# Exploring Lottery Prompts for Pre-trained Language Models

**Yulin Chen**[1*], **Ning Ding**[1,2*], **Xiaobin Wang**[3],
**Shengding Hu**[2], **Hai-Tao Zheng**[1,4†], **Zhiyuan Liu**[2,5,6†], **Pengjun Xie**[3]
[1]Shenzhen International Graduate School, Tsinghua University [2]DCST, Tsinghua University
[3]Alibaba Group, [4]Pengcheng Laboratory, Shenzhen, [5]BNRIST, IAI, Tsinghua University
[6]Jiangsu Collaborative Innovation Center for Language Ability,
Jiangsu Normal University, Xuzhou
{yl-chen21, dingn18}@mails.tsinghua.edu.cn
{zheng.haitao}@sz.tsinghua.edu.cn, {liuzy}@tsinghua.edu.cn

## 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 their generalizability. By searching through the prompt space, we first validate the assumption that for every instance, there is almost *always* a lottery prompt that induces the correct prediction from the PLM, and such prompt can be obtained at a low cost thanks to the inherent ability of PLMs. Meanwhile, we find that some strong lottery prompts have high performance over the whole training set, and they are equipped with distinguishable linguistic features. Lastly, we attempt to generalize the searched strong lottery prompts to unseen data with prompt ensembling method without any parameter tuning. Experiments are conducted on various types of NLP classification tasks and demonstrate that the proposed method can achieve comparable results with other gradient-free and optimization-free baselines.

## 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) to wrap input instances and trigger desired output tokens. Note that in this paper, the term "prompt" technically refers to the template that wraps the original input. 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 the zero-shot setting.

Despite the promising performance of prompting, it is often accompanied by drastic fluctuations among different prompts (Zhao et al., 2021). Given the observed sensitivity and context-dependent nature of the prompting method, it is intuitive to assign distinct prompts to each instance to trigger the desired output. Intrigued by this intuition, we explore a bold hypothesis:

> *Is it possible to find at least one instance-level prompt that induces correct output for every data point (lottery prompt) in classification tasks without any optimization?*

We empirically show that after building an automatic searching procedure with reasonable searching space on 13 representative classification datasets of up to 66 classes, **the existence of such lottery prompts can be validated (§ 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 updates our

[*] equal contributions
[†] corresponding authors



*Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*
*Volume 1: Long Papers*, pages 15428–15444

recognition of the limit of prompted knowledge in PLMs and demonstrates a promising upper bound of the PLMs' inference capability.

With the hypothesis verified, we conduct further analysis on the internal mechanisms and properties of lottery prompts to explore how the lottery prompts relate to model capability and how lottery prompts generalize to unseen data without any optimization.

(1) We first find that the search cost of lottery prompts is low for most datasets (under 30 API calls), and could reflect task difficulty and model capacity (§ 3.1). Search success rate increases and search cost decreases for larger PLMs and PLMs pre-trained for more steps, demonstrating that lottery prompts are a unique consequence of the expanded model capacity, rather than a mere stroke of luck. (2) Among these lottery prompts, we also find that there are a number of "strong prompts" that perform non-trivially on the whole training set, and interpretable linguistic features can be identified among them (§ 3.2). Strong prompts demonstrate considerable potential to be generalized to unseen data, i.e., test dataset, of the current task. We develop a mutual-information-based prompt ensembling method and show that strong prompts could be effectively generalized to unseen data in an optimization-free manner (§ 4). Without any parameter update, the ensembling of strong prompts could achieve on-par or better performance with many competitive baselines.

In summary, we validate the existence of lottery prompts and conduct an in-depth analysis of the properties of lottery prompts. We also show that by directly ensembling the strong prompts, prominent performance can be achieved on test data without any optimization. Our study points to the great potential of PLMs and is hoped to inspire future works in more efficient ways in searching and ensembling lottery prompts as an optimization-free adaptation of PLMs.

