AdaPrompt: Adaptive Model Training for Prompt-based NLP

Yulong Chen[♠♡], **Yang Liu**[♣], **Li Dong**[♢], **Shuohang Wang**[♣],
Chenguang Zhu[♣], **Michael Zeng**[♣], **Yue Zhang**[♡]
[♠] Zhejiang University
[♡] School of Engineering, Westlake University
[♣] Microsoft Cognitive Services Research Group
[♢] Microsoft Research Asia
yulongchen1010@gmail.com *yaliu10@microsoft.com* *yue.zhang@wias.org.cn*

Abstract

Prompt-based learning, with its capability to tackle zero-shot and few-shot NLP tasks, has gained much attention in community. The main idea is to bridge the gap between NLP downstream tasks and language modeling (LM), by mapping these tasks into natural language prompts, which are then filled by pre-trained language models (PLMs). However, for prompt learning, there are still two salient gaps between NLP tasks and pretraining. First, prompt information is not necessarily sufficiently present during LM pretraining. Second, task-specific data are not necessarily well represented during pretraining. We address these two issues by proposing AdaPrompt, adaptively retrieving external data for continual pretraining of PLMs by making use of both task and prompt characteristics. In addition, we make use of knowledge in Natural Language Inference models for deriving adaptive verbalizers. Experimental results on five NLP benchmarks show that AdaPrompt can improve over standard PLMs in few-shot settings. In addition, in zero-shot settings, our method outperforms standard prompt-based methods by up to 26.35% relative error reduction.

1 Introduction

Prompt-based methods (Brown et al., 2020; Liu et al., 2021; Schick and Schütze, 2021a; Li and Liang, 2021) have received increasing attention in Natural Language Processing (NLP) recently. The main idea is to make the most use of pre-trained language models (PLMs) by adapting an NLP task into a natural language prompt, which can then be filled by PLMs. Take sentiment classification (Socher et al., 2013) for example. Given the sentence "*I love the movie.*", the standard task is to make a binary classification on its sentiment polarity (i.e., `positive` or `negative`). Prompt-based methods first transform the sentence into "*I love the movie. The movie is* ⟨`mask`⟩." (the underlined text is called prompt), and then identify its polarity by checking whether PLMs tends to predict "*good*" or "*bad*" for the ⟨`mask`⟩ token (where the predicted words are then verbalized into class labels). The prompt-based task formulation is close to masked language modeling (Schick and Schütze, 2021a,b), which is the mainstream pretraining strategy, allowing PLMs to provide rich language knowledge seamlessly. Prompt-based methods have been shown particularly useful in zero-shot and few-shot settings (Petroni et al., 2019; Yin et al., 2019; Min et al., 2021), where with limited direct task data, prompt-based inference benefits more from large-scale pretraining than task-oriented finetuning.

Existing methods, however, still suffer from several potential limitations. First, large raw text data used for pretraining do not necessarily contain sufficient patterns that are directly related to task specific prompts (Illustrated in Figure 1). For

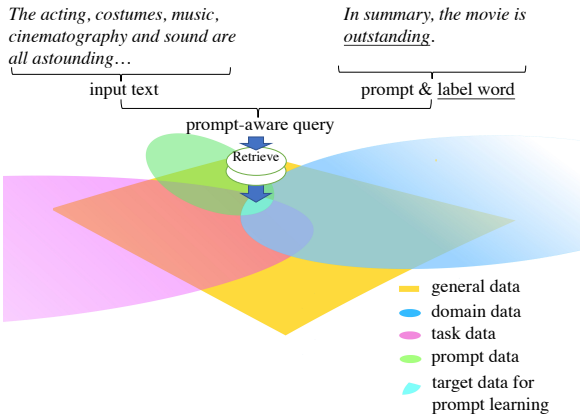

Figure 1: The distributions of data in prompt-based models. Task data, domain data, prompt data, and general data (for LM pretraining) are usually sampled from different distributions while remaining certain overlap (target data for prompt training). We aim to explore data from the overlapping area of these distributions to bridge the gap between PLM and downstream tasks in prompt-based systems.

*Yulong completed this work during his internship at Microsoft.

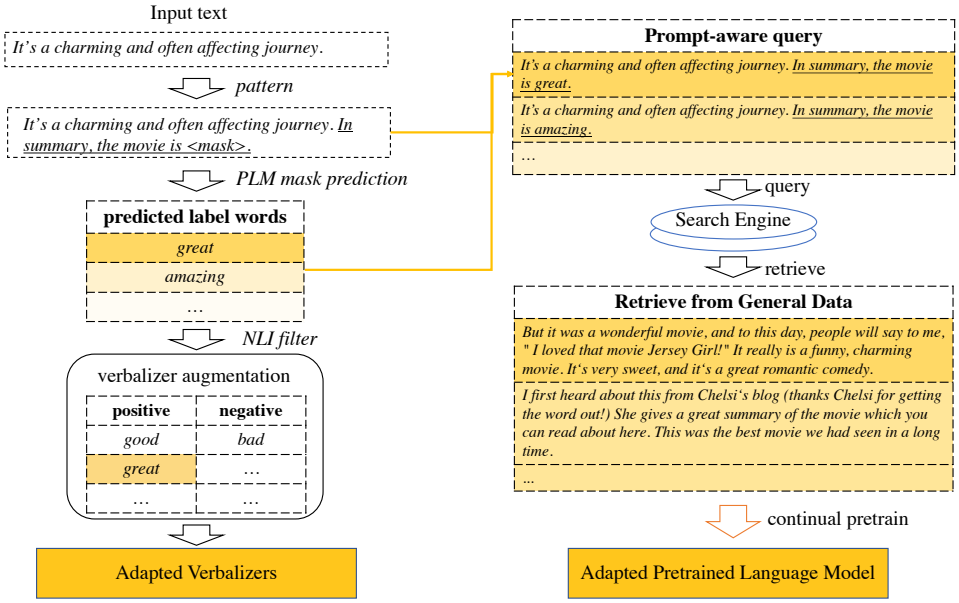

Figure 2: Overall procedure of AdaPrompt.

