Black-Box Prompt Learning for Pre-trained Language Models

Anonymous authors
Paper under double-blind review

Abstract

The increasing scale of general-purpose Pre-trained Language Models (**PLMs**) necessitates the study of more efficient adaptation across different downstream tasks. In this paper, we establish a Black-box Discrete Prompt Learning (**BDPL**) to resonate with pragmatic interactions between the cloud infrastructure and edge devices. Particularly, instead of fine-tuning the model in the cloud, we adapt PLMs by prompt learning, which efficiently optimizes only a few parameters of the discrete prompts. Moreover, we consider the scenario that we do not have access to the parameters and gradients of the pre-trained models, except for its outputs given inputs. This black-box setting secures the cloud infrastructure from potential attack and misuse to cause a single-point failure, which is preferable to the white-box counterpart by current infrastructures. Under this black-box constraint, we apply a variance-reduced policy gradient algorithm to estimate the gradients of parameters in the categorical distribution of each discrete prompt. In light of our method, the user devices can efficiently tune their tasks by querying the PLMs bounded by a range of API calls. Our experiments on RoBERTa and GPT-3 demonstrate that the proposed algorithm achieves significant improvement on eight benchmarks in a cloud-device collaboration manner. Finally, we conduct in-depth case studies to comprehensively analyze our method in terms of various data sizes, prompt lengths, training budgets, optimization objectives, prompt transferability, and explanations of the learned prompts.[1]

1 Introduction

Methods	Frozen	Black-Box	Discrete	Interpretable	Learnable
Vanilla FineTuning			N/A	N/A	N/A
GPT-3's FineTuning[2]	✓	✓	N/A	N/A	N/A
FeatureProbe (Peters et al., 2019)	✓				✓
ManualPrompt	✓	✓	✓	✓	
InContextLearning(Brown et al., 2020)	✓	✓	✓	✓	
PromptTuning (Lester et al., 2021)	✓				✓
P-Tuning v2 (Liu et al., 2021b)	✓				✓
AutoPrompt (Shin et al., 2020)	✓		✓	✓	✓
BBT (Sun et al., 2022)	✓	✓			✓
BDPL (ours)	✓	✓	✓	✓	✓

Table 1: Comparison of different tuning methods. **Frozen**: the pre-trained model is frozen and will not be updated. **Black-Box**: no access to the parameters and gradients from the pre-trained model. **Discrete**: the prompts are discrete tokens (compared with soft prompts). **Interpretable**: the prompts are readable and interpretable. **Learnable**: the prompt is parametric and learnable with explicit or estimated gradients (compared with manual prompt). **N/A**: not applicable since the corresponding descriptions are for prompt learning.

Large Pre-trained Language Models (PLMs) have demonstrated impressive versatility across a wide spectrum of downstream tasks, via either fine-tuning (FT) (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020; Zhang et al.,

---

[1] Our code and model will be released in the final version.

[2] `https://beta.openai.com/docs/guides/fine-tuning`

2020; Yang et al., 2020) or prompt-based learning (PL) (Gao et al., 2021; Liu et al., 2021c; Schick & Schütze, 2021; Li & Liang, 2021; Liu et al., 2021a). Traditionally, these two tuning paradigms are conducted at a white-box setting, where the parameters and gradients are accessible since the model is usually open-sourced and can be duplicated in user devices. Despite the fact that white-box methods have made remarkable progress, however, the increasing scale of PLMs renders this setting implausible. Nowadays, huge PLMs opt to serve users as commercial APIs deployed in the cloud, such as OpenAI GPT-3[3]. In particular, the service providers hide their model parameters and expose the query and prediction interface, which is termed the black-box setting in this paper.

Although solving NLP problems with APIs in a black-box setting is considerably challenging, it is indeed aligned with the new norm of the current interplay between the cloud infrastructure and edge devices. Specifically, from the position of cloud providers, it is reasonable to restrict the access of pre-trained model parameters since commercial, ethical, legal, security, and other concerns might be raised (Bommasani et al., 2021). First, under the white-box setting, the weaknesses and biases rooted in the underlying PLMs are at higher risk of being misused for harmful purposes. Second, the centralizing nature of PLMs exposes them to potential attacks to cause a single-point failure (Krishna et al., 2019). To this end, the black-box setting is more convincing to secure the cloud infrastructure from being condemned. As for the interests of user devices, the black-box paradigm grants them a more economical option. Otherwise, if we have access to the model's gradients, it requires transmitting gradients from cloud to device, causing high transmission costs.

With this basic setting in mind, we further elaborate on a more pragmatic scenario, i.e., the discrete prompt learning under the black-box constraint. Particularly, we opt for the prompt learning mechanism instead of the fine-tuning counterpart, partially due to the fact that the prompt learning is more cost-effective by tuning fewer parameters and can eliminate the gap between pre-training and downstream transfer. Moreover, black-box fine-tuning[4] requires users to upload their private labels and save the fine-tuned model on the server, which strongly relies on the cloud provider as the single-point trust for the security of private data and model. On the contrary, our black-box prompt learning allows users to store the private label and tune the prompt locally, preventing potential data leakage and protecting the users' commercial interests. For instance, in our setting, each user device can query the output from the cloud and then update its prompts separately on its own data.

It is noteworthy that we optimize discrete prompts, which are more interpretable to power users from different backgrounds to develop their own applications with PLMs. On the contrary, continuous prompt learning methods, e.g., BBT[5] (black box tuning (Sun et al., 2022)), are difficult to interpret their learned prompts. Moreover, these methods fail to be directly applied to prediction APIs because APIs only accept discrete inputs (e.g., GPT-3). However, the discrete nature of our BDPL allows commercial prediction APIs to directly take the learned prompt tokens without pain. Overall, our established Black-Box Discrete Prompt Learning (BDPL) is closely in accordance with the recent progress of huge PLMs, whose comparisons with the existing settings can be found in table 1. As shown in the table, BDPL can be specified under the constraints that the PLMs are frozen and both their parameters and gradients are invisible and share the virtue of optimizing discrete, interpretable, and learnable prompt tokens simultaneously.

