PETapter: Leveraging PET-style classification heads for modular few-shot parameter-efficient fine-tuning

Anonymous ACL submission

Abstract

Few-shot learning and parameter-efficient fine-tuning (PEFT) are crucial to overcome the challenges of data scarcity and ever growing language model sizes. This applies in particular to specialized domains such as argument mining, where complex and nuanced phrasing makes it difficult even for humans to distinguish not only between the stances but also whether a sentence contains a claim or an argument. We propose PETapter, a novel method that effectively combines PEFT methods with PET-style classification heads to boost few-shot learning capabilities without the significant computational overhead typically associated with full model training. We validate our approach on three established NLP benchmark datasets and one real-world argument mining dataset. We show that PETapter not only achieves comparable performance to full few-shot fine-tuning using pattern-exploiting training (PET), but also provides greater reliability and higher parameter efficiency while enabling higher modularity and easy sharing of the trained modules.

1 Introduction

Few-shot learning and parameter-efficient fine-tuning (PEFT) have emerged as pivotal disciplines in the realm of natural language processing (NLP), especially in applications where data scarcity poses significant challenges. Argument mining, an area focused on automatically detecting and analyzing arguments within text, is particularly affected by such challenges. Due to the nuanced and context-dependent nature of argumentative discourse, standard fine-tuning methods for pretrained language models (PLM) with only a few labeled data points usually perform poorly in these tasks (e.g., Rieger et al., 2024). Few-shot learning, with its promise to generalize from a limited number of training samples, offers an alternative pathway towards mitigating the data scarcity problem. However, the deployment of large-scale language models in a few-shot setting can be challenging, primarily due to the substantial computational resources required for training and fine-tuning, as they update all parameters of the model by default. PEFT methods, such as adapter modules, present a viable solution to this issue by enabling the adaptation of pretrained models to specific tasks with minimal modifications to the model architecture. Specifically, these methods freeze a large part of the model's parameters and train only small parts of the existing or few newly added layers (Poth et al., 2023). Further, PEFT methods using a standard linear layer perform significantly worse than few-shot methods when using few observations (cf. Sections 5.3 and 6.3).

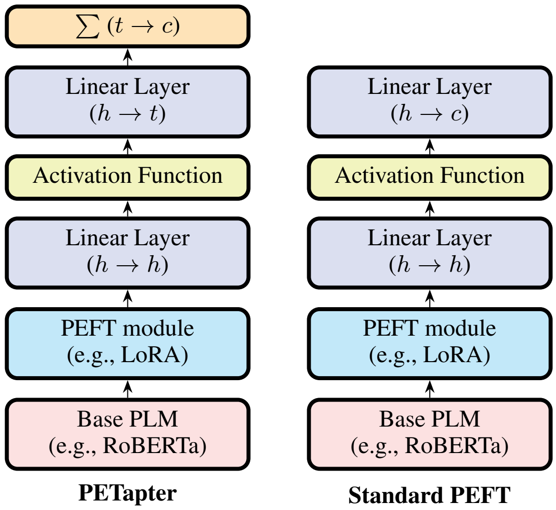

Figure 1: Schematic representation of the PETapter architecture compared to standard PEFT methods.

For this reason, this paper aims to delve into the innovative intersection of few-shot learning and parameter-efficient fine-tuning. By integrating few-shot learning paradigms with PEFT techniques, this paper explores how, e.g., the field of argument mining can leverage the strengths of both approaches to efficiently classify text sequences, even when little data is available. Through a study on three typical

NLP benchmark datasets and a study on a real-world argument mining dataset, we demonstrate that the combination of few-shot and PEFT methodologies can significantly reduce the reliance on extensive annotated datasets while achieving competitive performance (to full few-shot fine-tuning) with reduced computational effort. It thus combines the advantages of the two fields, few-shot learning and parameter-efficient fine-tuning.

As contribution, we propose a new method, PETapter, as a combination of PET-style classification heads with PEFT methods. We show that in most scenarios PETapter is as performant as PET itself, i.e. noticeable better than PEFT with a standard classification head. PETapter additionally offers all PEFT advantages, i.e. it requires less computational resources and is easier to share in the research community due to its modularity. We also show that PETapter provides more robust predictions than PET, especially on real-world datasets. By doing so, this paper not only contributes to the theoretical advancement of NLP techniques but also offers practical insights for researchers and practitioners working on supervised text classification problems such as argument mining.

In Section 2, we review relevant preliminary NLP work, in Section 3, we recapitulate the definition of pattern-verbalizer pairs (PVP) for usage in our new method PETapter in Section 4. In Sections 5 and 6, we conduct intensive NLP benchmark studies and a real-world study, which we finally discuss in Section 7 and summarize in Section 8. The data and code used will be made available in a GitHub repository after acceptance.

2 Related Work

Pretrained language models (PLMs) serve as the basis for most classification tasks in natural language processing (NLP). A frequently compared model is RoBERTa (Liu et al., 2019) due to its strong performance even with a comparatively small number of parameters (Base: 125 million, Large: 355 million). When using non-English texts, the multilingual alternative XLM-RoBERTa (Conneau et al., 2020) can be used, which was trained on 100 different languages and provides a reliable basis for achieving good performance, at least for high resource languages. As the next best model, (m)DeBERTa (He et al., 2023) forms a promising base PLM, but with 1.5 billion parameters it already has significantly higher requirements in terms of GPU capacity and computation time.

2.1 Parameter-Efficient Fine-Tuning (PEFT)

The ever-increasing sizes of base PLMs pose a great challenge for fully fine-tuning all parameters. Rebuffi et al. (2017), as well as Houlsby et al. (2019), were among the first to tackle this problem by freezing the PLM and instead inserting and training bottleneck feed-forward layers within the pretrained architecture. Pfeiffer et al. (2020) refined this idea into a slightly more parameter-efficient and robust method, though still sequential in its basic idea, which is named Pfeiffer adapters. Techniques such as low-rank adaptation (LoRA, Hu et al., 2021) and $(IA)^3$ (Liu et al., 2022) introduce minimal updates to the model weights, while their parallel approach makes it possible to combine the newly learned parameters with the frozen ones in such a way that there is no overhead during inference. Recent innovations such as considering different ranks for each linear layer in PRILoRA (Benedek and Wolf, 2024), sharing the low-rank matrices across all layers with layer-specific learned scaling vectors in VeRA (Kopiczko et al., 2024) or LoRA-like decomposition of pretrained weights in DoRA (Liu et al., 2024) are examples of further LoRA-like advancements in the field of parameter-efficient fine-tuning.