## 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 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 experiment.

### 2.1 Overview and Setup

Particularly, for every input instance in a classification task, we attempt to search through the prompt space and find a textual prompt that can make PLMs produce desired label words. 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[1] 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 $|\mathcal{T}| = 76725$. The major concerns of such designing is to restrict the prompt space and to fit with common syntactic order of words to ensure prompt plausibility to some extent. As for verbalizers, we follow the standard design of previous works (Sun et al., 2022). We use RoBERTa-large (Liu et al., 2019) and GPT-2 (Radford et al., 2019) as the backbones. The specific prompt format and verbalizers used are shown in Appendix C.

[1] https://sketchengine.co.uk

### 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

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

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.

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

$T(\cdot)$ means a wrapped input with empty string and $q$ is the corresponding output probability over the label words. $p$ 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 $y$.

### 2.3 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. The effect of pre-training will be further explored in Section 3.1, demonstrating that lottery prompts are a unique and consequent effect along with language model pre-training.

| 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 success rate (%) of lottery prompt search for each dataset's 1000 randomly sampled data. WNLI uses the whole training set with 635 instances.

## 3 Empirical Analysis

Since we have verified the existence lottery prompts, in this section, we conduct further analysis on search cost and the searched lottery prompts.

### 3.1 Search Cost Analysis

As aforementioned, the searching space in our experiment is $|\mathcal{T}| = 76725$, however, the practical cost to find a lottery prompt for one data point is significantly lower than the budget. As shown in Figure 2, the average search cost for each instance does not exceed 30 API calls on most datasets for both PLMs. In this section, we show that search cost correlates with data difficulty and model capacity with further analysis.

**Task Difficulty.** As shown in Figure 2, searching for a lottery prompt for a multi-class classification problem is more costly. The 66-class typing dataset Few-NERD requires a significantly higher search budget 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. As mentioned in the next part, it may be attributed to the designing of prompt format and label words. Meanwhile, NLI tasks with mixed domains are probably the most difficult

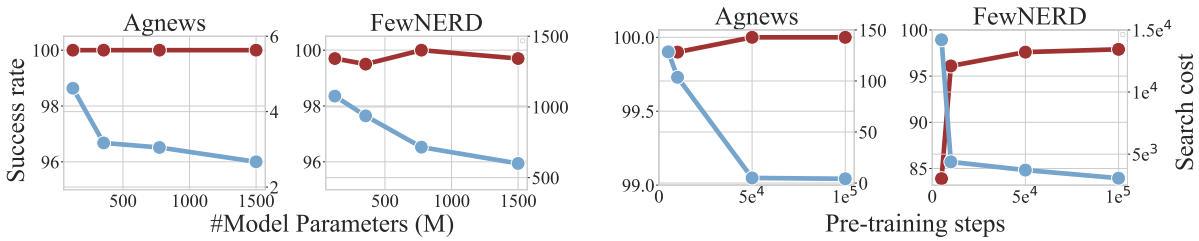


Figure 1: The change of search success rate and the mean search cost along with the model size (left) and the pre-training steps (right). Experiments are conducted with GPT-2, GPT-2-medium, GPT-2-large, GPT-2-xl, and RoBERTa-base pre-trained with 5000, 10000, 50000, and 100000 steps.

