Making Small Language Models Better Few-Shot Learners

Anonymous ACL submission

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 single-sentence 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 training objective. We conduct extensive experiments and show that SMASH can make a 6-layer DistilRoBRETa-base achieve comparable performance on few-shot datasets to a 12-layer RoBERTa-base at a low cost. [1]

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

To tackle with 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. Theoretically, 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

[1] We will make our code and models publicly available after publication.

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

Table 1: Fine-tuning (FT) and prompt-based fine-tuning (PT) performance on DistilRoBERTa-base (dR-b), RoBERTa-base (R-b) and RoBERTa-large (R-l) on 16-shot training and validation dataset. †: results from (Gao et al., 2020). **Bold results** indicates the largest improvement of PT comparing to FT.

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 performing 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 *single-sentence 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. Experiments on the GLUE benchmark (Wang et al., 2019) and several other single-sentence classification 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 single-sentence 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 serious 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). Prompt-based learning has been seen as a promising method for few-shot learning. PET (Schick and Schütze, 2021a,b) focuses on finding prompts under a semi-supervised setting using a small annotated dataset and a large set of unlabeled examples. (Gao et al., 2020) proposed methods to generate prompts and find demonstrations from few-shot datasets. The work most related to us is (Gu et al., 2021), which leveraged unsupervised data to pre-train representations of prompt tokens. Our work differs from previous works in two aspects: (1) we use extremely small models, compared to large models ranging from BERT-base to GPT-3 in previous works; (2) we study methods of training the language model instead of finding better prompts.

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. MiniLM (Wang et al., 2020) proposed self-attention distillation at the pre-training stage, and MiniLmv2 (Wang et al., 2021) proposed multi-head self-attention relation distillation that has no restrictions in terms of the number of teacher's and student's attention heads. TinyBERT (Jiao et al., 2020) proposed transformer distillation method that aligns embedding layer, hidden states, attention matrices, and output logits between teacher model and student model. 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

Fine-tuning. When fine-tuning on downstream tasks, we tokenize input x to a token sequence $\tilde{x}$, the language model $\mathcal{L}$ then maps $\tilde{x}$ to a sequence of hidden vectors $\{h_k \in \mathbb{R}^d\}$. For single-sentence classification tasks, we take $\tilde{x}_{single} =$ <s>x</s> and for sentence-pair tasks, we take $\tilde{x}_{pair} =$ <s>x_1</s></s>x_2</s>. For a downstream task with label space $\mathcal{Y}$, we train a task-specific classification head $softmax(W_o h_{\texttt{<s>}})$ by maximizing the log-likelihood of the correct label, where $W_o \in \mathbb{R}^{|\mathcal{Y}| \times d}$ is a set of randomly initialized parameters and $h_{\texttt{<s>}}$ is the hidden vector of <s>. However, it is challenging to train this new set of parameters from a small amount of annotated data.

Prompt-based Fine-tuning. An alternative approach is *prompt-based fine-tuning*, where input is formulated as a "blank-filling task" with natural language prompts. Take single-sentence classification task as an example, given an input sentence $x \in \mathcal{V}^*$ (where $\mathcal{V}$ is the vocabulary of $\mathcal{L}$, and $\mathcal{V}^*$ denotes a sequence of tokens from $\mathcal{V}$) and its label $y \in \mathcal{Y}$, a template $f : \mathcal{V}^* \rightarrow \mathcal{V}^*$ maps x into a new token sequence $f(x)$ containing the input sentence, several prompt tokens, and at least one <mask> token for $\mathcal{L}$ to predict. Then a verbalizer $v : \mathbb{R}^{|\mathcal{V}|} \rightarrow \mathcal{Y}$ is used to map p, the output distribution of $\mathcal{L}$ 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 $\mathcal{L}$ 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 $\mathcal{L}$ 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 $\mathcal{L}$ and $p(y_l|x) + p(y_u|x) = 1$. We train the model $\mathcal{L}$ to minimize the following objective function using KL divergence:

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

where

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

Knowledge Distillation. Knowledge distillation (KD) was first proposed by (Hinton et al., 2015), which aims at transferring knowledge from the teacher model to the student model. In the simplest form of KD, we train the student model using soft target distributions produced by the teacher model instead of hard labels on a transfer set. Formally, soft target distributions can be expressed as:

$$q_i = \frac{exp(z_i/\mathcal{T})}{\sum_{j \in \mathcal{Y}} exp(z_j/\mathcal{T})}$$

where $\mathcal{Y}$ is the label space, z is the teacher logits and $\mathcal{T}$ is the temperature hyper-parameter that controls the softness of probability distribution and is normally set to 1. The objective function is the cross-entropy loss of the student model's output distribution s and the soft target distribution q:

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

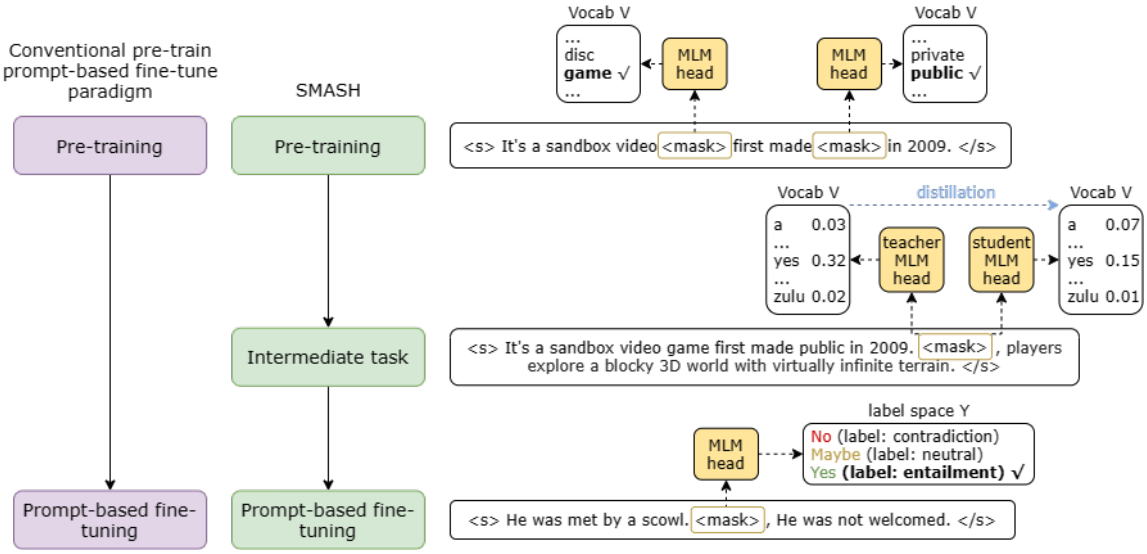


Figure 1: Overview of SMASH on sentence-pair tasks.

3.2 SMASH

(Gu et al., 2021) showed that the performance of downstream tasks can be improved by pre-training the representation of prompt tokens on an unsupervised corpus. Inspired by their work, we propose SMASH to leverage unsupervised corpus to transfer the ability to solve a group of downstream tasks from a pre-trained teacher model $\mathcal{L}_{tea}$ to a pre-trained student model $\mathcal{L}_{stu}$ 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 $\mathcal{L}_{stu}$ from $\mathcal{L}_{tea}$ with t^{dis}, we continue to train $\mathcal{L}_{stu}$ 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* and *single-sentence 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 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.