instance, the prompt for a question classification task is "*Can you tell me the* ⟨mask⟩*: What are the twin cities?*", where ⟨mask⟩ should be a class label word, *e.g., location, person* and etc (the correct label for this sample is *definition*). In contrast, LM pretraining data are typically BOOKCORPUS (Zhu et al., 2015) plus WIKIPEDIA corpus, where such prompts can occur scarcely in the literal or paraphrased form. In addition, texts of specific task, *e.g., movie review*, can differ from PLMs training data. As a result, directly using PLMs to fill such handcrafted prompts across domains can lead to poor performance. Second, to project mask prediction and label space, most existing work (Schick and Schütze, 2021a,b; Cui et al., 2021; Yin et al., 2019) uses a pre-defined verbalizer. However, it requires expert knowledge to build a verbalizer that can thoroughly cover candidate words and a poorly-designed verbalizer limits the accuracy of predictions. These problems become even more serious under zero-shot or very-few-shot settings, where prompt-based models highly rely on the generalization ability of PLMs to new tasks and domains.

We propose AdaPrompt, a framework that can adapt a PLM for the end task considering both the prompts and the verbalizer. We are interested in addressing the above issues under a zero-shot setting, where little or no labeled training data are available for a particular task. This setting is practically useful in that it involves manual labeling only for the prompt itself. The main idea is to adapt a PLM for the end task and prompt by making useful for raw test input data. In particular, as shown in Figure 2, given a test set, we expand each test input by augmenting it with different filled prompt templates (e.g., "*In summary, the movie is great.*"), and then use the resulting sentences (prompt-aware query) to retrieve relevant data from large raw texts. Using this methods, we can obtain large data that contain task and prompt characteristics, and then we adaptively continual pretrain (Gururangan et al., 2020) PLMs on the retrieved data, which can benefit prompt-based methods on NLP downstream tasks.

Given a specific task, different words can be verbalized into the same class labels. For example, a large number of adjectives can express the positive sentiment, and the best-performing candidates depend on the domain, PLM and context. In AdaPrompt, we propose to adaptively augment verbalizers by making use of knowledge from PLMs and Natural Language Inference (NLI) models. Take sentiment analysis for example, given *"good"* and *"bad"* as seed verbalizers, we first let PLMs to predict more candidate words, such as "*amazing*" and "*great*". Then, to identify if these candidates are suitable to verbalizer, we refer to a NLI model to predict whether *"This movie is amazing."* entails the meaning of *"This movie is good."*. Using this method, we are able to automatically expand the verbalizers.

Experiments on five text classification datasets show that AdaPrompt can outperform baseline prompt-based methods by 2.29%-5.79% under

very-few-shot setting and 2.46%-15.00% under zero-shot setting on accuracy. To our knowledge, we are the first to consider to bridge the gap between LM pretraining and NLP downstream tasks for prompt-based NLP.

2 Related work

2.1 Prompt-based NLP for Zero/Few-shot Learning

Although prompt-based methods have been used for machine translation (Brown et al., 2020; Raffel et al., 2020), summarization (Brown et al., 2020; He et al., 2020; Li and Liang, 2021), information extraction (Cui et al., 2021) and fact probing (Petroni et al., 2019), most of existing work focus on text classification (Shin et al., 2020; Gao et al., 2020; Min et al., 2021; Hu et al., 2021). A typical related work is PET (Schick and Schütze, 2021a), where Schick and Schütze (2021a) formally define *pattern-verbalizer pairs* that have been widely adopted by successive works. By using such pairs, Schick and Schütze (2021a,b) develop a series of work to explore the potential of PLMs, including annotating soft labels for raw training data, and data augmentation in iteratively. However, different from PET that assumes the availability of large silver training set for downstream tasks, we focus on zero and very-few-shot settings, where even unannotated task-relevant dataset is also limited (Perez et al., 2021). Therefore, following Hu et al. (2021), we simply focus on standard pattern-verbalizer pairs for text classification.

Prompt engineering (Jiang et al., 2020; Gao et al., 2020) focuses on how to create discrete or continuous prompts that can better induce PLMs to make correct predictions. Discrete prompt engineering works by replacing, deleting, inserting or paraphrasing parts of the prompt. For example, Wallace et al. (2019) use gradient-based search to explore phrases that can better trigger PLMs to make target predictions. Yuan et al. (2021) paraphrase prompts by replacing some phrases from a thesaurus. Such methods require a set of annotated data for validating the prompt quality. Continuous prompt (Lester et al., 2021; Li and Liang, 2021) performs prompting in the embedding space, which can be free of human language and reduce the cost of manually designing hard prompts. Methods with continuous prompts usually require tuning trainable parameters for prompts (Li and Liang, 2021; Shin et al., 2020). Gao et al. (2020) propose LM-BFF, a method that can automatically generate prompts and incorporate demonstrations into context, which are then used for prompt-based tuning using demonstrations. Those methods can efficiently adapt PLMs to downstream tasks, but they highly reply on annotated data for tuning parameters. Different from the above studies, we are interested in narrowing the gap between LM pretraining and NLP tasks for prompting learning in zero or very-few-shot settings.

It has been shown that using different verbalizers can be a key factor for prompt-based methods (Hu et al., 2021; Cui et al., 2021). However, manually exploring these label words is time-consuming and may neglect a lot of potential candidates. A straightforward idea to extend verbalizer by identifying synonyms of pre-defined words. Such a method can cause problems when a word has different meanings. To take the most advantage of knowledge in PLMs, we propose to automatically adapt verbalizers to be more relevant to prompt context. Recently, Hu et al. (2021) propose Knowledgeable Prompt-tuning that uses multiple external knowledge bases, including related words and sentiment dictionaries, to augment verbalizer words for corresponding tasks. Different from them, we focus on exploring knowledge in PLMs themselves. By making use of external NLI models AdaPrompt can select verbalizers automatically without the need of labeled task data, which is useful in zero-shot settings.

