LM-CPPF: Paraphrasing-Guided Data Augmentation for Contrastive Prompt-Based Few-Shot Fine-Tuning

Amirhossein Abaskohi[1], **Sascha Rothe**[2], **Yadollah Yaghoobzadeh**[1,3]
[1]School of Electrical and Computer Engineering
College of Engineering, University of Tehran, Tehran, Iran
[2]Google DeepMind, Zürich, Switzerland
[3] Tehran Institute for Advanced Studies, Khatam University, Iran
amir.abaskohi@ut.ac.ir, rothe@google.com, y.yaghoobzadeh@ut.ac.ir

Abstract

In recent years, there has been significant progress in developing pre-trained language models for NLP. However, these models often struggle when fine-tuned on small datasets. To address this issue, researchers have proposed various adaptation approaches. Prompt-based tuning is arguably the most common way, especially for larger models. Previous research shows that adding contrastive learning to prompt-based fine-tuning is effective as it helps the model generate embeddings that are more distinguishable between classes, and it can also be more sample-efficient as the model learns from positive and negative examples simultaneously. One of the most important components of contrastive learning is data augmentation, but unlike computer vision, effective data augmentation for NLP is still challenging. This paper proposes LM-CPPF, Contrastive Paraphrasing-guided Prompt-based Fine-tuning of Language Models, which leverages prompt-based few-shot paraphrasing using generative language models, especially large language models such as GPT-3 and OPT-175B, for data augmentation. Our experiments on multiple text classification benchmarks show that this augmentation method outperforms other methods, such as easy data augmentation, back translation, and multiple templates.[1]

1 Introduction

Pre-trained language models (PLMs) are trained on large-scaled corpora in a self-supervised fashion. They have fundamentally changed the NLP community in the past few years by achieving impressive results in various Tasks (Devlin et al., 2018; Radford et al., 2018; Yang et al., 2019; Chiang et al., 2022). However, when PLMs are fine-tuned on small datasets, their performance declines. Researchers have proposed various techniques to adapt PLMs to these scenarios (Snell et al., 2017; Sung et al., 2018). In addition to performance, fine-tuning PLMs to learn a new task is parameter inefficient, because an entirely new model is required for every task (Houlsby et al., 2019).

By the introduction of GPT-3 (Brown et al., 2020b) with 175B parameters, it has been shown that Large Language Models (LLMs) are efficient few-shot learners as they can use their knowledge more effectively. One of the key features of these LLMs is their ability to perform multiple tasks using prompts. A language prompt is a piece of text that is added to the input query to help the model make more accurate predictions. In addition, LLMs can be fine-tuned for specific tasks using few examples. This has made them powerful tools for NLP tasks, especially in few-shot scenarios. However, that might not be practical for many situations because of the model size. Therefore, there is a need to adapt smaller PLMs to work in a similar way to LLMs.

Prompt-based fine-tuning is a method for adapting PLMs to specific tasks or domains by providing a prompt (Schick and Schütze, 2020a,b). This approach has been shown to be effective in various NLP tasks, including text classification (Han et al., 2021; Wang et al., 2022) and question answering (Yao et al., 2022). However, it can be challenging to achieve strong performance when only a few examples are available for each task. Gao et al. (2020) introduced a prompt-based fine-tuning method called LM-BFF for RoBERTa (Liu et al., 2019) to tackle this issue. Their approach includes automated prompt generation and a more effective way of using task examples in fine-tuning.

Building on the success of LM-BFF and considering contrastive learning's promising results both in computer vision (Chen et al., 2020) and NLP (Chen et al., 2020; Miao et al., 2021), Jian et al. (2022) present a contrastive learning framework to improve LM-BFF. They propose a Supervised Contrastive Learning (SCL) approach (Khosla et al.,

[1] Our implementation is publicly available at: https://github.com/AmirAbaskohi/LM-CPPF

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 670–681
July 9-14, 2023 ©2023 Association for Computational Linguistics

2020) that classifies inputs using different augmented views of the data. These views are created using different templates for their demonstrations when building prompts.

In this paper, we show that while SCL at the feature space can be beneficial, the use of different templates can limit the full potential of this approach. We propose **LM-CPPF** (Contrastive Paraphrasing-guided Prompt-based Fine-tuning of Language Models), in which we integrate the knowledge of LLMs like GPT-3 and OPT-175B (Zhang et al., 2022) to build different views using paraphrasing. These models can generate paraphrases of a sentence with different syntax, not just by changing the lexicalization. Previous studies have considered generating paraphrases a challenging and costly NLP task (Siddique et al., 2020; Garg et al., 2021; Zhou and Bhat, 2021). However, PLMs can generate paraphrases easily and effectively using in-context learning with few examples. Although prior research has studied paraphrase generation with PLMs (Roy and Grangier, 2019; Hegde and Patil, 2020), to the best of our knowledge, this is the first time that large LLMs are utilized to generate paraphrases with prompts as an augmentation method. Our experiments on six different text classification tasks demonstrate that LM-CPPF outperforms the previous SOTA methods of data augmentation in prompt-based fine-tuning, including Easy Data Augmentation (EDA) (Wei and Zou, 2019), Back Translation (BT) (Sugiyama and Yoshinaga, 2019), and multiple templates (Jian et al., 2022).

