SMASH: Improving SMAll Language Models' Few-SHot Ability with Prompt-Based Distillation

Yueqian Wang[1,2], **Chang Liu**[1,3], **Kai Chen**[4], **Xi Wang**[4], **Dongyan Zhao**[1,3,5,6]*
[1] Wangxuan Institute of Computer Technology, Peking University
[2] School of Intelligence Science and Technology, Peking University
[3] Center for Data Science, Peking University
[4] School of Economics, Peking University
[5] Institute for Artificial Intelligence, Peking University
[6] State Key Laboratory of Media Convergence Production Technology and Systems
{wangyueqian,liuchang97,chen.kai,wang.x,zhaody}@pku.edu.cn,

Abstract

Large-scale language models coupled with prompts have shown remarkable performance on few-shot learning. However, through systematic experiments, we find that the few-shot performance of small language models is poor, and using prompts on them brings fewer improvements than on larger ones. In this paper, we propose **SMASH**, an approach to improve **SMA**ll language models' few-**SH**ot ability by training on intermediate tasks before prompt-based fine-tuning on downstream tasks. We design intermediate tasks for sentence-pair tasks and sentiment classification tasks by creating training examples with prompt templates similar to downstream tasks using sentences sampled from a large-scale unsupervised corpus, and apply knowledge distillation to distill from outputs of larger pre-trained models as the training objective. We conduct extensive experiments and show that SMASH can make a 6-layer DistilRoBRETa-base achieve comparable performance on few-shot datasets with a 12-layer RoBERTa-base at a low cost. [1]

		MNLI (acc)	QQP (f1)	RTE (acc)	SST-2 (acc)
Distil RoBERTa -base	PT	46.8	53.0	54.5	85.7
	FT	38.5	53.1	50.5	75.5
	Δ	8.3	-0.1	4.0	10.2
RoBERTa -base	PT	58.4	63.9	59.6	89.0
	FT	40.7	59.1	50.8	82.1
	Δ	17.7	**4.8**	8.8	6.9
RoBERTa -large	PT†	68.3	65.5	69.1	92.7
	FT†	45.8	60.7	54.4	81.4
	Δ†	**22.5**	**4.8**	**14.7**	**11.3**

Table 1: Fine-tuning performance (FT), prompt-based fine-tuning performance (PT) and relative improvement of PT comparing to FT (Δ) on models with different sizes on 16-shot training and validation dataset. †: results from (Gao et al., 2020). **Bold results** indicates the largest relative improvement.

1 Introduction

Language models at scale, such as GPT-3 (Brown et al., 2020), have shown remarkable performance on prompt-based few-shot learning on a wide variety of tasks given only a natural language prompt and few demonstrations. However, the ability of few-shot learning usually comes with heavy computation and a huge amount of parameters. Recent works (Gao et al., 2020; Li and Liang, 2021) investigated prompt-based few-shot learning on moderately-sized language models such as BERT-large (Devlin et al., 2019), RoBERTa-large (Liu et al., 2019) and GPT-2 (Radford et al., 2019), but these models are still difficult to be deployed on edge devices such as mobile phones.

In this paper, we investigate whether we can make small language models, such as DistilRoBERTa-base (Sanh et al., 2019), better few-shot learners. Prompt-based fine-tuning has been seen as a promising method for few-shot learning as it uses language modeling heads instead of introducing new parameters as task-specific classification heads during fine-tuning, thus narrowing down the gap between pre-training (e.g., masked language modeling for RoBERTa) and applying to downstream tasks. However, Table 1 shows that when annotated data is insufficient, prompt-based fine-tuning on DistilRoBERTa-base does not make improvements over fine-tuning as large as on RoBERTa-base or RoBERTa-large for most downstream tasks. We assume that's because the models' abilities to respond to gaps between tasks are proportional to their sizes, and the gap of transferring from pre-training to the prompt-based fine-tuning

* Corresponding author: Dongyan Zhao.
[1] Our model and code is publicly available at https://github.com/yellow-binary-tree/SMASH.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 6608–6619
December 7-11, 2022

on downstream tasks directly is still too wide for small language models. This suggests that additional adaptations are required when applying small language models on few-shot downstream tasks.

To tackle this problem, we propose **SMASH**, an approach of further training **SMA**ll language models on intermediate tasks before applying them to few-**SH**ot downstream tasks. Intuitively, if we can design intermediate tasks similar to both the pre-training task and downstream tasks, we can mitigate this problem by replacing one large gap (from the pre-training task to downstream tasks) with two smaller gaps (from the pre-training task to intermediate tasks, then to downstream tasks). Noticing that same manual prompt templates (or similar templates with minor differences) are often used when prompt-based fine-tuning models on a group of similar tasks (e.g., template <s>x_0?<mask>, x_1.</s> for sentence-pair tasks in GLUE benchmark (Wang et al., 2019), such as MNLI, QNLI, RTE, etc.), we consider using this prompt template and sample sentences from a large-scale unsupervised corpus to form the inputs of the intermediate task. To construct supervision signals we leverage knowledge distillation (Hinton et al., 2015) by feeding the inputs to a larger pre-trained language model and using its outputs as the training objective. In this way, the intermediate task can be both similar to the pre-training task (by training on similar distributions of data, e.g. large scale corpus from the web) and downstream tasks (by using similar prompt templates). From the perspective of knowledge distillation, the intermediate task can also be seen as a kind of data augmentation using a large-scale unsupervised corpus to transfer knowledge of solving a group of similar tasks from larger models to smaller models, which can be exploited later by prompt-based fine-tuning on downstream tasks.