To embrace this paradigm shift of tuning PLMs, we design a policy gradient inspired framework that can be optimized without relying on the parameters and gradients of the pre-trained models. Specifically, we characterize the prompt learning procedure as a discrete token selection problem, where the proper prompt tokens are sampled according to a categorical distribution. Because the API's parameters are invisible and gradients cannot be back-propagated, the categorical distribution needs to be optimized by some gradient-free algorithms. We resort to the policy gradient algorithm to estimate the gradients by no back-propagation. Moreover, to eliminate the high variance issue of policy gradient, we adopted a variance-reduced policy gradient estimator.

Experimental results on two kinds of datasets, *i.e.*, datasets without domain-shift and datasets with domain-shift, demonstrate the effectiveness of the proposed black-box discrete prompt learning, which significantly improves the performance over a generic pre-trained model and outperforms all baseline models on eight datasets. The results confirm that incorporating black-box prompt learning for pre-trained models is an effective and efficient solution to the PLM adaptation. We also present further analyses by investigating the effects of different training data sizes, prompt lengths, training budgets, and objectives. Our analyses demonstrated the robustness, scalability, and transferability of the proposed method.

The contributions of our work are as follows:

---

[3] `https://openai.com/api/`

[4] `https://beta.openai.com/docs/guides/fine-tuning`

[5] Our work was done concurrently with BBT. Our work studies this topic from different perspectives (discrete prompt learning with API calls) and achieves better results.

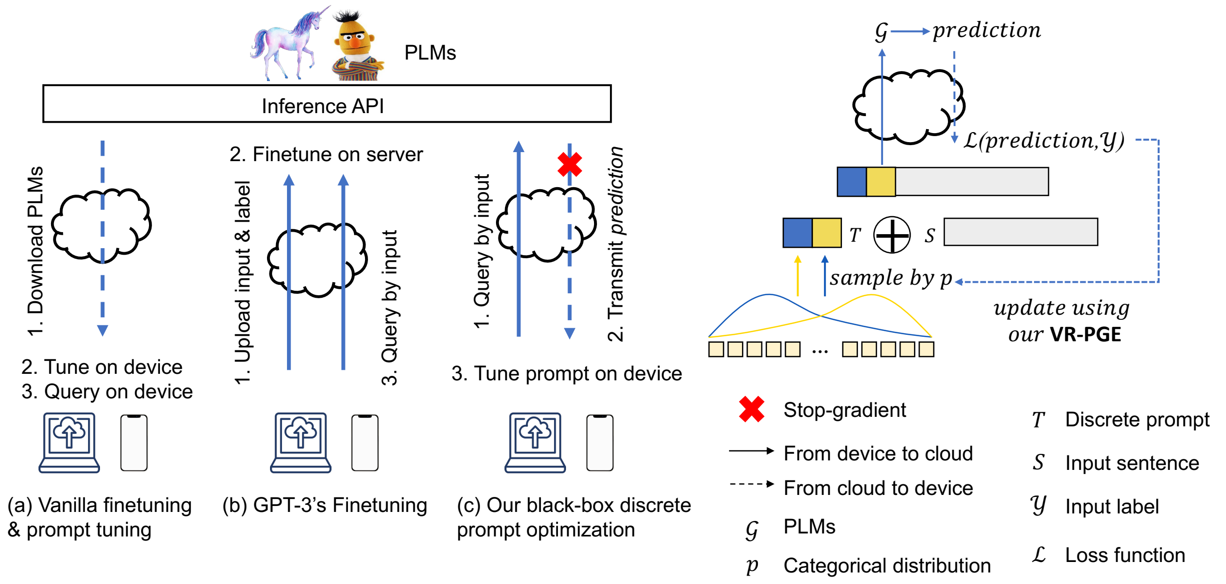


Figure 1: Schematic illustrations of the comparisons across various tuning paradigms and the cloud-device interplay at the training phase of our algorithm. **Left:** (a) Vanilla finetuning and prompt tuning can be conducted on user devices in a white-box manner since the PLMs are feasible to be duplicated at user devices. After tuning, users can still access services of PLMs on the device. (b) Increasing scale hinders the democratizing of PLMs. In GPT-3's finetuning, users have to upload the input and associated labels to the server. After finetuning, the model is saved on the server. (c) In our black-box discrete prompt learning setting, users send queries to the server and then rely on PLMs' predictions to update their discrete prompts on the devices using a gradient-free optimization. **Right:** In our framework, the user query is created by concatenating the discrete prompt and the input sentence, where the prompt tokens are sampled from their categorical distributions respectively. After calculating the loss between the PLMs' predictions and input labels, we apply a variance-reduced policy gradient algorithm to estimate the gradient of the categorical distribution and update it accordingly.

- We propose a new setting called black-box prompt learning, where we only have access to the output of prediction APIs without the need to access the PLM's parameters or gradients. The black-box prompt is optimized without the requirements of tuning pre-trained models, saving the fine-tuning costs.
- We propose a new black-box discrete prompt learning (BDPL) method to solve this new problem, and demonstrate its effectiveness in dealing with domain shifts on various tasks.
- We conduct comprehensive analyses on eight benchmark datasets under cloud-device collaboration settings, demonstrating its effectiveness for commercial APIs. BDPL has a much wider range of applications than previous methods, such as transfer learning, model personalization, and decentralized training.