In addition, ComPEFT (Yadav et al., 2023) and QLoRA (Dettmers et al., 2023) offer the possibility of even more effective parameter usage with condensation of information through quantization and desparsification. RoSA (Nikdan et al., 2024) combines low-rank adaptation with high sparsity training following the idea of robust principal component analysis. Several specialized approaches exist, one of which is AdaSent (Huang et al., 2023) focusing on PEFT for sentence representations.

While many implementations are often limited to the PEFT character of the methods (Mangrulkar et al., 2022; Hu et al., 2023a,b), the Python package Adapters (Poth et al., 2023) facilitates the modular use of PEFT methods, highlighting a trend towards modular deep learning (Pfeiffer et al., 2023) and holistic PEFT implementations (He et al., 2022; Sabry and Belz, 2023).

2.2 Few-Shot Text Classification

Few-shot text classification aims to maximize model performance with minimal labeled data. For this, a lot of approaches are based on a combination of prompt-based learning and meta learning. Schick and Schütze (2021) are the first to propose the utilization of so-called *prompt-based* cloze questions. In this pattern-exploiting training (PET), a masked token is predicted in a language model task style, similar to that used in pretraining. The authors show that due to the model's prior knowledge, it is able to predict these so-called verbalizers better than plain numbered class labels. Furthermore, the authors show that *meta-learning* on soft labels generated by augmentations using iterative PET (iPET) also leads to an improvement, so that a combination of prompt-based learning and meta-learning (Zhang et al., 2022) is proposed. Chen and Shu (2023) use such an approach and propose the use of label-guided data augmentation methods for prompt-based few-shot tuning.

Ling et al. (2023) demonstrate the possibility of an optimized automated search for verbalizers in prompt-based learning, while Karimi Mahabadi et al. (2022) suggest an approach without the need for prompts. Complementary to this, SetFit (Tunstall et al., 2022) form triplets of positive and negative examples and use contrastive learning to effectively train on few training samples. The recipe T-Few (Liu et al., 2022) as an additional dedicated few-shot model is based on the zero-shot model T0 (Sanh et al., 2022) and enriches it with a combination of losses, the PEFT method $(\text{IA})^3$ and a pretraining of it using out-of-domain data, which can be quite expensive with regard to the training time needed. PET, Setfit and T-Few represent key developments in the few-shot discipline and produce comparable performances on the RAFT benchmark dataset (Alex et al., 2021), each with strengths and weaknesses on different datasets

2.3 Argument Mining

In the field of argument mining, the following works are related to our idea of using PETapter to identify argumentative sentences from news media: Hüning et al. (2022) classify whether user-generated chat messages contain an argument using machine learning techniques utilizing sentence embeddings as features. For simplicity, they categorize claims as *no argument* in their setting. Jurkschat et al. (2022) consider the possibility of few-shot methods for classifying aspects of argumentative sentences and show that PET performs best among the methods considered. Likewise, PET is used by Rieger et al. (2024) and compared with the use of PEFT methods for the identification of claims and arguments from news media articles. The authors find that the task is challenging even for humans, so that only moderate inter-coder agreements are achieved.

In the following, we present the PETapter model, which combines few-shot and PEFT methods to achieve great performance, in particular in real-world data scarce argument mining settings.

3 Pattern-Verbalizer-Pair (PVP)

We consider a pretrained language model (PLM) M with an underlying vocabulary V, where $[\text{MASK}] \in V$, i.e. we assume a mask token to be contained in the vocabulary. In addition, let $\mathcal{L}$ be a set of target labels for a classification task and $x \in V^n$ an input (possibly consisting of several segments) that contains a total of n tokens from the vocabulary V. We define a pattern as a function $P^m : V^n \to V^{n+p}$, where $m \leq p$ is the number of [MASK] tokens contained in the pattern and p is the total number of vocabulary items added to the input x by the pattern function.

Furthermore, let $v^m : \mathcal{L} \to V^m$ be a verbalizer function that assigns m tokens from the vocabulary V to each label $\ell \in \mathcal{L}$. As the inventors of PET (Schick and Schütze, 2021) suggest, we refer to the combination (P^m, v^m) as a pattern-verbalizer pair (PVP). The pattern function P^m is used to transform the input x into a kind of cloze question and the verbalizer function v^m provides for each label the "speaking" and representative fill-in words for the corresponding cloze question, which are to be predicted by the model M. All PVPs used in this paper can be found in the Appendices A and B.

4 PETapter

PETapter can be seen as a modular classification head which can be added flexibly to a PEFT framework as an alternative to a classical linear layer classification head, cf. Figure 1. Here, the last (purple) linear layer of the classification head does not reduce the dimension directly to the number of classes c, but to the size t of the sub-vocabulary T of the verbalizer function used. Thus, each dimension of the output embedding represents a token from the reduced verbalizer vocabulary, similar to and motivated by the pattern-exploiting training (PET) by Schick and Schütze (2021). Then, the logits for the verbalizers (possibly composed of several tokens) can be calculated as a sum of the logits of the masked tokens in the pattern (cf. Equation 1).

In general, PETapter can be combined with any PEFT method (cf. the blue segment in Figure 1), i.e. it also features all its advantages over full fine-tuning: faster training, less resource requirements, better sharability and reusability due to modularity as well as robustification (at least) with regard to the elimination of catastrophic forgetting. Specifically, PETapter implements all these advantages over PET, which uses full fine-tuning, while performing on par (cf. Sections 5.3 and 6.3).

Let $\mathcal{L}$ be a set of target labels for a classification task with $|\mathcal{L}| = c$, let $P^m(x)$ be a pattern function inserting m [MASK] tokens to an input x, and let $v^m(\ell)$ be an injective function that maps each of the labels $\ell \in \mathcal{L}$ to m vocabulary tokens. Then, we obtain the subset of the vocabulary relevant for the verbalizers as $T = \bigcup_{\ell \in \mathcal{L}} \bigcup_{i=1}^{m} v^m(\ell)_i$ with $T \subset V$ and $t = |T|$ the corresponding number of relevant tokens, i.e. we can refine $v^m : \mathcal{L} \to T^m$.

Moreover, let $M(v^m(\ell) \mid P^m(x)) \in \mathbb{R}^m$ denote the logits for the m [MASK] tokens, the output of the top linear layer (purple) in Figure 1 on the left. The final score for each of the label candidates ℓ for a given input text x is then given as

$$s(\ell \mid x) = \sum_{i=1}^{m} M(v^m(\ell) \mid P^m(x))_i \qquad (1)$$

and the corresponding pseudo-probability as

$$q(\ell \mid x) = \frac{\exp(s(\ell \mid x))}{\sum_{\ell' \in \mathcal{L}} \exp(s(\ell' \mid x))}. \qquad (2)$$

Using this, we can calculate the cross-entropy loss over all observations as

$$\begin{aligned} L_{\text{CE}} &= -\sum_{(x,\ell^*)} \sum_{\ell \in \mathcal{L}} \mathbb{1}_{\{\ell = \ell^*\}}(x, \ell^*) \log[q(\ell \mid x)] \\ &= -\sum_{(x,\ell^*)} \log[q(\ell^* \mid x)] \qquad (3) \end{aligned}$$

if we consider $\ell^* \in \mathcal{L}$ to be the true label of x.