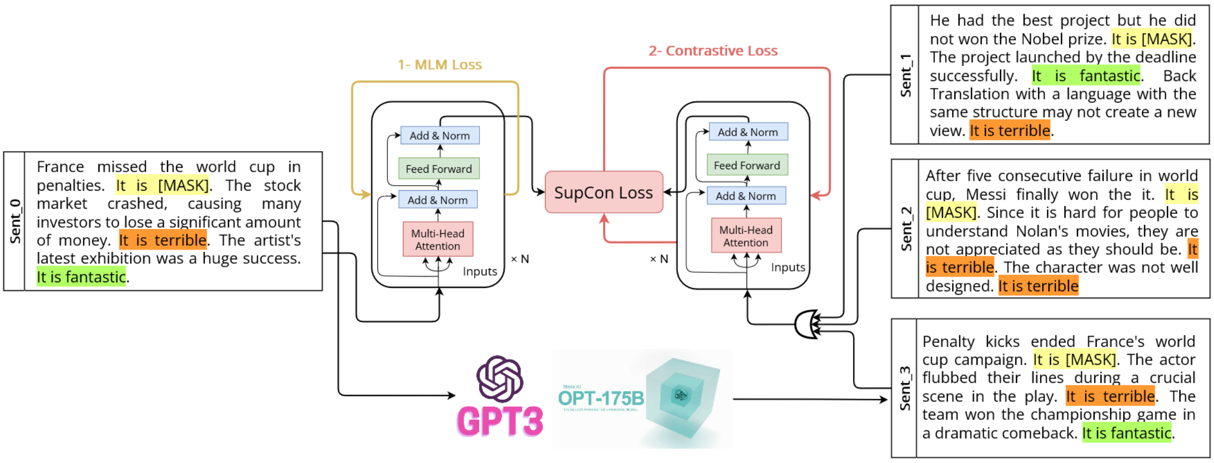
2 Related Works

LLMs like GPT-3 (Brown et al., 2020a) can perform NLP tasks with few examples and natural prompts. But smaller models are not efficient with this approach and there are data sparsity and prompt sensitivity issues. To address these challenges, Gao et al. (2021) propose LM-BFF, a framework that leverages a large PLM to automatically generate task-specific prompts for smaller models. It improves their few-shot performance on different NLP tasks. Some work have enhanced LM-BFF with different prompt tuning methods. For example, Zhou et al. (2022) present a dual context-guided continuous prompt tuning method that uses the language context and connects discrete and continuous prompt tuning. Jian et al. (2022) integrate contrastive learning and data augmentation with LM-BFF. In their contrastive part, in addition to comparing different instances from the same or different classes, they introduced a novel prompt-specific augmentation method. In their approach, they change the template of the prompt. In this paper, we use few-shot paraphrasing with LLMs for contrastive prompt-tuning, which fine-tunes models with natural prompts.

Paraphrasing is the task of expressing the same meaning with different words or structures. It can be used to create training data with increased diversity and naturalness for NLP tasks, such as text classification (Xie et al., 2020), natural language inference (Kumar et al., 2019), and text summarization (Loem et al., 2022), surpassing the limitations of traditional approaches. Paraphrasing helps with data scarcity and model generalization. There are different ways to generate paraphrases for data augmentation. One is back-translation (Sennrich et al., 2016), which uses a translation system to convert a sentence to another language and back. Another is to use paraphrasing models trained on parallel paraphrase datasets (Wieting and Gimpel, 2018; Zhu et al., 2022). PLMs can also generate paraphrases by using large-scale corpora, but they may produce paraphrases that are not semantically consistent or relevant. LLMs can reduce this problem as they encode and generate language better. In this paper, we generate paraphrases by carefully prompting LLMs and then use them for data augmentation.

3 Method

Background Contrastive learning's success relies on data augmentation, which creates new views of the input data. Contrastive learning has been utilized for various tasks in deep learning (Le-Khac et al., 2020; Conde and Turgutlu, 2021; Abaskohi et al., 2022); however, most NLP data augmentation methods may influence semantics which results in limited improvement. For instance, EDA's synonym substitution may create entirely new samples since words do not have equal senses (Keselj, 2009). In addition to these augmentation methods, the approach used in Jian et al. (2022) cannot be counted as data augmentation as the sample is still the same and only the template for the verbalizer changes. Although it is a creative approach designed specifically for the prompt-based method of LM-BFF, it is limited in performance even compared to EDA in several benchmarks. Furthermore, it requires an expert to create multiple templates



Sent_0: France missed the world cup in penalties. It is [MASK]. The stock market crashed, causing many investors to lose a significant amount of money. It is terrible. The artist's latest exhibition was a huge success. It is fantastic.