As the intermediate task depends on the input format of downstream tasks, it's not feasible to experiment with SMASH on all NLP tasks at once. In this paper, we only take *sentence-pair tasks* and *sentiment classification tasks*, two groups of tasks that are popular in NLP as an example, and design two intermediate tasks respectively. Note that practitioners can also use SMASH on other groups of downstream tasks by designing their own intermediate tasks under the training framework we proposed. Experiments on the GLUE benchmark and several other tasks show that using SMASH can make a 6-layer DistilRoBERTa-base achieve comparable performance with a 12-layer RoBERTa-base on few-shot datasets at a low cost. We find that SMASH provides more improvements on more complicated tasks like natural language inference and sentence similarity than easier tasks like sentiment classification, and is robust over different templates, verbalizers, and model structures. In summary, our key contributions are:

- Conducting systematic experiments to verify the effectiveness of existing few-shot learning methods on small language models;
- Proposing SMASH, a general method to improve few-shot prompt-based fine-tuning performance for small language models on a group of downstream tasks;
- Designing intermediate tasks for sentence-pair tasks and sentiment classification tasks, and showing their effectiveness on several downstream tasks.

2 Related Work

Prompt-based learning. Prompt-based learning has become a new paradigm in NLP fueled by the series work of GPT (Radford et al., 2018, 2019; Brown et al., 2020). There are a large body of works on mining sequences of tokens as discrete prompts (Jiang et al., 2020; Shin et al., 2020) or training continuous prompts (Li and Liang, 2021; Liu et al., 2021; Lester et al., 2021; Hambardzumyan et al., 2021; Zhang et al., 2022). Prompt-based learning has been seen as a promising method for few-shot learning. PET (Schick and Schütze, 2021a,b) focuses on prompt-based fine-tuning an ensemble of models to create a soft-labeled dataset from unlabeled in-domain examples and use it to fine-tune a classifier model. LM-BFF (Gao et al., 2020) proposes methods to generate prompts and find demonstrations from few-shot datasets. The work most related to us is PPT (Gu et al., 2021), which leverages unsupervised data to pre-train representations of prompt tokens.

General distillation of Pre-trained Language Models (PLMs). General Distillation aims at distilling student models at the pre-training stage using unsupervised data to foster its ability on solving various types of tasks. DistilBERT (Sanh et al., 2019) performs general distillation using soft cross-entropy loss over logits and cosine embedding loss. TinyBERT (Jiao et al., 2020) proposes transformer

distillation method that aligns embedding layer, hidden states, attention matrices, and output logits between teacher model and student model. CoDIR (Sun et al., 2020) adopts a contrastive learning framework where the student's representations are enforced to be close to the corresponding representations of the teacher and far apart from negative ones. MiniLM (Wang et al., 2020) proposes self-attention distillation at the pre-training stage, and MiniLMv2 (Wang et al., 2021) proposes multi-head self-attention relation distillation that has no restrictions in terms of the number of teacher's and student's attention heads. MGSKD (Liu et al., 2022) proposes to transfer the structural relations among multi-granularity representations to the student hierarchically. Though the effects of different training objectives have been studied extensively, most previous works conducted general distillation on the masked language modeling task using raw text input. To the best of our knowledge, our work is the first work that performs prompt-based general distillation using input sequences assembled by prompt templates.

3 Method

3.1 Preliminaries: Prompt-Based Fine-tuning

An burgeoning approach of fine-tuning pre-trained language models is *prompt-based fine-tuning*, where input is formulated as a "blank-filling task" with natural language prompts. Take sentiment classification task as an example, given an input sentence $x \in \mathcal{V}^*$ (where $\mathcal{V}$ is the vocabulary of the language model $\mathcal{M}$, and $\mathcal{V}^*$ denotes a sequence of tokens from $\mathcal{V}$) and its label $y \in \mathcal{Y}$, a template $f : \mathcal{V}^* \to \mathcal{V}^*$ maps x into a new token sequence $f(x)$ containing the input sentence, several prompt tokens, and at least one `<mask>` token for $\mathcal{M}$ to predict. Then a verbalizer $v : \mathbb{R}^{|\mathcal{V}|} \to \mathcal{Y}$ is used to map p, the output distribution of $\mathcal{M}$ to a label $v(p) = y \in \mathcal{Y}$. For example, we can formulate a sentiment classification task $t = (f, v)$ as:

$$f(x) = \texttt{<s>}\ x.\ \texttt{It was <mask>.</s>}$$

and let $\mathcal{M}$ decide whether it is more appropriate to fill in "*terrible*" (negative) or "*great*" (positive) for `<mask>`. Then the verbalizer $v = [great, terrible]$ maps the output distribution of $\mathcal{M}$ to a label:

$$v(p) = \begin{cases} positive & p(\text{“}great\text{”}) > p(\text{“}terrible\text{”}) \\ negative & otherwise \end{cases}$$

Same as (Gao et al., 2020), we treat regression tasks in a bounded interval $[l, u]$ as interpolation between two opposing words in verbalizer $v = [y_l, y_u]$. In this way, we calculate $\hat{y}$ as:

$$\hat{y} = l \times p(y_l|x) + u \times p(y_u|x)$$

where p is the output probability of model $\mathcal{M}$ and $p(y_l|x) + p(y_u|x) = 1$. We train the model $\mathcal{M}$ to minimize the following objective function using KL divergence:

$$\mathcal{L}_{kl} = KL(p_{pred}|p_{gold}),$$

where

$$p_{pred} = [p(y_l|x), p(y_u|x)]$$
$$p_{gold} = [\frac{u-y}{u-l}, \frac{y-l}{u-l}]$$

3.2 SMASH

In this subsection we propose SMASH to leverage unsupervised corpus to transfer the ability of solving a group of downstream tasks from a pre-trained teacher model $\mathcal{M}_{tea}$ to a pre-trained student model $\mathcal{M}_{stu}$ to further narrow down the gap between pre-training and prompt-based fine-tuning. Formally, suppose a group of downstream task $\mathcal{T}$ containing n downstream tasks $\{t_1, ..., t_n\}$, where $t_i = (f_i, v_i)$. We design an intermediate task $t^{dis} = (f^{dis}[, v^{dis}])$ for group $\mathcal{T}$. [2] After distilling $\mathcal{M}_{stu}$ from $\mathcal{M}_{tea}$ with t^{dis}, we continue to train $\mathcal{M}_{stu}$ for each task t in $\mathcal{T}$ with prompt-based fine-tuning on task-specific data.

