A Simple yet Efficient Prompt Compression Method for Text Classification Data Annotation Using LLM

Yiran Xie[1], Debin Xiao[2], Ping Wang[2], Shuming Liu[2],
[1]The Chinese University of Hong Kong, Shenzhen
[2]Guangdong OPPO Mobile Telecommunications Corp., Ltd., Shenzhen
223040060@link.cuhk.edu.cn, {xiaodebin, ping.wang, liushuming}@oppo.com

Abstract

Effectively balancing accuracy and cost is a critical challenge when using large language models (LLMs) for corpus annotation. This paper introduces a novel compression method based on keyword extraction (PCKE) that effectively reduces the number of prompt tokens in text classification annotation tasks, with minimal to no loss in accuracy. Our approach begins with an LLM that generates both category labels and relevant keywords from a small set of unannotated data. These outputs are used to train a BERT-based multi-task model capable of classification and keyword extraction. For larger unannotated corpora, this model extracts keywords which are then used in place of full texts for LLM annotation. The significant reduction in prompt tokens results in substantial cost savings. Furthermore, using a few well-chosen keywords ensures that classification accuracy is maintained. Extensive experiments validate that our method not only achieves a superior compression rate but also maintains high accuracy, outperforming existing general-purpose compression techniques. Our approach offers a practical and cost-efficient solution for large-scale text classification annotation using LLMs, particularly applicable in industrial settings.

1 Introduction

Large language models (LLMs) have demonstrated remarkable zero-shot learning capabilities across numerous NLP tasks (Kojima et al., 2022), including text classification (Sun et al., 2023a). However, due to high costs and time consumption, directly using LLMs to classify large-scale text in industry is often impractical. A common approach is to use LLMs to annotate a subset of the data, which is then used to train more efficient smaller models, typically based on BERT (Devlin et al., 2019; Sun et al., 2019; Han et al., 2021). Numerous studies have proposed methods to improve the accuracy of large language models (LLMs),

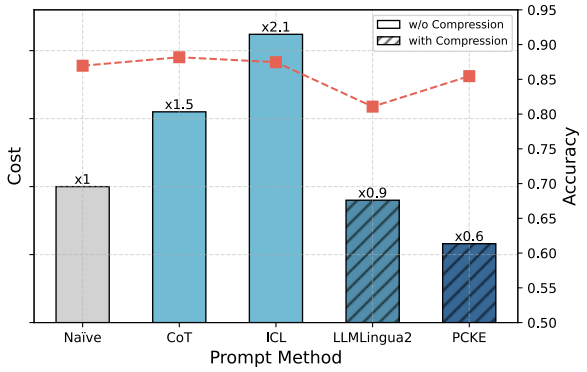


Figure 1: The average cost and accuracy for text classification datasets used in this work. The bar chart represents the average cost per GPT-4o API call, while the line chart shows the average accuracy of the LLM annotations. It can be seen that our method reduces token consumption while maintaining a very high annotation accuracy.

such as Chain-of-Thought (CoT) (Kojima et al., 2022), In-context Learning (ICL) (Brown et al., 2020; Xu et al., 2024), self-consistency (Wang et al., 2022; Huang et al., 2023a), automatic prompt optimization (Zhou et al., 2022; Pryzant et al., 2023; Yang et al., 2023). Although these methods have achieved notable results, they require longer prompts and result in increased computational and financial overhead, particularly in text classification with long texts. Prompt compression, therefore, has emerged as a critical yet sufficiently under-explored technique for shortening lengthy prompts.

Several studies have been conducted to enhance the efficiency of prompts for LLMs. Batch Prompting (Cheng et al., 2023; Lin et al., 2023) reduces the average system prompt consumption by annotating a batch of data in a single API call. On the other hand, general-purpose prompt compression methods aim to shorten the prompt length by removing redundant information and retaining essential information (Li et al., 2023; Jiang et al., 2023b,a; Pan

Proceedings of the 31st International Conference on Computational Linguistics: Industry Track, pages 511–521
January 19–24, 2025.

et al., 2024).