Sent_1: He had the best project but he did not won the Nobel prize. It is [MASK]. The project launched by the deadline successfully. It is fantastic. Back Translation with a language with the same structure may not create a new view. It is terrible.

Sent_2: After five consecutive failure in world cup, Messi finally won the it. It is [MASK]. Since it is hard for people to understand Nolan's movies, they are not appreciated as they should be. It is terrible. The character was not well designed. It is terrible

Sent_3: Penalty kicks ended France's world cup campaign. It is [MASK]. The actor flubbed their lines during a crucial scene in the play. It is terrible. The team won the championship game in a dramatic comeback. It is fantastic.

Figure 1: Our method, LM-CPPF, includes two objectives: (I) MLM and (II) Supervised Contrastive Learning. The target sentence is the first sentence in each prompt with a [MASK] token. The target sentence of Sent_0 is used to train our model and calculate the MLM loss. We build Sent_3, whose target sentence is a paraphrase of Sent_0's target sentence. Sent_1 and Sent_2, sampled from the dataset, have target sentences in the same and different classes as Sent_0, respectively.

for each task, which makes it challenging for newly emerged tasks. Here we propose leveraging LLMs to generate paraphrases and introduce LM-CPPF, a novel approach aimed at addressing the challenges associated with contrastive prompt-based fine-tuning of PLMs.

Few-shot paraphrasing Paraphrasing is one of the best methods for data augmentation in NLP. One of the most popular approaches for paraphrasing is back-translation (BT) (Sugiyama and Yoshinaga, 2019) due to its simplicity and efficiency. Nonetheless, BT's performance depends a lot on the intermediary language. In this paper, we, instead, use a combination of prompt-learning and LLMs for paraphrasing. In few-shot paraphrasing, an LLM rewrites a sentence given an instruction and a few examples. We believe that LLMs generate high-quality paraphrases due to their encoded semantic and sentence structure knowledge. We utilize GPT-3 (Brown et al., 2020b) or OPT-175B (Zhang et al., 2022) via their official APIs [2] for generating paraphrases.

To avoid violating the prompt-based fine-tuning settings, we do not include any additional task data in generating our paraphrases. Following the few-shot setting in LM-BFF, we assume to have access to a PLM M, datasets $\mathcal{D}_{train}$, and $\mathcal{D}_{test}$ with label space $\mathcal{Y}$ where there are only $\mathcal{K} = 16$ examples per class in $\mathcal{D}_{train}$. We use this setting for both prompt-based few-shot paraphrasing and fine-tuning. To generate paraphrases, excluding the one sample that we want to paraphrase, we use QuillBot[3] to create paraphrases for our prompts for the remaining 15 samples in the same class of $\mathcal{D}_{train}$. We leverage two types of prompts for paraphrasing: (I) **Only Demonstration:** Here, the samples and their paraphrased versions are given using the templates in Table C.3 to demonstrate the task of paraphrasing. (II) **Demonstrations with Instruction:** In addition to the previous method, this one includes instructions at the beginning of the prompt, defining paraphrasing before demonstrations. These instructions can be seen in Table C.4.

Contrastive prompt-based fine-tuning LM-CPPF consists of two steps. The first step involves calculating the Masked Language Modeling (MLM) loss by using the target sentence in the given template, the specific demonstrations in the prompt, and the verbalizer matched with the target sentence's label. We calculate the supervised contrastive loss in the second step by comparing the target prompt with another sample with the same template but different random demonstrations. This comparison sample can be in the same or a different class as the target prompt. When the comparison sample belongs to a different class, it is randomly sampled from the dataset. However, in cases where the comparison sample belongs to the same class, an alternative approach is employed. This involves either selecting another sample from the same class

[2] OPT-175B: opt.alpa.ai and GPT-3: openai.com/api

[3] quillbot.com

Task	LM-BFF	LM-BFF+ SupConLoss	LM-BFF+ Multi-templates	LM-CPPF GPT-3	LM-CPPF OPT	LM-CPPF GPT-2	LM-CPPF FT GPT-2
SST-2	89.5	90.3	91.0	**92.3**	91.8	91.1	91.4
SST-5	48.5	49.6	50.3	**52.8**	52.2	51.4	51.6
MNLI	62.3	63.2	64.8	**68.4**	66.2	65.6	65.8
CoLA	6.9	9.6	11.6	**14.1**	13.3	10.7	11.8
QNLI	61.2	65.4	67.2	**69.2**	68.5	67.5	67.8
CR	89.7	89.9	90.2	**91.4**	91.1	90.2	90.7

Table 1: Performance of LM-CPPF and our baselines in six datasets. LM-BFF+Multi-templates refers to Jian et al. (2022). LM-BFF+SupConLoss uses the same architecture of LM-BFF+Multi-templates, but without any data augmentation, just integrating supervised contrastive and MLM loss functions. Two cases are available for GPT-2: the pre-trained model and the GPT-2 fine-tuned (FT) on ParaNMT-50M (Wieting and Gimpel, 2018) dataset. LM-BFF, LM-BFF+Multi-template, and LM-CPPF (on average for all models used for paraphrasing) have 0.77 and 1.02, and 1.65 standard deviations on average for each task, respectively.

within the dataset or applying data augmentation techniques, paraphrasing in our case, to augment the target sample in order to create a new view of it. In both of these cases, the demonstrations are not the same. Figure 1 illustrates the fine-tuning process, and Algorithm D.1 shows our methodology when paraphrasing creates a new view of the target sample. See Appendix D for more information.