5 Benchmark Study

To evaluate how well our model performs in comparison to existing state-of-the-art methods, we consider three established NLP datasets in a first benchmark study. These have quite a laboratory character, as the distribution of the class labels is pretty much balanced. Furthermore, it is not finally clear to what extent (possibly implicit) information about the test data of such publicly available datasets is contained in PLMs (Li and Flanigan, 2024). Nevertheless, such datasets with predefined train-test splits offer the possibility of comparing methods across studies without the need of rerunning all the experiments.

5.1 Datasets: AG, Yahoo, Yelp

We follow the study by Schick and Schütze (2022) and use the three established NLP datasets *AG's News* (AG), *Yahoo Questions* (Yahoo), and *Yelp Full* (Yelp), which are presented by Zhang et al. (2015). In Appendix A, further information on the three datasets is given.

For the training, we create 5 stratified datasets with $n = 10$ randomly drawn observations and 5 datasets with $n = 100$ observations each for the problems AG, Yahoo, and Yelp. We evaluate all experiments with the corresponding entire test dataset from the predefined split.

5.2 Experimental Setup

As PLM, we make use of RoBERTa Base and Large. For fine-tuning methods, we consider the few-shot method PET, PEFT methods with a (standard) linear layer as classification head, and our method PETapter, i.e. PEFT methods in combination with a PET-style classification head. In both cases, we consider the methods $(\text{IA})^3$, LoRA and a Pfeiffer adapter as PEFT methods. For PET and PETapter, we compare the use of *prompt* or *Q&A* pattern, following Schick and Schütze (2022). In each dataset, we measure the change of using $n = 10$ or 100 observations. Moreover, we repeat each experiment five times, i.e. in Table 1 each cell corresponds to 25 experiments (5 repetitions $\times$ 5 datasets). In Appendix A, further information on all parameters as well as the used patterns is given.

5.3 Results

Table 1 shows the average accuracy of the settings and the standard deviation across the 25 experiments per cell. Overall, it can be seen that the best performance is achieved by PET using the prompt pattern, while PETapter achieves the best values in two scenarios and the linear layer classification head never performs best. As expected, RoBERTa Large consistently yields better results than the Base variant, although the PEFT methods generally benefit somewhat more from the use of the larger model. The accuracy achieved by PETapter is usually very close to that of PET. Merely in the setting $n = 10$ for Yahoo the values are significantly worse. On the other hand, for $n = 100$ on

			Prompt Pattern				Q&A Pattern				Linear Layer		
			PETapter				PETapter						
	n	Data	$(IA)^3$	LoRA	Pfeif.	PET	$(IA)^3$	LoRA	Pfeif.	PET	$(IA)^3$	LoRA	Pfeif.
RoBERTa Base	10	AG	0.668 ±.092	0.681 ±.080	0.683 ±.078	**0.804 ±.036**	0.596 ±.077	0.672 ±.077	0.680 ±.088	0.800 ±.028	0.297 ±.053	0.293 ±.045	0.316 ±.059
	10	Yahoo	0.236 ±.016	0.292 ±.033	0.271 ±.042	**0.545 ±.040**	0.274 ±.026	0.295 ±.041	0.285 ±.036	0.515 ±.036	0.105 ±.009	0.116 ±.020	0.123 ±.016
	10	Yelp	0.403 ±.037	0.434 ±.035	0.423 ±.038	0.441 ±.017	0.359 ±.037	0.412 ±.042	0.390 ±.047	**0.442 ±.025**	0.215 ±.017	0.219 ±.023	0.224 ±.027
	100	AG	0.856 ±.012	0.861 ±.012	0.860 ±.012	**0.875 ±.007**	0.863 ±.006	0.861 ±.010	0.862 ±.011	0.871 ±.008	0.837 ±.007	0.862 ±.008	0.864 ±.008
	100	Yahoo	0.622 ±.018	0.642 ±.009	0.642 ±.011	**0.666 ±.007**	0.622 ±.005	0.637 ±.007	0.633 ±.008	0.659 ±.007	0.418 ±.034	0.636 ±.013	0.633 ±.012
	100	Yelp	0.541 ±.010	0.553 ±.018	0.551 ±.016	0.552 ±.014	0.536 ±.014	0.553 ±.015	0.548 ±.014	**0.554 ±.015**	0.354 ±.022	0.538 ±.018	0.535 ±.020
RoBERTa Large	10	AG	0.641 ±.100	0.714 ±.070	0.702 ±.081	**0.842 ±.025**	0.611 ±.073	0.746 ±.054	0.738 ±.060	0.836 ±.032	0.305 ±.030	0.373 ±.049	0.443 ±.104
	10	Yahoo	0.242 ±.027	0.331 ±.040	0.290 ±.056	**0.574 ±.030**	0.323 ±.049	0.365 ±.049	0.346 ±.054	0.550 ±.040	0.124 ±.012	0.150 ±.027	0.169 ±.041
	10	Yelp	0.442 ±.040	0.470 ±.041	0.479 ±.035	0.475 ±.026	0.440 ±.049	0.472 ±.049	**0.490 ±.046**	0.486 ±.041	0.211 ±.010	0.221 ±.012	0.216 ±.014
	100	AG	0.868 ±.011	0.873 ±.010	0.875 ±.010	**0.877 ±.009**	0.876 ±.009	0.870 ±.010	0.873 ±.010	0.874 ±.009	0.833 ±.011	0.875 ±.008	0.875 ±.008
	100	Yahoo	0.654 ±.020	0.662 ±.014	0.661 ±.017	**0.680 ±.013**	0.655 ±.010	0.654 ±.008	0.656 ±.012	0.675 ±.013	0.364 ±.043	0.648 ±.016	0.647 ±.015
	100	Yelp	0.611 ±.011	0.613 ±.014	0.614 ±.010	0.593 ±.014	**0.626 ±.008**	0.622 ±.013	0.620 ±.013	0.595 ±.016	0.347 ±.019	0.551 ±.019	0.512 ±.043

Table 1: Mean accuracies (± standard deviation) of the experiments in the benchmark study.

RoBERTa	Architecture	AG	Yahoo	Yelp
Base	PETapter	0.33	0.33	0.32
Base	PET	0.38	0.39	0.39
Large	PETapter	0.65	0.64	0.65
Large	PET	1.00	1.00	1.00
Large	PET	6.1s	6.2s	6.2s

Table 2: Comparison of training times per iteration of $n = 100$ observations in the benchmark study. The last row shows the time for PET using RoBERTa Large. The other rows indicate the time relative to it. The times for using PETapter or a linear layer are identical, as are for the three architectures $(IA)^3$, LoRA, and Pfeiffer.