However, despite the impressive performance of these methods aimed at reducing the cost of LLMs, they may still fall short of expectations in text classification tasks. The main reasons are: (1) Methods like Batch Prompting, which annotates a batch of data in a single API call, only reduce the average length of the system prompt. However, in text classification tasks, the text often contains a significant amount of redundant information that contributes little to the classification. Therefore, the cost savings of Batch Prompting methods are still limited. (2) Many prompt compression methods focus on overcoming context window limitations, enabling LLMs to process longer texts (Jiang et al., 2023b; Li et al., 2023; Jung and Kim, 2023). However, our goal is to annotate high-quality data at a lower cost to train an excellent small language model as a text classifier. As a result, existing prompt compression methods are not well-suited to our task.

To address the aforementioned problem, we propose a **Prompt Compression method Based on Keyword-Extraction (PCKE)**. Specifically, our prompt compressor is a keyword extraction model based on a fine-tuned BERT. When invoking the large language model for annotation, we replace the original text with the extracted keywords to achieve compression. Meanwhile, to enhance the relevance of the extracted keywords to the sentence categories, we incorporated a classification task in the keyword extraction model.

The process of corpus annotation is conducted in tandem with the training of the compression model through an iterative approach. Initially, we call an LLM to extract keywords and category labels for a subset of the training set, thereby training an initial compression model. This model is then utilized to compress a new batch of unlabeled data, which is subsequently fed back into the LLM for annotation. The resulting annotated data is integrated into the final annotated corpus, while also serving as training data for an enhanced compression model. This iterative process continues until all the unlabeled data is annotated. It is noteworthy that the final compression model itself can serve as a baseline classification model for online deployment.

To evaluate the effectiveness of our approach, we conducted extensive experiments on multiple text classification datasets. The experimental results demonstrate that our method achieves the highest annotation accuracy among all the compared methods, closely matching the performance obtained with uncompressed prompts. This indicates that PCKE has a superior capability to perceive essential and task-specific information, resulting in minimal information loss. Moreover, it achieves superior accuracy while requiring fewer tokens, demonstrating its excellent compression capabilities. Compared to the current state-of-the-art method, LLMlingua2 (Pan et al., 2024), our approach achieves an average accuracy improvement of 4.4% at the same compression rates. Furthermore, it preserves 97.7% of the original uncompressed prompting performance while only consuming 33.1% of the tokens required by the original prompt.

Overall, our main contributions can be summarized as follows,

- To the best of our knowledge, we are the first to propose a prompt compression method specifically designed for text classification tasks.
- Our approach demonstrates significant advantages in both accuracy and compression rate compared to general compression methods.
- By substituting the extracted keywords for the original text, we have verified that a minimal set of keywords encompasses the majority of the information necessary for classification tasks, especially for LLM.

2 Related Works

Prompt Compression Methods aim to enhance the efficiency of LLMs by significantly reducing the length and complexity of prompts while preserving essential information for accurate responses. Based on the use of task-specific information, these methods can be classified into task-aware and task-agnostic approaches.

Task-agnostic compression methods do not rely on task-specific information but instead use general compression techniques to simplify input prompts. For example, information entropy-based methods (Li et al., 2023; Jiang et al., 2023a) use a Small Language Model (SLM) to evaluate the information entropy of each word to remove redundant content. LLM-based methods directly use a well-trained LLM for compression (Pan et al., 2024), or fine-tune the LLM (Ge et al., 2023). However, since the compressor is unable to discern which information is critical to the task, this method may lead to a decline in downstream task performance.

Moreover, compression methods based on LLMs are not suitable for text classification tasks because they require high computational overhead.

Task-aware compression achieves efficient compression by identifying and extracting information highly relevant to specific tasks, thereby providing more precise context and task-related information. This ensures that the model maintains high performance even when handling complex tasks. For example, LongLLMLingua (Jiang et al., 2023b) employs a question-aware compression technique that operates in a coarse-to-fine manner to estimate the information entropy of tokens. It dynamically adjusts the estimation based on the specific question. Reinforcement Learning (RL) is also utilized to train a model for prompt compression using reward signals from current prompt (Jung and Kim, 2023) or downstream tasks (Huang et al., 2023b).

However, these methods do not achieve the mutual promotion between downstream tasks and prompt compression. Our approach differs in that we simultaneously train a prompt compressor and a classifier for downstream tasks, allowing them to mutually reinforce each other, thereby enhancing the performance of both tasks.