2 Approach

In our setting, the input is a sentence $S = s_1 s_2 \cdots s_l \cdots s_m$ with s_l indicating the l-th token and the output corresponds to its category y. Our goal is to learn n discrete prompt tokens $T = t_1 t_2 \cdots t_i \cdots t_n = \mathcal{V}[j_1]\mathcal{V}[j_2]\cdots\mathcal{V}[j_i]\cdots\mathcal{V}[j_n]$, which are prepended to the input sentence to create the user query $[T, S]$. Note that $\mathcal{V}$ represents the vocabulary list consisting of a total of N tokens, and $t_i = \mathcal{V}[j_i]$ is the i-th token in T and j_i-th token in $\mathcal{V}$.

The overall architecture is shown in Figure 1. During the black-box training, we freeze the prediction model $\mathcal{G}$ with a stop-gradient strategy, and only optimize the discrete prompts T. Here, we assume independent categorical distribution for each prompt index $j_i \sim \text{Cat}(\boldsymbol{p}_i)$, where the random variable j_i is sampled with the probability distribution $\boldsymbol{p}_i = [p_{i,1}, \cdots, p_{i,N}]$ over the N token indexes, where $\boldsymbol{p}_i \in \mathcal{C}$ and $\mathcal{C} = \{\boldsymbol{p} : \|\boldsymbol{p}\|_1 = 1, 0 \preceq \boldsymbol{p} \preceq 1\}$. Since $\boldsymbol{p}_i$ is independent of each other, the joint probability of the whole discrete prompt is $P(T) = \Pi_{i=1}^{n} P(t_i) = \Pi_{i=1}^{n} p_{i,j_i}$.

Because the prediction model's parameters are invisible and gradients cannot be back-propagated to the prompts, it is no longer possible to directly update the prompts by back-propagating through $\nabla_{\boldsymbol{p}_i}\mathcal{L}(\mathcal{G}([T, S], y))$, where y is the label. Inspired by the policy gradient algorithm in discrete optimization, we resort to estimating the gradients by no back-propagation to accomplish **black-box** training.

For abbreviation, we denote the $\mathcal{L}(\mathcal{G}([T, S], y))$ as $\mathcal{L}(T)$ since S, y can be deemed as constants here. By the virtue of the policy gradient estimator (PGE), we can optimize the loss function via forward propagation with:

$$\mathbb{E}_T\left[\mathcal{L}(T)\right] = \int \mathcal{L}(T) P(T)\,\mathrm{d}T, \tag{1}$$

and estimate the gradient of $\boldsymbol{p}_i$ by:

$$\begin{aligned}
\nabla_{\boldsymbol{p}_i}\mathbb{E}_T\left[\mathcal{L}(T)\right] &= \int \mathcal{L}(T)\nabla_{\boldsymbol{p}_i} P(T)\,\mathrm{d}T \\
&= \int \mathcal{L}(T)\frac{P(T)}{P(T)}\nabla_{\boldsymbol{p}_i} P(T)\,\mathrm{d}T \\
&= \int P(T)\mathcal{L}(T)\nabla_{\boldsymbol{p}_i}\log P(T)\,\mathrm{d}T \\
&= \mathbb{E}_{P(T)}\left[\mathcal{L}(T)\nabla_{\boldsymbol{p}_i}\log \Pi_{j=1}^{n} P(t_j)\right] \\
&= \mathbb{E}_{P(T)}\left[\mathcal{L}(T)\nabla_{\boldsymbol{p}_i}\log P(t_i)\right]
\end{aligned} \tag{2}$$

The j-th component of $\nabla_{\boldsymbol{p}_i}\log P(t_i)$ could be solved explicitly by:

$$\nabla_{p_{i,j}}\log P(t_i) = \nabla_{p_{i,j}}\log p_{i,j_i} \tag{3}$$

When $j = j_i$, it is obvious that $\nabla_{p_{i,j}}\log P(t_i) = \frac{1}{p_{i,j_i}}$. When $j \neq j_i$, equation (3) is calculated by:

$$\begin{aligned}
\nabla_{p_{i,j}}\log P(t_i) =& \nabla_{p_{i,j}}\log(1 - \sum_{k=1, k\neq j_i}^{N} p_{i,k}) \\
=& -\frac{1}{1-\sum_{k=1,k\neq j_i}^{N} p_{i,k}} \\
=& -\frac{1}{p_{i,j_i}}
\end{aligned} \tag{4}$$

However, consistent with previous policy gradient applications (Sutton et al., 1999; Rezende et al., 2014; Jang et al., 2017; Zhou et al., 2021), we observed that conventional PGE suffers from high variance, which makes it challenging to converge in practice. Therefore, we adopted a variance-reduced policy gradient estimator (VR-PGE) as described in Williams (1992); Dong et al. (2020); Zhou et al. (2021). The estimated gradient is calculated by:

$$\boldsymbol{g}_{\boldsymbol{p}_i}^{vr} = \frac{1}{I-1}\sum_{k=1}^{I}\left(\mathcal{L}(T^{(k)}) - \frac{1}{I}\sum_{j=1}^{I}\mathcal{L}(T^{(j)})\right)\nabla_{\boldsymbol{p}_i}\log P(t_i) \tag{5}$$

where $T^{(k)}, k = 1, \cdots, I$ are sampled independently from $P(T)$.

Thus, the prompt token distribution $\boldsymbol{p}_i$ can be updated by a projected stochastic gradient descent algorithm:

$$\boldsymbol{p}_i \leftarrow \mathrm{proj}_{\mathcal{C}}(\boldsymbol{p}_i - \eta\cdot\boldsymbol{g}_{\boldsymbol{p}_i}^{vr}), i = 1, \cdots, n \tag{6}$$

where η is the learning rate of prompt learning, I is the sample size, and $\mathrm{proj}_{\mathcal{C}}$ is the projection calculation (details are presented in the Appendix).