Yelp, all three PEFT architectures combined with our PETapter method are noticeably better than PET itself. $(IA)^3$ with Q&A pattern even leads to the best global performance in this case, while it produces rather low scores in most scenarios. Our explanation for this is that IA3 is just one component of the generally in benchmarks competitively performing T-Few. Overall, prompt and Q&A pattern perform similarly. Thus, we can confirm these findings from Schick and Schütze (2022) regarding PET and show that they generally hold for PETapter as well. Basically, it can be seen that the linear layer rarely comes close to the performance of PETapter; using linear layer classification heads in combination with $(IA)^3$ leads to the worst results overall.

As a result, it is evident that PETapter is to be preferred over the use of a classical linear layer as a classification head in use cases where the benefits of PEFT methods are desired. At the same time, PETapter achieves comparable performance to PET in most cases.

In addition, as a PEFT method (cf. Section 2.1), PETapter can be trained more effectively than standard PET. In Table 2 the training times of the models are displayed. According to this, PETapter (no matter the architecture) requires $\approx 65\%$ of the time of regular PET per iteration. Compared to a regular PEFT approach using a linear layer, PETapter does not produce any overhead in training time.

6 Real-World Study

As indicated in Section 5, the evaluation based on AG, Yahoo, and Yelp represents a lab situation, e.g., because of their balancedness and (implicit) inclusion in the pretraining of language models. Complementary to this, in a second study, we compare the performance of the methods on a (so far unpublished) dataset with real-world challenges. It is a dataset in the context of argument mining with argumentative sentences on the topic of *arms deliveries to Ukraine* with a total of 7301 thematically relevant articles in 2022 from 22 German media outlets. The composition is explained in detail in the study of Rieger et al. (2024). Here, we consider a version of the dataset consisting of 1766 labeled data with the four possible labels claim/argument for/against, with non-relevant sentences already removed.

6.1 Dataset: Ukraine Arms Deliveries

The German language Ukraine dataset consists of 766 train (294 articles) and 1000 test (369 articles) observations. Using a two-stage sampling (first article, then sentence) we intend to tackle the problem that including very related sentences from the same text in both splits could lead to an overoptimistic estimation of the error. Here, a single observation consists of a target sentence to be classified as well as two contextual sentences before and after it. Based on the 766 observations, we draw training sets of sizes $n = 10, 100, 250$ according to three different sampling strategies.

We draw the same number of observations from the subsets of the four labels in *equal* sampling, i.e. 25 observations each in the scenario of $n = 100$. Using *random* sampling, we make a simple random selection from the total number of all possible training examples and with *stratified* sampling we ensure that the label distribution of the entire training set is replicated as accurately as possible even in smaller samples. We create 5 datasets for each combination of sampling strategy (Equal, Random, Stratified) and number of shots ($n = 10, 100, 250$), where we do not consider to random sample only 10 observations in order to ensure a minimum of one observation per label in all training sets. Table 3 provides the label distributions of the training and test dataset and the corresponding expected numbers under random sampling with $n = 100, 250$.

6.2 Experimental Setup

For the comparison of our PETapter model to PET and PEFT with a linear layer as classification head, we use the XLM-RoBERTa Large model due to the German dataset. In addition to the different architectures (cf. Section 5.2), we compare the effect of different sampling strategies and the number of training observations $n = 10, 100, 250$. We again repeat each experiment five times. Thus, a single cell in Table 4 corresponds to 25 experiments (5 repetitions $\times$ 5 datasets). In Appendix B, further information on all parameters as well as the used patterns is given.

Label	Train				Test
	10*	100*	250*	All	
argumentagainst	1.2	11.8	29.4	92	118
argumentfor	1.9	19.4	48.6	152	162
claimagainst	2.4	23.5	58.8	184	248
claimfor	4.6	45.2	113.2	354	456

Table 3: Label distribution in the train and test data split of the Ukraine dataset. *Expected distribution for random selection.

6.3 Results

Table 4 shows the macro-F1 scores from the real-world study. According to this, there is no significant difference between the performance of PET and PETapter. The scores using the linear layer, on the other hand, are significantly worse in all scenarios and combinations. The classification task appears to be so hard that for $n = 10$ neither any scenario nor any model could achieve meaningful performance. For $n = 100, 250$, it can be seen that PET benefits greatly from the use of a balanced training set (equal sampling); this can be seen in a strong reduction of uncertainty measured by the standard deviation. In principle, however, PETapter produces consistently more reliable performances in the sense that the standard deviation of the scores is consistently lower for all settings. It can be seen that even for $n = 100$ random and stratified sampling lead to similar results.

In Table 5, we also present label-specific performance scores (precision, recall, macro-F1) for each of the four classes in the dataset. Here, we focus on the presentation of the results for $n = 250$ and the Pfeiffer adapters as PEFT method. The results of the random sampling are shown in Table 8 in the Appendix. It is of little surprise that the rarest class *argumentagainst* generates the worst precision, recall, and macro-F1 scores in the stratified sampling. In the case of equal sampling, this remains true only for the precision; the recall can be increased through the oversampling for PET from previously 51% to the then highest value of the four classes of 81%. The fact that an equal sampling in the training set leads to higher overall macro-F1 scores and lower uncertainty than a stratified sampling can be explained by the fact that the corresponding recall values of otherwise rarely occurring classes can be increased in this way.

In general, PETapter provides the best overall performance in this real-world study. While it yields a similar level of performance, it produces

n	Sampling	PETapter			PET	Linear Layer		
		(IA)^3	LoRA	Pfeiffer		(IA)^3	LoRA	Pfeiffer
10	Equal	0.28 ±.046	0.31 ±.043	**0.33 ±.057**	0.33 ±.080	0.14 ±.039	0.13 ±.042	0.15 ±.041
10	Stratified	0.19 ±.023	0.27 ±.039	0.33 ±.027	**0.40 ±.055**	0.16 ±.000	0.16 ±.001	0.17 ±.021
100	Equal	0.49 ±.023	0.57 ±.020	0.57 ±.028	**0.59 ±.027**	0.23 ±.041	0.26 ±.029	0.29 ±.030
100	Random	0.41 ±.036	**0.56 ±.036**	0.55 ±.036	0.56 ±.053	0.16 ±.000	0.20 ±.041	0.26 ±.037
100	Stratified	0.40 ±.029	0.58 ±.042	0.57 ±.035	**0.59 ±.054**	0.16 ±.000	0.20 ±.030	0.26 ±.035
250	Equal	0.57 ±.015	0.67 ±.014	0.68 ±.018	**0.70 ±.025**	0.28 ±.027	0.46 ±.050	0.49 ±.075
250	Random	0.50 ±.031	**0.67 ±.021**	0.67 ±.024	0.67 ±.109	0.16 ±.000	0.38 ±.031	0.45 ±.086
250	Stratified	0.48 ±.036	0.67 ±.019	**0.67 ±.018**	0.67 ±.109	0.16 ±.003	0.37 ±.040	0.46 ±.082

Table 4: Mean macro-F1 scores (± standard deviation) of the experiments in the real-world study (Ukraine).