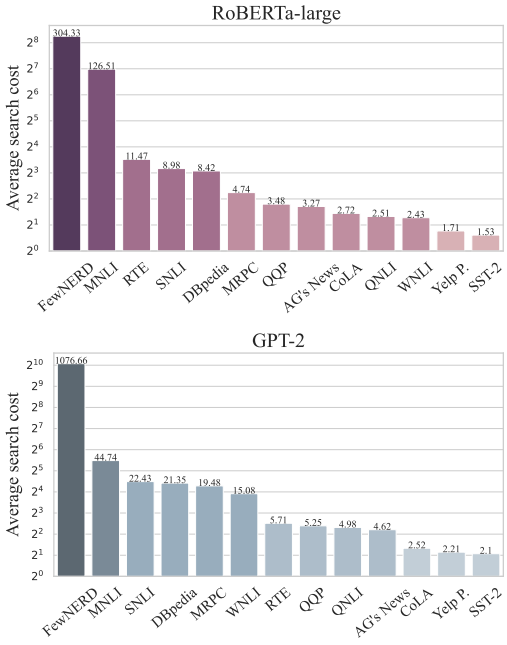


Figure 2: The average search cost (number of API calls for each instance) on each dataset.

sentence-pair tasks, given that MNLI, RTE, and SNLI are more costly than paraphrase tasks and other domain-specific NLI datasets. Comparing across 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**.

**Hard Instances.** Beyond task difficulty, we are also interested in some of the hard instances, i.e. instances that require a significant number of searches or fail to match any lottery prompt in the given search space. We gather the 5 instances that require the most searches or ultimately observe a failure in searching from both PLMs. The examples from 3 datasets are presented in Appendix Table 8. It can be seen that for SST-2, the presented cases are intuitively difficult, as many of them involve vague expressions and complex reasoning that can be misleading. On the other hand, the hard cases in MNLI and SNLI seem more counter-intuitive. Most "entailment" cases have considerable vocabulary overlap between the premise and hypothesis statements. The three failed cases 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 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 erroneous prediction. The effect of negation has also been studied with standard PLM finetuning and proved to be a challenge (Hossain et al., 2022; Hosseini et al., 2021). The analysis shows that although for most instances, the lottery prompts can be easily found, **the prompting method is still disadvantaged when it comes to complex text that requires advanced understanding ability**. Also, prompting method is sensitive to verbalizer designs and **can be easily influenced by statistical correlation between label words and input texts.**

**Impact of Model Size and Pre-training.** To explore the effect of model capacity on the easiness to search for lottery prompts, we conduct the same searching process as described in § 2 on AG's News and FewNERD with PLMs of different scales and pre-training status. Specifically, we use GPT-2, GPT-2-medium, GPT-2-large and GPT-2-xl for model size ablation and RoBERTa-base pre-trained for 5000∼100000 steps for pre-training ablation, respectively. Figure 1 shows the variation of search success rate and average search cost per instance. For models of different scale, the success rate is similar but the search cost consistently decreases as models scale up, which shows that large PLMs generally have a larger feasible solution space for spe-

cific instances. Meanwhile, finding lottery prompts for PLM at their early pre-training stage is much harder. As the pre-training progresses, a significant reduction in search cost and increase in success rate follow. This indicates that **the existence of lottery prompts is not merely a stroke of luck, but a consequence of pre-training that expands model capacity and can be further strengthened as PLMs scale up.**

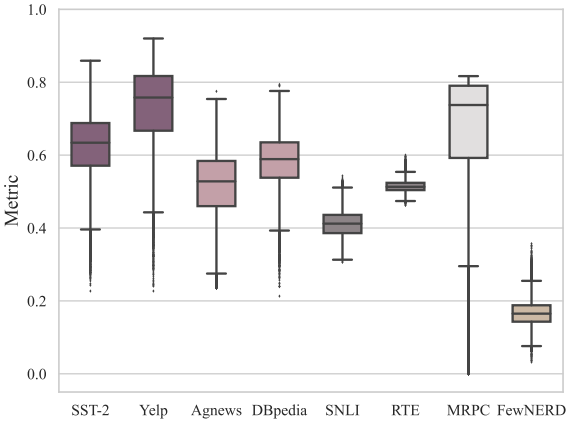


Figure 3: Prompt performance and variation on each dataset using RoBERTa-large. The vertical axis represents the metric of each prompt over $\mathcal{X}_{\text{train}}$. MRPC uses f1 metric, and others use accuracy.