Here we introduce the detailed training procedure for updating the prompts using our proposed VR-PGE, whose mini-batch version is displayed in Algorithm 1. Assuming the input data is divided into B batches, and within each batch, we will perform I iterations of sampling to reduce the variance of estimation. Specifically, at the k-th iteration within each batch, we first sample the sequence of prompt tokens $T^{(k)} = \mathcal{V}[j_1^{(k)}]\mathcal{V}[j_2^{(k)}]\cdots\mathcal{V}[j_n^{(k)}]$ according to the joint distribution $P(T)$. When $T^{(k)}$ is created, we will prepend it to the input sentence S and feed the query $[T^{(k)}, S]$ into the black-box pre-trained language model $\mathcal{G}$, which will return back the prediction. In light of the model prediction and ground-truth label Y, we then calculate the loss $\mathcal{L}(\mathcal{G}[T^{(k)}, S], Y)$. Then the estimated gradients $\boldsymbol{g}_{\boldsymbol{p}_i}^{vr}$ for each p_i could be obtained by executing Equation (5) after sampling all I prompt sequences for the training batch. Finally, the categorical distributions are updated by Equation (6).

Algorithm 1 The black-box discrete optimization procedures.

```
Require: Input batch $S$, Label batch $Y$, Parameter of categorical distribution $\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n$, Prediction model $\mathcal{G}$, Loss function $\mathcal{L}$.
for $k \leq I$ do
    Sample $j_1^{(k)} \sim \mathrm{Cat}(\boldsymbol{p}_1), \cdots, j_n^{(k)} \sim \mathrm{Cat}(\boldsymbol{p}_n)$
    $T^{(k)} = t_1^{(k)} \cdots t_n^{(k)} = \mathcal{V}[j_1^{(k)}] \cdots \mathcal{V}[j_n^{(k)}]$
end for
$\mathcal{L}_{\text{avg}} = \frac{1}{I}\sum_{k=1}^{I} \mathcal{L}(\mathcal{G}[T^{(k)}, S], Y)$
for $i \leq n$ do
    $\boldsymbol{g}_{\boldsymbol{p}_i}^{vr} = \frac{1}{I-1}\sum_{k=1}^{I} \nabla_{\boldsymbol{p}_i} \log P(t_i^{(k)})(\mathcal{L}(\mathcal{G}[T^{(k)}, S], Y) - \mathcal{L}_{\text{avg}})$
    $\boldsymbol{p}_i \leftarrow \mathrm{proj}_{\mathcal{C}}(\boldsymbol{p}_i - \eta \cdot \boldsymbol{g}_{\boldsymbol{p}_i}^{vr})$
end for
return $\boldsymbol{p}_1, \cdots \boldsymbol{p}_n$
```

Vocabulary Construction A natural question is how to construct the vocabulary V and what is the size N. Inspired by the observation in Diao et al. (2021), which revealed the importance of domain-specific and task-specific words and ngrams in representation learning, we introduce such important ngrams as prompt candidates. Therefore, we adopt pointwise mutual information (PMI) to construct the vocabulary of candidate prompt tokens in an unsupervised way. For each sentence in the training set, we calculate the PMI by

$$\mathrm{PMI}(\bar{x}, \widetilde{x}) = \log \frac{p(\bar{x}\widetilde{x})}{p(\bar{x})p(\widetilde{x})}, \tag{7}$$

where $\bar{x}$ and $\widetilde{x}$ are two adjacent words in the sentence, and $p(x)$ is the probability of an n-gram x. If the PMI score between these two adjacent words is high, they have a high probability of co-occurrence and are more likely to form an n-gram, suggesting they are good collocation pairs. If the PMI score is lower than a threshold σ, a delimiter is inserted between $\bar{x}$ and $\widetilde{x}$. As a result, the sentence will be segmented by several delimiters. Finally, we obtain a list of ngrams V by extracting those consecutive words after segmentation and with a frequency of at least f. As for the size N, we observe that large N will cause an unstable optimization process and even divergence. Therefore, we keep N between 50 and 200.

3 Experimental Settings

In this section, we first introduce the datasets and evaluation metrics (§3.1), followed by the baseline models (§3.2). Lastly, we describe the implementation details (§3.3).

3.1 Datasets and Evaluation Metrics

In order to examine the model's ability in regular classification tasks as well as domain-specific classification tasks, we include four datasets from the GLUE benchmark (Wang et al., 2019): MRPC (Dolan & Brockett, 2005), CoLA (Warstadt et al., 2019), QNLI (Wang et al., 2019), RTE (Dagan et al., 2005; Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), and four domain-specific datasets: CitationIntent (Jurgens et al., 2018), SciERC (Luan et al., 2018), RCT (Dernoncourt & Lee, 2017), HyperPartisan (Kiesel et al., 2019) from specific domains including computer science, biomedical science and news following Gururangan et al. (2020); Diao et al. (2021). The statistics of these datasets are shown in Table 2. Considering the data sparsity issue and large query costs[6] in cloud-device collaboration, we conduct our experiments on a popular and more realistic setting — few-shot learning, where huge models have shown their powerful ability (Brown et al., 2020). We follow Perez et al. (2021) to simulate a true k-shot learning setting. We randomly sample k data from the original training set for each class to construct the training set and another different k data to construct the validation set. The original validation set will be used as the test set. Because the size of the RCT validation set and test set is too large (30K), we randomly sample 1K data to save costs. We adopt Matthews Correlation Coefficient for CoLA, F1-score for MRPC, CitationIntent, SciERC, HyperPartisan, RCT, and accuracy for RTE, QNLI, SST-2, IMDB, CR, MR, and MPQA following Wang et al. (2019); Diao et al. (2021).

---

[6] For example, only one training epoch on 10, 000 sentences with 300, 000 tokens will cause 6 USD costs for GPT-3-Davinci. Not to mention tens of hundreds of rounds of training.