3 Method

Our goal in this study is to develop a text classifier that achieves high accuracy while being cost-efficient. To this end, we introduce PCKE, a novel approach capable of simultaneously executing prompt compression and text classification. The subsequent sections will detail our method, which is structured around two essential components.

3.1 Initial Annotation for Prompt Compression and Text Classification

Firstly, we have a purely unsupervised training set $D_{unlabeled}$. At the initial round of PCKE, we randomly sample D_0 from $D_{unlabeled}$ and annotate it, resulting in $D_0^{'}$. The remaining data is denoted by D_{rest}. Then we incorporate $D_0^{'}$ to the training set D_{train}. For the annotation method, inspired by Chain-of-Thought (Kojima et al., 2022) and CARP (Sun et al., 2023b), we enable LLMs to annotate category labels and extract keywords simultaneously. This not only helps to improve the annotation accuracy but also makes it possible to train the SLM for subsequent prompt compression tasks. In the annotation step, we may directly perform zero-shot In-Context Learning (ICL). However, the quality of annotated data has an essential influence on the subsequent step of training SLM. To enhance the quality of our annotation, we employ the self-consistency method as described by Wang et al. (2022). In this approach, the keywords are extracted multiple times, and the final set of keywords is obtained by taking the union of these multiple extractions. This ensures a more comprehensive and reliable set of keywords, ultimately leading to better training outcomes for the SLM.

Algorithm 1 PCKE Procedure

Input: D_i denotes unlabeled dataset sampled from $D_{unlabeled}$. D_{rest} denotes the rest of the unlabeled dataset, SLM_i denotes the SLM after i-th iteration. k denotes the amount of data required in each round and N denotes the number of iterations. CMP_D_i denotes the compressed dataset.

Output: Cmp_i denotes the prompt compressor, Clf_i denotes the text classifier. They are one SLM that can perform two tasks.

$D_0, D_{rest} \leftarrow D_{unlabeled}$
$D_0^{'} \leftarrow \textsc{Annotate}(D_0, LLM)$
$D_{train} \leftarrow D_0^{'}$
$Cmp_0, Clf_0 \leftarrow \textsc{Train}(D_{train}, SLM)$
for $i = 1 \rightarrow N$ **do**
  $D_i, D_{rest} \leftarrow \textsc{Split}(D_{rest}, k)$
  $CMP\_D_i \leftarrow \textsc{Compress}(D_i, Cmp_{i-1})$
  $D_i^{'} \leftarrow \textsc{Annotate}(CMP\_D_i, LLM)$
  $D_{train} \leftarrow D_{train} + D_i^{'}$
  $Cmp_i, Clf_i \leftarrow \textsc{Train}(D_{train}, SLM)$
end for
return Cmp_N, Clf_N

3.2 Knowledge Distillation to SLMs

After obtaining the annotated dataset from LLM, the subsequent step is to train an SLM that can perform both classification and prompt compression tasks simultaneously. For the classification task, it is straightforward. We use an MLP classifier on the embedding vectors from the SLM encoder. As for the prompt compression task, we consider it as the extraction of keywords that are beneficial for classification. Consequently, we carry out a word-level binary classification to decide whether each word should be kept or not. Eventually, the redundant information that makes no contribution to classification can be removed, and the length of the prompt will be significantly reduced.

3.2.1 Architecture of SLMs

Formally, we utilize a BERT (Devlin et al., 2019) encoder as the feature encoder f_θ and two MLPs for sentence-level classification and word-level classification. Given an original text x_i consisting of N words $x_i = \{x_i^1, x_i^2, ..., x_i^N\} = \left\{x_i^j\right\}_{j=1}^N$, let $\tilde{y_i}$ denote the category label of x_i and $y_i = \left\{y_i^j\right\}_{j=1}^N$ denote the corresponding labels for all words in x_i.

$$h_i = f_\theta(x_i)$$
$$h_i^0 = \sum_{j=1}^{N} h_i^j$$

where $h_i = \left\{h_i^j\right\}_{j=1}^N$ denotes feature vectors for all words that are used for prompt compression and h_i^0 represents the CLS vector used for classification.