2.2 Continual Pretraining for Domain Adaptation

Continual pretraining (Gururangan et al., 2020) has shown benefit of optimizing a PLM to a target domain before further finetuning. It can be categorised into domain adaptive continual pretraining and task adaptive continual pretraining. The difference is that, domain adaptive pre-training (DAPT) uses domain relevant data while task adaptive pretraining (TAPT) uses task-specific data. For example, given the AGNews dataset for the news classification, DAPT uses NEWS dataset (Zellers et al., 2019) to continual pretrain a language model, while TAPT uses the training set of AGNews.

Similar to continual pretraining, many recent methods highlight the merits of relying on language modeling objectives for domain adaptation. Chronopoulou et al. (2019) and Radford et al.

(2018) propose to train task-specific parameters for PLMs by using an auxiliary LM loss on target domain. Works like SciBERT (Beltagy et al., 2019) and DialogLM (Zhong et al., 2021) are PLMs that are continually pretrained on large amounts of scientific and dialogue corpora. UDALM (Karouzos et al., 2021) first trains PLMs by masked language modeling (MLM) on the target domain and then trains a target classifier with source domain labeled data, while keeping the MLM objective on unlabeled target domain data.

Data selection is a common practice in domain adaption for NLP models (Moore and Lewis, 2010; Ruder and Plank, 2017; Van Der Wees et al., 2017). It has been used in machine translation (Van Der Wees et al., 2017; Wang et al., 2018), parsing (Plank and Van Noord, 2011; Ruder and Plank, 2017), named entity recognition (Dai et al., 2019) and sentiment analysis (Ruder et al., 2017). The main idea is to have a selection model that can distinguish in-domain and out-of-domain data. The selection model can be a supervised classifier (Aharoni and Goldberg, 2020), similarity-based metric (Plank and Van Noord, 2011) or language model perplexity (Moore and Lewis, 2010). Very recently, Yao et al. (2021) propose to retrieve a small set of training data from general corpora with labeled task data as queries, finding that using language modeling objective on this data as an auxiliary loss can help train task-specific NLP models without pretraining.

3 Method

Our method is based on prompt-based text classification methods (Section 3.1). The overall procedure of AdaPrompt is shown in Figure 2, which can be divided into two parts: PLM domain adaptation (Section 3.2) and verbalizer adaptation (Section 3.4). In Section 3.3, we introduce a method that adapts both PLMs and verbalizers in an iterative way for continual improvements.

3.1 Prompt-based Text Classification

Given an input text, $\mathbf{x} = (x_0, x_1, ..., x_n)$, we consider various tasks to classify the sentence into a class label $l \in \mathcal{L}$. As mentioned in Section 1, the standard prompt-based method reformulates the input into a cloze-style question and identifies its label by checking PLMs' predictions. Table 1 shows the prompt templates and verbalizer patterns for the SST-2 (Socher et al., 2013), Yelp (Zhang et al., 2015), AGNews (Zhang et al., 2015), TREC (Voorhees and Tice, 2000) and DBPedia (Lehmann et al., 2015) datasets, which cover sentiment classification, topic classification and question classification tasks. Formally, let $\mathcal{M}$ be a language model pre-trained on large-scale general data, and $\langle \texttt{mask} \rangle$ be the mask token. The prompt-based method first defines a *pattern* function, $Prompt$, that converts $\mathbf{x}$ into a cloze-style question containing $\langle \texttt{mask} \rangle$. Then, it defines a *verbalizer* function v, which maps a small set of pre-defined verbalizer words ($\mathcal{Y}$) predicted at the position of <mask> into class labels, i.e., $v : \mathcal{Y} \mapsto \mathcal{L}$.

Take sentiment classification for movie review for instance. The task is to classify the sentiment polarity, where $\mathcal{L} = \{\texttt{positive}, \texttt{negative}\}$. For an input $\mathbf{x}$, we choose the pattern:

$Prompt$ ="$\mathbf{x}$. *In summary, the movie is* $\langle \texttt{mask} \rangle$."

Then we define a verbalizer that maps $\mathcal{Y} =$ {*"good", "bad"*} into $\mathcal{L}$:

$v($*"good"*$) = \texttt{positive}$;
$v($*"bad"*$) = \texttt{negative}$

Given an example:

$\mathbf{x} =$ *"It's a charming and often affecting journey."*,

we can convert the input into a cloze-style question using $Prompt$:

$Prompt(\mathbf{x}) =$ *"It's a charming and often affecting journey. In summary, the movie is* $\langle \texttt{mask} \rangle$*."*

Using such *pattern-verbalizer* pairs, we ask $\mathcal{M}$ to directly give scores s for each label $l \in \mathcal{L}$ as:

$$s(l|\mathbf{x}) = Pr[\texttt{<mask>} = y | Prompt(\mathbf{x}), \mathcal{M}] \quad (1)$$

where $l = v(y)$. The predicted label is:

$$\hat{l} = \arg\max_{l \in \mathcal{L}} s(l|\mathbf{x}) \quad (2)$$

3.2 Adaptively Retrieve Data for Continual Pretraining

As discussed in the Section 1, the lack of domain adaptation can be a potential challenge for prompt-based NLP models, especially under zero-shot and very-few-shot settings. To tackle this problem, we propose to build a continual pretraining dataset by retrieving from general corpora, with unannoated test texts, designed prompts and label words as queries. In this way, we can obtain task-relevant data for any tasks or domains, using only test input. Meanwhile, prompt and verbalzier information is