In this work, we focus on two groups of downstream tasks: *sentence-pair classification* and *sentiment classification*, and provide the design of their intermediate tasks respectively. We emphasize that SMASH can be applied to any downstream tasks, and practitioners can design their own corresponding intermediate tasks by themselves.

Sentence-Pair Tasks. Sentence-pair tasks such as natural language inference take two sentences (x_1, x_2) as input. Following (Gu et al., 2021), we construct a dataset from unlabeled raw text documents, and set the two sentences from different documents as label 0, those from the same document but not adjacent as label 1, and those next to each other as label 2. We sampled the

[2] The verbalizer of intermediate tasks is optional as $\mathcal{M}_{stu}$ can learn from the output distribution of the whole vocabulary from $\mathcal{M}_{tea}$ directly.

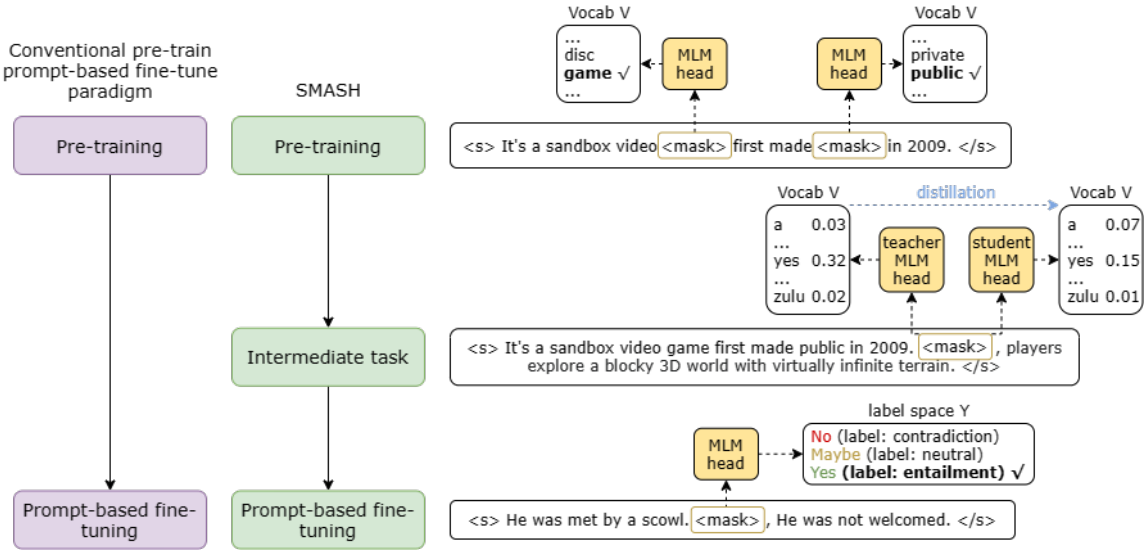


Figure 1: Overview of SMASH on sentence-pair tasks.

same amount of training examples for each label to avoid distribution bias. We use template f^{dis} = <s>x_1? <mask>, x_2</s>. Figure 1 is an overview of SMASH on sentence-pair tasks.

Sentiment Classification Tasks. Sentence classification tasks take only one sentence x as input. In order to comply with the setting where annotated data is difficult to obtain, we filter sentences with low-classification probability with a pre-trained RoBERTa-large model instead of a model fine-tuned on another sentiment analysis task in (Gu et al., 2021). To filter the low-classification probability sentences, we use template

$$f^{filter} = \texttt{<s>}x_1\ \texttt{It was <mask>. </s>}$$

and verbalizer

$$v^{filter} = [terrible, bad, okay, good, great]$$

After filtering we distill $\mathcal{M}_{stu}$ using template $f^{dis} = f^{filter}$.

We experimented on three transformer distillation objectives: prediction distillation, hidden states distillation, and multi-head attention distillation.

For **prediction distillation**, we train $\mathcal{M}_{stu}$ using the output distribution of $\mathcal{M}_{tea}$ by minimizing the following objective function:

$$\mathcal{L}_{ce} = -\sum_{i \in \mathcal{V}} t_i \times log(s_i) \quad (1)$$

where t_i and s_i are probabilities of token i estimated by $\mathcal{M}_{tea}$ and $\mathcal{M}_{stu}$ respectively.

For **hidden states distillation**, the objective is defined as:

$$\mathcal{L}_{hidn} = \sum_{m=0}^{M^{stu}} MSE(H_m^{stu}, H_{g(m)}^{tea}) \quad (2)$$

where MSE denotes mean squared error loss, M^{stu} is the number of layers of the student model, H_m refers to the hidden states of mth layer, and H_0 denotes the representations after embedding layer. $g(m)$ denotes the layer mapping function from student to teacher, which means mth layer of $\mathcal{M}_{stu}$ learns from $g(m)$th layer of $\mathcal{M}_{tea}$. We use uniform strategy described in (Jiao et al., 2020), where $g(m) = m \times (M^{tea}/M^{stu})$.