4 Experiments

Evaluation datasets and protocol Our method is evaluated on six different classification tasks from LM-BFF (Liu et al., 2021). The reported numbers represent the average accuracy from five runs using Roberta-base (Liu et al., 2019). In Section 4.1 where LLMs are compared for paraphrasing, we also employed pre-trained and fine-tuned GPT-2 as an additional model for paraphrasing, allowing us to leverage smaller models in our experiments. For the fine-tuning of GPT-2 specifically for paraphrasing, we utilized the ParaNMT-50M (Wieting and Gimpel, 2018) dataset. More details regarding the training process can be found in Appendix A.

4.1 Paraphrasing in Prompt Fine-tuning

This section presents the results of our fine-tuning approach using paraphrasing on various NLP tasks. As shown in Table 1, LM-CPPF improves the model's accuracy on all tasks compared to the baseline method of LM-BFF+Multi-templates (Jian et al., 2022). Comparing the standard deviation of our model in five runs and the standard deviations of LM-BFF and LM-BFF + Multi-templates, we see that LM-CPPF has a higher standard deviation as it uses an intermediary model for generating paraphrases. In contrast, LM-BFF + Multi-templates integrates templates that have nearly equal performance (Jian et al., 2022).

We also compare the effect of using GPT-3, OPT-175B, and GPT-2 as our language model for few-shot paraphrasing. We did two experiments with GPT-2 large: (I) Using a pre-trained version of GPT-2 where the weights are not tuned at all (II) Fine-tuned GPT-2 where the model has been fine-tuned on the ParaNMT-50M dataset. The results in Table 1 indicate that GPT-3 outperforms OPT-175B in all tasks and GPT-2 has a lower performance, which was predictable since it has significantly fewer parameters. Also, fine-tuned GPT-2 shows a better performance which suggests that GPT-2's knowledge after pre-training is not enough for doing a task like paraphrasing. About the LLMs, although both models have 175B parameters, OPT-175B has a $1/7$ carbon footprint of GPT-3, and it is also freely available (Zhang et al., 2022). Consequently, we base our further analysis on OPT-175B.

4.2 Few-shot Paraphrasing vs. Other Data Augmentation Methods

In this section, we present an experimental comparison of the performance of the few-shot paraphrasing approach and other data augmentation methods, including BT and EDA. The results are shown in Table 2. The BT approach is evaluated using different intermediary languages (Arabic, French, Deutsch, Chinese, and Hindi). The results indicate that BT's performance is slightly different across languages, with Chinese showing the highest performance. In general, paraphrasing approaches, including BT, are better in comparison to EDA. In SST-2 and CR, where the samples are usually simple sentences, BT shows weaker performance

Task	Few-shot	Back Traslation					SR	RI	RS	RD	EDA
	Paraphrasing	AR	FR	DE	ZH	HI					
SST-2	**91.8**	90.8	90.6	90.4	90.7	90.3	90.5	89.5	90.8	91.3	90.4
SST-5	**52.2**	49.2	49.3	49.1	49.6	48.3	47.9	49.3	49.3	48.2	48.2
MNLI	**66.2**	64.3	63.1	63.8	65.4	62.2	62.9	63.2	61.7	60.2	60.3
CoLA	**13.3**	6.7	6.8	6.4	7.1	5.9	6.3	5.8	5.8	5.1	5.1
QNLI	**68.5**	66.5	66.2	65.8	66.6	64.3	66.1	65.9	66.3	65.6	63.3
CR	**91.1**	88.5	88.6	88.4	88.7	87.9	89.8	89.1	89.3	89.6	89.7

Table 2: Comparing the accuracy of our few-shot paraphrasing approach with the Back Translation (BT) and Easy Data Augmentation (EDA) methods. EDA includes Synonym Replacement (SR), Random Insertion (RI), Random Swap (RS), and Random Deletion (RD). EDA in the results is combination of all of the four mentioned methods. BT and EDA standard deviations are 1.31 and 1.4 on average, respectively, while our approach has a standard deviation of 1.65.