Dataset	Class	Classification	Prompt Template	Verbalizer
SST-2	2	sentiment	`Text` *In summary, this movie is* ⟨*mask*⟩.	*"good"*, *"bad"*
Yelp	2	sentiment	`Text` *In summary, this restaurant is* ⟨*mask*⟩.	*"good"*, *"bad"*
AGNews	4	news topic	*[Category:* ⟨*mask*⟩*]* `Title , Body`	*"Sport"*, *"Tech"*, *"Business"*, *"World"*
TREC	6	question type	*Can you tell me the* ⟨*mask*⟩ `Text`	*"explanation"*, *"description"*, *"person"*, *"location"*, *"number"*, *"entity"*
DBPedia	14	ontology	*Introduction to the* ⟨*mask*⟩ `Text`	*"company"*, *"school"*, *"artist"*, *"film"*, *"book"*, *"plan"*, *"building"*, *"village"*, *"animal"*, *"sport"*, *"album"*, *"officer"*, *"scenery"*, *"transportation"*

Table 1: Datasets used in this paper with seed prompts and verbalizer words. Each seed verbalizer word corresponds to a class label.

also considered during the retrieval process, leading to a more comprehensive dataset for prompt-aware continual pretraining.

Formally, given a retrieval query q, a retrieval engine $\mathcal{E}_D$ indexed on a large general dataset $\mathcal{D}$ can return a set of similar text $d_q = \mathcal{E}_D(q)$. In order to obtain prompt-aware data that can not only adapt PLMs to target domains but also make PLMs more sensitive to prompts, we include both task and prompt characteristics when building queries. As shown in Figure 2, for an unannotated input text $\mathbf{x}$ in text data, we first convert it into $Prompt(\mathbf{x})$, and obtain a set of predicted label words using a PLM $\mathcal{M}$:

$$\mathcal{O} = \mathcal{M}(Prompt(\mathbf{x})) \tag{3}$$

where $\mathcal{O} = \{o_1, o_2, ..., o_{|\mathcal{O}|}\}$ are the top-$|\mathcal{O}|$ predictions. We replace the mask token in $P(\mathbf{x})$ with o_i, to form a list Q of queries. For example:

$$Q = \{q_1, ..., q_{|\mathcal{O}|}\} \tag{4}$$

$$q_i = \text{"}\mathbf{x}\text{. } \textit{In summary, the movie is } o_i\text{."} \tag{5}$$

With this set of prompt-based queries, we retrieve prompt-aware data $\mathcal{D}_p$, which is a small subset of the general data. In this work, we use ElasticSearch[1] as the search engine since it is fast and training-free. ElasticSearch is a distributed search engine that can return a list of top-k texts that match the query. We consider TF-IDF as the similarity metric and take BOOKCORPUS (Zhu et al., 2015) plus WIKIPEDIA, CCNEWS (Nagel, 2016), STORIES (Trinh and Le, 2018), and OPENWEBTEXT (Gokaslan and Cohen, 2019) as the set of raw text data to query from. We first index general corpora on sentence level, and use prompt-aware query to retrieve relevant data. To enrich the context information, we also extract the previous and following sentences of the one that is invoked by prompt-aware queries, and concatenate them in their natural order as a line of data. As shown in Figure 2, one test input can lead to multiple prompt-aware queries because the masked token in the prompt can be replaced by the $|\mathcal{O}|$ predictions. In addition, one query can lead to multiple outputs returned by ElasticSearch engine as demanded.

[1]https://www.elastic.co

Algorithm 1 Verbalizer Adaptation

Input: prompt P, seed verbalizer words $y \in \mathcal{Y}_l$, candidate words $c \in \mathcal{C}$ and an NLI system $\mathcal{N}$
for c **in** $\mathcal{C}$ **do**
  if $\mathcal{N}(f(P, y), fill(P, c)) = Entail$
    or $\mathcal{N}(fill(P, c), f(P, y)) = Entail$ **then**
    add c in $\mathcal{Y}_l$
  end if
end for
Return $\mathcal{Y}_l$

We continue to pretrain the PLM $\mathcal{M}$ on $\mathcal{D}_p$ with masked language modeling loss and obtain an adapted PLM $\mathcal{M}_{\mathcal{D}_p}$. $\mathcal{M}_{\mathcal{D}_p}$ now contains richer knowledge of both the target domain and the prompts. It can be used to replace $\mathcal{M}$ in Eq. 1 for zero-shot text classification.

3.3 Iterative Adaptation

After obtaining $\mathcal{M}_{\mathcal{D}_p}$ for a specific task (Section 3.2), we can iterate the process by replacing $\mathcal{M}$ with $\mathcal{M}_{\mathcal{D}_p}$ in Eq. 3, and obtain an iterative set of predicted words and a list of queries marked as $\mathcal{O}'$ and Q'. Given that $\mathcal{O}'$ contains more in-domain knowledge, we can retrieve higher quality pretraining data with more task relevant information, using Q' to query the $\mathcal{E}_D$. In this way, we obtain a new version of $\mathcal{D}'_p$, and a new continual pretrained PLM $\mathcal{M}'_{\mathcal{D}_p}$, which can also be used to make zero-shot predictions using Eq. 1. In this work, we conduct this procedure twice.

Dataset	Testset	Top-$\|\mathcal{O}\|$	E_{space}	Resulting Data
TREC	500	20	100	60k
SST-2	872	20	100	205k
AGNews	7,600	10	50	414k
YELP	38,000	1	50	267k
DBPedia	70,000	1	50	1,301k

Table 2: Data statistics for datasets. E_{space} corresponds to the ElasticSearch space. Note that the resulting data size is calculated after data de-duplication.