For **multi-head attention distillation**, the objective is defined as:

$$\mathcal{L}_{attn} = \sum_{m=1}^{M^{stu}} \sum_{h=1}^{H} MSE(A_{mh}^{stu}, A_{g(m)h}^{tea}) \quad (3)$$

where H is the number of attention heads, A_{mh} refers to the hth attention head of mth layer. We use the same layer mapping function as hidden states distillation, i.e. $g(m) = m * (M^{tea}/M^{stu})$. Note that hidden states distillation and multi-head attention distillation requires $\mathcal{M}_{stu}$ and $\mathcal{M}_{tea}$ to be the same type of transformer language model with the same number of attention heads and hidden size, while prediction distillation has no requirements for the model structure of $\mathcal{M}_{stu}$ and $\mathcal{M}_{tea}$.

After conducting experiments on all the above-mentioned distillation objectives described in detail in Section 4.4, we only use prediction distillation in

other experiments due to its superior performance and simplicity.

4 Experiment

4.1 Setup

Datasets. We evaluate our methods on SNLI (Bowman et al., 2015), a number of tasks from the GLUE benchmark (Wang et al., 2019), and three more sentiment analysis datasets: SST-5 (Socher et al., 2013b), MR (Pang and Lee, 2005) and CR (Hu and Liu, 2004). Our few-shot dataset is the same as (Gao et al., 2020), with $K = 16$ examples per label sampled from the original training set as our training set and validation set, and use the original validation set as our test set. For each task, we sampled 5 different few-shot datasets and report the average (standard deviation) metric of 5 trials. For intermediate tasks, we sampled 2560k sentence pairs for sentence-pair task and 2560k sentences for sentiment classification task from Wikipedia [3], so every training example is used only once.

Implementation Details. During the intermediate task, we use RoBERTa-large as the teacher model and DistilRoBERTa-base [4] as the student model. We set batch size as 128 and learning rate as 1e-4. During prompt-based fine-tuning, we perform grid search and take learning rates from {1e-5, 2e-5, 5e-5} and batch sizes from {2, 4}. We set the max length of input sequences as 64 for sentiment classification tasks and 128 for sentence-pair tasks. We use templates and verbalizers same as (Gao et al., 2020). For more implementation details please refer to Appendix A.

Baselines. We compare to several baselines including: (1) prompt-based fine-tuning DistilRoBERTa-base (**prompt-based fine-tune**); (2) prompt-based zero-shot performance of DistilRoBERTa-base (**zero-shot**); (3) fine-tuning DistilRoBERTa-base (**fine-tune**); (4) knowledge distillation by adding the soft label loss (Eq. 1) between logits from RoBERTa-large and DistilRoBERTa-base when prompt-based fine-tuning DistilRoBERTa-base (**distill**), and use back-translation (Sennrich et al., 2015) to augment the training set 10 times larger (**distil bt10**); (5) **LM-BFF** (Gao et al., 2020) which uses automatically-searched prompts and demonstrations; (6) **Vanilla PPT** (Gu et al., 2021) which leverages unsupervised corpus to pre-train soft-prompts; and (7) prompt-based fine-tuning **(PT) RoBERTa-base, RoBERTa-large**, and prompt-based **zero-shot** performance of **RoBERTa-base**.

[3] https://huggingface.co/datasets/wikipedia

[4] We use model checkpoint from https://huggingface.co/distilroberta-base, with 6 layers, 768 dimension, 12 heads, and 82M parameters.

4.2 Main Results

Table 2 shows our main results. On most tasks SMASH is comparable with the $2\times$ larger RoBERTa-base model on both prompt-based fine-tuning and zero-shot performance, and it also outperforms LM-BFF and Vanilla PPT, which shows the effectiveness of SMASH.

For sentence-pair tasks, prompt-based zero-shot performance of DistilRoBERTa-base is worse than or slightly better than the majority baseline, but it can be improved by SMASH; For sentiment classification tasks, prompt-based zero-shot performance of DistilRoBERTa-base is much better than the majority baseline, and SMASH cannot provide further improvements. We suppose that's because in the pre-training stage DistilRoBETa-base only sees one sentence at a time, so transferring from pre-training to sentence-pair tasks is harder, thus training on SMASH intermediate task is more beneficial.

We observed that distilling directly from RoBERTa-large using prompt-based methods performs worse than prompt-based fine-tuning on few-shot datasets, as it's hard to train a strong task-specific teacher model and transfer knowledge from teacher model to student model due to lack of training data. It is counter-intuitive that the student model performs even worse with the use of back-translation, we suppose that's because the teacher model is not powerful enough to resolve the noise introduced by the augmented examples.

We also observed that on most tasks, prompt-based fine-tuning improvements of SMASH are significantly greater than zero-shot improvements (e.g., for RTE, prompt-based fine-tuning improvement is $59.4 - 54.5 = 4.9$, and zero-shot improvement is $53.8 - 51.3 = 2.5$). This verifies our assumption that instead of making the student model learn to solve downstream tasks directly, SMASH works like general distillation that transfers the potential ability to solve a group of similar tasks to the student model, which can be exploited later by prompt-based fine-tuning on downstream tasks.