### 3.2 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 perform well 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 3. **It could be concluded that for all datasets, there are a handful of "strong prompts" that can perform satisfactorily on the dataset.** Note that despite having altogether 66 classes, the best-performing prompt almost achieves an accuracy of 0.4 on Few-NERD. Meanwhile, different tasks show distinct patterns. Text classification 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.

To inspect into the linguistic characteristics of the strong prompts, we present the top-5 prompts for some of the representative datasets and their corresponding metrics on the training set of 1000 instances in Table 2. While many prompts may not seem syntactical on the whole, certain linguistic characteristics can still be identified, which fit with our language intuition, both syntactically and semantically, and reveal some of the most contributive words in prompts for distinctive datasets. For example, the top prompts for the 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 topic classification, the words like "other" and "such" naturally lead to nominal label words like "sports" and "artist". As for natural language inference task, although language entailment is subjective, it is common that personal pronouns are often involved when we express our opinions on entailment, like "I think it means", "Do you think", etc. Therefore appearance pf pronouns is in top prompts is reasonable. Meanwhile, we do observe that good prompts are not always interpretable. It may imply that the PLM's internal language ability and understanding deviates from human beings, which is why prompt engineering is important. Above all, we see that **"strong prompts" do exist and they are equipped with distinct linguistic features depending on label words and task type.**

## 4 Explore the Generalizability of Strong Prompts

In § 2 and § 3, 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, and that there are a handful of strong prompts that perform non-trivially on the training set. In this section, we first describe the ensembling method and then present the generalization results.

### 4.1 Prompt Ensembling Method

We gather a set of feasible prompts $\mathcal{T}^*$ with the searching result on $\mathcal{X}_{\text{train}}$ and conduct inference with designed prompt ensembling method for each instance in $\mathcal{X}_{\text{test}}$. 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 selected 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 presented as

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

where $p_k = p(x; T_k, \mathcal{M})$ and is calculated as equation 2, and $\Phi$ is the ensembling method used.

| 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 2: An example of Top-5 prompts over 1000 training instances for each dataset and their individual performance on training sets. The model used is RoBERTa-large.

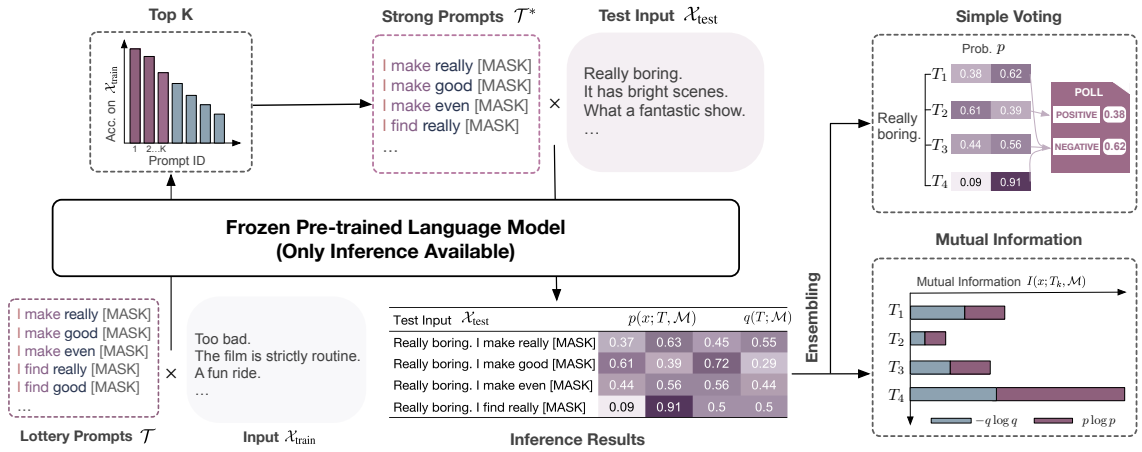


Figure 4: The complete process of searching for $\mathcal{T}^*$ and ensembling with $\Phi$.