3.4 Adaptive Verbalizer Augmentation

As described in Section 3.1, the regular prompt-based method defines the *verbalizer* that maps predicted label word into task classes, such as "*good*" for `positive` and "*bad*" for `negative`. In particular, as shown in Figure 2, for an unannotated test input, we infer top-$|\mathcal{O}|$ label words at mask token position. We then collect these labels over testset, filter by word frequency, and obtain a set of words $\mathcal{C}$ than can be used as candidates for verbalizer augmentation. We find that Natural Language Entailment models are good protocols to further select these candidates. It can explore entailment relation between candidate words and seed verbalizer words under certain context.

Specifically, given a seed verbalizer word $y_l \in \mathcal{Y}_l$ for label l , and a candidate word $c \in \mathcal{C}$, we compare whether a prompt filled by y_l is entailed with the prompt filled by c. The pseudo code is shown in Algorithm 1, where $\mathcal{N}$ is an NLI system, and $\mathcal{N}(s_1, s_2) = Entail$ means sentence s_1 entails sentence s_2. $fill(P, w)$ is a function that fills the mask token with a word w in the prompt P. If entailment relation holds for this pair, we then add c add in to $\mathcal{Y}_l$, which can be used as an augmented verbalizer.

After obtaining the augmented set of verbalizer words, Eq. 1 can be rewritten as:

$$s(l|\mathbf{x}) = \frac{1}{|\mathcal{Y}_l|} \sum_{y \in \mathcal{Y}_l} Pr[\texttt{<mask>} = y | Prompt(\mathbf{x}), \mathcal{M}] \quad (6)$$

and we can still use Eq. 2 for prediction.

3.5 Patterns and Verbalizers

To evaluate our methods, we conduct experiments on five benchmarks: SST-2 (Socher et al., 2013), Yelp (Zhang et al., 2015), AGNews (Zhang et al., 2015), TREC (Voorhees and Tice, 2000) and DBPedia (Lehmann et al., 2015) datasets. Table 1 shows prompt templates and seed verbalizer words that we use for each dataset. For AGNews and YELP, we adapt patterns and verbalizers from PET (Schick and Schütze, 2021a) since it is the basic prompt-based method that has been mostly widely used.

AGNews is a text classification dataset in the domain of News. Given a headline and a main text body, the model is require to classify the news into one of the classes: (1) World, (2) Sports, (3) Business or (4) Science/Tech.

YELP is a sentiment analysis dataset. Given a restaurant review, the task is to predict whether the review is positive or negative.

SST-2 is a sentiment analysis dataset similar to YELP but its domain is movie reviews. Thus, we use the same seed prompt and verbalizer words as for YELP, but change "*restaurant*" in prompt template to "*movie*".

DBPedia 2014 is an ontology classification dataset, extracted from DBPedia 2014 with 14 non-overlap classes, such as Educational Institution and Office Holder. We define two patterns for this task:

$P1(\mathbf{x})$ = "*Description to the* $\langle \texttt{mask} \rangle$ ***x***"
$P2(\mathbf{x})$ = "*Introduction to the* $\langle \texttt{mask} \rangle$ ***x***"

and we use $P2$ as the seed pattern.

TREC-10 is a question classification dataset. Given a question, the task is identify the objective that the question asks, and classify it into one of six classes, such as a definition question or a numeric question. We define two patterns for this task:

$P1(\mathbf{x})$ = "*Tell me the* $\langle \texttt{mask} \rangle$ ***x***"
$P2(\mathbf{x})$ = "*Can you tell me the* $\langle \texttt{mask} \rangle$*:* ***x***"

and $P2$ as the seed prompt.

4 Experiments

4.1 Settings

Since AdaPrompt is completely free of annotated data and it aims to enhance the domain knowledge of PLMs, it can be easily applied to existing prompt-based methods that use discrete prompts. In this work, we take ROBERTA-large (Liu et al., 2019) as our foundation PLM and adopt pattern-verbalizer pairs from (Schick and Schütze, 2021a) (Section 3.1) as the baseline setting, because it is the basic prompt-based method that has been mostly widely used and can be easily extended to other methods, such as AutoPrompt (Shin et al., 2020).

Since we consider zero-shot settings, the choice of hyper-parameters is based previous work (Gao et al., 2020; Schick and Schütze, 2021a,b) and empirical consideration. For all continual pretraining, we use a learning rate of $1e^{-5}$, batch size of 12 per

Models	SST-2	Yelp	AGNEWS	DBPedia	TREC	Avg.
Channel	$77.10 \pm NA(NA)$	$--$	$61.80 \pm NA(NA)$	$51.40 \pm NA(NA)$	$30.50 \pm NA(NA)$	$--$
GPT-3	$75.80 \pm 0.00(75.80)$	$--$	$73.90 \pm 0.00(73.90)$	$59.70 \pm 0.00(59.70)$	$57.40 \pm 0.00(57.40)$	$--$
ROBERTA	$64.56 \pm 16.77(88.99)$	$72.63 \pm 16.34(87.97)$	$69.52 \pm 6.96(78.76)$	$56.32 \pm 0.49(56.67)$	$45.50 \pm 0.14(45.60)$	61.71
Ada	$75.92 \pm 17.36(91.28)$	$75.09 \pm 17.57(89.25)$	$76.55 \pm 7.28(84.95)$	$70.95 \pm 8.80(77.17)$	$60.50 \pm 3.54(63.00)$	71.80
iAda	$77.18 \pm 17.96(91.74)$	$75.81 \pm 18.05(90.41)$	$74.28 \pm 9.00(83.37)$	$73.01 \pm 6.70(77.92)$	$61.10 \pm 1.27(62.00)$	72.28

Table 3: Zero-shot results. We report average accuracy and standard deviation of different patterns here. Results of the best patterns are shown in brackets. The Avg. reports the overall averaged results. Ada and iAda denote to AdaPrompt and iterative AdaPrompt based on ROBERTA-large, respectively. Channel (Min et al., 2021) is based on GPT-2 large. GPT-3 results are reported by Zhao et al. (2021), using GPT-3 ($175B$). NA denotes to that results are not reported. For GPT-3 (Zhao et al., 2021), they only use a fixed prompt format.

