Finding a Lottery Prompt: Optimization-free Prompting for Pre-trained Language Models

Anonymous ACL submission

Abstract

Consistently scaling pre-trained language models (PLMs) imposes substantial burdens on model adaptation, necessitating more efficient alternatives to conventional fine-tuning. Given the advantage of prompting in the zero-shot setting and the observed performance fluctuation among different prompts, we explore the instance-level prompt and the achievable upper-bound performance. We first validate the assumption that for every instance, there is almost *always* a lottery prompt that induces the correct prediction from the given PLM without tuning a single parameter. Meanwhile, it is shown that some strong prompts have high performance over the whole training set. Then, we attempt to generalize the strong prompts from the training set to the test set with ensembling methods. Experiments are conducted on various types of NLP classification tasks and demonstrate that the proposed method can achieve considerably better performance than strong optimization-based baselines by at least 3% (up to 17%) in average metric.

1 Introduction

Since pre-trained language models (PLMs) became the de-facto standard in modern NLP researches (Devlin et al., 2019; Liu et al., 2019; Han et al., 2021a), the pretraining-finetuning paradigm has been prevailing until recent years when models keep scaling (Radford et al., 2019; Brown et al., 2020; Rae et al., 2021) and become too expensive to be optimized. To this end, researchers are actively seeking more effective strategies that require little or even no optimization to harness PLMs.

Among these exploratory studies of advanced model adaptation, prompting (Brown et al., 2020; Schick et al., 2020; Schick and Schütze, 2021a; Gao et al., 2021) is gaining popularity in the community, which uses additional context (prompts[1]) to wrap input instances and trigger desired output tokens. In classification tasks, these tokens are further mapped to particular labels by a verbalizer. Such a paradigm is verified to be effective in a variety of downstream tasks, even when annotations are insufficient. Particularly, empirical evidence shows that coincidental prompts could achieve extraordinary performance in the zero-shot setting, i.e., no training examples are presented. For example, simple manual prompt can achieve an f1 score of over 60 on 46-class entity typing dataset (Ding et al., 2021a), and reaches 73% accuracy on DBpedia with 14 classes (Hu et al., 2021) in zero-shot setting.

Although such effectiveness is often accompanied by drastic fluctuations with prompt selection (Zhao et al., 2021), it still implies that prompting PLMs may implicitly have an exceedingly high upper capability even without model tuning. And it may be possible to tap into the potential by assigning different prompts for distinct data points. Intrigued by this intuition, we explore a bold hypothesis: *Is it possible to find at least one instance-level prompts (lottery prompts) that induce correct output for every data point in classification tasks without any optimization?* Surprisingly, after building an automatic searching procedure with reasonable searching space on 13 representative classification datasets, we validate the existence of such lottery prompts (§ 2). That is, the combination of just a few discrete tokens can make a PLM output correct labels for almost any classification data. This finding refreshes our recognition of the limit of prompted knowledge in PLMs and demonstrates a promising upper capability of the PLMs' inference.

Then, we further analyze the lottery prompts and find there are a considerable number of "strong prompts" among them (§ 2.3), i.e., prompts that could perform well on the whole training data, not just the individual data point. This offers a natural opportunity to generalize such strong prompts to

[1]In this paper, the term "prompt" technically refers to the template that wraps the original input.

unseen (test) data (§ 3). We develop both straightforward and sophisticated strategies to establish a group of strong prompts (§ 3.2) for each dataset and to ensemble the predictions (§ 3.3) within the group. Among them, the mutual information-based strategy we developed is verified to be consistently effective, indicating unique correlations between prompts and instances in terms of shared information. Although we did not introduce any optimization in this study, our method remarkably outperforms strong gradient-based and gradient-free (optimization still needed) baselines by at least 3% (up to 17%) in average metric on all the evaluation datasets (§ 4). We also analyze the effect of ensembling strategy and training data size on the test set performance (§ 4.3) to explore the conditions for better generalizability of the lottery prompts. For simple tasks, the strong prompts are shown to be transferable (§ 4.3). All the codes will be publicly available for reproducibility.

2 The Existence of Lottery Prompts for Every Data Point

Considering the extraordinary performance observed on zero-shot classification and the large variance brought by the prompt selection, we make an assumption as follows: Given a pre-trained language model and a classification dataset, for each instance, at least one lottery prompt exists that can induce the desired label from the PLM, without the need to update the PLM parameters.

To validate the assumption, we conduct pilot experiments that attempt to find the lottery prompt for every data point on 13 classification tasks. Note that for different instances, the prompt may be different, and our goal is to verify the existence of such prompts in this pilot experiment.

2.1 Overview and Setup

Particularly, for every input instance in a classification task, we attempt to search and combine textual tokens from a discrete search space into a prompt. And the model is expected to produce desired label words for correct classification of the wrapped instance. We choose 13 datasets of various NLP tasks for assumption validation. Most of them come from GLUE benchmark (Wang et al., 2018), and others include Yelp Polarity (Zhang et al., 2015), SNLI (Bowman et al., 2015), AG's News (Zhang et al., 2015), DBpedia (Zhang et al., 2015), and Few-NERD (Ding et al., 2021b). SST-2 (Socher et al., 2013) and Yelp Polarity are datasets for binary sentiment classification. CoLA (Warstadt et al., 2019) is for acceptibility judgment of single sentence. SNLI, RTE (Wang et al., 2018), QNLI (Wang et al., 2018), WNLI (Levesque, 2011) and MNLI (Williams et al., 2018) target at language inference detection given a sentence pair. QQP (Iyer et al., 2017) and MRPC (Schick et al., 2020) are for paraphrase judgment. AG's News and DBpedia are used for text theme classification. Few-NERD is an entity typing dataset. As for prompt search space, 200 words with top frequency in English[2] are gathered and grouped according to part-of-speech tag with NLTK package (Loper and Bird, 2002) into nouns, verbs, prepositions, adjectives and adverbs. The designed prompt search space is the Cartesian product of three word sets $\mathcal{T} = \texttt{NOUNS} \times \texttt{VERBS} \times (\texttt{PREP} \cup \texttt{ADJ} \cup \texttt{ADV}) \times \{\texttt{<MASK>}\}$, and $|\mathcal{T}| = 76725$. We use RoBERTa-large (Liu et al., 2019) and GPT-2 (Radford et al., 2019) as the backbones.