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 take the top-k best-performing prompts over the training set as $\mathcal{T}^*$. In the experiments, we empirically use $k = 10$. A naive ensembling method is to take the average output as the final prediction by simple voting, where $\Phi(p_1, p_2, ..., p_t) = \frac{1}{t}\sum_{k=1}^{t} p_k$. While a more sophisticated strategy that echoes the spirit of "lottery prompt" is to select one most "reliable" prompt for each instance $x \in \mathcal{X}_{\text{test}}$. 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 $x$,

$$\begin{aligned} I(x;T_k,\mathcal{M}) &= H(q|T_k(\cdot)) - H(p|T_k(x)) \\ &= -\sum_i q_i(T_k;\mathcal{M}) \log q_i(T_k;\mathcal{M}) + \\ &\quad \sum_i p_i(x;T_k,\mathcal{M}) \log p_i(x;T_k,\mathcal{M}), \end{aligned} \tag{4}$$

where $q$ and $p$ are the predicted probability vectors as in Equation 2. So the overall objective is

$$\begin{aligned} T^* &= \arg\max_{T\in\mathcal{T}^*} I(x;T,\mathcal{M}), \\ \Phi(p_1,p_2,...,p_t) &= p(x;T^*,\mathcal{M}). \end{aligned} \tag{5}$$

Specifically, maximization of mutual information entails that a good prompt itself should contain no bias towards the label set, so $q$ should be close to 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 $p$. Experiments show that under few-shot settings, our mutual-information-based ensembling strategy is more advantageous than direct simple voting (§ 4.2). The complete searching and ensembling process is shown in Figure 4.

### 4.2 In-Dataset Generalization

**Experimental Setup.** We comprehensively evaluate the generalizability of strong prompts on 8 representative datasets. Following previous works (Sun et al., 2022), we conduct experiments under few-shot settings. Specifically, we choose