	MNLI-m (acc)	MRPC (f1)	QQP (f1)	SNLI (acc)	QNLI (acc)	RTE (acc)
majority†	32.7	81.2	0.0	33.8	49.5	52.7
prompt-based fine-tune	46.8 (1.5)	70.0 (6.5)	53.0 (1.5)	50.1 (5.0)	54.1 (2.2)	54.5 (3.9)
zero-shot	44.8	73.0	50.1	37.5	50.6	51.3
fine-tune	38.5 (1.5)	65.4 (15.1)	53.1 (5.5)	39.8 (2.0)	54.6 (2.5)	50.5 (1.4)
distill	46.6 (1.5)	67.1 (9.9)	53.3 (1.9)	49.9 (3.7)	53.7 (2.1)	54.9 (2.5)
distill bt10	44.2 (3.4)	72.3 (2.8)	53.0 (2.5)	48.4 (2.9)	52.8 (2.8)	54.9 (1.8)
LM-BFF	50.3 (2.0)	64.3 (4.4)	52.5 (5.5)	51.0 (6.7)	54.8 (2.4)	52.8 (3.1)
Vanilla PPT	39.6 (0.7)	75.3 (2.9)	59.8 (2.6)	44.4 (2.7)	62.1 (3.6)	51.5 (2.0)
SMASH	**56.8 (1.1)**	74.7 (5.5)	**64.4 (2.4)**	**59.7 (4.7)**	**66.9 (3.8)**	**59.4 (3.2)**
SMASH zero-shot	46.7	**76.0**	52.8	47.3	50.9	53.8
RoBERTa-base PT	58.4 (1.7)	72.1 (10.9)	63.9 (4.0)	61.6 (4.2)	61.3 (5.0)	59.6 (6.6)
RoBERTa-base zero-shot	48.1	53.0	51.6	48.6	50.8	53.1
RoBERTa-large PT†	68.3 (2.3)	74.5 (5.3)	65.5 (5.3)	77.2 (3.7)	64.5 (4.2)	69.1 (3.6)

	MNLI-mm (acc)	STS-B (pear.)	SST-2 (acc)	SST-5 (acc)	MR (acc)	CR (acc)
majority†	33.0	-	50.9	23.1	50.0	50.0
prompt-based fine-tune	48.2 (3.3)	42.3 (15.4)	85.7 (1.4)	**42.7 (1.5)**	80.4 (1.4)	85.8 (1.8)
zero-shot	45.7	-5.7	81.8	28.4	78.0	84.0
finetune	39.3 (1.5)	56.7 (10.1)	75.5 (4.0)	35.5 (0.9)	67.9 (3.8)	74.5 (4.2)
distill	48.7 (1.8)	42.2 (15.6)	86.3 (1.0)	41.2 (1.1)	80.4 (1.6)	86.7 (1.6)
distill bt10	46.6 (3.2)	33.6 (13.3)	82.9 (2.2)	40.2 (1.5)	78.9 (1.6)	84.1 (1.9)
LM-BFF	52.9 (1.9)	42.0 (15.3)	86.1 (0.7)	38.4 (2.6)	79.8 (2.4)	84.7 (2.7)
Vanilla PPT	39.8 (1.4)	55.7 (3.0)	83.6 (0.9)	36.8 (1.8)	80.2 (0.6)	86.5 (0.6)
SMASH	**58.8 (0.9)**	**64.3 (7.4)**	**88.3 (0.4)**	42.4 (2.6)	**81.9 (2.2)**	**88.0 (0.8)**
SMASH zero-shot	47.9	19.5	75.7	31.1	74.0	77.6
RoBERTa-base PT	60.6 (1.4)	69.1 (3.0)	89.0 (1.2)	44.5 (1.6)	84.3 (1.3)	89.2 (1.4)
RoBERTa-base zero-shot	49.2	14.5	77.8	32.8	72.3	79.7
RoBERTa-large PT†	70.5 (1.9)	71.0 (7.0)	92.7 (0.9)	47.4 (2.5)	87.0 (1.2)	90.3 (1.0)

Table 2: Main results. †: results from (Gao et al., 2020). **Bold Results** indicates the best result achieved using DistilRoBERTa-base and no extra in-domain training data.

4.3 Comparisons to Baselines with Stronger Data Requirements

In this subsection we also compare our methods to **iPET** (Schick and Schütze, 2021a), which iteratively trains ensembles of PET models to label in-domain unlabeled examples. We use RoBERTa-large as PET models and DistilRoBERTa-base as the final classifier. Note that this comparison is not fair as SMASH only requires web-scraped corpus, while iPET requires up to 10000 unlabeled examples per label from the same downstream task, which can play a key role in few-shot learning.

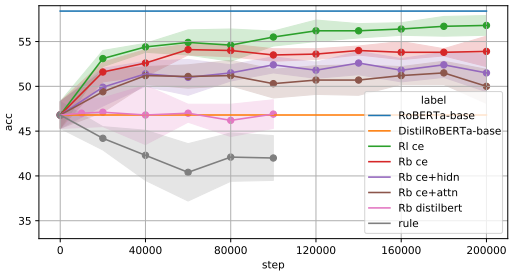
Results of some representative tasks are shown in Table 3 (see Table 5 in Appendix for more results). SMASH results in inferior performance than iPET on some of the tasks, as iPET fine-tunes the final classifier on soft-labeled in-domain examples several orders of magnitude larger than the 16-shot dataset. However, we find that SMASH and iPET are compatible, as simply changing the training method of iPET final classifier from fine-tuning to prompt-based fine-tuning **(+PT)** and using models trained on SMASH intermediate tasks as initialization **(+SMASH init)** brings further improvements.