Then, by applying Softmax function and MLPs, we can get

$$p_{cls}(x_i) = \text{softmax}(\text{MLP}_1(h_i^0))$$
$$p_{kwd}(x_i^j) = \text{softmax}(\text{MLP}_2(h_i))$$

where $p_{cls}(x_i) \in \mathbb{R}^C$ represents the probability distribution of the original text category and $p_{kwd}\left(x_i^j\right) \in \mathbb{R}^2$ denotes the probability distribution of labels $\{preserve, discard\}$ for j-th word of text x_i. For a sample x_i, the classification loss and keyword loss can be defined as follows:

$$l_{cls} = \text{CrossEntropy}(\tilde{y_i}, p_{cls}(x_i))$$
$$l_{kwd} = \text{CrossEntropy}(y_i, p_{kwd}(x_i))$$

Where l_{cls} represents the classification loss and l_{kwd} represents the keyword loss.

3.2.2 Robust Training of SLMs

The total loss for training SLM can be calculated by directly summing the classification loss and the keyword loss. Nevertheless, datasets annotated by LLMs may be noisy, which will lead to the performance degradation of SLM. Fortunately, previous studies (Han et al., 2018; Li et al., 2020) in weakly supervised learning have demonstrated that deep models have the potential to detect noisy samples during the training process. Therefore, we adopt a selection-based technique (Li et al., 2020) to develop a robust SLM for classification and prompt compression. Specifically, after several warm-up epochs with standard training on the noisy dataset, the cross-entropy l_i can indicate how well the SLM fits each sample x_i. We use a Gaussian Mixture Model (GMM) with two components to model the loss l_i, allowing us to distinguish between clean and noisy samples. Let $w_i = p(g|l_i)$ denote the probability of x_i belonging to the Gaussian component with smaller mean g, which is interpreted as its clean probability. By setting a threshold τ on w_i, the training set can be divided into a clean subset D_{clean} and anoisy subset D_{noisy}

$$D_{clean} = \{(x_i, y_i, \tilde{y_i}) \mid x_i \in D_{train}, w_i \geq \tau\}$$
$$D_{noisy} = \{(x_i, y_i) \mid x_i \in D_{train}, w_i < \tau\}$$

During the training process, we adopt different loss strategies for noisy samples and clean samples, under the assumption that the keywords extracted by the LLM are less likely to be wrong. Specifically, for samples from the clean subset D_{clean}, we calculate both classification loss and keyword loss, while for samples in the noisy subset D_{noisy}, we only calculate keyword loss. This can be formulated as:

$$L_{clean} = \frac{1}{|D_{clean}|} \sum_{x_i \in D_{clean}} (l_{cls}(x_i) + l_{kwd}(x_i))$$
$$L_{noisy} = \frac{1}{|D_{noisy}|} \sum_{x_i \in D_{noisy}} l_{kwd}(x_i)$$
$$L_{total} = L_{clean} + \alpha L_{noisy}$$

Where l_{cls} and l_{kwd} are classification loss and keyword loss which have been defined in section 3.2.1. L_{clean} and L_{noisy} are the total loss of clean samples and noisy samples respectively. α is the loss weight parameter which can be learned automatically to balance L_{clean} and L_{noisy}.

3.3 Low-cost Annotation and SLM Training

After the initial round of annotation and Knowledge Distillation to SLMs, we have an SLM that can be used for prompt compression. Then, we perform inference on the remaining train set D_{rest} and obtain compressed text. If the remaining training set is not large, we will compress and annotate all the remaining data, merge the resulting data with D_{train}, and train the SLM again. Alternatively, we can also compress and annotate part of the remaining training set, then incorporate the resulting data to D_{train} and train the SLM. This process can be executed several times. It is worth noting that by doing so, we can obtain the SLM with higher classification accuracy at a lower annotation cost.

4 Experiment

In this section, we present comprehensive experimental results to validate the efficacy of our proposed method (PCKE). For more detailed information, please refer to Appendix A.

4.1 Datasets and Implementation Details

We conduct experiments on four classification datasets: MR (Pang and Lee, 2005), Amazon-531 (McAuley and Leskovec, 2013), AGNews (Zhang et al., 2015), and Inshort-News (Xu et al., 2020). Table 1 provides the basic statistics for these datasets. MR is a movie review sentiment classification dataset with two categories. Amazon-531 contains 142.8 million product reviews, we sampled 22,000 reviews from six product categories for training and testing. AGNews includes news articles in four categories, with 30,000 training samples and 1,900 test samples per class. We randomly selected 5,000 articles from each category for the training set, totaling 20,000 articles. Inshort-News offers brief summaries of news articles across seven categories, with 4,000 entries for training and 1,000 for testing. For the MR and Inshort-News datasets, we utilized all texts from the original training sets.