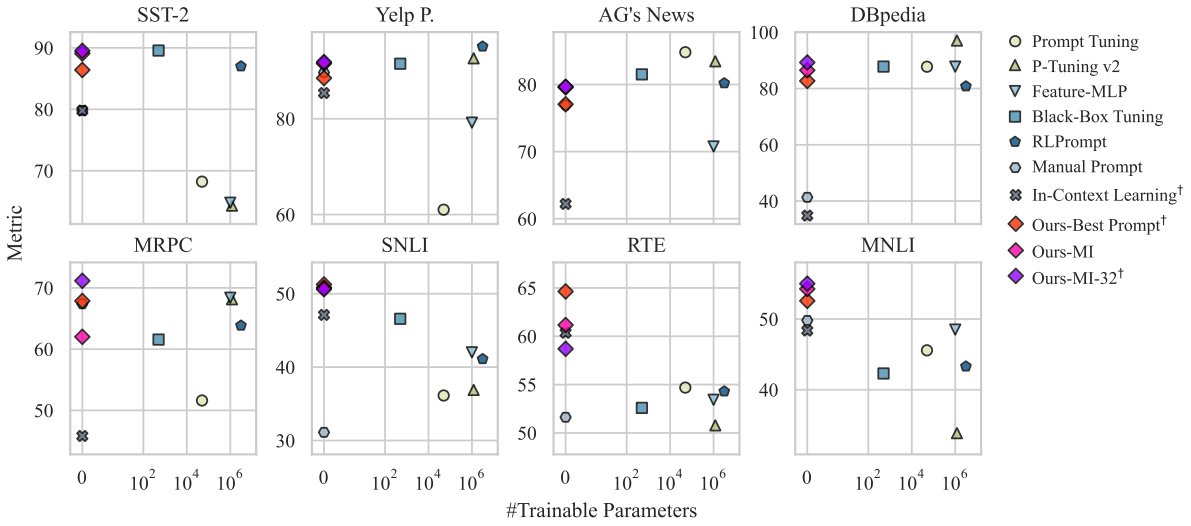


Figure 5: Performance on datasets under few-shot setting with RoBERTa-large. "Best Prompt" means directly using the top-1 performing searched prompt for test. † means using 32-shot data as training set and no extra data as validation set, and manual prompt uses no training data. Other baseline methods use 16-shot data as training and validation set. Our method uses mutual-information-based prompt ensembling method (indicated as "MI") and average results over 5 runs are reported. The baseline results mainly follow Sun et al. (2022).

the top-10 prompts as $\mathcal{T}^*$ and obtain the final prediction and test metrics with mutual-information-based ensembling as $\Phi$ on the test set. We keep the verbalizers aligned with Sun et al. (2022) for fair comparison. The description of experimental details and baselines can be found in Appendix A.

**Overall Results.** Figure 5 shows the in-dataset generalization results on each dataset. Overall, our method performs comparably to the existing baselines and requires the fewest trainable parameters. For some datasets, the searched strong prompts are shown to be more effective than baselines. 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 many 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 MI over best prompt shows that ensembling a set of strong prompts is beneficial. Comparing across datasets, our method is more advantageous for 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.

**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 with both simple voting and mutual-information-based ensembling. As shown in Figure 6, 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 data 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-based ensembling method yields higher results than simple voting. But as more training data are available, the gap is narrowed and the two ensembling strategies converge to similar levels.

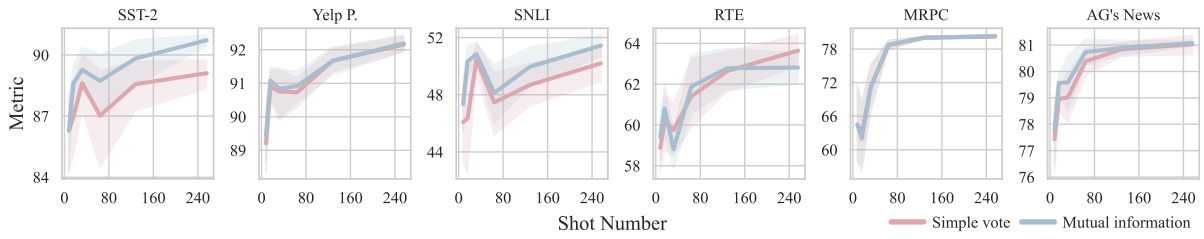


Figure 6: Average performance and standard deviation of the simple vote and mutual information under few-shot settings. We use 8, 16, 32, 64, 128, and 256-shot settings for training data. Top-10 prompts over each training set are adopted as $\mathcal{T}^*$. For each setting, the experiments are run with 5 different random seeds.

| Task | Setting | Metrics |
|---|---|---|
| *Sentiment Analysis* | SST-2 → Yelp P. | 90.27 ( 1.58 ↓) |
| | Yelp P. → SST-2 | 84.15 ( 5.37 ↓) |
| *Natural 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 strategy is used as $\Phi$.

### 4.3 Cross-Dataset Generalization

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, which can be attributed to the inconsistency in class number. MNLI is shown to be a robust dataset for NLI task and the searched prompts perform satisfactorily on both RTE and SNLI. 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 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), prompting 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; Li et al., 2022). 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 the 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 work 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 tun-

ing 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 explore the existence of lottery prompts for every single instance and the adaptation of them for various classification tasks in an optimization-free manner. We propose a large prompt space composed of common words as the search space to verify the assumption. We also identify the searched strong prompts and the relation between model capacity and search cost and demonstrate the effectiveness of ensembling the strong prompts on the test set. Our proposed optimization-free method achieves satisfactory results on various NLP tasks under few-shot settings. 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 and utilizing lottery prompts.