Single-Sentence Classification Tasks. Single sentence classification tasks such as sentiment classification take 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 (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 $\mathcal{L}_{stu}$ 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 $\mathcal{L}^{stu}$ using the output distribution of $\mathcal{L}^{tea}$ by minimizing

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

the following objective function:

$$\mathcal{L}_{ce} = -\sum_{i \in \mathcal{V}} t_i \times log(s_i) \quad (2)$$

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

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

$$\mathcal{L}_{hidn} = \sum_{m=0}^{M^{stu}} MSE(H_m^{stu}, H_{g(m)}^{tea}) \quad (3)$$

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 student model learns from $g(m)$th layer of teacher model. 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:

$$\mathcal{L}_{attn} = \sum_{m=1}^{M^{stu}} \sum_{h=1}^{H} MSE(A_{mh}^{stu}, A_{g(m)h}^{tea}) \quad (4)$$

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 $\mathcal{L}^{stu}$ and $\mathcal{L}^{tea}$ 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 $\mathcal{L}^{stu}$ and $\mathcal{L}^{tea}$.

After conducting experiments on all above-mentioned distillation objectives described in detail in Section 4.3, 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 GLUE benchmark (Wang et al., 2019) and three more sentiment analysis datasets: SST-5 (Socher et al., 2013), MR (Pang and Lee, 2005) and CR (Hu and Liu, 2004). Our few-shot dataset is 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. We sample datasets for intermediate tasks from Wikipedia [3]. We sampled 1300k sentence pairs for sentence-pair task and 2560k sentences for single-sentence task. Training for 200k steps takes roughly 2 epochs for sentence-pair task and 1 epoch for single-sentence task.

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 use DistilRoBERTa-base as our small language model to study. 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 single-sentence 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 a number of baselines, namely (1) majority in full training set (majority); (2) fine-tuning DistilRoBERTa-base (dR-b finetune); (3) prompt-based zero-shot performance of DistilRoBERTa-base (dR-b zero-shot) and RoBERTa-base (R-b zero-shot); (4) prompt-based fine-tuning DistilRoBERTa-base (dR-b), RoBERTa-base and RoBERTa-large; (5) 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 (dR-b distill); and (6) same as (5) but use back-translation (Sennrich et al., 2015) to augment the training set 10 times larger for KD (dR-b distill bt10).

4.2 Main Results

Table 2 shows our main results. On most tasks SMASH outperforms DistilRoBERTa-base and is comparable with the 2× larger RoBERTa-base model on both prompt-based fine-tuning and zero-shot performance, which shows the effectiveness of SMASH. As the input of CoLA may be a non-grammatical sentence that is out of the language

[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.

	MNLI-m (acc)	MRPC (f1)	QQP (f1)	CoLA (matt.)	SNLI (acc)	QNLI (acc)	RTE (acc)
majority†	32.7	81.2	0.0	0.0	33.8	49.5	52.7
dR-b	46.8 (1.5)	70.0 (6.5)	53.0 (1.5)	3.6 (4.5)	50.1 (5.0)	54.1 (2.2)	54.5 (3.9)
dR-b zero-shot	44.8	73.0	50.1	-1.2	37.5	50.6	51.3
dR-b finetune	38.5 (1.5)	65.4 (15.1)	53.1 (5.5)	**5.5 (3.0)**	39.8 (2.0)	54.6 (2.5)	50.5 (1.4)
dR-b distill	46.6 (1.5)	67.1 (9.9)	53.3 (1.9)	0.9 (4.4)	49.9 (3.7)	53.7 (2.1)	54.9 (2.5)
dR-b distill bt10	44.2 (3.4)	72.3 (2.8)	53.0 (2.5)	1.4 (3.5)	48.4 (2.9)	52.8 (2.8)	54.9 (1.8)
SMASH	**55.9 (1.6)**	**73.9 (6.4)**	**61.3 (3.0)**	4.1 (3.1)	**62.0 (3.6)**	**66.7 (2.9)**	**61.7 (2.9)**
SMASH zero-shot	46.6	76.2	52.5	-0.7	49.8	51.3	53.8
RoBERTa-base	58.4 (1.7)	72.1 (10.9)	63.9 (4.0)	7.2 (6.0)	61.6 (4.2)	61.3 (5.0)	59.6 (6.6)
R-b zero-shot	48.1	53.0	51.6	-3.6	48.6	50.8	53.1
RoBERTa-large†	68.3 (2.3)	74.5 (5.3)	65.5 (5.3)	9.3 (7.3)	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
dR-b	48.2 (3.3)	42.3 (15.4)	85.7 (1.4)	**42.7 (1.5)**	80.4 (1.4)	85.8 (1.8)
dR-b zero-shot	45.7	-5.7	81.8	28.4	78.0	84.0
dR-b 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)
dR-b 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)
dR-b 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)
SMASH	**58.2 (1.6)**	**64.2 (8.0)**	**88.3 (0.4)**	42.4 (2.6)	**81.9 (2.2)**	**88.0 (0.8)**
SMASH zero-shot	47.9	19.7	75.7	31.1	74.0	77.6
RoBERTa-base	60.6 (1.4)	69.1 (3.0)	89.0 (1.2)	44.5 (1.6)	84.3 (1.3)	89.2 (1.4)
R-b zero-shot	49.2	14.5	77.8	32.8	72.3	79.7
RoBERTa-large†	70.5 (1.9)	71.0 (7.0)	92.7 (0.9)	47.4 (2.5)	87.0 (1.2)	90.3 (1.0)