Datasets	MR	Amazon-531	AGNews	Inshort-News
#Categories	2	6	4	7
#Train	8,530	20,000	20,000	4,000
#Test	1,066	2,000	7,600	1,000

Table 1: Basic statistics of the four classification datasets.

We employ the OpenAI GPT-4o model for all experiments. In Round1 (Initial Annotation for Prompt Compression and Text Classification), we randomly selected and annotated 2000 samples from $D_{unlabeled}$ as the initialized dataset D_{train}, while the remaining data are assigned to D_{rest}. The decoding temperature is set to 0. Following Cheng et al. (2023), we set batch size to 20. The compression rate τ is defined as the quotient of the number of words in the compressed text and the number of words in the original text.

4.2 Compared Methods and Metrics

To validate the efficacy of our proposed method (PCKE), we compare our method against several state-of-the-art methods. **Selective-Context** (Li et al., 2023) is a model-agnostic approach, which identifies and prunes redundant content using self-information computed by LLaMa-2-7B. **KeyBERT** (Grootendorst, 2020) is a general keyword extraction method. We adopted it directly for prompt compression as one of the baselines. **LLMLingua2** (Pan et al., 2024) identify preserved tokens by LLMs such as GPT-4 and train a Transformer encoder to compress prompts.

For the evaluation metrics, we utilized classification accuracy. Specifically, we assessed the classification performance of the SLM by using its classification accuracy. To evaluate the compression performance of the SLM, we compared the accuracy of the LLM's annotations on the text before and after compression.

4.3 Main Results

Given a fixed compression rate, we utilized these methods to generate compressed prompts for data annotation using GPT-4o. Subsequently, we compared the annotation accuracy and the number of input tokens consumed. The experimental results, presented in Table 2, demonstrate that our method achieved the highest annotation accuracy among all the compression methods, closely matching the performance obtained with uncompressed prompts. This indicates that PCKE has a superior capability to perceive task-specific information, resulting in minimal information loss in the compressed prompts. Furthermore, our method maintains higher accuracy while requiring fewer tokens, demonstrating superior compression capabilities.

Moreover, the result of employing our method for data annotation and subsequently training a classifier with this annotated data is shown in Table 3. The performance of the SLM trained using our annotated dataset is comparable to that of the model trained with data annotated using the original prompts. Through the iterative training, we observed a gradual and mutual enhancement between the compressor and the classifier. The compressor is aware of the task-specific information, which helps to identify the most significant content. Meanwhile, with explicit supervision from the keywords, the classifier can better concentrate on the critical elements of the text, thereby improving its classification ability. More detailed results of the iterative process can be found in section 4.4.

4.4 Effect of Increase Iterations

In PCKE, we first annotate a small portion of the training set and train the SLM. After that, we perform the prompt compression task, annotate the remaining data, and train the SLM again. However,

Method	MR			Amazon-531			AGNews			Inshort-News		
	acc	τ	Tokens	acc	τ	Tokens	acc	τ	Tokens	acc	τ	Tokens
Original Prompt	94.22	-	28.6	88.78	-	93.5	88.74	-	57.7	78.21	-	88.2
Compressed Prompt												
Selective-Context	76.61	0.3	14.6	80.43	0.15	20.0	85.84	0.3	20.2	73.24	0.3	27.6
KeyBERT	77.13	0.3	16.7	84.57	0.15	25.2	86.39	0.3	23.7	76.05	0.3	32.2
LLMLingua2	80.45	0.3	15.8	82.44	0.15	25.1	87.30	0.3	24.3	74.14	0.3	32.3
PCKE(ours)	**89.62**	0.3	15.2	**86.73**	0.15	20.5	**88.10**	0.3	23.2	**77.39**	0.3	29.9

Table 2: Comparison of Text Classification Accuracy using GPT-4o with Different Prompt Compression Methods. The PCKE Method Utilizes a Prompt Compressor that is Trained in Round 2. τ Refers to the Compression Rate, Indicating the Proportion of Words in the Compressed Prompt Relative to the Original Prompt. $Tokens$ Indicates the Average Number of Tokens Consumed for Annotating each Individual Instance.