GPU. We train each model for 3 epochs and use the 500 training step model for evaluation.

For few-shot settings, we evaluate our models using regular prompt tuning and sequence classifier with 10, 50, 100 training samples. Given that few-shot learning can be influenced greatly by training samples, we follow previous work (Hu et al., 2021; Schick and Schütze, 2021a; Gao et al., 2020) and repeat the training and evaluation for 5 times using different seed, and report the averaged scores for each datasets.

Prompt-Aware Data Retrieval Table 2 presents the basic statistics of datasets used in this paper. TREC and SST datasets contain smaller testsets, while YELP and DBPedia contain much larger testsets. To balance the retrieved data size, we set different top-$|\mathcal{O}|$ for predicted words and ElasticSearch space (E_{space}) for different datasets based on our practical experience. In other words, given one test input, we have $|\mathcal{O}| \times E_{space}$ data. After de-duplication, the resulting retrieved data sizes are shown in Table 2.

We also retrieve data using only input sentences **x** instead of **q**, and use those data to continue pre-train a PLM, referred as *in-domain* adaptation. Since we cannot augment queries by using top-$|\mathcal{O}|$ predicted words, we enhance the Elasticsearch space by top-$|\mathcal{O}|$ times, and obtain retrieved data at similar sizes of resulting data as reported in Table 2 for a fair comparison.

Verbalizer Augmentation To obtain possible verbalizers that can better represent classes, we first obtain top-N predicted words given a test sample ($N = 20$ for SST-2 and TREC, $N = 10$ for AGNews and $N = 5$ for YELP and DBPedia, considering their testset sizes). We set the number of candidate words $|\mathcal{C}| = 20 \times |\mathcal{L}|$, where $|\mathcal{L}|$ is number of classes. We use a ROBERTA-LARGE model finetuned on MNLI (Williams et al., 2018), a Multi-Genre Natural Language Inference dataset, as the entailment model for identifying potential verbalizer words for augmentation. For each candidate word, we apply this NLI model to infer its probability of having an entailment relationship with any seed verbalizer word. Candidate with probability higher than a threshold k is then added to the augmented verbalizer. We set $k = 0.4$ by experiments.

For comparison, we also use Word2Vec (Mikolov et al., 2013) to obtain word vectors and explore potential verbalizer words by their similarity with the seed verbalizer words.

4.2 Results

We conduct experiments under zero-shot and few-shot settings. Under the zero-shot setting, we directly use PLMs to infer label words at masked positions. Under the few-shot setting, we follow Schick and Schütze (2021a) and Hu et al. (2021) and use prompt-tuning, which directly finetunes a LM given a small set of annotated data and prompts.

4.2.1 Zero-shot Setting

In zero-shot setting, we compare AdaPrompt with prompt-based methods using ROBERTA (Schick and Schütze, 2021a), GPT-2 (Gao et al., 2020) and GPT-3 (Zhao et al., 2021), respectively. The Channel refers to noisy channel model, which is based on GPT-2 and uses channel models to enhance the LM's understanding of prompts. GPT-3 model is a huge PLM that contains $175B$ parameters. Table 3 presents the results under zero-shot setting. Following previous work (Schick and Schütze, 2021a,b), we report average accuracy, standard deviation and accuracy of the best pattern over different patterns.

First, compared with our foundation model, ROBERTA-large, we see that AdaPrompt consistently outperforms regular prompt-based methods on all datasets with better average performance and best pattern performance, bringing a $2.46 \sim 14.63$ improvement. It is noticeable that with AdaPrompt,

$\|T\|$	Models	SST-2	Yelp	AGNEWS	DBPedia	TREC	Avg.
10	ROBERTA	84.97 ± 9.88	86.84 ± 16.08	78.42 ± 6.23	86.78 ± 1.10	45.56 ± 9.55	76.51
	AdaPrompt	90.42 ± 1.63	89.13 ± 13.30	84.21 ± 2.00	91.68 ± 1.84	57.56 ± 7.85	82.60
50	ROBERTA	92.56 ± 1.31	95.87 ± 0.57	85.50 ± 1.36	94.72 ± 0.49	73.88 ± 3.13	88.51
	AdaPrompt	92.75 ± 1.03	95.74 ± 0.89	86.29 ± 0.80	94.59 ± 0.71	78.42 ± 6.17	89.56
100	ROBERTA	92.40 ± 1.04	95.89 ± 0.68	87.29 ± 1.31	95.59 ± 0.52	86.30 ± 2.14	91.49
	AdaPrompt	92.75 ± 0.68	95.93 ± 0.95	87.98 ± 0.65	95.60 ± 0.51	87.58 ± 1.38	91.97

Table 4: Average accuracy and standard deviation on SST-2, YELP, AGNews, DBPedia and TREC under few-shot settings. $|T|$ is the training set size. Each experiment is repeated 5 times using different seeds.

Model	Size	SST-2
Albert	$17M$	$54.67 \pm 3.30(58.94)$
Albert+AdaPrompt	$17M$	$58.51 \pm 5.79(63.99)$
Bert	$340M$	$58.03 \pm 6.18(63.53)$
Bert+AdaPrompt	$340M$	$68.89 \pm 16.11(85.67)$
ROBERTA	$355M$	$64.56 \pm 16.77(88.99)$
ROBERTA+AdaPrompt	$355M$	$77.18 \pm 17.96(91.74)$

Table 5: We report average accuracy and standard deviation here. Results of the best pattern are shown in the bracket.

we can outperform GPT-3 in zero-shot setting, which is a huge model pretrained on a gigantic corpus. This confirms the effectiveness of AdaPrompt in domain adaptation. We also observe that iterative AdaPrompt can further bring improvements on most datasets (SST-2, YELP and DBPedia). This directly demonstrates that PLMs continual pretrained on the retrieved data can be more adaptive to downstream tasks, and thus generate more task relevant label words, which can serve as a source to find better texts. Performance of iterative AdaPrompt decreases on AGNEWS, we believe this is because this news dataset is similar with general data used for pretraining ROBERTA, and thus continual pretraining on such retrieved data can be less useful. Finally, we see that AdaPrompt improves over 10.09 accuracy of the overall performance.