4.4 Comparisons of Different Distillation Settings

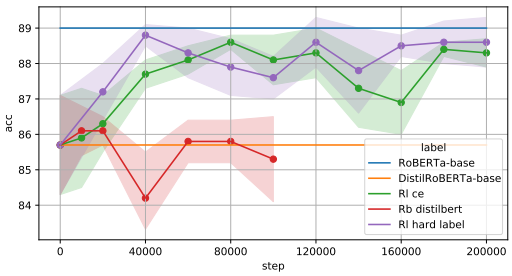
To verify the effectiveness of different distillation settings, we compare the following settings for

	MNLI (acc)	MRPC (f1)	QNLI (acc)	QQP (f1)	SST-2 (acc)	CR (acc)
SMASH	56.8 (1.1)	**74.7 (5.5)**	66.9 (3.8)	**64.4 (2.4)**	88.3 (0.4)	88.0 (0.8)
iPET	66.5 (3.0)	71.0 (10.7)	77.2 (2.8)	59.9 (4.5)	90.9 (0.3)	87.9 (0.7)
+ PT	67.2 (2.7)	69.3 (10.8)	77.9 (3.3)	63.7 (4.4)	90.6 (0.3)	88.5 (0.7)
+ SMASH init	**69.5 (2.9)**	69.9 (10.8)	**79.0 (2.6)**	63.5 (4.3)	**91.2 (0.2)**	**89.2 (0.6)**

Table 3: Results of iPET, and modifications of iPET. **Bold results** indicates the best results.



(a) MNLI-m



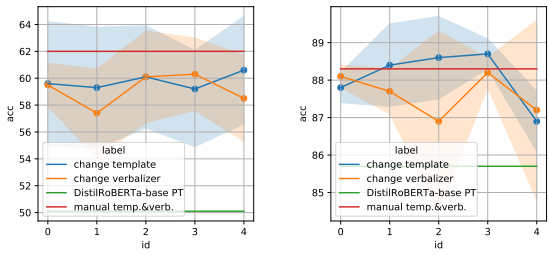
(b) SST-2

Figure 2: Few-shot Performance with different distillation settings. Shaded area indicates standard deviation. We omit standard deviation of RoBERTa-base and DistilRoBERTa-base for simplicity.

sentence-pair tasks: (1) SMASH with objective $\mathcal{L}_{ce}$ (Eq.1) and RoBERTa-large teacher **(Rl ce)**; (2) SMASH with objective $\mathcal{L}_{ce}$ (Eq.1) and RoBERTa-base teacher **(Rb ce)**; (3) SMASH with objective $\mathcal{L}_{ce} + 0.01\mathcal{L}_{hidn}$ (Eq.2) and RoBERTa-base teacher **(Rb ce+hidn)**; (4) SMASH with objective $\mathcal{L}_{ce} + 0.01\mathcal{L}_{attn}$ (Eq.3) and RoBERTa-base teacher **(Rb ce+attn)**; (5) perform further pre-training using raw text from Wikipedia on masked language modeling task and training objective same as DistilBERT (Sanh et al., 2019) and RoBERTa-base teacher **(Rb distilbert)**; and (6) using rule-based label described in Section 3.2 as objective and minimize cross-entropy loss using prompt-based fine-tuning with template same as (1) and verbalizer $v^{dis} = [No, Maybe, Yes]$ **(rule)**. We trained all these models for up to 200k steps and prompt-based fine-tune on MNLI.

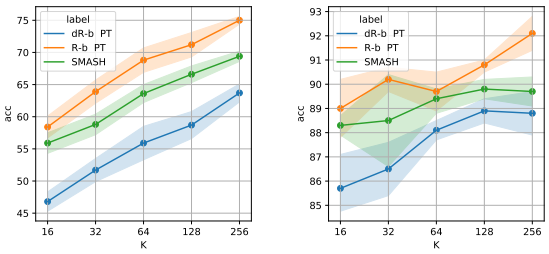
Figure 2(a) shows the results of the comparison. Performance of all settings stabilizes at around 100k steps, and fluctuations after 100k steps are probably due to high variance caused by small training and validation sets of downstream tasks. For settings (2)-(4) with RoBERTa-base teacher, distilling only with $\mathcal{L}_{ce}$ achieves better performance than using other transformer distillation objectives such as $\mathcal{L}_{hidn}$ or $\mathcal{L}_{attn}$ consistently. We suppose that's because during the pre-training stage of DistilRoBERTa-base, its self-attention heads and hidden states are not trained to imitate RoBERTa-base, so its self-attention heads (and hidden states) may capture different information (and lie in different spaces) from RoBERTa-base. Hence adding these objectives actually introduces noise to the distillation process. Setting (5) shows that further pre-training using (Sanh et al., 2019) objective does not make further improvements, as this setting still uses the masked language modeling task and can not narrow down the gap between the pre-training task and downstream tasks. This observation makes sense as DistilRoBERTa-base has been trained with setting (5) for millions of steps during the pre-training stage and already converges. Based on previous observations we find that using $\mathcal{L}_{ce}$ as training objective gets the best results despite its simplicity and compatibility, as it has no requirements of the structure of the teacher model. So in setting (1), we use RoBERTa-large as a stronger teacher with $\mathcal{L}_{ce}$ objective, and it outperforms all other settings.

Setting (6) shows that training using rule-based labels results in inferior performance, indicating that without knowledge distillation, these rule-based labels are not appropriate optimization targets for natural language inference downstream tasks. In summary, choosing an intermediate task similar to downstream tasks and using knowledge distillation are both essential.



(a) SNLI (b) SST-2

Figure 3: Prompt-based fine-tuning using different templates and verbalizers. Shaded area indicates standard deviation. We omit standard deviation of SMASH and DistilRoBERTa-base for simplicity.



(a) MNLI (b) SST-2

Figure 4: Comparisons of different downstream dataset sizes. Shaded area indicates standard deviation.