2.2 The Searching Process

For each dataset, we randomly sample 1000 instances from the training set as $\mathcal{X}_{\text{train}} = \{(x_i, y_i)\}$ and apply each prompt $T \in \mathcal{T}$ to each instance and use the PLM $\mathcal{M}$ to produce the prediction. Specifically, a prompt T composed of a noun, a verb and an adjective may be *"it was really"*. Applying it to an instance x:*"A fun movie."* will yeild the input text $T(x)$:*"A fun movie. it was really <MASK>"*. For each of such pair $T(x) \in \mathcal{X}_{\text{train}} \times \mathcal{T}$, the score for each class can be obtained as

$$o(x; T, \mathcal{M}) = \texttt{Softmax}(\mathrm{V}(\mathcal{M}(T(x)))), \quad (1)$$

where V denotes the projection from output logits over PLM vocabulary to the class label set. Specifically, to reduce the impact from the prompt, we use calibration (Zhao et al., 2021) to rescale the scores before making the final prediction.

$$\begin{aligned} q(T; \mathcal{M}) &= \texttt{Softmax}(\mathrm{V}(\mathcal{M}(T(\cdot)))), \\ p(x; T, \mathcal{M}) &= \texttt{Normalize}(\frac{o(x; T, \mathcal{M})}{q(T; \mathcal{M})}). \end{aligned} \quad (2)$$

$T(\cdot)$ means a pure prompt without input x is fed into the PLM and q is the output probability over the label set. p is the final calibrated probability over the class labels. For every $(x, y) \in \mathcal{X}_{\text{train}}$, we enumerate over each $T \in \mathcal{T}$ and see if the output $\hat{y} = \arg\max p$ will give the correct prediction y. The specific prompt format and label words used are shown in Appendix C.

[2]`https://sketchengine.co.uk`

Datasets	#Classes	RoBERTa-large	GPT-2
SST-2	2	100.00	100.00
Yelp P.	2	100.00	100.00
SNLI	3	100.00	99.90
RTE	2	100.00	100.00
MRPC	2	100.00	100.00
CoLA	2	100.00	100.00
MNLI	3	99.90	99.90
QNLI	2	100.00	100.00
QQP	2	100.00	100.00
WNLI	2	100.00	100.00
AG's News	4	100.00	100.00
DBpedia	14	100.00	100.00
Few-NERD	66	100.00	99.70

Table 1: The percentage of instances with lottery prompts for each dataset's 1000 randomly sampled data. WNLI uses the whole training set with 635 instances.

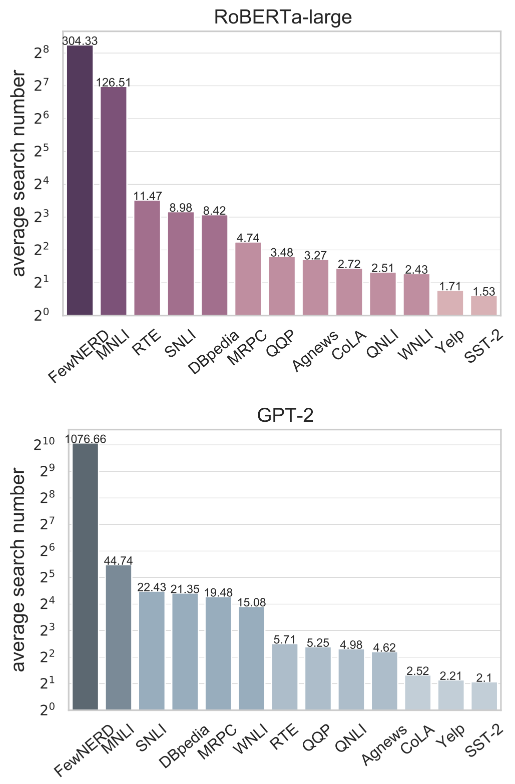


Figure 1: The average search number of each dataset.

2.3 Results and Analyses

Verification of the Assumption. Table 1 reports the basic searching results. Each instance x is considered correctly predicted if there exists $T \in \mathcal{T}$ such that $y = \arg\max p$. It is shown that for all datasets, a lottery prompt that induces the correct prediction from $\mathcal{M}$ exists for almost all 1000 instances. The assumption is thus validated, that is, in a finite search space composed of textual tokens, we can almost always find at least one combination of common words as a prompt to make the prediction correct. While it may not be surprising to see a success on binary classification tasks, achieving 100% coverage on Few-NERD, a 66-class dataset for entity typing, is worth noting. It indicates that the particular semantics distributed in PLM can be triggered by certain contexts even without any further fine-tuning.

Naturally, the phenomenon is not observed when the model is not pre-trained. We conduct the same searching process for Few-NERD on a randomly initialized RoBERTa-large, and only 33.1% instances could find the corresponding lottery prompts. This further shows that the existence of lottery prompts for each instance is not merely a stroke of luck, but also a unique and consequent effect along with language model pre-training.

The Cost of the Searching Process. As aforementioned, the searching space in our pilot experiment is $|\mathcal{T}| = 76725$, however, the practical cost to get a lottery prompt for one data point is significantly lower than the budget. As shown in Figure 1, the average number of searches for each instance does not exceed 30 on most datasets for both PLMs. Naturally, searching for a lottery prompt for a multi-class classification problem is more costly. The 66-class typing dataset Few-NERD requires a significantly larger search number than the rest of the datasets, most of which only contain 2 or 3 classes. Another reasonable observation is that single sentence classification tasks are generally easier than tasks involving sentence pairs. Meanwhile, NLI tasks with mixed domains are probably the most difficult sentence-pair tasks, given that MNLI, RTE, and SNLI are more costly than paraphrase tasks and other domain-specific NLI datasets. In terms of the genre of models, the auto-regressive model (GPT-2) generally takes more searches than the auto-encoding model (RoBERTa-large). Despite the differences in individual datasets, they show similar trends, which can roughly reflect how difficult the dataset is for PLMs.

The Strong Prompts. After searching for lottery prompts for all instances, we are interested in if there are "strong prompts" among them, i.e., prompts that win on the whole $\mathcal{X}_{\text{train}}$. We measure the performance of each prompt over $\mathcal{X}_{\text{train}}$ with standard metrics on some representative datasets from each task category. The metric statistics and variation of all prompts are shown in Figure 2. Different tasks show distinct patterns. Text classifica-

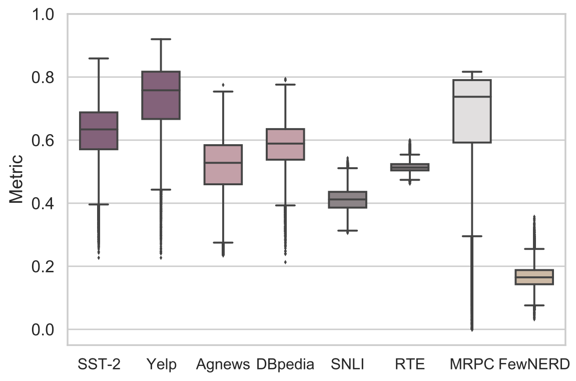


Figure 2: Prompt performance and variation on each dataset using RoBERTa-large. The vertical axis represents the metric of each prompt over $\mathcal{X}_{\text{train}}$. MRPC uses f1 metric, and others use accuracy.