Dataset	**|L|**	**|Train|**	**|Dev|**	**|Test|**	**Type**	**Metrics**	**Domain**
Generic Domain Tasks							
MRPC	2	3.7K	408	1.7K	paraphrase	F1	news
CoLA	2	8.6K	1K	1K	acceptability	Matthews corr.	misc.
QNLI	2	105K	5.5K	5.5K	NLI	acc.	Wikipedia
RTE	2	2.5K	277	3K	NLI	acc.	news, Wikipedia
Domain-Specific Tasks							
CI	6	1.6K	114	139	citation intent	F1	computer science
SE	7	3.2K	455	974	relation classification	F1	computer science
RCT	5	180K	30K	30K	abstract sentence roles	F1	biomedical
HP	2	516	64	65	review helpfulness	F1	reviews

Table 2: The statistics of eight datasets in the main experiment and five datasets in transfer learning settings. CI, SE, HP denote CITATIONINTENT, SCIERC, HYPERPARTISAN, respectively. $|\mathbf{L}|$: number of classes for classification tasks.

3.2 Baselines

For GPT-3-based models, because previous white-box tuning methods and black-box continuous prompt tuning methods (e.g., BBT) cannot be applied to GPT-3, we compare our model with the following baselines.

- **GPT-3's FineTuning**[7]: a GPT-3 inference API that is fine-tuned entirely on a labeled dataset (black-box).
- **ManualPrompt**: a GPT-3 inference API with manually composed prompts to conduct the zero-shot evaluation. The human-written prompts are shown in Appendix A.5 (black-box).
- **InContextLearning** (Brown et al., 2020): a frozen ROBERTA-large model with a set of examples containing training sentences and labels as the input prefix, which is then prepended to the input texts (black-box).

For RoBERTa-based models, we adopt the ROBERTA-large as the backbone and the following tuning methods.

- **Vanilla FineTuning** (Liu et al., 2019): a ROBERTA-large model that is fine-tuned entirely on a labeled dataset (white-box).
- **PromptTuning** (Lester et al., 2021): a frozen ROBERTA-large model with continuous prompt embeddings prepended to the input, and learned by gradients (white-box).
- **P-Tuning v2** (Liu et al., 2021b): a frozen ROBERTA-large model with continuous prompt embeddings prepended to each layer, and learned by gradients (white-box).
- **AutoPrompt** (Shin et al., 2020): a frozen ROBERTA-large model with discrete prompts optimized based on gradient-guided search (white-box).
- **FeatureProbe** (Peters et al., 2019): a frozen ROBERTA-large model outputs the features given inputs and a newly added classification layer is trained with the gradients (white-box).
- **ManualPrompt**: a frozen ROBERTA-large model with manually composed prompts to conduct the zero-shot evaluation. The human-written prompts are shown in Appendix A.5 (black-box).
- **InContextLearning** (Brown et al., 2020): a frozen ROBERTA-large model with a set of examples containing training sentences and labels as the input prefix, which is then prepended to the input texts (black-box).
- **BBT** (Sun et al., 2022): a frozen ROBERTA-large model with continuous prompts that are optimized by covariance matrix adaptation evolution strategy (black-box).

3.3 Implementation

For GPT-3 experiments, we conduct experiments with four variants: GPT-3-Ada, GPT-3-Babbage, GPT-3-Curie, and GPT-3-Davinci. The batch size of training and evaluation is set to 4 to fulfill the query length limit (i.e., 2048). We call the APIs directly from OpenAI's services[8]. For ROBERTA-large experiments, we initialize it with pre-trained weights by Huggingface's Transformers library[9]. The batch size of training and evaluation is set to 16 and 32, respectively. The number of API calls is limited to 8000 across all datasets.

