals making them. In addition to booths offering traditional Japanese favorites such as udon and maguro sashimi, there were plenty of innovative twists, such as dorayaki , a sweet snack made of two pancakes and a red-bean filling, that came in coffee and tomato flavors. While I was there I was lucky to catch the World Sushi Cup Japan 2013, where top chefs from around the world were competing and presenting a wide range of styles that you would not normally see in Japan, like the flower makizushi above.

True keywords: foodex

Predicted keywords: foodex, japan, food

6.) openKP:
"Quickly create pointandclick games and virtual tours for Windows native PSP iPhone and iPod Touch web apps No programming required very easy to use Free edition contains all the main features Includes free drawing tool and music composer NEW Create Games for iPhone DOWNLOAD FREE GAMES CREATED WITH ADVENTURE MAKER WATCH THE GUIDED TOUR Whats New in version 45 Whats New in version 44

True keywords: virtual tours, adventure maker, no programming required

Predicted keywords: virtual tours

D Prompt Used for LLM-based Keyword Extraction

For each document processed in our experiments, we applied the same prompt to ensure consistency in the extraction of relevant keywords. We condition the generation to return valid json prompts, so we can uniformly parse and compare the results. The prompt used for both the GPT4o-mini and Gemini-1.5-flash-8B models is provided below:

```
You are an assistant that specializes in text
 ↪  analysis. Your task is to extract the most
 ↪  relevant keywords from the following
 ↪  document.
Extract only keywords present in the document.
 ↪  Please analyze the text and provide a
 ↪  comma-separated list of keywords ordered
 ↪  from most important to least important.
Return a maximum of 10 keywords.

Document:
\"\"\"{document}\"\"\"

Return only the list of keywords, formatted as a
 ↪  json list (i.e. with square brackets [ and
 ↪  ]).
```

E Performance of our approaches trained solely on training sets

To make a comparison between our approach and other baselines (i.e., especially non-sequence labelling supervised baselines, which were not fine-tuned on the validation data) more comprehensive, Table 5 presents results when training of our approaches is done solely on KP20k, KPTimes, and OpenKP, with no validation data. Note that in this training setting our approaches outperform all baselines in 7 out of 13 cases. On the other hand, a large drop in performance can be observed for all our approaches on the Inspec dataset, where the keyword labelling regime is very different from the labelling regime of the KP20k training set, on which the models were trained.

	KP20k		Inspec		Krapivin		KPTimes		JPTimes		openKP		
	F1@5	**F1@10**	**F1@5**	**F1@10**	**F1@5**	**F1@10**	**F1@5**	**F1@10**	**F1@5**	**F1@10**	**F1@1**	**F1@3**	**F1@5**
Unsupervised baselines													
TF-IDF	7.2	9.4	16.0	24.4	6.7	9.3	17.9	15.1	26.6	22.9	19.6	22.3	19.6
TextRank	18.0	15.0	28.6	33.9	18.5	16.0	2.2	3.0	1.2	2.6	5.4	7.6	7.9
Large Language Model baselines													
ChatGLM2-6b	/	/	25.1	30.1	/	/	/	/	/	/	16.0	11.0	8.6
ChatGPT	23.2	10.8	35.2	33.9	12.8	11.7	27.9	/	/	/	20.8	20.4	16.6
GPT4o-mini	22.7	19.8	34.4	42.1	22.1	20.5	19.5	16.6	29.3	25.2	34.0	26.8	26.8
Gemini-1.5-flash-8B	26.5	23.6	35.9	42.1	26.6	25.9	20.4	26.9	31.8	27.5	33.1	25.5	25.6
Supervised baselines													
CopyRNN	31.7	27.3	24.4	28.9	30.5	26.6	40.6	39.3	25.6	24.6	21.7	23.7	21.0
CatSeqD	34.8	29.8	27.6	33.3	32.5	28.5	42.4	42.4	23.8	23.8	/	/	/
GPT-2	27.5	27.8	41.3	46.9	25.3	25.3	42.1	42.3	33.1	33.6	/	/	/
GPT-2+BiLSTM-CRF	35.5	36.0	46.2	52.4	28.7	28.8	47.8	47.9	38.6	38.9	/	/	/
TNT-KID	33.6	33.8	46.0	53.6	31.0	32.0	48.5	48.5	35.9	36.1	/	/	/
JointKPE	**41.7**	34.4	35.2	35.0	**36.0**	29.2	/	/	/	/	36.4	39.1	33.8
Diff-KPE	**41.7**	34.3	32.3	35.0	35.0	31.4	/	/	/	/	**37.8**	38.5	32.7
HybridMatch	/	/	/	/	/	/	/	/	/	/	37.3	39.4	34.1
Our approaches - just pre-training													
DeBERTa	35.2	35.3	14.5	14.5	28.8	28.9	55.6	55.7	37.4	37.5	34.1	**42.9**	**43.2**
DeBERTa MoE	33.9	34.0	14.7	14.8	28.5	28.8	56.4	56.5	39.1	39.3	34.7	42.7	43.0
DeBERTa RNN	33.6	33.7	13.9	13.9	27.5	27.7	53.2	53.2	36.8	36.9	33.4	39.7	39.8
DeBERTa MoE RNN	36.3	**36.5**	15.1	15.1	29.6	30.3	**59.7**	**59.8**	**39.3**	**39.5**	29.6	41.5	42.2

Table 5: Performance comparison between our approaches trained solely on KP20k, KPTimes, and OpenKP, and the baseline approaches. Only one seed is used in this scenario. Bold represents the best score according to a specific criterion on a specific dataset.