tion tasks with single sentence are more sensitive to prompt choice and often observe larger performance variation over the prompt space. For SST-2, while the best-performing prompt reaches an accuracy of 0.8, the worst prompts can barely get to 0.3. For NLI tasks, prompt performance is more stable however mediocre. It should be noted that, despite having altogether 66 classes, the average accuracy on Few-NERD is still over 0.1. Many of the top-performing prompts fit with our language intuition, both syntactically and semantically. For example, the top prompts for sentiment analysis task are compatible with chosen label words. Adverbs that enhance the statement (e.g. just, really, very) appear frequently in sentiment analysis tasks. For entity typing, the words like "other" and "such" naturally lead to noun-like words or category words. A detailed case study of strong prompts and difficult instances can be found in Appendix D and E. The most important indication from the results is that some strong prompts could yield significant performance on $\mathcal{X}_{\text{train}}$, and such prompts could be obtained without any optimization process. At this point, a natural question arises: *How to generalize these strong prompts to unseen (test) data?*

3 Generalize Strong Prompts to Test Dataset

In Section 2, we have empirically verified that conditioned on a pre-trained model and a classification task, it is possible to find a lottery prompt for almost every data point. We also find that a few strong prompts could perform non-trivially on the whole dataset $\mathcal{X}_{\text{train}}$. In this section, we attempt to generalize the strong prompts to unseen (test) dataset $\mathcal{X}_{\text{test}}$. Our considerations are:

- A strong prompt that performs well on the training set can naturally be assumed to perform relatively well on the test set.
- It is empirically difficult to estimate which prompt will perform well on which instance for test data. Therefore, we attempt to combine the predictions of selected strong prompts $\mathcal{T}^* = \{T_1, T_2, ..., T_t\} \in \mathcal{T}$. Under this circumstance, we investigate strategies to select $\mathcal{T}^*$ and ensemble the predictions of prompts in $\mathcal{T}^*$.

3.1 Overview

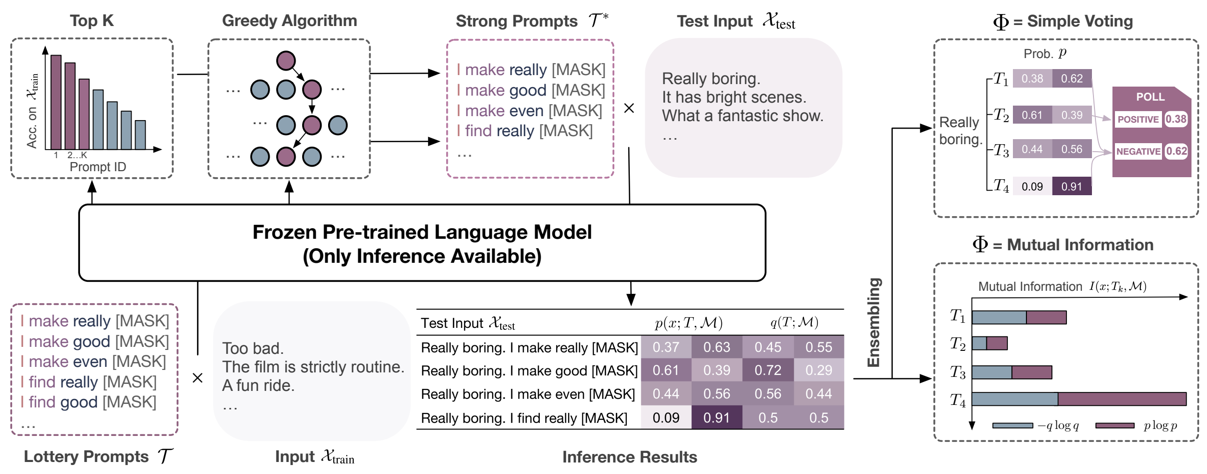
The objective is to identify a set of feasible prompts that can perform well on $\mathcal{X}_{\text{test}}$. We divide the procedure into two stages, where we first find a group of strong prompts (denoted as $\mathcal{T}^*$), then use ensembling methods to get the final prediction for $\mathcal{X}_{\text{test}}$. Since the choice of $\mathcal{T}^*$ is solely based on inference results on $\mathcal{X}_{\text{train}}$, the process uses no validation set. Formally, given the chosen group of strong prompts $\mathcal{T}^* = \{T_1, T_2, ..., T_t\} \subset \mathcal{T}$, the prediction for each data point $x \in \mathcal{X}_{\text{test}}$ is obtained by

$$p(x; \mathcal{T}^*, \mathcal{M}) = \Phi(p_1, p_2, ..., p_t), \qquad (3)$$

where $p_k = p(x; T_k, \mathcal{M})$ and is calculated as equation 2, and Φ is the ensembling method used. Both the prompt group $\mathcal{T}^*$ and the ensembling strategy Φ would have considerable impact on final predictions on the test dataset $\mathcal{X}_{\text{test}}$.

3.2 The Choice of $\mathcal{T}^*$

As for the choice of $\mathcal{T}^*$, two main strategies are developed: (1) *Top-k*. With the assumption that strong prompts over $\mathcal{X}_{\text{train}}$ are also expected to perform well on $\mathcal{X}_{\text{test}}$, these best-performing prompts are regarded as the most reliable for predicting the unseen data. So we naively take the top-k best performing prompts over the training set as $\mathcal{T}^*$. In the experiments, we use k=5, 10, and 20. (2) *Greedy best*. Since our final objective is to maximize the metrics on $\mathcal{X}_{\text{test}}$, it is natural that we should attempt to find a $\mathcal{T}^*$ that achieves the highest ensembling results on $\mathcal{X}_{\text{train}}$. While the combinatorial optimization problem of evaluating each $\mathcal{T}' \in \mathcal{T}$ is intractable, the greedy algorithm is adopted to approximate the solution. As shown in Algo. 1, at each iteration the prompt that will boost the performance most will be added into $\mathcal{T}^*$, which is measured by the average of predicted probability, until no improvement can be made by any new prompt.



Test Input $\mathcal{X}_{\text{test}}$	$p(x;T,\mathcal{M})$		$q(T;\mathcal{M})$	
Really boring. I make really [MASK]	0.37	0.63	0.45	0.55
Really boring. I make good [MASK]	0.61	0.39	0.72	0.29
Really boring. I make even [MASK]	0.44	0.56	0.56	0.44
Really boring. I find really [MASK]	0.09	0.91	0.5	0.5

Figure 3: The complete searching process of $\mathcal{T}^*$ and ensembling for final prediction.