---

[7] `https://beta.openai.com/docs/guides/fine-tuning`

[8] `https://openai.com/api/`

[9] `https://github.com/huggingface/transformers`

DATASET	MRPC	COLA	QNLI	RTE	CI	SE	RCT	HP	AVG.	$COST
GPT-3 Ada										
GPT-3's FineTuning	$45.68_{4.31}$	$0.02_{0.01}$	$49.78_{2.13}$	$52.71_{1.21}$	$27.69_{3.24}$	$3.48_{0.32}$	$56.95_{4.23}$	$24.41_{1.43}$	32.59	3.74
ManualPrompt	$35.58_{2.12}$	$0.03_{0.01}$	$45.60_{1.52}$	$47.29_{2.03}$	$26.88_{1.98}$	$1.21_{0.59}$	$15.76_{2.43}$	$15.22_{1.84}$	23.44	0.17
InContextLearning	$40.52_{1.52}$	$1.25_{1.44}$	$48.75_{1.65}$	$48.74_{1.32}$	$28.18_{0.32}$	$2.52_{0.35}$	$22.65_{2.56}$	$20.04_{2.31}$	26.58	2.07
BDPL	$43.24_{2.52}$	$1.98_{0.35}$	$51.23_{0.42}$	$52.65_{1.28}$	$28.34_{1.47}$	$3.77_{0.28}$	$45.65_{2.88}$	$22.41_{1.67}$	31.16	2.45
GPT-3 Babbage										
GPT-3's FineTuning	$66.43_{1.68}$	$0.29_{0.11}$	$50.94_{0.21}$	$52.34_{1.02}$	$5.19_{0.41}$	$4.14_{1.02}$	$61.14_{5.24}$	$33.26_{1.32}$	34.22	5.60
ManualPrompt	$62.38_{3.15}$	$0.23_{0.98}$	$48.78_{1.42}$	$51.21_{0.58}$	$31.44_{2.77}$	$1.74_{0.52}$	$21.67_{2.33}$	$27.21_{1.52}$	30.58	0.21
InContextLearning	$65.41_{1.67}$	$2.55_{0.01}$	$48.29_{0.88}$	$51.52_{0.37}$	$13.08_{1.52}$	$2.51_{0.86}$	$36.71_{1.82}$	$32.17_{1.40}$	31.53	2.59
BDPL	$67.72_{1.23}$	$2.77_{0.11}$	$52.08_{0.31}$	$53.07_{1.04}$	$40.23_{2.47}$	$3.21_{0.75}$	$45.21_{2.24}$	$30.43_{2.27}$	36.84	3.07
GPT-3 Curie										
GPT-3's FineTuning	$76.30_{2.11}$	$3.41_{1.27}$	$48.97_{1.27}$	$54.51_{1.74}$	$28.37_{1.90}$	$5.07_{0.82}$	$50.55_{1.30}$	$43.32_{1.54}$	38.81	28.02
ManualPrompt	$73.28_{1.21}$	$2.01_{0.57}$	$47.17_{0.92}$	$44.04_{1.18}$	$19.18_{1.34}$	$2.77_{0.34}$	$31.04_{1.42}$	$37.14_{1.47}$	32.08	0.87
InContextLearning	$80.99_{1.32}$	$2.78_{0.01}$	$46.84_{1.13}$	$46.21_{1.55}$	$15.17_{1.87}$	$4.81_{1.32}$	$50.10_{2.31}$	$38.96_{2.33}$	35.73	10.37
BDPL	$82.56_{1.07}$	$3.98_{1.29}$	$50.08_{0.77}$	$55.76_{1.34}$	$25.45_{1.84}$	$3.36_{1.79}$	$49.55_{2.38}$	$39.44_{1.59}$	38.77	12.27
GPT-3 Davinci										
GPT-3's FineTuning	$84.56_{1.31}$	$55.33_{1.45}$	$54.22_{2.45}$	$57.04_{1.21}$	$35.44_{1.65}$	$10.32_{2.31}$	$51.55_{2.71}$	$60.12_{1.76}$	51.07	280.2
ManualPrompt	$69.71_{2.13}$	$55.21_{2.44}$	$28.01_{1.32}$	$55.29_{1.91}$	$25.62_{2.01}$	$4.88_{0.97}$	$26.76_{1.88}$	$52.58_{1.44}$	39.76	8.70
InContextLearning	$82.39_{1.74}$	$56.69_{2.03}$	$17.92_{1.66}$	$56.58_{2.31}$	$30.12_{2.97}$	$9.15_{1.48}$	$44.38_{2.15}$	$55.43_{2.48}$	44.08	103.7
BDPL	$83.44_{1.42}$	$58.44_{1.38}$	$56.22_{1.48}$	$57.21_{0.79}$	$34.59_{1.95}$	$6.55_{2.06}$	$48.77_{2.47}$	$58.47_{2.37}$	50.46	122.7

Table 3: The overall performance of black-box prompt and the comparison on eight datasets with GPT-3. We report average scores across three random seeds, with standard deviations as subscripts. AVG. denotes the average score across all tasks. $COST denotes the money cost in US Dollar for calling GPT-3's API during training and inference.

For BDPL, we optimize the prompts by AdamW (Loshchilov & Hutter, 2019) for 30 epochs with a learning rate of 1×10^{-4}. The prompt length is 50, and the size of the candidate prompt list N is 100. Other hyper-parameters are detailed in the Appendix A.3.

4 Experimental Results

The overall results on eight datasets are reported in Tables 3 and 4.

We first verify our proposed method's effectiveness on a purely black-box setting with GPT-3 APIs. From Table 3, BDPL shows great superiority across eight datasets. Compared with ManualPrompt and InContextLearning, BDPL demonstrates significant improvements, which are 7.84% and 4.83% on average of Ada, Babbage, Curie, and Davinci models. BDPL also achieves comparable performance with GPT-3's fine-tuning, which requires large money costs and uploading user's data. In Babbage, BDPL even outperforms GPT-3's fine-tuning. As the experiments are conducted on the few-shot setting, where a small number of data are available to fine-tune the models' parameters, for large models like GPT-3, overfitting could be a serious problem that deteriorates the performance so that fine-tuning is inferior to BDPL. Although careful adjustment of the fine-tuning algorithm may mitigate the overfitting and improve the accuracy, it needs a lot of manual efforts and money costs, which is implausible for cloud-device collaboration. Moreover, it is observed that ManualPrompt and InContextLearning with less capable versions of GPT-3 (e.g., Ada and Babbage) fail in some challenging datasets (e.g., COLA and SE) but BDPL performs well on them. With the increase of the model capacity (from Ada to Davinci), we observed ManualPrompt and InContextLearning could also solve them, which is consistent with recent observations of large model's emergent abilities (Wei et al., 2022). From this perspective, our method offers another option in addition to increasing model size, which is an efficient solution for less capable models to perform challenging tasks.

In addition to the auto-regressive model, we also conduct additional experiments with an encode-only model, RoBERTa-large. Because the weights of RoBERTa-large are released and gradients can be leveraged, several white-box baseline models are introduced for comparison. First, our model outperforms all black-box methods, demonstrating the effectiveness of our proposed black-box discrete prompt optimization. Second, BDPL achieves comparable performance compared with white-box prompt-based methods including both discrete and continuous prompts. Given that white-