Model	Round	Cmps/Annos	MR (τ=0.50)	Amazon-531 (τ=0.15)	AGNews (τ=0.30)	Inshort-News (τ=0.30)
GPT4o	1	Original Prompt	94.22	89.38	88.74	78.21
	3	Compressed Prompt	91.68	88.43	88.10	77.39
BERT	2	Annotated by Round 1	88.00	73.35	87.50	78.00
	4	Annotated by Round 2	89.50	74.90	88.45	79.30

Table 3: Comparison of Annotation Accuracy of PCKE and Classification Accuracy of Fine-tuned BERT on Test Set Across Different Rounds.

when the dataset is too large, instead of annotating all the remaining data at once, we can annotate the data and train the SLM in multiple rounds. This approach may lead to better performance of the SLM at a lower cost. In this section, we conduct experiments on the Amazon-531 dataset and the MR dataset to verify this conjecture. We set the step size to 2000, meaning we randomly chose 2000 pieces of data to compress and annotate in each iteration. The results in Table 4 demonstrate that PCKE consistently enhances the classifier's performance over successive iterations. This indicates that our proposed PCKE method has the potential to significantly reduce costs, with the savings becoming increasingly pronounced as the amount of large-scale annotated data grows.

Iterations	Amazon-531	MR
1	0.733	0.880
2	0.738	0.894
3	0.750	0.895
4	0.750	0.896
5	0.754	-
6	0.752	-
7	0.760	-

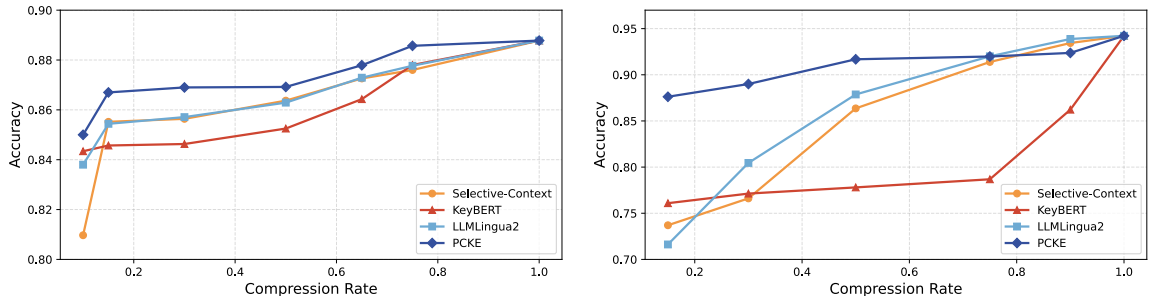
Table 4: Accuracy of the BERT-based Classifier Across Multiple Iterations on Amazon-531 and MR Datasets. The MR dataset lacks sufficient data to support 5, 6, and 7 rounds of iterations.

4.5 Effect of Different Compression Rates

To evaluate the impact of varying compression rates on annotation accuracy, we conduct comprehensive experiments using the Amazon-531 and MR datasets. Consistent with the main experiment, we first randomly select and annotate 2000 data from the training set to train the SLM. After that, we perform compression and annotation on the test set sequentially. By adjusting the threshold for token retention, we can achieve varying compression ratios. Finally, the annotation accuracy is used as the evaluation metric. The result is shown in Figure 2. It can be seen that, as the compression rate decreases, all methods exhibit a decreasing trend, indicating that each compression method incurs a certain degree of information loss. However, our method demonstrates the smallest and slowest decline among all methods. This is particularly noticeable when the compression rate is very low, our method demonstrates great preservation of annotation accuracy in such cases. This observation proves that our approach is better at capturing the essential information from the original prompt, thereby achieving more effective compression performance.

5 Conclusion

We propose a prompt compression method for leveraging LLM in text classification data annotation tasks, significantly reducing annotation costs by minimizing token usage. A fundamental ob-



(a) Amazon-531 (b) MR

Figure 2: Compression Rate vs. Accuracy of Different Methods.

servation that drives this methodology is that for LLMs with powerful reasoning, a minimal set of core keywords is adequate to identify the sentence category. The compression model is a multi-task framework based on BERT, which performs both keyword extraction and classification. By concurrently annotating and optimizing the compressor, we ultimately obtain a fully annotated dataset as well as a baseline classification model suitable for online deployment. Experimental results demonstrate that our compression method surpasses general approaches in both accuracy and compression ratio, making it particularly beneficial for teams operating under strict budget constraints.