Task	**Template Number**					
	1	2	3	4	5	6
SST-2	91.8	91.2	91.4	89.1	92.1	**92.4**
SST-5	52.2	53.1	52.7	53.4	53.6	**54.1**
MNLI	66.2	65.9	**66.9**	66.1	66.2	66.4
CoLA	13.3	12.7	13.2	13.8	13.4	**13.6**
QNLI	68.5	68.4	68.6	68.5	68.8	**69.3**
CR	91.1	91.2	91.3	91.5	91.7	**92.2**

Table 3: Performance of different paraphrasing prompt demonstration templates.

Task	w/o Instruct	**Template Number**				
		1	2	3	4	5
SST-2	92.4	93.1	93	92.8	**93.2**	92.7
SST-5	54.1	54.7	54.5	54.2	**54.9**	54.3
MNLI	66.9	67.8	67.5	67.1	**68.2**	67.2
CoLA	13.6	13.1	13.2	12.6	**13.3**	12.8
QNLI	69.3	69.8	70.1	69.5	**70.2**	69.6
CR	92.2	93.1	92.8	92.6	**93.3**	92.4

Table 4: Performance of different paraphrasing prompt instruction templates on various NLP tasks.

than EDA. We believe the reason is that BT can be more effective for longer sequences because longer sequences usually contain more context and nuanced meaning. Moreover, EDA employs additional knowledge from another PLM in certain actions, such as synonym substitution, similar to BT and few-shot paraphrasing.

The few-shot paraphrasing approach introduced in this work outperforms both BT and EDA. This confirms that using PLM's knowledge properly in paraphrasing is an effective and efficient data augmentation method. In few-shot paraphrasing, we instruct the model to generate paraphrases that differ in lexicalization and sentence structure.

4.3 Prompt Template Evaluation

As the heart of our method is the few-shot paraphrase generation done by LLMs, we investigate the impact of different paraphrasing prompt demonstrations and instruction templates on the performance of our model. Table 3 shows that the last template presented in Table C.3 is better in almost all tasks. This template, "<Original Text>, in other words <Paraphrased>", uses a complete and concrete sentence, unlike other templates, which use specific tokens, such as "[Original]", to distinguish between the original and the paraphrased version. Also, we compare different instruction templates presented in Table C.4. As we aimed to report our best result in each task here, we used the best demonstration template for any particular task, which was determined in Table 3. Table 4 shows that the fourth template achieves the best performance, as it precisely describes the task with its instruction "Generate a paraphrase of the following text using different words and sentence structures while still conveying the same meaning".

5 Conclusion

Our experiments demonstrated the effectiveness of using few-shot paraphrasing as a data augmentation method for contrastive prompt-based fine-tuning of PLMs. It outperformed other data augmentation methods in text classification tasks, such as EDA, multiple templates, and back translation. We also found that our approach is effective with GPT-3 or OPT-175b models in generating paraphrases. Overall, LM-CPPF improves the performance of LM-BFF by large margins using contrastive learning applied on paraphrases generated by LLMs.

Limitations

Our approach relies on the performance of the few-shot paraphrasing. This results in two limitations for our approach. One limitation is the difficulty in accessing GPT-3 and OPT-175b models. These models currently need to be more widely available. OPT-175B has a free version but it is very slow. Another limitation is the need for annotated demonstrations for few-shot paraphrasing. While there are available models and tools, like QuillBot, that can be used for this purpose, their quality is not comparable to GPT-3 and OPT-175b. This can limit the power of these tools in our approach. Using human knowledge to paraphrase the demonstration can help these large models generate high-quality paraphrases but it is expensive.

Ethics Statement

The research conducted in this paper has been carried out in accordance with the ethical principles of ACL. We have ensured that our experiments do not harm any individuals or groups and have obtained informed consent from all participants. As mentioned in the paper, we also tried to base our main experimentation on the more environmentally-friendly option, OPT-175B.

References

Amirhossein Abaskohi, Fatemeh Mortazavi, and Hadi Moradi. 2022. Automatic speech recognition for speech assessment of persian preschool children. *arXiv preprint arXiv:2203.12886*.

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

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020b. Language models are few-shot learners.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Cheng-Han Chiang, Yung-Sung Chuang, and Hung-yi Lee. 2022. Recent advances in pre-trained language models: Why do they work and how do they work. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: Tutorial Abstracts*, pages 8–15, Taipei. Association for Computational Linguistics.

Marcos V Conde and Kerem Turgutlu. 2021. Clip-art: Contrastive pre-training for fine-grained art classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3956–3960.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Sonal Garg, Sumanth Prabhu, Hemant Misra, and G Srinivasaraghavan. 2021. Unsupervised contextual paraphrase generation using lexical control and reinforcement learning. *arXiv preprint arXiv:2103.12777*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification.

Chaitra Hegde and Shrikumar Patil. 2020. Unsupervised paraphrase generation using pre-trained language models. *arXiv preprint arXiv:2006.05477*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Yiren Jian, Chongyang Gao, and Soroush Vosoughi. 2022. Contrastive learning for prompt-based few-shot language learners. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5577–5587, Seattle, United States. Association for Computational Linguistics.