## Acknowledgements

This research is supported by the National Natural Science Foundation of China (Grant No.62276154 and No.62236004), Research Center for Computer Network (Shenzhen) Ministry of Education, Beijing Academy of Artificial Intelligence (BAAI), the Natural Science Foundation of Guangdong Province (Grant No. 2023A1515012914), Basic Research Fund of Shenzhen City (Grant No. JCYJ20210324120012033 and JSGG20210802154402007), the Major Key Project of PCL for Experiments and Applications (PCL2021A06), Overseas Cooperation Research Fund of Tsinghua Shenzhen International Graduate School (HW2021008), Major Project of the National Social Science Foundation of China (No. 22&ZD298), and Institute Guo Qiang at Tsinghua University. Finally, we thank Xiaozhi for providing valuable comments.


## 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 2 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.

Another aspect is that not all strong prompts are interpretable as presented in 2. While recently emerged larger models like ChatGPT demonstrate superb language understanding ability and can almost always answer yes or no questions correctly given a human-interpretable prompt. This gap observed between small PLMs like RoBERTa and large language models like ChatGPT is yet another interesting research topic.

## 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 tokens unavailable for output.

## 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.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint*, abs/2205.12548.

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.

Md Mosharaf Hossain, Dhivya Chinnappa, and Eduardo Blanco. 2022. An analysis of negation in natural language understanding corpora. In *Proceedings of ACL*, pages 716–723.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R. Devon Hjelm, Alessandro Sordoni, and Aaron C. Courville. 2021. Understanding by understanding not: Modeling negation in language models. In *Proceedings of NAACL-HLT*, pages 1301–1312.

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.

Yiyuan Li, Tong Che, Yezhen Wang, Zhengbao Jiang, Caiming Xiong, and Snigdha Chaturvedi. 2022. SPE: symmetrical prompt enhancement for fact probing. In *Proceedings of EMNLP*, pages 11689–11698.

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.

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. In *Proceedings of ICML*, pages 20841–20855.

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

## A.1 Experimental Settings

We conduct experiments under few-shot settings on 8 datasets: SST-2, Yelp P., AG's News, DBpedia, MRPC, SNLI, RTE and MNLI. We experiment under both 16-shot and 32-shot data as the training set as our method requires no validation set. The total seen labeled data number does not exceed 32-shot across all methods. The original validation set is used as the test set following (Sun et al., 2022). The detailed training and test set statistics for experiments in Figure 5 are shown in Table 4. All datasets are distributed under either CC BY license or CC BY-SA license, or subject to specified term of use. We have read and complied to the terms during experiments. The label words used follow (Sun et al., 2022) and are the same across all methods.

| Datasets | Classes | $\lvert\mathcal{X}_{\text{test}}\rvert$ |
|---|---|---|
| SST-2 | 2 | 872 |
| Yelp P. | 2 | 38000 |
| AG's News | 4 | 7600 |
| DBpedia | 14 | 70000 |
| MRPC | 2 | 1725 |
| SNLI | 3 | 10000 |
| RTE | 2 | 277 |
| MNLI | 3 | 9815 |

Table 4: Statistics of the training and test set for experiments in Figure 5.

## A.2 Baselines

We choose comparable baselines that do not update the PLM parameters, including 1) Gradient-based methods: Prompt Tuning and P Tuning v2; 2) Optimization-based methods: Feature MLP, Black-box Tuning, and RLPrompt; and 3) Optimization-free methods: Manual Prompt, and In-Context learning. The details are as follows: **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** uses 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. **RLPrompt** (Deng et al., 2022) optimizes for discrete prompts with reinforcement learning. **Manual Prompt** is a zero-shot method that directly uses a hand-crafted textual prompt for each dataset. **In-Context Learning** (Brown et al., 2020) is an optimization-free method that uses a few samples as demonstrations prepended to the test sample. Table 5 lists the features and trainable parameter number of baselines and our method.