6 Discussion

In our experiments, we validated our method on text classification datasets with up to seven categories. However, in real-world industrial applications, the number of text categories can be significantly larger, and the classification performance after compression may be influenced by the number and distribution of classes in the dataset. Since our method focuses on compressing the original text based on classification, achieving more accurate classification results leads to better compression outcomes. This issue can be addressed by designing more powerful classifiers, such as using a larger BERT-style model or a model fine-tuned with domain-specific data. Nonetheless, the effectiveness of this approach requires further exploration in the future.

Despite the good performance of PCKE, our method does have some limitations. Firstly, the model does not show significant advantages in short text classification tasks. At the same time, our approach is particularly effective for tasks that can be distinguished based on keyword entities, such as e-commerce, news, or game classification. However, for tasks that require understanding the entire sentence to make a judgment, such as humor detection or more nuanced emotional classification, its performance may be average. For these cases, a potential compression method could involve using an LLM to generate a brief summary and then training a summarization-based SLM. This needs to be explored in future work. Additionally, poor compression may result in the LLM being unable to determine the category. If this issue frequently occurs with a specific pattern of samples, the resulting annotated dataset will lack such samples, leading to poor performance of the final classifier on these types of patterns.

References

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 neural information processing systems*, 33:1877–1901.

Zhoujun Cheng, Jungo Kasai, and Tao Yu. 2023. Batch prompting: Efficient inference with large language model APIs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 792–810, Singapore. Association for Computational Linguistics.

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.

Tao Ge, Jing Hu, Xun Wang, Si-Qing Chen, and Furu Wei. 2023. In-context autoencoder for con-

text compression in a large language model. *ArXiv*, abs/2307.06945.

Maarten Grootendorst. 2020. Keybert: Minimal keyword extraction with bert.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. 2018. Co-teaching: robust training of deep neural networks with extremely noisy labels. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 8536–8546, Red Hook, NY, USA. Curran Associates Inc.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. 2021. Pre-trained models: Past, present and future. *AI Open*, 2:225–250.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023a. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Xijie Huang, Li Lyna Zhang, Kwang-Ting Cheng, and Mao Yang. 2023b. Boosting llm reasoning: Push the limits of few-shot learning with reinforced in-context pruning. *arXiv preprint arXiv:2312.08901*.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023a. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023b. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In *The 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*. ACL.

Hoyoun Jung and Kyung-Joong Kim. 2023. Discrete prompt compression with reinforcement learning. *IEEE Access*, 12:72578–72587.

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.

Junnan Li, Richard Socher, and Steven C. H. Hoi. 2020. Dividemix: Learning with noisy labels as semi-supervised learning. *ArXiv*, abs/2002.07394.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore. Association for Computational Linguistics.

Jianzhe Lin, Maurice Diesendruck, Liang Du, and Robin Abraham. 2023. Batchprompt: Accomplish more with less. *arXiv preprint arXiv:2309.00384*.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, Dongmei Zhang, Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, and Reiichiro Nakano. 2024. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. *ArXiv*, abs/2403.12968.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with" gradient descent" and beam search. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Chinese computational linguistics: 18th China national conference, CCL 2019, Kunming, China, October 18–20, 2019, proceedings 18*, pages 194–206. Springer.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023a. Text classification via large language models. *arXiv preprint arXiv:2305.08377*.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023b. Text classification via large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8990–9005.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Song Xu, Haoran Li, Peng Yuan, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. Self-attention guided copy mechanism for abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1355–1362, Online. Association for Computational Linguistics.

Xin Xu, Yue Liu, Panupong Pasupat, Mehran Kazemi, et al. 2024. In-context learning with retrieved demonstrations for language models: A survey. *arXiv preprint arXiv:2401.11624*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *ArXiv*, abs/2309.03409.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

A Appendix

A.1 Prompts

In this part, we show the prompts for Amazon-531 dataset, which are shown in Figure3. Prompts for other datasets are similar to those for Amazon-531 so we omit these details. It should be noted that prompts used for the initial round of annotation and subsequent rounds of annotation differ slightly. Additionally, in the initial round, we input the original text to the LLM, while in subsequent rounds, we input a set of keywords separated by commas to the LLM.