Dataset	MRPC	CoLA	QNLI	RTE	CI	SE	RCT	HP	Avg.
				White-Box Methods					
Vanilla FineTuning	78.44$_{1.31}$	20.38$_{1.91}$	53.16$_{1.82}$	55.60$_{2.51}$	37.39$_{1.65}$	23.06$_{1.62}$	45.24$_{5.23}$	55.47$_{2.33}$	46.09
PromptTuning	52.73$_{3.41}$	7.99$_{0.68}$	53.52$_{1.57}$	56.32$_{1.63}$	34.39$_{2.56}$	28.64$_{2.52}$	36.71$_{3.11}$	47.35$_{3.24}$	39.70
P-Tuning v2	62.41$_{1.98}$	8.93$_{2.68}$	51.53$_{1.34}$	53.07$_{1.72}$	31.38$_{4.23}$	24.55$_{2.47}$	35.38$_{3.88}$	55.42$_{4.17}$	40.33
AutoPrompt	63.76$_{3.10}$	5.42$_{2.31}$	50.23$_{1.32}$	52.12$_{1.56}$	27.88$_{2.94}$	21.52$_{2.54}$	29.55$_{2.53}$	40.56$_{3.77}$	36.38
FeatureProbe	68.93$_{1.65}$	15.58$_{1.22}$	50.54$_{0.23}$	54.11$_{2.48}$	22.33$_{1.98}$	20.76$_{3.62}$	31.17$_{4.65}$	60.12$_{2.59}$	40.44
				Black-Box Methods					
ManualPrompt	70.41$_{1.58}$	0.58$_{0.02}$	49.19$_{1.14}$	48.22$_{0.57}$	12.32$_{2.35}$	9.55$_{1.43}$	11.7$_{1.52}$	35.66$_{1.63}$	29.70
InContextLearning	72.10$_{2.33}$	1.11$_{0.39}$	50.81$_{0.53}$	49.31$_{2.27}$	14.56$_{1.68}$	9.19$_{1.52}$	25.81$_{1.59}$	38.52$_{2.41}$	32.67
BBT	66.42$_{3.74}$	5.52$_{2.66}$	55.41$_{3.24}$	52.59$_{2.21}$	17.40$_{5.37}$	16.42$_{0.93}$	31.71$_{1.50}$	47.19$_{4.83}$	36.58
BDPL	78.12$_{3.69}$	4.58$_{1.22}$	53.10$_{1.07}$	53.47$_{0.92}$	23.95$_{1.25}$	21.45$_{2.04}$	36.55$_{3.17}$	45.62$_{3.41}$	39.33

Table 4: The overall performance of black-box prompt and the comparison on eight datasets with RoBERTa-large. We report average scores across three random seeds, with standard deviations as subscripts. Avg. denotes the average score across all tasks.

box prompt tuning methods cannot be applied in black-box settings when the gradients are unavailable, previous black-box methods such as InContextLearning and BBT can achieve 2.97% and 6.88% improvement on average over ManualPrompt. BDPL outperforms the previous best black-box method (*i.e.*, BBT) by an average of 2.75%. Compared with BBT, our method, BDPL, not only outperforms it but is also more practical considering its discrete nature. BBT is optimizing continuous prompts and cannot be directly fed into current prediction APIs. We also notice that there is still a large gap between FineTuning and all other methods. FineTuning updates the full model with gradients and huge parameters, serving as an upper bound for all methods. Across eight tasks, it is observed that the BDPL on domain-specific datasets is as effective as on generic datasets. While it is known that domain shift introduced difficulty for models to deal with, BDPL offers an effective solution to domain-specific datasets.

5 Analysis

We analyze several aspects of BDPL, including the effects of different training data sizes, prompt lengths, training budgets, and learning objectives. In addition, we examine the transferability of our learned prompts under the transfer learning setting and the explanation of prompts. We choose GPT-3 Babbage as the backbone model in the following discussion. The details are illustrated in this section.

5.1 Ablation Study

Effects of Training Data Size First, we analyze the effects brought by four different training data sizes: 4-shot, 8-shot, 16-shot, and 32-shot. Experiments are conducted on MRPC and RCT datasets. As shown in the left part of Figure 2 (a) and (b), with the increase in training data size, the performance of FT, InContextLearning and BDPL is improved on both MRPC and RCT, which is consistent with the assumption that more data brings sufficient training. Compared with baseline models, our model achieved consistent improvement over ManualPrompt and InContextLearning, verifying its effectiveness and scalability under different data sizes.

Effects of Prompt Length It is known that prompt-based methods are sensitive to many aspects of prompts, including contexts (Jiang et al., 2020), orders (Lu et al., 2021) and lengths (Lester et al., 2021), and inappropriately designed prompts lead to bad performance. Here we study the effects of different prompt lengths on MRPC and RCT. Considering the maximum number of tokens is 2048 for the input of GPT-3 API, too many prompt tokens (e.g., more than 100) cause additional costs and even failure of queries. Therefore, we conduct experiments with length = 10, 25, 50, and 75. The results are shown in the Figure 2 (c) and (d). With the increase in prompt length, the performance increases at first and decreases when the prompt length reaches 75 in most cases. We conclude that the approximate best prompt length is 50, since a shorter prompt length limits the representation capacity while a longer prompt length might involve noises contained in the training data and is hard to optimize.

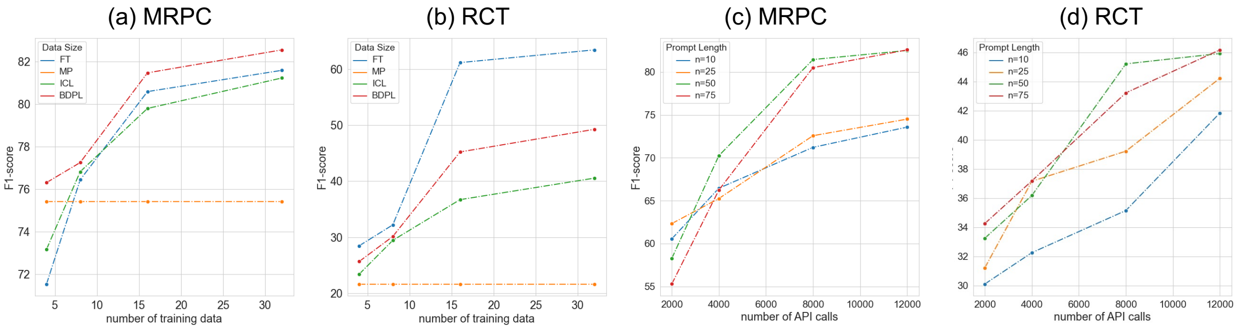


Figure 2: The effects of training data size, prompt length and number of API calls on MRPC and RCT. FT, MP, and ICL denote GPT-3's FineTuning, ManualPrompt, and InContextLearning, respectively.