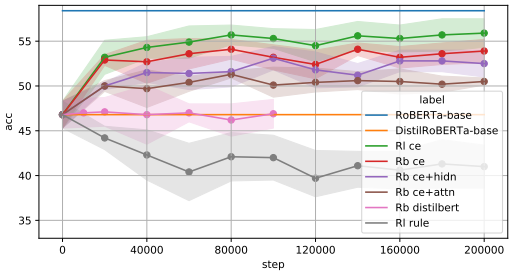
Table 2: Main results. †: results from (Gao et al., 2020). **Bold results** indicates the best result achieved using DistilRoBERTa-base.

model's distribution, discussions in the rest of this paper are based on tasks other than CoLA unless otherwise specified.

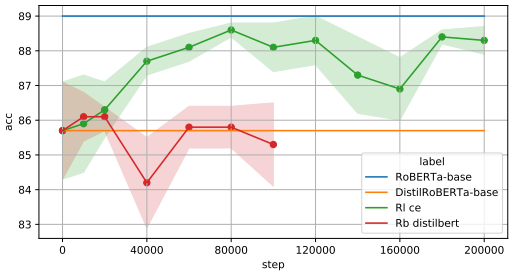
First, for sentence-pair tasks, prompt-based zero-shot performance of DistilRoBERTa-base is worse than or only slightly better than majority baseline, but it can be improved by SMASH; For single-sentence classification tasks, prompt-based zero-shot performance of DistilRoBERTa-base is much better than majority baseline, and SMASH cannot provide further improvements. This indicates that without next sentence prediction (NSP) objective in the pre-training stage, DistilRoBERTa-base does not encode enough knowledge to understand the relationship between two sentences, and SMASH can remedy this problem by training the model with an intermediate task that resembles NSP. As for single-sentence classification tasks, DistilRoBERTa-base encodes its ability to understand a single sentence during pre-training, so SMASH cannot improve its zero-shot performance.

Second, distilling directly from RoBERTa-large using prompt-based method perform worse than prompt-based fine-tuning on a few-shot dataset, 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.

Finally, we observed that on most tasks, prompt-based fine-tuning improvements are significantly greater than zero-shot improvements (e.g., for RTE, prompt-based fine-tuning improvement is $61.7 - 54.5 = 7.2$, 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 one certain downstream task, SMASH works like general distillation that transfers the potential ability to solve a group of similar tasks to student model, which can be exploited later by prompt-based fine-tuning on downstream tasks.



(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.