	Label	Equal Sampling			Stratified Sampling		
		PETapter	PET	Lin. Layer	PETapter	PET	Lin. Layer
Precision	argumentagainst	0.50 ±.031	0.51 ±.040	0.34 ±.068	**0.54 ±.037**	0.54 ±.122	0.39 ±.092
	argumentfor	0.58 ±.045	0.63 ±.064	0.45 ±.075	0.64 ±.032	**0.66 ±.145**	0.47 ±.135
	claimagainst	0.70 ±.033	**0.71 ±.045**	0.50 ±.078	0.68 ±.036	0.66 ±.143	0.49 ±.065
	claimfor	0.87 ±.022	**0.90 ±.023**	0.69 ±.076	0.84 ±.022	0.83 ±.080	0.65 ±.064
Recall	argumentagainst	0.75 ±.043	**0.81 ±.061**	0.46 ±.113	0.53 ±.064	0.57 ±.140	0.16 ±.098
	argumentfor	0.66 ±.048	**0.70 ±.047**	0.65 ±.060	0.63 ±.053	0.61 ±.138	0.42 ±.147
	claimagainst	0.75 ±.026	0.76 ±.051	0.51 ±.090	**0.78 ±.028**	0.75 ±.161	0.54 ±.121
	claimfor	0.67 ±.034	0.67 ±.047	0.48 ±.121	0.78 ±.022	**0.80 ±.051**	0.74 ±.051
Macro-F1	argumentagainst	0.60 ±.029	**0.62 ±.031**	0.38 ±.082	0.53 ±.045	0.55 ±.123	0.22 ±.100
	argumentfor	0.62 ±.026	**0.66 ±.040**	0.53 ±.061	0.63 ±.024	0.63 ±.134	0.44 ±.138
	claimagainst	0.72 ±.019	**0.73 ±.025**	0.50 ±.072	0.72 ±.028	0.70 ±.148	0.51 ±.084
	claimfor	0.76 ±.018	0.77 ±.031	0.56 ±.105	**0.81 ±.012**	0.81 ±.040	0.69 ±.049

Table 5: Mean precision, recall, and macro-F1 scores per label (each ± standard deviation) of the experiments in the real-world study (Ukraine). We consider the Pfeiffer adapter as the PEFT method of PETapter and $n = 250$. We omit the results using random sampling as they are quite similar to those of the stratified datasets, cf. Table 8.

more reliable performance values overall. Thus, PETapter makes the idea of PET easily accessible in PEFT settings without loss of performance. While PET on our system[1] processes 7 observations per second (obs/s) during training and 8 obs/s during testing, PETapter processes 25 obs/s during training and 51 obs/s ((IA)^3/LoRA) or 42 obs/s (Pfeiffer) during testing. This shows a meaningful speed-up of PETapter compared to PET for the training as well as the inference phase.

6.4 PVP-Experiments

A major criticism of PET-like models is the need for manual generation of patterns and verbalizers. To address this, we tested the use of automated PVPs (*No Pattern*, *Alpha* verbalizer) and badly chosen verbalizers (*Shuffle*). For reasons of complexity, we limit the experiment to the combination of LoRA and PETapter. As a result from Table 6, it can be concluded that the pattern should at best be chosen manually, as there are notable differences between the performances in all scenarios. However, the choice of verbalizer in our task has hardly any influence on the performance already for $n = 100$. In fact, for *No Pattern*, *Alpha* mostly performs better than the manually selected verbalizers, which may be because *Alpha* consists of only one token each, while the other two scenarios require two tokens per verbalizer. Moreover, *Shuffle* does not result in any noteworthy differences in label-specific performance values compared to *Normal* (table not included due to space constraints).

As an extension, Table 7 indicates that a combination of the five repetitions using a majority vote leads to superior and more stable results. In particular in the scenario of limited human resources, this offers the possibility of boosting the performance of automatically selected PVPs by simply stacking independent repetitions.

7 Discussion

The results show that PET benefits greatly from equal sampling with unbalanced data. We were not able to achieve this gain to the same extent using PETapter. Therefore, even though equal sampling is not a realistic scenario in (few-shot) real-world settings, it could be promising to develop PEFT methods in such a way that they benefit from bal-

[1] 48 GB NVIDIA RTX 6000 Ada, Intel Xeon W7-3445 20×2.6 GHz, 256 GB ECC DDR5-4800 RAM

n	Sampling	No Pattern Alpha	No Pattern Normal	No Pattern Shuffle	Pattern Alpha	Pattern Normal	Pattern Shuffle
10	Equal	0.22 ±.039	0.25 ±.040	0.22 ±.039	0.23 ±.039	0.31 ±.043	0.28 ±.043
10	Stratified	0.20 ±.025	0.22 ±.031	0.22 ±.033	0.23 ±.067	0.27 ±.039	0.27 ±.043
100	Equal	0.47 ±.026	0.43 ±.041	0.41 ±.037	0.57 ±.041	0.57 ±.020	0.56 ±.033
100	Random	0.43 ±.046	0.39 ±.040	0.39 ±.043	0.53 ±.035	0.56 ±.036	0.54 ±.044
100	Stratified	0.40 ±.027	0.38 ±.035	0.37 ±.027	0.52 ±.051	0.58 ±.042	0.54 ±.048
250	Equal	0.62 ±.022	0.61 ±.024	0.60 ±.022	0.67 ±.021	0.67 ±.014	0.68 ±.019
250	Random	0.58 ±.035	0.57 ±.054	0.56 ±.054	0.65 ±.029	0.67 ±.021	0.66 ±.020
250	Stratified	0.60 ±.025	0.59 ±.030	0.58 ±.032	0.65 ±.019	0.67 ±.019	0.66 ±.017

Table 6: Mean macro-F1 scores (± standard deviation) of the PVP-experiments in the real-world study (Ukraine) using LoRA as PEFT method and PETapter as classification head.