Vlado Keselj. 2009. Speech and language processing daniel jurafsky and james h. martin (stanford university and university of colorado at boulder) pearson prentice hall, 2009, xxxi+ 988 pp; hardbound, isbn 978-0-13-187321-6.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Alex Krizhevsky. 2014. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*.

Ashutosh Kumar, Satwik Bhattamishra, Manik Bhandari, and Partha Talukdar. 2019. Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. 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 3609–3619, Minneapolis, Minnesota. Association for Computational Linguistics.

Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. 2020. Contrastive representation learning: A framework and review. *Ieee Access*, 8:193907–193934.

Shikun Liu, Shuaifeng Zhi, Edward Johns, and Andrew J Davison. 2021. Bootstrapping semantic segmentation with regional contrast. *arXiv preprint arXiv:2104.04465*.

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*.

Mengsay Loem, Sho Takase, Masahiro Kaneko, and Naoaki Okazaki. 2022. ExtraPhrase: Efficient data augmentation for abstractive summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*, pages 16–24, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Deshui Miao, Jiaqi Zhang, Wenbo Xie, Jian Song, Xin Li, Lijuan Jia, and Ning Guo. 2021. Simple contrastive representation adversarial learning for nlp tasks. *arXiv preprint arXiv:2111.13301*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Aurko Roy and David Grangier. 2019. Unsupervised paraphrasing without translation. *arXiv preprint arXiv:1905.12752*.

Timo Schick and Hinrich Schütze. 2020a. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Timo Schick and Hinrich Schütze. 2020b. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

AB Siddique, Samet Oymak, and Vagelis Hristidis. 2020. Unsupervised paraphrasing via deep reinforcement learning. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1800–1809.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Amane Sugiyama and Naoki Yoshinaga. 2019. Data augmentation using back-translation for context-aware neural machine translation. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 35–44.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2020. Contrastive multiview coding. In *European conference on computer vision*, pages 776–794. Springer.

Jianing Wang, Chengyu Wang, Fuli Luo, Chuanqi Tan, Minghui Qiu, Fei Yang, Qiuhui Shi, Songfang Huang, and Ming Gao. 2022. Towards unified prompt tuning for few-shot text classification.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

John Wieting and Kevin Gimpel. 2018. ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Melbourne, Australia. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. 2020. Unsupervised data augmentation for consistency training.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Yuan Yao, Bowen Dong, Ao Zhang, Zhengyan Zhang, Ruobing Xie, Zhiyuan Liu, Leyu Lin, Maosong Sun, and Jianyong Wang. 2022. Prompt tuning for discriminative pre-trained language models. *arXiv preprint arXiv:2205.11166*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5086.

Jie Zhou, Le Tian, Houjin Yu, Zhou Xiao, Hui Su, and Jie Zhou. 2022. Dual context-guided continuous prompt tuning for few-shot learning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 79–84, Dublin, Ireland. Association for Computational Linguistics.

Hongyu Zhu, Yan Chen, Jing Yan, Jing Liu, Yu Hong, Ying Chen, Hua Wu, and Haifeng Wang. 2022. DuQM: A Chinese dataset of linguistically perturbed natural questions for evaluating the robustness of question matching models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7782–7794, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

A Evaluation Setting

We used a learning rate of $1e^{-5}$ for MLM loss like LM-BFF. Although contrastive learning algorithms often perform better with larger batch training, due to resource limitations, we had to use half the batch size suggested in Jian et al. (2022) for various tasks in the SCL phase. As recommended in Krizhevsky (2014), we used $sqrt(0.5) \approx 0.7$ of the learning rates mentioned in Jian et al. (2022) for this phase. Therefore, we report baselines with our smaller batch size. Our method uses a single template for each task's prediction. The primary prompts are listed in Appendix B. For the prompts used in the paraphrasing phase, with the exception of experiments in Section 4.3, we used randomly selected templates from the suggested prompts listed in Table C.3. In all of the experiments, we used OPT-175B, except one of the results mentioned in Section 4.1, where we compared OPT-175B and GPT-3 in paraphrasing.

We show the batch size and learning rate for SupCon in Table A.1. It is important to note that the results of LM-BFF presented in the main paper were obtained using the same large batch size as our method to ensure fair comparisons.

We fine-tuned with a batch size that fits into GPU memory and is divisible by the total number of examples in the task. Experiments were conducted on one NVIDIA RTX-3090 with 24 GB memory using the RoBERTa-base model. Furthermore, as per LM-BFF, we fine-tuned for a maximum of 1000 steps.

Task	Batch Size	Learning Rate
SST-2	8	$7e^{-7}$
SST-5	20	$7e^{-6}$
MNLI	12	$7e^{-6}$
CoLA	8	$7e^{-6}$
QNLI	8	$7e^{-6}$
CR	16	$7e^{-6}$

Table A.1: Batch size and learning rate for SupCon loss used for each task.