4.2.2 Few-shot Setting

Table 4 reports the experimental results under few shot setting. Each experiments is repeated 5 times using different seed, and we report the average accuracy and standard deviation here. To explore whether AdaPrompt can consistently bring improvement to ROBERTA, we conduct experiments using 10, 50, 100 samples, respectively.

We can see that compared with ROBERTA-large baseline, under few-shot setting, AdaPrompt can still improve model performance. Although the relative improvement decreases as the size of training set improves, we can see that AdaPrompt outperforms ROBERTA over tasks in all few-shot settings. In particular, AdaPrompt outperforms standard ROBERTA models by $2.29 \sim 5.79\%$ in 10-shot setting, showing that it is useful in very-few-shot setting.

4.2.3 Ablation Study

AdaPrompt with different PLMs In this ablation study, we apply AdaPrompt with different PLMs (Bert-large, Albert-large and ROBERTA-large). We report experimental results on the SST-2 dataset. Table 5 shows the results. Although the performance of different models varies, we still observe that AdaPrompt can consistently bring huge improvement over all models. We also find that model performance increases with model size. AdaPrompt using ROBERTA-large outperforms other models overall performance by a large margin ($8.29 \sim 18.67$) and achieves 91.74 accuracy with the best pattern.

Prompt-Aware Continual Pretrain and Verbalizer Augmentation To study the effectiveness of continual pretraining on prompt-aware data and verbalier augmentation, we conduct ablation experiments with the combination of with or without continual pretraining (`CP`) or verbalizer augmentation (`va`). As shown in Table 6, We can see that compared with foundation model (`-CP-va`, 61.71 acc. on average), continual pretraining and verbalizer augmentation can both bring improvement to model performance (5.31 and 5.89 acc. on average, respectively), and the model has the best results two methods are combined together (AdaPrompt), suggesting these two methods can benefit each other.

In addition, we consider the influence of query text on retrieving textual data for continual pretraining. Table 6 shows the results using only test input versus results using the prompts and verbalizer. With different verbalizers, our method generates multiple queries from a single test input. However, for both methods compared, the retrieved text from ElasticSearch are confined to k sequences (Section 3.2. From the table we can see that on all datasets, using prompt-augmented

Models	SST-2	Yelp	AGNEWS	DBPedia	TREC	Avg.
AdaPrompt	75.92 ± 17.36	75.09 ± 17.57	76.55 ± 07.28	70.95 ± 08.80	60.50 ± 03.54	71.80
-va	71.07 ± 13.58	71.04 ± 15.57	72.16 ± 05.78	65.90 ± 02.71	45.40 ± 01.13	65.11
-CP	72.16 ± 16.35	75.72 ± 17.79	75.70 ± 07.88	50.95 ± 00.09	58.70 ± 03.25	66.65
-PR	71.22 ± 15.55	74.85 ± 17.51	75.12 ± 05.71	70.40 ± 07.48	58.60 ± 00.57	70.04
-CP-va	64.56 ± 16.77	72.63 ± 16.34	69.52 ± 06.96	56.32 ± 00.49	45.50 ± 00.14	61.71

Table 6: Experimental results of ablation study. - means "without" here. va: verbalizer augmentation, CP: Continual Pretraining, PR: Prompt-aware Retrieval. Note that -PR means we do not use prompt-aware retrieval, but simply use test input data for retrieval and continual pretraining.

SST-2				
E_{space}	1	10	50	100
Size	3k	23k	98k	205k
Accuracy	73.54 ±16.77	75.06 ±17.34	75.95 ±17.73	75.92 ±17.36
DBPedia				
E_{space}	1	5	25	50
Size	58k	235k	708k	1,301k
Accuracy	70.64 ±9.66	71.39 ±10.78	74.13 ±7.51	70.95 ±8.80

Table 7: Analysis on retrieved data size. Data sizes are calculated after de-duplication.

queries (AdaPrompt) give substantially stronger results. Take SST-2 for example, the accuracy is 71.22 (SST-2 -PR) given only test input queries, but 75.92 with prompt-augmented queries, with a 4.7 absolute improvement. This shows that continual pretraining using prompt-aware data is highly beneficial to zero-shot prompt-based NLP.

4.3 Analysis

4.3.1 Size of Retrieved Data

As stated, Elasticsearch returns data in the order of matching scores. Using a smaller size, the retrieved data are more textual related to the query, while using a larger size, the retrieved data can contain noisy. To compare the effects of different sizes of retrieved data for continual pretraining, We retrieve 1, 10, 50 and 100 texts using each query (i.e., E_{space}) for the SST-2 and 1, 5, 25, 50 for DBPedia, respectively As shown in Table 7, we see that accuracy rises in the beginning when retrieval size increases. But as the retrieval size grows bigger, the accuracy starts to decrease slightly. This can be explained by that the lower-ranked retrieved data have a lower relevance to the target task, which introduces more noise in continual pretraining. We used fixed E_{space} sizes for our experiments in zero-shot settings (Section 4.1), due to lack of a validation set. In few-shot settings, in practice, the size can be considered as a hyperparameter and tuned over validation data.

4.3.2 The Effect of Verbalizer Strategies

Table 8 compares the model performance when using different verbalizer augmentation strategies, namely using NLI model and word similarity (Section 4.1). Additional, we adopt a verbalizer augmentation method using knowledge base (KB) (Hu et al., 2021) [2]. To set a fair comparison, we limit the verbalizer word set for each label within 5. We report average accuracy and standard deviation here.