Algorithm 1 The greedy algorithm for choosing the best performing prompt group on $\mathcal{X}_{\text{train}}$. $P^{(k)}$ is the set of scores for each instance $(x, y) \in \mathcal{X}_{\text{train}}$ given $T_k \in \mathcal{T}$, where $P_i^{(k)} = p(x_i; T_k, \mathcal{M})$. Eval$(\cdot,\cdot)$ calculates the metric for specific dataset.

```
Input: $\mathcal{X}_{\text{train}}, \mathcal{M}, \mathcal{T}$; Output: $\mathcal{T}^*$
$\mathcal{T}^* \leftarrow \emptyset, \mathcal{P} \leftarrow \emptyset, k^* \leftarrow$ None, $s^* \leftarrow 0$
while $|T^*|$ is 0 or $k^*$ is not None do
    if $k^*$ is not None then
        Add$(\mathcal{T}^*, T_{k^*})$; Add$(\mathcal{P}, P^{(k^*)})$
    end
    $k^* \leftarrow$ None; $s' \leftarrow s^*$;
    for $k = 1$ to $|\mathcal{T}|$ do
        if $T_k \in \mathcal{T}^*$ then continue ;
        Add$(\mathcal{P}, P^{(k)})$ $\hat{Y} = \arg\max(\text{average}(\mathcal{P}))$
        if Eval$(\hat{Y}, \mathcal{X}_{train}) > s'$ then
            $s' \leftarrow$ Eval$(\hat{Y}, \mathcal{X}_{\text{train}})$; $k^* \leftarrow k$;
        end
        Delete$(\mathcal{P}, P^{(k)})$; $s^* \leftarrow s'$;
    end
end
return $\mathcal{T}^*$
```

3.3 Ensembling Method Φ

The most naive way of ensembling is *simple voting* by predictions of individual prompts in $\mathcal{T}^*$,

$$\Phi(p_1, p_2, ..., p_t) = \frac{1}{t}\sum_{k=1}^{t} p_k. \tag{4}$$

However, it is not always true that each prompt should weigh equally with respect to each instance. For some data, some prompts may be more reliable, while for the others the effect may be different. It will be ideal if we can further select the most suitable prompt out of $\mathcal{T}^*$ for each instance. Therefore, we devise a more sophisticated method termed as *mutual-information-based choice*. Intuitively, the more reliable a prompt T is, the more confident the model $\mathcal{M}$ will be about instance x. Inspired by Sorensen et al. (2022), we measure the confidence with the mutual information between x and y, T, which is defined by the reduction in entropy of predicted probability brought by x,

$$\begin{aligned} I(x;T_k,\mathcal{M}) &= H(q|T_k(\cdot)) - H(p|T_k(x)) \\ &= -\sum_i q_i(T_k;\mathcal{M})\log q_i(T_k;\mathcal{M}) + \\ &\quad \sum_i p_i(x;T_k,\mathcal{M})\log p_i(x;T_k,\mathcal{M}), \end{aligned} \tag{5}$$

where q and p are the predicted probability vectors as in Equation 2, and the subscript i is the class index. It is expected that if a prompt is reliable for x, the uncertainty in prediction should be greatly reduced compared with making a prediction based on the prompt solely. Specifically, it entails that a good prompt itself should contain no bias towards the label set, so q should be a uniform distribution. On the other hand, a suitable prompt for a specific instance should induce an almost certain prediction on the desired class, corresponding to a near one-hot vector p. So the objective is to find the most confident prompt $T^* \in \mathcal{T}^*$ for each instance in $\mathcal{X}_{\text{test}}$, and produce the prediction p with T^*.

$$\begin{aligned} T^* &= \arg\max_{T\in\mathcal{T}^*} I(x;T,\mathcal{M}), \\ \Phi(p_1,p_2,...,p_t) &= p(x;T^*,\mathcal{M}). \end{aligned} \tag{6}$$

4 Experiments

In this section, we comprehensively evaluate the generalization of the selected prompts on 7 representative datasets. By comparing with other gradient-based or gradient-free baselines, we find that our optimization-free method could yield surprising superiority. Further, we investigate the effect of $\mathcal{T}^*$, Φ, and the size of training data.

4.1 Baselines

Given that our method optimizes zero parameters, few existing methods are directly comparable. We choose some of the optimization-based methods that optimize only a small number of parameters instead. **Prompt Tuning** (Lester et al., 2021) optimizes the continuous prompt at the input level. **P-Tuning v2** (Liu et al., 2021a) is a variant of prompt tuning that pre-pends trainable parameters to each layer of the PLM and optimizes them in a multi-task setting. **Feature-MLP** and **Feature-BiLSTM** (Peters et al., 2019) both use pre-trained features output by PLMs and train a lightweight classifier offline. **Black-Box Tuning** (Sun et al., 2022) is a gradient-free method that optimizes the projected extra 500 parameters at the input layer with Covariance Matrix Adaptation Evolution Strategy. **Manual Prompt** is a zero-shot method that directly uses a hand-crafted textual prompt for each dataset. **Best Prompt** uses the searched Top-1 prompt on training data directly. **In-Context Learning** (Brown et al., 2020) is an optimization-free method that uses a few samples as demonstrations prepended to the test sample.

4.2 Experimental Settings

We conduct experiments under few-shot settings on 7 datasets: SST-2, Yelp P., AG's News, DBpedia, MRPC, SNLI, and RTE. While Sun et al. (2022) adopts a 16-shot setting and uses the same number of instances as the validation set, we directly use 32-shot data as the training set as our method requires no validation set. The total seen labeled data number is the same across all methods. Specifically, we randomly select 32 instances for each class from the training set and obtain the predicted scores combined with each $T \in \mathcal{T}^*$. Then different strategies are applied to choose $\mathcal{T}^*$ and obtain the final prediction and evaluation metrics with specific Φ on the test set. The original validation set is used as the test set following (Sun et al., 2022). For each dataset, the experiments are run with 5 different random seeds, and the mean metrics and standard deviations are reported. The baseline results are taken from Sun et al. (2022). All methods use RoBERTa-large (Liu et al., 2019) as the PLM backbone. The label words used follow (Sun et al., 2022) and are the same across all methods.

4.3 Results and Analysis

Overall Results Table 2 shows the main results on each dataset. The reported results use Top-10 prompts as $\mathcal{T}^*$ and mutual-information-based choice as Φ. Overall the proposed method performs the best among all methods. It points to the fact that with a reasonable prompt search space and a few training instances, strong prompts can be identified and generalized effectively to unseen data. Best prompt on 32-shot data surprisingly overtakes other baselines. This, jointly with the mediocre performance of manual prompts, indicate that a human-comprehensible prompt may not always be the best choice for PLMs and may fail to probe a considerable amount of intrinsic knowledge in PLMs. Meanwhile, the success of our method over best prompt (where a single prompt is used) shows that ensembling a set of strong prompts is beneficial. Comparing across datasets, our method is more advantageous over black-box tuning in harder tasks, including natural language inference (SNLI and RTE) and paraphrasing (MRPC). For single-sentence classification tasks, the improvement is minor. This finding fits with our intuition, as tasks involving two sentences often require more abstract abilities like reasoning and the contexts are more diverse across instances. Designing or optimizing for one unified prompt for such datasets is admittedly harder. Above all, it is exciting that ensembling a proper set of prompts composed of textual tokens may surpass network optimization on a dataset in an era of pre-trained language models and points to the values of mining and tapping into an optimal usage of plain textual prompt.