For the GPT-2 experiments in Table 1, we followed the same intructions for generating paraphrases as we used for GPT-3 and OPT-175. In fine-tuning GPT-2, we fine-tuned our model on ParaNMT-50M (Wieting and Gimpel, 2018) with the batch size of 32 and learning rate of $1e^{-3}$ for 5 epochs.

B Task Prompts

The primary prompts utilized for each task in our experiments are displayed in Table B.2. They were handpicked by LM-BFF (Gao et al., 2021).

C Paraphrasing Prompts

To find the best prompt for paraphrasing, we checked different corpus available online and found out how the paraphrasing examples are introduced. We generated our prompts by using this information and our manual modification in these templates.

In this demonstration prompt, we did not provide any explanations or descriptions for the specific transformation applied to the input to produce the output. Instead, we labeled the original sample and its paraphrase. For instance, we used the token **[Original]** to indicate the original sentence in the dataset and the token **[Paraphrase]** to indicate the

Task	Template	Verbalizers
SST-2	<S1>It was [MASK] .	positive: great, negative: terrible
SST-5	<S1>It was [MASK] .	v.positive: great, positive: good, neutral: okay, negative: bad, v.negative: terrible
MNLI	<S1>? [MASK] , <S2>	entailment: Yes, netural: Maybe, contradiction: No
CoLA	<S1>This is [MASK] .	grammatical: correct, not_grammatical: incorrect
QNLI	<S1>? [MASK] , <S2>	entailment: Yes, not_entailment: No
CR	<S1>It was [MASK] .	positive: great, negative: terrible

Table B.2: Primary templates and verbalizers (label words) used in our experiments.

paraphrased sample. Table C.3 shows the templates we used for this approach.

Demonstration Template
Original:<Original Text> Paraphrase:<Paraphrased Text>
[Original]:<Original Text> [Paraphrase]:<Paraphrased Text>
Original:<Original Text> Rewrite:<Paraphrased Text>
[Original]:<Original Text> [Rewrite]:<Paraphrased Text>
Here is the original source: <Original Text> Here is the paraphrase: <Paraphrased Text>
<Original Text>, in other words <Paraphrased Text>

Table C.3: The templates that were used to give examples of how the paraphrasing should be done to the pre-trained language model.

In instruction for prompts, we provided examples and simple instructions to the language models. The instructions were used to ask the model to generate paraphrases before presenting them with examples. Table C.4 shows the instructions we used to explain the task to the model at the beginning of our prompts.

Instructions
Summarize the following text in your own words
Rewrite the following text that expresses the same idea in a different way
Generate a paraphrase of the following text that expresses the same ideas in a different way
Generate a paraphrase of the following text using different words and sentence structures while still conveying the same meaning
Generate a summary or paraphrase of the following text that captures the essence of the ideas in a concise manner

Table C.4: The instructions that were used before giving examples to the language model to describe the paraphrasing task.

D Contrastive Prompt-based Fine-tuning Details

Contrastive prompt-based fine-tuning contains two main steps: (1) Masked Language Modeling and (2) Contrastive Learning.

Masked Language Modeling (MLM) Loss. A classification task is approached as a Masked Language Modeling(MLM) problem in prompt-based methods. The input consists of a sentence (sent) and a template with a mask (temp) (i.e., $x_{prompt} = sent, temp([\text{MASK}])$), and the goal is to determine the best token to fill in the [MASK]. This results in a MLM loss, represented as $\mathcal{L}_{MLM} = MLM(x_{prompt}, y)$, where y is the word label associated with x_{prompt}. LM-BFF (Gao et al., 2021) uses demonstrations of label words to improve the results. The input for this approach includes the sentence ($sent_0$) and the masked template ($temp_0$) with a mask ([MASK]. The input also contains an additional sentence ($sent_i$) with the same template ($temp_0$) with its own verbalizer ($word_i$) for those sentences. The label words are sampled from the training set. The classification loss is then calculated using this input.

The language model first encodes the input sentence x_{in} into a sequence of tokens, which are then mapped to a sequence of hidden states $h_1, h_2, ..., h_L$. L denotes the length of the sequence, and the dimension of the hidden states is denoted by d. For example, in prompt-based fine-tuning, if the input sentence (x_{in}) is "France missed the world cup in penalties," the corresponding prompt x_{prompt} would be [CLS] x_{in}, [MASK].[SEP]. The model then determines whether it is more likely to place the appropriate verbalizer at the [MASK] position. It has been found that fine-tuning with this fill-in-the-blank framework is superior to standard fine-tuning. The prediction of the model $\mathcal{M}$ for a class $y \in \mathcal{Y}$ can be expressed by mapping the label space Y to the