Results show that, compared with using word similarity to select candidate words and directly using KBs to augment verbalizer words, using NLI to augment verbalizer words gives more better performance on most tasks, and is more stable. We also find that using KBs to augment verbalizer words gives better performance on the DBPedia tasks, but much worse performance on the TREC task. This can be because TREC is less close to topic classification (Min et al., 2021), and directly using most related words can add noisy. This also suggests that more sophisticated strategy that cares of task and prompt information can be useful, which we leave for future work.

4.3.3 Generalization Capability

For experiments in section 4.2.1 and section 4.2.2, we use task testset as the sources to build queries for retrieving pretraining data. However, in a more general setting, we want to learn whether AdaPrompt can also generalize to new samples that are different from query data. To this end, we take the original training set of SST-2 and DBPedia as an *unseen* test set, evaluating models trained using data retrieved from the original test data. As shown in Table 9, we can see that AdaPrompt achieves 73.05 and 70.97 accuracy on SST-2 and DBPedia, respectively. Compared with performance on testset (Table 3), we find that although the performance

[2]For sentiment analysis tasks, we take sentiment words shown in (Hu et al., 2021), which are adopted from https://www.enchantedlearning.com/wordlist/; for other tasks, we use most related words: https://relatedwords.org/

Dataset	SST-2	YELP	AGNEWS	DBPedia	TREC	Avg.
va_w	74.91 ± 11.71	75.39 ± 17.47	69.07 ± 06.70	55.32 ± 11.33	60.60 ± 03.39	67.06
va_m	75.92 ± 17.36	75.09 ± 17.57	76.55 ± 07.28	70.95 ± 08.80	60.50 ± 03.54	71.80
va_k	69.07 ± 15.80	74.64 ± 17.55	60.15 ± 07.79	74.85 ± 17.50	24.00 ± 00.57	60.54

Table 8: Model performance of AdaPrompt using different verbalizer augmentation strategies. va_w: using word2vec similarity. va_m: using ROBERTA trained on MNLI. va_k: using most related words/sentiment dictionary. Avg. refers to overall averaged results.

Model	SST-2	DBPedia
ROBERTA	64.82 ± 11.62	56.49 ± 00.41
AdaPrompt	73.05 ± 13.08	70.97 ± 08.87

Table 9: Model performance when tested on *full training set*. We report averaged accuracy and standard deviation here.

of AdaPrompt sightly decreases when evaluated on full SST-2 training set, it can still outperform ROBERTA by a large margin (+8.23). It demonstrates that although AdaPrompt uses a small part of data to retrieve related text for continual pretraining, it can bring improvements over the full dataset, and shows good generalization ability.

5 Conclusion

We investigated AdaPrompt, a zero-shot prompt-based method for NLP that makes use of test input data and prompts for adaptive continual pretraining and verbalizer selection. Results on five classification datasets show that AdaPrompt improves over a standard prompt method by large margins. In particular, retrieving relevant data for continual pre-training of a language model can serve to warm-up the model for both domain adaptation and prompt-filling tasks. In addition, an NLI model allows effective selection of filled tokens to achieve improved performance.

References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. *arXiv preprint arXiv:2004.02105*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Alexandra Chronopoulou, Christos Baziotis, and Alexandros Potamianos. 2019. An embarrassingly simple approach for transfer learning from pretrained language models. 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 2089–2095, Minneapolis, Minnesota. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

Xiang Dai, Sarvnaz Karimi, Ben Hachey, and Cecile Paris. 2019. Using similarity measures to select pretraining data for ner. *arXiv preprint arXiv:1904.00585*.

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

Aaron Gokaslan and Vanya Cohen. 2019. Openweb-text corpus.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Junxian He, Wojciech Kryściński, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2020. Ctrl-sum: Towards generic controllable text summarization. *arXiv preprint arXiv:2012.04281*.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Juanzi Li, and Maosong Sun. 2021. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. *arXiv preprint arXiv:2108.02035*.

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.

Constantinos Karouzos, Georgios Paraskevopoulos, and Alexandros Potamianos. 2021. UDALM: Unsupervised domain adaptation through language modeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2579–2590, Online. Association for Computational Linguistics.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

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. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

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*.

Tomas Mikolov, Kai Chen, Greg S Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Noisy channel language model prompting for few-shot text classification. *arXiv preprint arXiv:2108.04106*.

Robert C Moore and Will Lewis. 2010. Intelligent selection of language model training data.

Sebastian Nagel. 2016. Cc-news.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *arXiv preprint arXiv:2105.11447*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Barbara Plank and Gertjan Van Noord. 2011. Effective measures of domain similarity for parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1566–1576.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Sebastian Ruder, Parsa Ghaffari, and John G Breslin. 2017. Data selection strategies for multi-domain sentiment analysis. *arXiv preprint arXiv:1702.02426*.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with bayesian optimization. *arXiv preprint arXiv:1707.05246*.

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.

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.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Eliciting knowledge from language models using automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.

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.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

Marlies Van Der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. *arXiv preprint arXiv:1708.00712*.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. 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)*.

Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. 2018. Denoising neural machine translation training with trusted data and online data selection. *arXiv preprint arXiv:1809.00068*.

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. Association for Computational Linguistics.

Xingcheng Yao, Yanan Zheng, Xiaocong Yang, and Zhilin Yang. 2021. Nlp from scratch without large-scale pretraining: A simple and efficient framework. *arXiv preprint arXiv:2111.04130*.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. 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 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *arXiv preprint arXiv:2106.11520*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 9054–9065.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Tony Z Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. *arXiv preprint arXiv:2102.09690*.

Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Dialoglm: Pre-trained model for long dialogue understanding and summarization. *arXiv preprint arXiv:2109.02492*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.