Comparison of $\mathcal{T}^*$ and Φ To compare different strategies in choosing $\mathcal{T}^*$ and ensembling methods Φ, we run experiments with 1000 instances as the training set on each of the 7 datasets. The full training set is used to ensure stability. Figure 4 shows metrics for each combination of $\mathcal{T}^*$ and Φ. In terms of $\mathcal{T}^*$, generally, a larger k will produce better performance for the top-k strategy. For most datasets, $k = 10$ is already good enough. The magnitude of increase brought by larger k is also closely related to the distribution of strong prompts. In reference to Figure 2, datasets with a large number of strong prompts (Yelp P. and MRPC) observe little impact from the strategies chosen. While for

Method	#Tunable Param.	SST-2 Acc.	Yelp P. Acc.	AG's News Acc.	DBpedia Acc.	MRPC F1	SNLI Acc.	RTE Acc.	Avg.
Gradient-based Methods									
Prompt Tuning	50K	68.23 ± 3.78	61.02 ± 6.65	84.81 ± 0.66	87.75 ± 1.48	51.61 ± 8.67	36.13 ± 1.51	54.69 ± 3.79	63.46
P-Tuning v2	1.2M	64.33 ± 3.05	92.63 ± 1.39	83.46 ± 1.01	97.05 ± 0.41	68.14 ± 3.89	36.89 ± 0.79	50.78 ± 2.28	70.47
Gradient-free Methods									
Feature-MLP	1M	64.80 ± 1.78	79.20 ± 2.26	70.77 ± 0.67	87.78 ± 0.61	68.40 ± 0.86	42.01 ± 0.33	53.43 ± 1.57	66.63
Feature-BiLSTM	17M	65.95 ± 0.99	74.68 ± 0.10	77.28 ± 2.83	90.37 ± 3.10	71.55 ± 7.10	46.02 ± 0.38	52.17 ± 0.25	68.29
Black-Box Tuning	500	89.56 ± 0.25	91.50 ± 0.16	81.51 ± 0.79	87.80 ± 1.53	61.56 ± 4.34	46.58 ± 1.33	52.59 ± 2.21	73.01
Optimization-free Methods									
Manual Prompt‡	0	79.82	89.65	76.96	41.33	67.40	31.11	51.62	62.56
In-Context Learning†	0	79.79 ± 3.06	85.38 ± 3.92	62.21 ± 13.46	34.83 ± 7.59	45.81 ± 6.67	47.11 ± 0.63	60.36 ± 1.56	59.36
Best Prompt†	0	86.40 ± 4.28	88.5 ± 1.65	77.06 ± 0.19	82.70 ± 0.00	73.63 ± 3.53	51.26 ± 1.75	64.64 ± 0.00	74.88
Ours†	0	89.52 ± 1.73	91.85 ± 0.42	79.67 ± 1.73	89.26 ± 0.77	73.97 ± 3.75	50.61 ± 0.98	58.70 ± 1.32	**76.23**

Table 2: Performance on datasets under few-shot setting. † means using 32-shot data as training set and no extra data as validation set. ‡ means no labeled data are used for training or validation. Other methods use 16-shot data as training and validation set. The baseline results follow Sun et al. (2022).

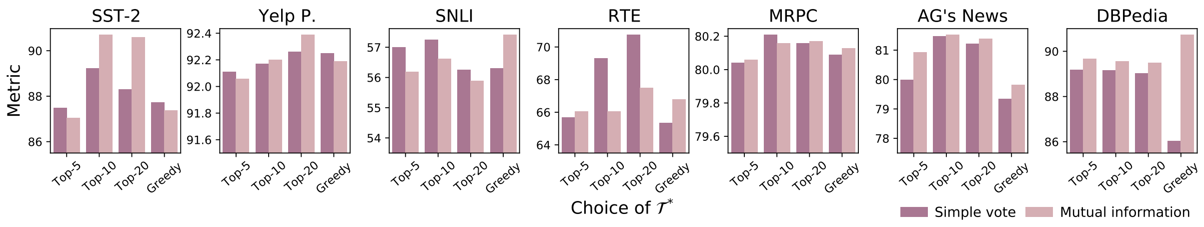


Figure 4: Performance of different choices of $\mathcal{T}^*$ and Φ. 1000 training instances are used for each dataset.

datasets like RTE, where most prompts perform badly, ensembling with larger k will greatly boost the prediction accuracy. It is surprising that despite having a higher performance on training sets, the greedy strategy performs even worse on most test sets. It indicates that the boost in training sets is often linked to some corner cases, which is hard to generalize to test set. As for Φ, distinct patterns are observed for different datasets. Measuring prediction confidence with mutual information tends to be more reliable for single-sentence classification tasks. This also reflects the difficulty of the tasks to some extent.

Impact of Training Data Size To further explore the property of our method, experiments are conducted under few-shot settings ranging from 8 shots to 256 shots. Each is run 5 times, and metrics on the test sets are reported in Figure 5. We can see that performance varies a lot when different instances are sampled as the training set under low-shot settings. It suggests the importance of choosing the proper training data for our method. When more shots are provided, metrics get higher and variance gets smaller. As the volume climbs up to 128 shots and 256 shots, the increase in metrics becomes minor for most datasets. It can also be concluded that for low-shot settings, mutual information can yield higher results on the test set. But as more training data are available, the gap is narrowed and the two ensembling strategies converge to similar levels.

Task	Setting	Metrics
Sentiment Analysis	SST-2 → Yelp P.	90.27 (1.58 ↓)
	Yelp P. → SST-2	84.15 (5.37 ↓)
Language Inference	RTE → SNLI	40.48 (10.13 ↓)
	SNLI → RTE	54.51 (4.19 ↓)
	MNLI → SNLI	47.96 (2.65 ↓)
	MNLI → RTE	55.81 (2.89 ↓)

Table 3: Transferability test of $\mathcal{T}^*$ across datasets with similar tasks. Prompts are searched on 32-shot training data from source dataset and evaluated on test set of target dataset. Top-10 prompts are used as $\mathcal{T}^*$ and mutual-information-based choice is used as Φ.