Data	n	PVP	Mean	Majority
AG	10	Manual	0.71 ±.080	0.71 ±.071
AG	100	Manual	0.87 ±.010	0.87 ±.009
Yahoo	10	Manual	0.30 ±.044	0.32 ±.034
Yahoo	100	Manual	0.66 ±.013	0.66 ±.012
Yelp	10	Manual	0.45 ±.055	0.45 ±.053
Yelp	100	Manual	0.61 ±.013	0.62 ±.009
Ukraine	10	Manual	0.27 ±.039	0.27 ±.029
Ukraine	100	Manual	0.58 ±.042	0.59 ±.025
Ukraine	250	Manual	0.67 ±.019	0.69 ±.018
Ukraine	10	Autom.	0.20 ±.025	0.20 ±.021
Ukraine	100	Autom.	0.40 ±.027	0.41 ±.018
Ukraine	250	Autom.	0.60 ±.025	0.62 ±.025

Table 7: Mean (majority) macro-F1 scores (± standard deviation) using LoRA and PETapter. For Ukraine, we present the results using the *Stratified* sampling. *Manual* PVP means *Prompt* pattern for AG, Yahoo, and Yelp; for Ukraine, it represents the combination of *Pattern* using the *Normal* verbalizers, *Autom.* represents the combination of *No Pattern* and *Alpha*.

anced data as much as PET. Moreover, as a follow-up study, we want to investigate to what extent or at what level of unbalancedness it is worth simply omitting observations from a stratified sampling in order to obtain a more balanced training set.

Following the mathematical definition of PETapter (cf. Equation 1), it is recommended to always use the same number of verbalizer tokens for each class. We also recommend using as simple patterns as possible. Initial experiments have shown that PET in particular leads to poor results with small n if the pattern is too complex. PETapter was slightly more robust against the choice of pattern in these experiments. In addition, although grammatical correctness of the pattern is desirable, it is not essential to achieve satisfactory results.

For the activation function, we decided to use GELU in combination with LayerNorm (cf. Figure 1). We will further investigate this decision in future studies by examining the influence of using alternative activation functions.

8 Conclusion

We show that our novel method PETapter combines the advantages of few-shot learning and parameter-efficient fine-tuning (PEFT). PETapter combines PEFT methods with PET-style classification heads. In this way, it makes the idea of PET easily accessible in PEFT settings. As a result, it achieves PET performance and increased reliability of the results while offering all the advantages of PEFT. It can be trained faster with higher parameter efficiency and without catastrophic forgetting. Furthermore, it offers a modularity that makes it easier to share models because the newly learned parameters require, e.g., only 16 MB compared to 2.1 GB of disk space. Due to the implementation in the Adapters (Poth et al., 2023) framework, it allows easy execution and combination with existing/new PEFT methods like QLoRA (Dettmers et al., 2023), mixture of experts models (Zadouri et al., 2024), or others (cf. Section 2.1).

Better performance can probably also be achieved by using DeBERTa (He et al., 2023) or dedicated pretrained models, e.g. DeBERTa pretrained on the PoliStance Affect dataset[2]. In addition, the use of a dataset-independent pretraining of the first linear layer after the PEFT module (lower purple segment in Figure 1) as in T-Few (Liu et al., 2022) might lead to better results as well, or at least to requiring fewer iterations to get to the same performance. Furthermore, examining the implementation of other elements of T-Few, e.g. the use of the unlikelihood or length-normalization loss, in combination with PETapter or implementing automated verbalizer search or including meta learning via proxy tasks/soft labels similarly to iPET may also be beneficial.

[2] https://huggingface.co/mlburnham/deberta-v3-large-polistance-affect-v1.0

Limitations

As mentioned in Sections 7 and 8, not all promising models and combinations of model ingredients can be evaluated. Therefore, it is expected that better performance scores can be achieved by optimizing the combination of all mentioned ingredients, cf. Section 2. Instead, we show that the PETapter idea is a promising method overall, if the underlying task is possible to formulate as classification tasks. In addition, our analyses are limited to four data sets (also for reasons of sustainable NLP), which are restricted to the languages English and German. In return, we pay a lot of attention to a reliable evaluation through 5 repetitions $\times$ 5 sampled datasets for each of the considered scenarios in Sections 5 and 6.

References

Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C. Jess Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, Michael Noetel, and Andreas Stuhlmüller. 2021. RAFT: A real-world few-shot text classification benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Nadav Benedek and Lior Wolf. 2024. PRILoRA: Pruned and rank-increasing low-rank adaptation. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 252–263, St. Julian's, Malta. Association for Computational Linguistics.

Canyu Chen and Kai Shu. 2023. PromptDA: Label-guided data augmentation for prompt-based few shot learners. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 562–574, Dubrovnik, Croatia. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTav3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *The Eleventh International Conference on Learning Representations*.

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 *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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.

Shengding Hu, Ning Ding, Weilin Zhao, Xingtai Lv, Zhen Zhang, Zhiyuan Liu, and Maosong Sun. 2023a. OpenDelta: A plug-and-play library for parameter-efficient adaptation of pre-trained models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 274–281, Toronto, Canada. Association for Computational Linguistics.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023b. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.

Yongxin Huang, Kexin Wang, Sourav Dutta, Raj Patel, Goran Glavaš, and Iryna Gurevych. 2023. AdaSent: Efficient domain-adapted sentence embeddings for few-shot classification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3420–3434, Singapore. Association for Computational Linguistics.

Hendrik Hüning, Lydia Mechtenberg, and Stephanie Wang. 2022. Detecting arguments and their positions in experimental communication data. Available at SSRN.

Lena Jurkschat, Gregor Wiedemann, Maximilian Heinrich, Mattes Ruckdeschel, and Sunna Torge. 2022. Few-shot learning for argument aspects of the nuclear energy debate. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 663–672, Marseille, France. European Language Resources Association.

Rabeeh Karimi Mahabadi, Luke Zettlemoyer, James Henderson, Lambert Mathias, Marzieh Saeidi, Veselin Stoyanov, and Majid Yazdani. 2022. Prompt-free and efficient few-shot learning with language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), pages 3638–3652, Dublin, Ireland. Association for Computational Linguistics.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. 2024. VeRA: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*.

Changmao Li and Jeffrey Flanigan. 2024. Task contamination: Language models may not be few-shot anymore. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):18471–18480.

Tongtao Ling, Lei Chen, Yutao Lai, and Hai-Lin Liu. 2023. Evolutionary verbalizer search for prompt-based few shot text classification. In *Knowledge Science, Engineering and Management*, pages 279–290, Cham. Springer Nature Switzerland.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 1950–1965. Curran Associates, Inc.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. DoRA: Weight-decomposed low-rank adaptation.

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.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Mahdi Nikdan, Soroush Tabesh, Elvir Crnčević, and Dan Alistarh. 2024. RoSA: Accurate parameter-efficient fine-tuning via robust adaptation.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Sebastian Ruder, Ivan Vulić, and Edoardo Maria Ponti. 2023. Modular deep learning.