A.2 Case Study

In this part, we show two compression examples using the above four methods(Selective-Context, KeyBERT, LLMLingua2 and our proposed PCKE) on Amazon-531 dataset and AGNews dataset. The results can be found in Figure 4. For Amazon-531 dataset, we set the compression rate to 0.15 while for AGNews dataset, we set the compression rate to 0.30. Words that are useful for classification are highlighted in blue. As can be seen, only a few words in the original text are helpful for classification. This also shows that it is reasonable to compress the original text before annotating it. Comparing the four methods, our method obtains the most useful words, thus achieving higher annotation accuracy.

Prompts Used for Amazon-531 Dataset

The system prompt for the initial round of annotation:
You are a data annotation expert in the field of e-commerce reviews. Please extract keywords based on the original text, and then classify the data into the following 6 categories based on the keywords: 'toys games', 'health personal care', 'beauty', 'baby products', 'pet supplies', 'grocery gourmet food'

Input and output format description:
Input (one text to be annotated per line):
1, xxx
Output (one result per line, expressed in legal json format):
{"id": 1, "keyword": "keyword1,keyword2,...", "label":""}
Notes:
Please output one of the given labels in the label field, and do not output other labels.

The system prompt for the subsequent round of annotation:
You are a data annotation expert in the field of e-commerce reviews. You are given some keywords for e-commerce reviews. These keywords come from e-commerce reviews. Please classify the data into the following 6 categories based on these keywords: 'toys games', 'health personal care', 'beauty', 'baby products', 'pet supplies', 'grocery gourmet food' and filter out the keywords that can support your judgment.

Input and output format description:
Input (one text to be annotated per line):
1, xxx
Output (one result per line, expressed in legal json format):
{"id": 1, "keyword": "keyword1,keyword2,...", "label":""}
Notes:
Please output one of the given labels in the label field, and do not output other labels.
The keyword field can only output keywords that are helpful for classification and does not output keywords that are not helpful for classification.

Figure 3: The Prompts We Used for Amazon-531 Dataset.

Examples of four Compression Methods on the Amazon-531 Dataset and AGNews Dataset

Original Text: I bought this tent on 6/6/2005 for my 10 month old boy and used it once at the the rest of the time it sat in my dining room. I went to bed one night and it was up, but the next morning it was halfway down. When I went to see what was wrong I saw that one of the rods had snapped off in the metal holder and cannot be fixed. This could actually suffocate a baby if noone was in the room or the baby could hurt themselves if they found the broken rod. I was very disappointed because the size is great. I also agree with the other reviews it is hard to put together because the rods are so tight you have to have a lot of strenth to get it set up. Please don't buy this we are going to return but it's such a hassle to spend 39.99 plus tax on something and it breaks in less then a month

Category: baby products

Selective Context: sed up halfway could suffocate noone hurt the broken rod it alot strenth Please we return spend breaks

KeyBERT: broken,rod was,pool the,snapped off,suffocate baby,pool,set up,rods are,this tent

LLMLingua2: bought tent 6 2005 10 month used snapped suffocate baby hurtdisappointed size hard rods. 39. 99 tax breaks

PCKE(Ours): tent, 10 month old boy, pool, dining room, rods, baby, noone

---

Original Text: S and P watching Shell for possible debt downgrade LONDON : Standard and Poor #39;s Ratings Services said it had its eye on Royal Dutch/Shell for a possible downgrade of the oil company #39;s debt rating in case of a further restatement of its reserves.,

Category: Business

Selective Context: watching Shell possible debt : Standard and Poor #39;s Ratings Services its eye Royal Dutch/Shell case

KeyBERT: 9 debt rating,watching shell,dutch shell for,watching shell for,downgrade london standard,debt downgrade

LLMLingua2: P Shell debt LONDON Standard Poor 39 Dutch Shell downgrade reserves

PCKE(Ours): shell, debt downgrade, poor, ratings services, royal dutch/shell, oil company, reserves

Figure 4: Two Examples of 4 Compression Methods on Amazon-531 Dataset and AGNews Datase. Words that are useful for Classification are Highlighted in Blue.