4.3 Comparisons of Different Distillation Settings

To verify the effectiveness of different distillation settings, we compare the following settings for sentence-pair tasks: (1) SMASH with objective $\mathcal{L}_{ce}$ (Eq.2) and RoBERTa-large teacher (Rl ce); (2) SMASH with objective $\mathcal{L}_{ce}$ and RoBERTa-base teacher (Rb ce); (3) SMASH with objective $\mathcal{L}_{ce}+0.01\mathcal{L}_{hidn}$ (Eq.3) and RoBERTa-base teacher (Rb ce+hidn); (4) SMASH with objective $\mathcal{L}_{ce}+0.01\mathcal{L}_{attn}$ (Eq.4) 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 (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 $v^{dis} = [No, Maybe, Yes]$ (Rl rule). We trained all these models for up to 200k steps and prompt-based fine-tune on MNLI.

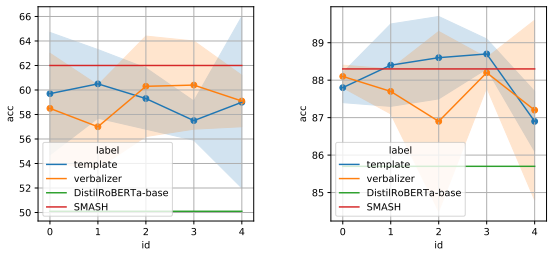
Figure 2(a) shows the results of the comparison. Performance of all settings stabilizes at around 100k steps, and fluctuations after 100k steps are probably due to high variance caused by small training and validation sets of downstream tasks. For settings (2)-(4) with RoBERTa-base teacher, distilling 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.

Figure 2(b) shows comparisons of setting (1) and (5) on SST-2, and its results is consistent with Figure 2(a).

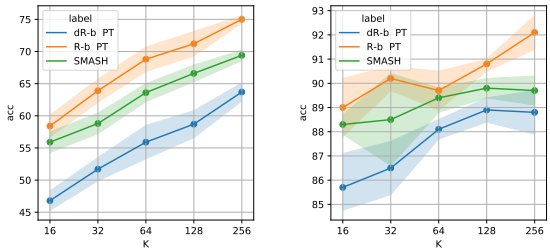
4.4 Robustness Under Different Templates and Verbalizers

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 using



(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-m (b) SST-2

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

5 best templates/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.

4.5 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 reduces as K increases.

4.6 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 3 shows that SMASH improves the few-shot performance of T5-small on several sentence-pair tasks.

	MNLI-m (acc)	QQP (f1)	MRPC (f1)
T5-small	42.5 (2.0)	54.0 (3.5)	65.2 (5.8)
SMASH	**48.4 (1.9)**	**58.0 (4.7)**	**70.0 (5.2)**
T5-base	52.9 (2.1)	63.8 (0.6)	71.3 (4.3)

Table 3: Few-shot performance of T5 models. **Bold results** indicates the best result achieved using T5-small.

5 Conclusion

In this paper, we present SMASH, an approach of using knowledge distillation on an unsupervised corpus to improve small language model's 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 single-sentence classification tasks. We show that our approach results in significant improvements on few-shot datasets, especially for harder tasks like natural language inference. We analyze 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.

References

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*.

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.

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.

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*.

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 Y Ng, and Christopher Potts. 2013. 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.

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.

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 single-sentence task. We distill for 200k steps, which takes about 4 days for sentence-pair task and 2 days for single-sentence task on 2 GTX 1080 Ti GPUs. We sampled 1300k sentence pairs from Wikipedia for sentence-pair task, training for 200k steps takes roughly 2 epochs; and 2560k sentences from Wikipedia for single sentence task, training for 200k steps takes 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]

Experiments in Section 4.3 For settings (1)-(4), we use the same hyper-parameters as Section 4.2. For setting (5), we use learning rate = 1e-5, max 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 to compare with other settings. For setting (6), we use learning rate = 1e-5.

Experiments in Section 4.5 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 4.6 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.

[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.

A.2 Templates and Verbalizers

Table 4 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>`.

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}
CoLA	<s>x_0 This is <mask>.</s>	{0:incorrect, 1:correct}
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 4: Manual templates and verbalizers used.