Figure 2(b) shows comparisons on SST-2 using setting (1), (5), and a new setting (7) using the hard label annotated during the filter process in Sec.3.2 and verbalizer $v^{dis} = [terrible, bad, okay, good, great]$ as supervision **(Rl hard label)**. Note that the verbalizer of SST-2 $v = [terrible, great]$ is a subset of v^{dis}. Though setting (7) seems similar to (6), labels in setting (7) are acquired using a RoBERTa-large instead of rules, so it can be viewed as a kind of "knowledge distillation" which transfers knowledge using hard labels on a 5-class single-sentence classification task. The performance of setting (7) is comparable with setting (1), which shows that using prompt-based fine-tuning as intermediate task also works, as long as the label is given by a teacher model.

4.5 Robustness Under Different Templates and Verbalizers

Previous works (Gao et al., 2020; Gu et al., 2021; Liu et al., 2021) mentioned that the choice of template and verbalizer leads to substantial differences in performance. Note that the superior results in Table 2 are achieved using manual downstream task templates and verbalizers, in this subsection we validate the robustness of SMASH when changing templates or verbalizers, especially when using templates that are different from the intermediate task. Figure 3 presented results changing the manual template to one of the 5 best templates or changing the manual verbalizer to one of the 5 best verbalizers from the generated prompts provided by (Gao et al., 2020). Note that these templates/verbalizers are generated based on RoBERTa-large and may not be the best ones for DistilRoBERTa-base. Results show that SMASH provides consistent improvements even the template of the downstream task is different from the intermediate task.

	MNLI (acc)	**QQP** (f1)	**MRPC** (f1)
T5-small	42.5 (2.0)	54.0 (3.5)	65.2 (5.8)
SMASH	**48.3 (2.0)**	**60.0 (3.4)**	**69.4 (6.3)**
T5-base	52.9 (2.1)	63.8 (0.6)	71.3 (4.3)

Table 4: Few-shot performance of T5 models. **Bold results** indicates the best result achieved using T5-small.

4.6 Impact on Downstream Datasets of Different Sizes

Figure 4 illustrates comparisons of prompt-based fine-tuning DistilRoBERTa-base (dR-b PT), RoBERTa-base (R-b PT) and SMASH on DistilRoBERTa-base when K increases. SMASH still provides improvements when using training set and validation set up to $K = 256$ samples per label, but the improvements reduce as K increases.

4.7 SMASH on Different Language Models

To explore the impact of SMASH on language models other than RoBERTa, We use T5-large (Raffel et al., 2019) as the teacher model and T5-small as the student model. We distill for 200k steps and prompt-based fine-tune on downstream tasks. We compare with two prompt-based fine-tuning baselines: T5-small and T5-base. Table 4 shows that SMASH improves the few-shot performance of T5-small on several sentence-pair tasks.

5 Conclusion

In this paper, we present SMASH, an approach of using knowledge distillation on an unsupervised corpus to improve small language models' few-shot

performance. The principle of this approach is distilling the model using input similar to downstream tasks sampled from unsupervised corpus as an intermediate task to transfer knowledge of solving these tasks and further narrow down the gap between pre-training and prompt-based fine-tuning. We design intermediate tasks for sentence-pair tasks and sentiment classification tasks. We show that our approach results in significant improvements on few-shot datasets, especially for harder tasks like natural language inference. We analyse SMASH on different distillation objectives, and verify its robustness over different templates, verbalizers, and model structures.

Possible future directions of this work include: apply SMASH on more types of downstream tasks, especially those that can not be easily formulated using prompts or are difficult to simulate using unsupervised corpus (e.g., text-to-SQL); or explore intermediate tasks that are more data-efficient.

Limitations

Like many other prompt-based approaches, SMASH requires expert knowledge when designing templates and verbalizers for different groups of tasks, and the performance can be much or less affected by the choices of them. Though in this paper we demonstrated the effectiveness of SMASH on several classification tasks, it is non-trivial to apply SMASH on tasks that are difficult to simulate using unsupervised corpus like machine translation, text-to-SQL, dependency parsing, etc. Due to computational constraints, we only experimented with a 6-layer DistilRoBERTa-base as our small model. Theoretically SMASH is also applicable to larger "small models" such as RoBERTa-base or RoBERTa-large with the usage of a larger teacher, but the effects on these models remain unexplored.

Ethical Statement

The risks and potential harms of pre-trained language models are widely discussed in papers such as (Bender et al., 2021; Bender and Koller, 2020). As models in this work are trained under a few-shot learning setting, these models may have biases due to the lack of diversity of training data. The performance of these models can also be strongly influenced by prompts, and inattentively designed prompts may cause the model to exhibit unexpected behaviors.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive feedback and comments for the improvement of this work. This work was supported by the National Key Research and Development Program of China (No. 2020AAA0106600).

References

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? . *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases.

In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

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

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Chang Liu, Chongyang Tao, Jiazhan Feng, and Dongyan Zhao. 2022. Multi-granularity structural knowledge distillation for language model compression. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1001–1011, Dublin, Ireland. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too.

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*.

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

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Timo Schick and Hinrich Schütze. 2021a. 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. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013a. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Siqi Sun, Zhe Gan, Yuwei Fang, Yu Cheng, Shuohang Wang, and Jingjing Liu. 2020. Contrastive distillation on intermediate representations for language model compression. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 498–508, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019*.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2140–2151, Online. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022. Differentiable prompt makes pre-trained language models better few-shot learners. In *International Conference on Learning Representations*.