Clifton Poth, Hannah Sterz, Indraneil Paul, Sukannya Purkayastha, Leon Engländer, Timo Imhof, Ivan Vulić, Sebastian Ruder, Iryna Gurevych, and Jonas Pfeiffer. 2023. Adapters: A unified library for parameter-efficient and modular transfer learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 149–160, Singapore. Association for Computational Linguistics.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jonas Rieger, Kostiantyn Yanchenko, Mattes Ruckdeschel, Gerret von Nordheim, Katharina Kleinen-von Königslöw, and Gregor Wiedemann. 2023. Few-shot learning for automated content analysis (FLACA) in the German media debate on arms deliveries to Ukraine. In *Digital Total*, page 19.

Jonas Rieger, Kostiantyn Yanchenko, Mattes Ruckdeschel, Gerret von Nordheim, Katharina Kleinen-von Königslöw, and Gregor Wiedemann. 2024. Few-shot learning for automated content analysis: Efficient coding of arguments and claims in the debate on arms deliveries to Ukraine. *Studies in Communication and Media*, 13:72–100.

Mohammed Sabry and Anya Belz. 2023. PEFT-Ref: A modular reference architecture and typology for parameter-efficient finetuning techniques.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2022. True few-shot learning with Prompts—A real-world perspective. *Transactions of the Association for Computational Linguistics*, 10:716–731.

Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. 2022. Efficient few-shot learning without prompts.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Prateek Yadav, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023. ComPEFT: Compression for communicating parameter efficient updates via sparsification and quantization.

Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. 2024. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Haoxing Zhang, Xiaofeng Zhang, Haibo Huang, and Lei Yu. 2022. Prompt-based meta-learning for few-shot text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1342–1357, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.

A Details of Benchmark Study

For the benchmark study, we compare the use of "roberta-base" and "roberta-large" from HuggingFace's Transformers (Wolf et al., 2020). For PET (`https://github.com/timoschick/pet`), we use the two parameters `pet_num_train_epochs=10` and `pet_per_gpu_train_batch_size=1` that differ from the default. For all experiments using the package Adapter (Poth et al., 2023), we use the parameters

- `c_rate=16` (Pfeiffer),
- `r=8` (LoRA),
- `alpha=16` (LoRA),
- `learning_rate=5.0e-5`,
- `max_epochs=30`,
- `per_device_train_batch_size=2`

and alternate `arch` with `ia3`, `lora`, and `pfeiffer`. The patterns in the following subsections are motivated by the results of the study of Schick and Schütze (2022). Due to the limited input length of PLMs, potential truncations of the input elements in the pattern are indicated with *, potentially as group within { } brackets.

A.1 AG's News

This English language dataset is available at `https://huggingface.co/datasets/ag_news` (Zhang et al., 2015) and consists of 120 thousand training and 7.6 thousand test observations.

Prompt Pattern [MASK] News: [text]*

Q&A Pattern [text]* [SEP] Question: What is the topic of this article? Answer: [MASK].

Verbalizers

World World
Sports Sports
Business Business
Sci/Tech Tech

A.2 Yahoo Questions

This English language dataset is available at `https://huggingface.co/datasets/yahoo_answers_topics` (Zhang et al., 2015) and consists of 1.4 million training and 60 thousand test observations.

Prompt Pattern [MASK] Question: {[question_title] [question_content] [best_answer]}*

Q&A Pattern {[question_title] [question_content] [best_answer]}* [SEP] Question: What is the topic of this question? Answer: [MASK].

Verbalizers

Society & Culture Society
Science & Mathematics Science
Health Health
Education & Reference Education
Computers & Internet Computer
Sports Sports
Business & Finance Business
Entertainment & Music Entertainment
Family & Relationships Relationship
Politics & Government Politics

	Label	Random Sampling PETapter	Random Sampling PET	Random Sampling Lin. Layer	Stratified Sampling PETapter	Stratified Sampling PET	Stratified Sampling Lin. Layer
Precision	argumentagainst	0.57 ±.042	0.57 ±.126	0.37 ±.228	0.54 ±.037	0.54 ±.122	0.39 ±.092
	argumentfor	0.64 ±.050	0.66 ±.141	0.45 ±.117	0.64 ±.032	0.66 ±.145	0.47 ±.135
	claimagainst	0.68 ±.030	0.66 ±.141	0.52 ±.063	0.68 ±.036	0.66 ±.143	0.49 ±.065
	claimfor	0.81 ±.027	0.81 ±.077	0.65 ±.065	0.84 ±.022	0.83 ±.080	0.65 ±.064
Recall	argumentagainst	0.51 ±.059	0.53 ±.147	0.12 ±.093	0.53 ±.064	0.57 ±.140	0.16 ±.098
	argumentfor	0.61 ±.056	0.61 ±.140	0.42 ±.169	0.63 ±.053	0.61 ±.138	0.42 ±.147
	claimagainst	0.75 ±.039	0.74 ±.157	0.54 ±.175	0.78 ±.028	0.75 ±.161	0.54 ±.121
	claimfor	0.80 ±.029	0.82 ±.047	0.76 ±.054	0.78 ±.022	0.80 ±.051	0.74 ±.051
Macro-F1	argumentagainst	0.53 ±.037	0.55 ±.128	0.16 ±.109	0.53 ±.045	0.55 ±.123	0.22 ±.100
	argumentfor	0.62 ±.046	0.63 ±.134	0.43 ±.142	0.63 ±.024	0.63 ±.134	0.44 ±.138
	claimagainst	0.71 ±.025	0.70 ±.146	0.52 ±.124	0.72 ±.028	0.70 ±.148	0.51 ±.084
	claimfor	0.81 ±.016	0.81 ±.039	0.70 ±.036	0.81 ±.012	0.81 ±.040	0.69 ±.049

Table 8: Mean precision, recall, and macro-F1 scores per label (each ± standard deviation) of the experiments in the real-world study (Ukraine). We consider the Pfeiffer adapter as the PEFT method of PETapter and $n = 250$. In addition to the results from Table 5, we present the results of random sampling in comparison to stratified sampling.

A.3 Yelp Full

This English language dataset is available at `https://huggingface.co/datasets/yelp_review_full` (Zhang et al., 2015) and consists of 650 thousand training and 50 thousand test observations.

Prompt Pattern [text]* [SEP] All in all, it was [MASK].

Q&A Pattern [text]* [SEP] Question: What does the customer think of this restaurant? Answer: [MASK].