Transferability Test We test the prompt transferbility across datasets with similar tasks under 32-shot setting. Experiments are conducted on sentiment analysis and language inference tasks. We also use MNLI as the source dataset as many previous works do. Table 3 shows that prompts chosen by our method can be transferable. While SST-2 and Yelp observe mutual transferability, transfering RTE to SNLI is relatively hard. MNLI is shown to be a robust dataset for NLI task and the searched

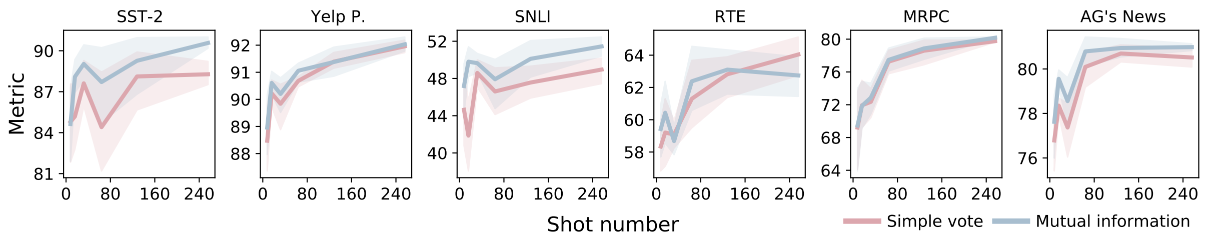


Figure 5: Average performance and standard deviation of the simple vote and mutual information under few-shot settings. We use 8, 16, 32, 64, 128, and 256-shot settings for training data. Top-10 prompts over each training set are adopted as $\mathcal{T}^*$. For each setting, the experiments are run with 5 random seeds.

prompts perform satisfyingly on both RTE and SNLI even with only a few training instances. It is also in line with previous research findings that using prompts pretrained on MNLI could greatly boost performance on other NLI datasets. Above all, the results demonstrate that our proposed strategy for choosing and ensembling prompts can effectively extract representative prompts for a specific kind of task, which can be further utilized to reduce search cost.

5 Related Work

Prompting, as an alternative to standard finetuning, is originally inspired by GPT-3 (Brown et al., 2020) and knowledge probing (Petroni et al., 2019; Jiang et al., 2020). With a similar form to pre-training tasks, it stimulates the intrinsic knowledge in PLMs more efficiently. Following several of the earliest works (Schick and Schütze, 2021a,b), promoting has been applied in various NLP tasks (Han et al., 2021b; Li and Liang, 2021; Sainz et al., 2021; Ding et al., 2021a). It is also discovered that the specific prompt used has a great impact on task performance. Therefore, efforts have been devoted to prompt engineering and automatic prompt generation. Optimizing for a good prompt has been conducted at both discrete token level (Shin et al., 2020; Gao et al., 2021) and continuous embedding level (Li and Liang, 2021; Zhang et al., 2021; Liu et al., 2021b). Some also focus on the choice and representation of label words (Schick et al., 2020; Hu et al., 2021; Zhang et al., 2021). Experiments show that a well-optimized or pre-trained (Gu et al., 2022) prompt can be beneficial.

Given the striking performance of prompting under few-shot settings especially, recently, more works are focusing on more efficient tuning of PLMs based on prompts. Prompt tuning (Lester et al., 2021) tunes the pre-pended token embedding only. Other works enhance PLMs' zero-shot learning ability with prompts. Studies show that large PLMs with proper prompts (Wei et al., 2021) and training with diverse prompts (Sanh et al., 2021) can advance zero-shot performance. This line of works emphasizes the efficient tuning and steering process of large PLMs. Black-box tuning (Sun et al., 2022) optimizes the pre-pended continuous prompt in a projected low-dimensional space without PLM gradient information.

This work is among the first few efforts (Jin et al., 2022; Wu et al., 2022) in mining instance-level prompts, and is the first to propose and prove the existence of a lottery prompt composed of a few textual tokens for each instance. In contrast to tuning a small number of parameters or tuning without gradients, an optimization-free method is proposed to generalize the searched prompts to the test sets.

6 Conclusion

In this work, we emphasize on the existence of a lottery prompt for every single instance and the adaptation of them in various classification tasks in an optimization-free manner. We propose a large prompt space composed of common words as the search space, devise multiple strategies to choose a set of prompts based on the training set, and demonstrate the effectiveness of ensembling the set of chosen prompts on the test set. Our proposed optimization-free method achieves satisfactory results on various NLP tasks under few-shot settings. It is also found that the specific training instances used and the ensembling strategy adopted is crucial to the generalization effect. Above all, this work illuminates the fact that the great potential of PLMs can be successfully harnessed and prompted by plain textual prompts mined from PLM vocabulary without parameter optimization and thus points to the need for future efforts in more efficient ways in mining lottery prompts.

Ethical Considerations

This work shows that with proper plain textual prompts, instance-level desired results can be prompted from PLMs. This inherent feature of PLMs means attacks can be launched to produce rude or discriminated words. On the other hand, however, we believe it can also be a technique used for debiasing a PLM. Overall, this effect depends on the intention of the users and the pre-training corpus of the corresponding PLMs. The analysis of this study can be used to facilitate the community to develop more specifications for the rational use of PLMs (especially the super-large ones), and more approaches to effectively prevent potential ethical issues. For example, we can use this technique to analyze which outputs that may have ethical issues are easily triggered by which contexts (prompts), and develop a set of intervention methods to make these

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of EMNLP*, pages 632–642.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris 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. In *Proceedings of NeurIPS*, volume 33, pages 1877–1901.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021a. Prompt-learning for fine-grained entity typing. *arXiv preprint*, 2108.10604.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Haitao Zheng, and Maosong Sun. 2022. OpenPrompt: An open-source framework for prompt-learning. In *Proceedings ACL*, pages 105–113.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021b. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of ACL*, pages 3198–3213.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of ACL/IJCNLP*, pages 3816–3830.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. PPT: Pre-trained prompt tuning for few-shot learning. In *Proceedings of ACL*, pages 8410–8423.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Liang Zhang, Wentao Han, Minlie Huang, Qin Jin, Yanyan Lan, Yang Liu, Zhiyuan Liu, Zhiwu Lu, Xipeng Qiu, Ruihua Song, Jie Tang, Ji-Rong Wen, Jinhui Yuan, Wayne Xin Zhao, and Jun Zhu. 2021a. Pre-trained models: Past, present and future. *AI Open*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021b. Ptr: Prompt tuning with rules for text classification. *arXiv preprint*, 2105.11259.

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*, 2108.02035.

Shankar Iyer, Nikhil Dandekar, , and Kornel Csernai. 2017. First quora dataset release: Question pairs.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *TACL*, 8:423–438.

Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. 2022. Instance-aware prompt learning for language understanding and generation. *arXiv*, abs/2201.07126.