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

A Experimental Details

A.1 Hyper-Parameters

Experiments in Section 4.2. During distillation stage, we use batch size = 128, learning rate = 1e-4, max input length = 128 for sentence-pair task and 64 for sentiment classification task. We distill for 200k steps, which takes about 4 days for sentence-pair task and 2 days for sentiment classification task on 2 GTX 1080 Ti GPUs. We sampled 2560k sentences (or sentence pairs) from Wikipedia for sentiment classification (or sentence-pair) task, and training for 200k steps takes exactly 1 epoch. During prompt-based fine-tuning stage, we perform grid search and take learning rates from {1e-5, 2e-5, 5e-5} and batch sizes from {2, 4}. We prompt-based fine-tune the model for up to 1000 steps and save checkpoints every 100 steps. We take the best-performing checkpoint on validation set to get test set results. [5]

For LM-BFF baseline, we use auto templates and manual verbalizers with SBERT (Reimers and Gurevych, 2019) to select demonstrations. For Vanilla PPT baseline, we use the same data as SMASH to pre-train soft-prompts for sentence-pair tasks, and a RoBERTa-base fine-tuned on Yelp-full (Zhang et al., 2015) to filter the pre-training corpus for sentiment classification tasks. For iPET baseline, we made minor modifications on the provided prompts to train PET models on the tasks not experimented in (Schick and Schütze, 2021a). We use 3 generations in iPET, in each generation we train 3 different models for each of the 4 patterns. We follow (Schick and Schütze, 2021a) to set other hyper-parameters.

Experiments in Section 4.4 For settings (1)-(4), we use the same hyper-parameters as Section 4.2. For setting (5), we use learning rate = 1e-5, max

[5] When the validation set is small, the best checkpoint tends to over-fit to the validation set. We observed in some cases the test set performance of the best grid-searched hyper-parameter is even worse than an arbitrary hyper-parameter.

	MNLI-mm (acc)	SNLI (acc)	RTE (acc)	SST-5 (acc)	MR (acc)
SMASH	58.8 (0.9)	59.7 (4.7)	**59.4 (3.2)**	42.4 (2.6)	81.9 (2.2)
iPET	68.5 (3.0)	69.8 (3.8)	52.6 (2.9)	46.6 (0.8)	86.3 (0.6)
+ PT	69.0 (2.9)	70.7 (3.7)	46.4 (3.1)	47.8 (1.1)	86.7 (0.4)
+ SMASH init	**70.6 (2.9)**	**72.1 (3.9)**	46.9 (2.2)	**48.1 (1.0)**	**87.3 (0.3)**

Table 5: Results of iPET and modifications of iPET. We do not report results on STS-B as iPET do not apply to regression tasks. **Bold results** indicates the best results.

Task	Template	Verbalizer
MNLI	<s>x_0?<mask>, x_1.</s>	{contradiction:No, entailment:Yes, neutral:Maybe}
MRPC	<s>x_0<mask>, x_1.</s>	{0:No, 1:Yes}
QQP	<s>x_0<mask>, x_1.</s>	{0:No, 1:Yes}
SNLI	<s>x_0?<mask>, x_1.</s>	{contradiction:No, entailment:Yes, neutral:Maybe}
QNLI	<s>x_0?<mask>, x_1.</s>	{not entailment:No, entailment:Yes}
RTE	<s>x_0?<mask>, x_1.</s>	{not entailment:No, entailment:Yes}
STS-B	<s>x_0<mask>, x_1.</s>	{0:No, 1:Yes}
SST-2	<s>x_0 It was <mask>.</s>	{0:terrible, 1:great}
SST-5	<s>x_0 It was <mask>.</s>	{0:terrible, 1:bad, 2:okay, 3:good, 4:great}
MR	<s>x_0 It was <mask>.</s>	{0:terrible, 1:great}
CR	<s>x_0 It was <mask>.</s>	{0:terrible, 1:great}

Table 6: Manual templates and verbalizers used.

input length = 512, weight of $\mathcal{L}_{ce} = 5$, weight of $\mathcal{L}_{mlm} = 2$ and weight of $\mathcal{L}_{cos} = 1$. We use batch size = 4 and gradient accumulation steps = 32, and consider each gradient update as a training step for a fair comparison with other settings. For setting (6) and (7), we use learning rate = 1e-5.

Experiments in Section 4.6 We prompt-based fine-tune for up to {1000, 1000, 2000, 2000, 4000} steps for K = {16, 32, 64, 128, 256} respectively. Other hpyer-parameters are same as Section 4.2.

Experiments in Section 4.7 When distilling and prompt-based fine-tuning T5, we format sentence-pair tasks as replace corrupted spans task same as the pre-training stage by using template x_1? <extra_id_0>, x_2</s>. We use the output probability of the second generated token, as the first generated token is always <extra_id_0>. Other hyper-parameters are same as Section 4.2.

A.2 Tasks Used in the GLUE Benchmark

We use MNLI (Williams et al., 2018), QQP [6], QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2006), MRPC (Dolan and Brockett, 2005), STS-B (Cer et al., 2017) and SST-2 (Socher et al., 2013a) from the GLUE benchmark in our experiments. We ommited results on WNLI [7] due to its adversarial dev set and CoLA (Warstadt et al., 2018) due to its input may be a non-grammatical sentence and is out of the distribution of the pre-training corpus, as mentioned in (Gao et al., 2020).

A.3 Templates and Verbalizers

Table 6 shows our templates and verbalizers used on RoBERTa models, which is the same as (Gao et al., 2020). For T5 models, we use the same verbalizers and similar templates by removing <s> and replacing <mask> with <extra_id_0>.

[6] https://quoradata.quora.com/

[7] https://cs.nyu.edu/ davise/papers/WinogradSchemas/WS.html