Verbalizers

1 star terrible

2 stars bad

3 stars okay

4 stars good

5 stars great

B Details of the Real-World Study

The dataset will be made available to researchers after acceptance of the paper. Since this dataset is in German language, we use "xlm-roberta-large" from HuggingFace's Transformers (Wolf et al., 2020). For PET (`https://github.com/timoschick/pet`), we use the two parameters `pet_num_train_epochs=10` and `pet_per_gpu_train_batch_size=1` (for a fair training time comparison in Table 2, we use `pet_per_gpu_train_batch_size=2`) that differ from the default. For all experiments using the package Adapter (Poth et al., 2023), we use the parameters

- `c_rate=16` (Pfeiffer),
- `r=8` (LoRA),
- `alpha=16` (LoRA),
- `learning_rate=5.0e-5`,
- `max_epochs=30`,
- `per_device_train_batch_size=2`

and alternate `arch` with `ia3`, `lora`, and `pfeiffer`.

Due to the limited input length of PLMs, potential truncations of the input elements in the pattern are indicated with *, potentially as group within {} brackets. We place the target sentence at the beginning to ensure that it is never truncated due to restrictions regarding the model's input length of 512 tokens. Experiments with target_sentence in the middle returned slightly worse results. In the same way, experiments with an equivalent German pattern yield slightly worse results.

Pattern This sentence contains [MASK] [MASK] arms deliveries to Ukraine: {[target_sentence] [SEP] [context_before] [SEP] [context_after]}*

No Pattern [MASK] [MASK]: {[target_sentence] [SEP] [context_before] [SEP] [context_after]}*

Alpha Verbalizers
The alphanumeric verbalizers only consist of one token each, so that in this case the used pattern is reduced to one [MASK] token only.

argumentagainst a

argumentfor b

claimagainst c

claimfor d

nostance e

Normal Verbalizers

argumentagainst argument against
argumentfor argument for
claimagainst claim against
claimfor claim for
nostance nothing regarding

Shuffle Verbalizers

argumentagainst claim for
argumentfor claim against
claimagainst nothing regarding
claimfor argument against
nostance argument for

C RAFT Benchmark

The RAFT Leaderboard (Alex et al., 2021, `https://raft.elicit.org/`) has been under maintenance for some time now. We made a submission a while ago, but unfortunately still got no score. If the scores are published during the review process, we will include a table with comparative values for T-Few (Liu et al., 2022), Setfit (Tunstall et al., 2022), PET (Schick and Schütze, 2021) and the human baseline at this point.

The 11 datasets of RAFT are in English language and available at `https://huggingface.co/datasets/ought/raft`. We use the model "microsoft/deberta-v2-xxlarge" from HuggingFace's Transformers (Wolf et al., 2020) and for all 11 datasets the parameters

- `arch=lora`,
- `r=8`,
- `alpha=16`,
- `learning_rate=5.0e-5`,
- `max_epochs=30`,
- `per_device_train_batch_size=2`,
- `number_of_runs=5`,

where we select the majority vote out of the five runs as the final submission.

The patterns in the following subsections are motivated by the results of the study of Schick and Schütze (2022). Due to the limited input length of PLMs, potential truncations of the input elements in the patterns are indicated with *, potentially as group within { } brackets.

C.1 ade_corpus_v2

Pattern [Sentence]* [SEP] Question: Is this sentence related to an adverse drug effect (ADE)? Answer: [MASK].

Verbalizers

not ADE-related No
ADE-related Yes

C.2 banking_77

For the processing of the *banking_77* dataset, some preprocessing is necessary. As there are only 50 observations in each of the training sets of the RAFT setting, but at the same time there are 77 different classes in this specific dataset, the model misses 27 of the classes in the training set. To overcome this, we augment the training data such that each of the 50 observations is combined with all 77 classes using the yes/no pattern below. As a result, the 27 classes that were previously not included in the training data now become part it each 50 times in combination with the label *no*.

Pattern The following is a banking customer service query. [SEP] {[Query] [SEP] Is [Label]}* the correct category for this query? Answer: [MASK].

Verbalizers

No No
Yes Yes

C.3 neurips_impact_statement_risks

Pattern {Title: [Paper title] [SEP] Statement: [Impact statement]}* [SEP] Question: Does this impact statement mention a harmful application? Answer: [MASK].

Verbalizers

doesn't mention a harmful application No
mentions a harmful application Yes

C.4 one_stop_english

Pattern [Article]* [SEP] Question: Is the level of this article 'elementary', 'intermediate' or 'advanced'? Answer: [MASK].

Verbalizers

elementary elementary
intermediate intermediate
advanced advanced

C.5 overruling

Pattern In law, an overruling sentence is a statement that nullifies a previous case decision as a precedent. [SEP] [Sentence]* [SEP] Question: Is this sentence overruling? Answer: [MASK].

Verbalizers

not overruling No

overruling Yes

C.6 semiconductor_org_types

Pattern {Title: [Paper title] [SEP] Organization name: [Organization name]}* [SEP] Question: What is the category of this institution? Answer: [MASK].

Verbalizers

company Company

research institute Institute

university University

C.7 systematic_review_inclusion

Pattern {Journal: [Journal] [SEP] Title: [Title] [SEP] Abstract: [Abstract]}* [SEP] Question: Should this paper be included in a meta-review which includes the findings of systematic reviews on interventions designed to promote charitable donations? Answer: [MASK].

Verbalizers

not included No

included Yes

C.8 tai_safety_research

Pattern Transformative AI (TAI) is defined as AI that precipitates a transition comparable to (or more significant than) the agricultural or industrial revolution [SEP] {Journal: [Publication Title] [SEP] Title: [Title] [SEP] Abstract: [Abstract Note]}* [SEP] Question: Is this paper a TAI safety research paper? Answer: [MASK].

Verbalizers

not TAI safety research No

TAI safety research Yes

C.9 terms_of_service

Pattern The following sentence is from a Terms of Service. [SEP] [Sentence]* [SEP] Question: Is this sentence potentially unfair? Answer: [MASK].

Verbalizers

not potentially unfair No

potentially unfair Yes

C.10 tweet_eval_hate

Pattern [Tweet]* [SEP] Question: Does this tweet contain hate speech against either immigrants or women? Answer: [MASK].

Verbalizers

not hate speech No

hate speech Yes

C.11 twitter_complaints

Pattern [Tweet text]* [SEP] Question: Does this tweet contain a complaint? Answer: [MASK].

Verbalizers

no complaint No

complaint Yes

D GPT Comparison

A different version of the Ukraine dataset, in which not only the four classes outlined in this paper are considered, but also *irrelevant* sentences are contained, is studied by Rieger et al. (2023) in a comparison between full fine-tuning methods, PET and PEFT methods. The authors show that PETapter already with only 272 observations outperforms zero-shot GPT-4. The authors further observe that GPT-4 performs better on this real-world and unpublished dataset in a zero-shot manner than using an in-context learning prompt with few (up to 10) examples. Moreover, they found out that GPT-3.5 performs significantly worse than GPT-4 on this task.