Algorithm D.1 Learning from MLM and SupCon with Paraphrasing

```
Input:
Training set: $\mathcal{D}_{train}$
MLM model: $\mathcal{M}$
Function to concatenate two strings: $Concat$
Cross Entropy loss: $CE$
Supervised Contrastive loss: $SupCon$
Paraphrase function: $Paraphrase$
Function that samples from a dataset and puts
it in the specific template: $Sample$
// The third parameter of this function specifies
// whether to pus [MASK]or the verbalizer of
// the label
Template For Prompts: $Template$
$MaxStep = 1000$
Preparing Samples:
for i < MaxStep do
    $sent, y$=Sample($\mathcal{D}_{train}, Template$, false)
    $demo_1$=Sample($\mathcal{D}_{train}, Template$, true)
    $demo_2$=Sample($\mathcal{D}_{train}, Template$, true)
    $x_{in_1}$=Concat($sent, demo_1$)
    $x_{in_2}$=Concat(Paraphrase($sent$), $demo_2$)
    ▷ MLM Learning:
    $output_1 = \mathcal{M}(input_1)$
    $\mathcal{L}_{MLM}$ = CE($output_1, y$)
    $\mathcal{L}_{MLM}$.backward()
    optimizer.step()
    ▷ Contrastive Learning:
    $output_2 = \mathcal{M}(input_2)$
    $\mathcal{L}_{SupCon}$ = SupCon($output_1, output_2$)
    $\mathcal{L}_{SupCon}$.backward()
    optimizer.step()
end for
```

label words, where $\mathcal{V}(y)$ represents the label word for class y. This can be written as:

$$\begin{aligned} p(y|x_{in}) &= p([MASK] = \mathcal{V}(y)|x_{prompt}) \\ &= \frac{exp(w_{\mathcal{V}(\dagger)}.h_{[MASK]})}{\sum_{y' \in \mathcal{Y}} exp(w_{\mathcal{V}(y')}.h_{[MASK]})} \end{aligned} \quad (1)$$

where the weight vector of the MLM head is denoted by w.

In LM-BFF, the authors add demonstrations to the input x_{prompt} to improve the model's understanding of verbalizers. As a result, the input to LM-BFF is in the following form:

$$\mathcal{T}(x_{in}) \oplus \mathcal{T}(x_{in}^1, y^1) \oplus ... \oplus \mathcal{T}(x_{in}^k, y^k) \quad (2)$$

where $\mathcal{T}(x_{in}^i, y^i)$ illustrates the i-th demonstration in the template $mathcalT$ with where the actual verbalizer of the samples replaces the [MASK]. Also, k is the number of demonstrations we want to use in our prompts. This paper uses random sampling to select demonstrations from the training set. The MLM loss is calculated as follows:

$$\mathcal{L}_{MLM} = \sum_{(x_{in}, y) \in \mathcal{D}_{train}} -log[p(y|x_{in})] \quad (3)$$

Supervised Contrastive Loss. Supervised Contrastive Learning is a specific form of contrastive learning (Chen et al., 2020; Tian et al., 2020; Liu et al., 2021) that clusters two augmented batches at the class level in the feature space and calculates the contrastive loss using Equation 4:

$$\mathcal{L}_{SupCon} = (x_1', x_2', y) \quad (4)$$

where x_1' and x_2' are the augmented version of the input batch x and y is the actual label of the batch.

To use SupCon on multiple views of an input text, we first need to obtain two views of the text:

$$x_1 = T(sent), T(sent_i, verb_i) \quad (5)$$

$$x_2 = T(Par(sent)), T(sent_j, verb_j) \quad (6)$$

where x_1 is the same as $x_{prompt+demo}$ in LM-BFF and $\mathcal{T}$ is a function that formats the sentence according to a specific template. Instead of using a new template in which the newly generated sample does not provide a new perspective, we use the few-shot paraphrasing (Par) function. Also, $verb$ stands for the verbalizer used for the actual label of the sample. Now using Equation 4 on two views, we can calculate the total loss:

$$\mathcal{L}_{Total} = \mathcal{L}_{SupCon} + \mathcal{L}_{MLM} \quad (7)$$

Algorithm D.1 shows an overview of our method which uses contrastive few-shot fine-tuning with few-shot paraphrasing. It is important to mention that learning from $\mathcal{L}_{SupCon}$ requires one additional forward and backward pass, which increases the computational cost by a factor of 1.5. However, the cost is still the same as Jian et al. (2022)'s model due to the $O(1)$ time complexity of the $Paraphrase$ function. Figure 1 shows the fine-tuning procedure for one prompt sample and its new view created using few-shot paraphrasing.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
There is a Limitations section after the conclusion

□ A2. Did you discuss any potential risks of your work?
Not applicable. Our project does not have any potential risk

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Yes. Section abstract and section 1.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

□ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☑ Did you run computational experiments?

Section 3 and Appendix A

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
We did mention the model we used whose number of parameters is available online. Also, we mentioned the GPU resource we used.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☒ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
As we wanted to make a fair comparison with other models, we tried not to change the hyperparameters as possible, considering it has been done before. We used the information from the previous studies for setting the hyperparameters. Moreover, limited resources made the domain for different hyperparameters small for us.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 3. In addition to mentioning the average accuracy, we mentioned the standard deviation of our result.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 2 and appendix A and D. We also attached our implementation.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.