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

Hector J. Levesque. 2011. The winograd schema challenge. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of ACL*, pages 4582–4597.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021a. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint*, abs/2110.07602.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

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*, abs/1907.11692.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, page 63–70.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of RepL4NLP*, pages 7–14.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of EMNLP-IJCNLP*, pages 2463–2473.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Oscar Sainz, Oier Lopez de Lacalle, Gorka Labaka, Ander Barrena, and Eneko Agirre. 2021. Label verbalization and entailment for effective zero- and few-shot relation extraction. *arXiv preprint*, abs/2109.03659.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M. Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint*, abs/2110.08207.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of COLING*, pages 5569–5578.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of EACL*, 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 NAACL*, pages 2339–2352.

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 EMNLP*, pages 4222–4235.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642.

Taylor Sorensen, Joshua Robinson, Christopher Rytting, Alexander Shaw, Kyle Rogers, Alexia Delorey, Mahmoud Khalil, Nancy Fulda, and David Wingate. 2022. An information-theoretic approach to prompt engineering without ground truth labels. In *Proceedings of ACL*, pages 819–862.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. *arXiv preprint*, abs/2201.03514.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *TACL*, 7:625–641.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint*, abs/2109.01652.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL*, pages 1112–1122.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers:

State-of-the-art natural language processing. In *Proceedings of EMNLP: System Demonstrations*, pages 38–45.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, V. G. Vinod Vydiswaran, and Hao Ma. 2022. IDPG: an instance-dependent prompt generation method. *arXiv preprint*, abs/2204.04497.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable prompt makes pre-trained language models better few-shot learners. *arXiv preprint*, abs/2108.13161.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of NeurIPS*, page 649–657.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of ICML*, pages 12697–12706. PMLR.

A Experimental Details

RoBERTa-large contains 354 million parameters and GPT-2 has 1.5 billion parameters. There is no extra parameter added in our method. All experiments are conducted on NVIDIA A100 and GeForce RTX 3090 GPUs with CUDA. The search process in § 4 with 32-shot data takes about 2 hours with 40 GB maximum memory. The test process takes 5∼30 minutes depending on the size of $\mathcal{T}^*$ and $\mathcal{X}_{\text{test}}$. The detailed training and test set statistics for experiments in Table 2 are shown in Table 4. Our method is developed by OpenPrompt (Ding et al., 2022), an open-source prompt-learning framework based on PyTorch (Paszke et al., 2019). The models are obtained from the Huggingface Transformers library (Wolf et al., 2020).

Datasets	$\lvert\mathcal{X}_{\textbf{train}}\rvert$	$\lvert\mathcal{X}_{\textbf{test}}\rvert$
SST-2	64	872
Yelp P.	64	38000
AG's News	128	7600
DBpedia	448	70000
MRPC	64	1725
SNLI	96	10000
RTE	64	277

Table 4: Statistics of the training and test set for experiments in Table 2.

B Efficiency Analysis

The results reported in § 4 all search through the whole prompt space $\mathcal{T}^*$, i.e. every combination of an instance and a prompt is covered. Since it would require up to 4 hours with a single NVIDIA A100, we seek to optimize the process by pruning the search space. Our strategy is as follows: (1) randomly sample a batch of *valid prompts* (in our experiments we use batch size 16) from $\mathcal{T}^*$ and apply them to the whole training set $\mathcal{X}_{\text{train}}$; (2) record the performance of each prompt word, i.e. if a prompt is *"it was really"* and achieves 0.8 accuracy on $\mathcal{X}_{\text{train}}$, then for each word in the prompt (*"it", "was", "really"*) 0.8 is recorded; (3) update the set of *valid prompts*; (4) repeat until there is no remaining *valid prompt*. A *valid prompt* means the average score of the three words is over a predefined threshold. In our experiments, the threshold is set to 0.7 on SST-2 dataset and achieves a mean test accuracy of 87.6%.

As shown in Table 5, with the pruning strategy, the average time cost can be greatly reduced to 10 minutes with a still satisfying performance on test data. In our experiments, prompts are randomly sampled and grouped into batches. We believe a better-designed and heuristically informed batching strategy will further boost the searching efficiency and test performance.

Method	**Time Cost**
Prompt Tuning	15.9 mins
Feature-MLP	7.0 mins
Feature-BiLSTM	9.3 mins
Black-box Tuning	10.1 mins
Ours	10.3 mins

Table 5: Training and searching time cost on SST-2. Following Table 2, our method searches on 32-shot training data and other baselines are trained on 16-shot training data and evaluates on 16-shot validation data. The max sequence length is set uniformly to 47. Our method is run for 5 times on a single NVIDIA A100 and the mean time cost is reported.

C Details of Prompts and Label Words

Table 6 displays the specific prompt format and label words used for searching for lottery prompt for each dataset. Note that for auto-regressive PLMs like GPT-2, the "<mask>" token should be placed at the end of the input.

D Analysis of Strong Prompts

Table 7 shows the top-5 prompts for each dataset and their corresponding metrics on the training set of 1000 instances. It can be seen that some words are especially contributive to correct predictions. For example, "just" and "find" for SST-2, "think" and "like" for RTE, "very" and "had" for MRPC, etc. We also get the 100 best prompts out of $\mathcal{T}$ for SST-2, and visualize the frequent words at each position, as shown in Figure 6. From the number of unique words at each position, we can conclude that words more adjacent to the "<mask>" token has a larger impact on the prediction. Most of the strong prompts for both PLMs fit with language intuition, as words like "even" and "really" naturally go with adjectives like "good" and "bad". GPT-2 demonstrates better fluency and interpretability compared to RoBERTa-large, as some high-frequency words found for RoBERTa-large like "without" are hard to comprehend.

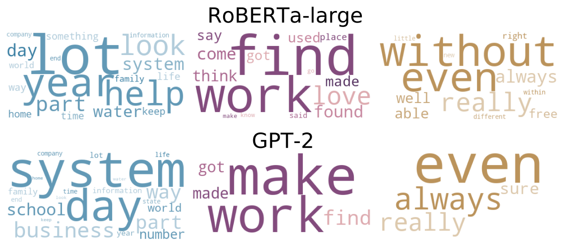


Figure 6: Frequent words of 100 top-performing prompts at each position on SST-2. The 3 positions are [NOUN], [VERB], [PREP|ADJ|ADV] from left to right.

E A Case Study of Hard Instances

Though a lottery prompt can be found for most instances in § 2, some individual cases still fail to match such prompts and some instances require a relatively large number of searches. We gather the 5 instances that require the most search numbers or observe a failure in searching from both PLMs. The examples in 3 datasets are presented in Table 8. It can be seen that for SST-2, the presented cases are intuitively difficult. Many of them involve complex logic, so naively identifying keywords will lead to definite failure in prediction. On the other hand, the hard cases in MNLI and SNLI seem more counter-intuitive. Most "entailment" cases have considerable vocabulary overlap between the first and the second sentence. The three cases that fail to match a lottery prompt among $\mathcal{T}$ are short sentences with almost identical expressions. We believe it is the negative effect from prompt template and label word chosen. For MNLI, both the two high-lighted cases contain negation auxiliaries that rarely follow a "Yes" statement. This tendency drives the PLMs to always favor the choice of "No", which leads to error prediction.