| Method | Gradients | Tuning | #Tunable Param. |
|---|---|---|---|
| Prompt Tuning | Yes | Yes | 50K |
| P-Tuning v2 | Yes | Yes | 1.2M |
| Feature-MLP | No | Yes | 1M |
| Black-Box Tuning | No | Yes | 500 |
| RLPrompt | No | Yes | 3.1M |
| Manual Prompt$^{\ddagger}$ | No | No | 0 |
| In-Context Learning$^{\dagger}$ | No | No | 0 |
| Best Prompt$^{\dagger}$ | No | No | 0 |
| Ours | No | No | 0 |

Table 5: A summary of features of baselines methods and our method. "Gradients" refers to whether gradients of PLMs are required, and "Tuning" means whether updates of parameters are performed.

## A.3 Implementation Details

RoBERTa-large contains 354 million parameters and GPT-2 has 1.5 billion parameters. There is no extra parameter added in our method. For each dataset, the experiments are run with 5 different random seeds, and the mean metrics are reported. Most baseline results are taken from Sun et al. (2022) and Deng et al. (2022), while we re-run RLPrompt for MRPC and all baselines for MNLI with original code. All experiments are conducted on NVIDIA A100 and GeForce RTX 3090 GPUs with CUDA. The search process in § 4.2 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 $\mathcal{X}_{\text{test}}$. 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).

# B Efficiency Analysis

The results reported in § 4.2 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 prun-

ing 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 6, 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 |
| Black-box Tuning | 10.1 mins |
| Ours | 10.3 mins |

Table 6: Training and searching time cost on SST-2. Following Figure 5, 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 7 displays the specific prompt format and label words used for searching for lottery prompt for each dataset with RoBERTa-large. For autoregressive PLMs like GPT-2, the "<mask>" token are removed and the prediction of the next token by PLM will be extracted.

## D Visualization of Words in Strong Prompts

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 7. From the variation of words at each position, we can conclude that words more adjacent to the "<mask>" token has a larger impact on the prediction, which fits with our intuition. In addition, 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.

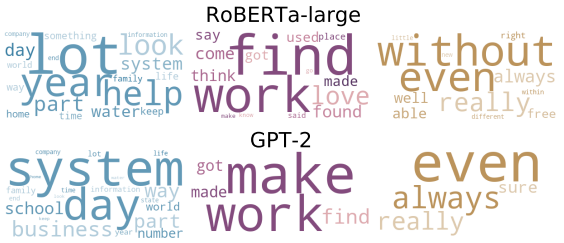


Figure 7: Frequent words of 100 top-performing prompts at each position on SST-2. The 3 positions are [NOUN], [VERB], [PREP|ADJ|ADV] from left to right.

| Dataset | Prompt | Label words |
|---|---|---|
| SST-2 | <Text> [Prompt] <mask> | great, bad |
| Yelp P. | <Text> [Prompt] <mask> | great, 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 7: 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>".

| 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.

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*In section "Limitations" at the end of the paper*

☑ A2. Did you discuss any potential risks of your work?
*In section "Ethical Considerations" at the end of the paper*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*See Abstract and Section 1 Introduction*

☒ A4. Have you used AI writing assistants when working on this paper?
*Left blank.*

### B ☑ Did you use or create scientific artifacts?

*Section 2, 3 and 4*

☑ B1. Did you cite the creators of artifacts you used?
*Section 2.1*

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*Appendix A*

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*Not applicable. Left blank.*

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*Not applicable. Left blank.*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*Not applicable. Left blank.*

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*Appendix A*

### C ☑ Did you run computational experiments?

*Left blank.*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Appendix A*

---

*The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.*

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*Appendix A*

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
*Section 4.2*

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
*Section 2.1*

**D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?**

*Left blank.*

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
*No response.*

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
*No response.*

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
*No response.*

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*No response.*

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*No response.*