F Limitations

The current method works with a large prompt search space $\mathcal{T}$, which means a tremendous number of inference API calls are required. Though Figure 1 shows that the average cost of finding a lottery prompt for each instance is low, the searching process is highly randomized and there is no guarantee of the performance of searched prompts over the test dataset. Therefore, finding strong prompts over the training set can still be laborious. How to use PLM inference calls more efficiently and leverage the generalization ability of $\mathcal{T}^*$ within a reasonable cost is of future research interest. Our acceleration strategy can be found in Appendix B.

Dataset	Prompt	Label words
SST-2	<Text> [Prompt] <mask>	good, bad
Yelp P.	<Text> [Prompt] <mask>	good, bad
CoLA	<Text> [Prompt] <mask>	reasonable, unreasonable
SNLI	<Text1> [Prompt]? <mask>, <Text2>	Yes, Maybe, No
RTE	<Text1> [Prompt]? <mask>, <Text2>	Yes, No
MNLI	<Text1> [Prompt]? <mask>, <Text2>	Yes, Maybe, No
QNLI	<Text1> [Prompt]? <mask>, <Text2>	Yes, No
WNLI	<Text1> [Prompt]? <mask>, <Text2>	Yes, No
MRPC	<Text1> [Prompt]? <mask>, <Text2>	Yes, No
QQP	<Text1> [Prompt]? <mask>, <Text2>	Yes, No
AG's News	<Text> [Prompt] <mask>	world, sports, business, technology
DBpedia	<Text> [Prompt] <mask>	company, school, artist, athlete, politics, transportation, building, river, village, animal, plant, album, film, book
Few-NERD	<Text> <Entity> [Prompt] <mask>	water, law, broadcast/program, media/newspaper, restaurant, artist/author, film, award, park, event, government/agency, person, educational/degree, education, director, game, sports/facility, protest, car, language, airport, organization, building, location, athlete, show/organization, sports/league, geopolitical, scholar/scientist, library, hotel, road/railway/highway/transit, painting, hospital, election, written/art, religion, company, train, ship, attack/battle/war/military/conflict, sports/event, disaster, currency, weapon, living, sports/team, politician, god, political/party, music, art, actor, theater, biology, software, island, medical, disease, chemical, product, airplane, food, mountain, astronomy, soldier

Table 6: The prompt format and label words used for each dataset. [Prompt] represents the sequence of "[NOUN] [VERB] [PREP|ADJ|ADV]". For GPT-2, "<Text1> [Prompt]? <mask>, <Text2> " is changed into "<Text1> <Text2> [Prompt]? <mask>".

Dataset	Top-5 Prompts	Metrics
SST-2	he work just, I find very, I find really, help are for, she work just	85.9, 85.6, 85.2, 84.6, 84.0
Yelp P.	look place really, you place also, look was also, I were very, they place also	92.0, 91.3, 91.3, 91.2, 91.2
SNLI	I get really, I like through, I said always, keep love through, you found that	56.9, 56.0, 55.8, 55.8, 55.7
RTE	keep like always, way think such, life think same, end think such, end like always	60.0, 59.7, 59.6, 59.6, 59.4
MRPC	money had very, something had very, I been very, help had very, life had very	70.9, 70.5, 70.4, 70.4, 70.2
AG's News	lot say on, I said other, time think other, state say on, you think other	79.7, 78.8, 78.1, 78.0, 77.3
DBpedia	you said such, something know then, life make of, home said such, information is that	87.5, 86.6, 86.0, 85.9, 85.8

Table 7: An example of Top-5 prompts over 1000 training instances for each dataset and their individual performance on training sets.

Datasets	Instance Text		Label
SST-2	it falls far short of poetry , but		negative
	will be best appreciated by those willing to endure its extremely languorous rhythms , waiting for happiness		negative
	expiration date		negative
	gut-wrenching , frightening war scenes since " saving private ryan "		positive
	sit through – despite some first-rate performances		positive
	largely flat and uncreative		negative
	all of dean 's mannerisms and self-indulgence , but		negative
	if oscar had a category called best bad film you thought was going to be really awful but was n't		positive
MNLI	It would be nice if more of the newcomers were artists, artisans, and producers, rather than lawyers and lobbyists, but head for head, I'll stack up Washington's intellectual capital against any competitor's.	It would be nice if there were more lawyers instead of artistic people.	contradiction
	i just couldn't watch that much TV	I couldn't watch that much TV	entailment
	yeah uh well we did well we did you know we really did i mean i just don't understand these people that think taking an RV and parking it and sitting inside and watching TV and having your microwave it's not camping	I don't think it's camping if you hang out in an RV.	entailment
	Of course	Maybe.	contradiction
	I think not!	I do not think so.	entailment
	Exhibit 10 Adjustment Factors Used to Account for Projected Real Income Growth through 2010 and 2020	See Exhibit 10 for Adjustment Factors Used to Account for Projected Real Income Growth through 2010 and 2020	neutral
	In the dark of night, their aim must be true.	Their aim must be accurate in the dark, or else they will not succeed.	neutral
	now we quit that about two years ago no three years ago when we got China mugs for everybody	We stopped doing that three years ago, after we got everyone China mugs.	entailment
SNLI	Two men are playing a game of chess, one is standing and the other is sitting.	A crowd watches a concert.	contradiction
	A green jeep with men who are manning guns, with a crowd in the background on the street.	Video game fans in cosplay outfits.	contradiction
	A man has a pink ribbon around his arm.	A guy with a strip of cloth around his bicep.	entailment
	Large amounts of people walk around near a large, silver, reflective display.	People are singing.	contradiction
	Man playing the accordion on a sidewalk during the day.	The Pope speed dials.	contradiction
	People walk and bike in front of a box office.	People are carrying about their business nearby a box office	entailment
	Three naked little boys are playing in a river and are covered in mud; one is standing up.	the boys had no clothes on in the river	entailment
	A person wearing a dark blue covered up attire from head to toe, with a mask and vest, holding a thin sword.	Someone with a sword	entailment
	Four children are in an industrial kitchen looking at a recipe with the ingredients on the table in front of them.	Four people are in the kitchen	entailment
	Two guys getting a drink at a store counter.	two guys get a drink	entailment

Table 8: The most difficult instances for RoBERTa-large and GPT-2, measured by number of searches required to get the lottery prompt out of $\mathcal{T}$. Instances in purple indicate failure to find a lottery prompt for GPT-2, and instances in blue are failure instances for RoBERTa-large.