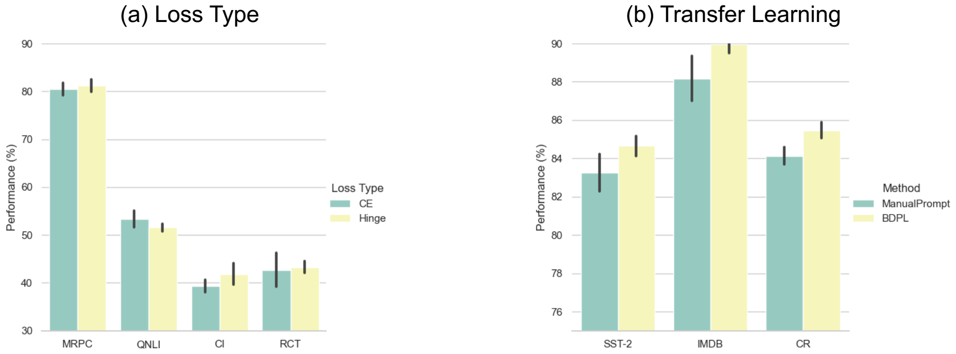
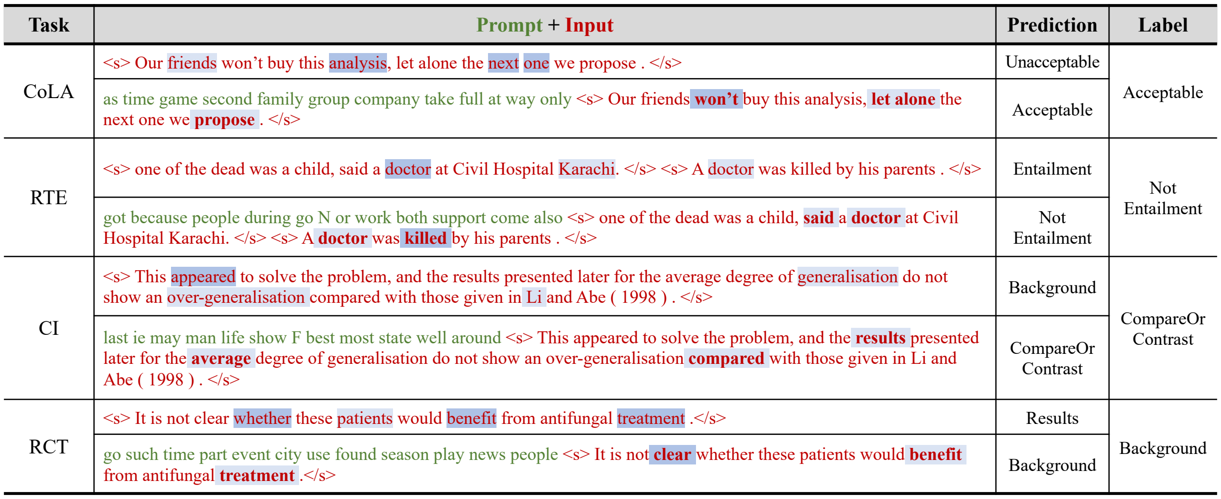


Figure 3: **(a) Ablations of loss function.** CE and Hinge represent cross-entropy loss and hinge loss, respectively. **(b) Transfer learning performance.** SST-2 is the source task while IMDB and CR are two tasks in the target domain.

Effects of Training Budgets Training budgets are essential factors for efficient-tuning methods. An efficient method is expected to achieve good performance with as few as training budgets. We measure the budgets by the number of prediction API calls and report the performance of our models with different numbers of API calls in Figure 2 (c) and (d). It is observed that with the increase of API calls, BDPL under different settings obtains performance gains because of sufficient training. All settings could converge within $12,000$ API calls. We also find that the 50-prompt performs well at first but the gap between 75-prompt and 50-prompt narrows down when the training budget grows, and finally, 75-prompt achieves competitive performance. It told us that if we do not have a sufficient training budget, it would be better to apply fewer prompt tokens, which are easier to optimize.

Effects of Different Objectives In the previous experiments, the prompts are optimized with the cross-entropy loss, and here we explore the effectiveness of our model with different objectives. With the same setting as our main experiment, we conduct further experiments with hinge loss on four datasets: MRPC, QNLI, CI, and RCT. We find that our model with both objectives can achieve comparable results. As shown in Figure 3 (a), the model with hinge loss outperforms that with cross-entropy loss on MRPC, CI and RCT, but underperforms it on QNLI. On average, our approach with hinge loss works as well as that with cross-entropy loss. It is flexible enough to work with different objectives, and we hope to extrapolate to any kinds of human-designed objectives.

Effects of Transfer Learning A critical advantage of discrete prompts is the possibility of transferring prompts learned from one task to another because discrete tokens share the same text space instead of specific latent space for continuous prompts. To verify the transferability of black-box optimized prompt tokens, we conduct experiments on three sentiment analysis datasets (*i.e.*, SST-2 (Socher et al., 2013), IMDB (Maas et al., 2011), and CR (Hu & Liu, 2004)) with GPT-3-Babbage model in the 16-shot setting. First, we use SST-2 as the source task following Vu et al. (2021) and optimize the discrete prompt tokens with our proposed BDPL. Then we obtain those selected prompt tokens and simply prepend them to the beginning of the input of the target task, IMDB and CR. Our setting assumes no training data in the target domain, so we directly test the performance in the target task. Following Wang et al. (2021), for CR, we randomly



Task	Prompt + Input	Prediction	Label
CoLA	<s> Our friends won’t buy this analysis, let alone the next one we propose . </s>	Unacceptable	Acceptable
	as time game second family group company take full at way only <s> Our friends **won’t** buy this analysis, **let alone** the next one we **propose** . </s>	Acceptable	
RTE	<s> one of the dead was a child, said a doctor at Civil Hospital Karachi. </s> <s> A doctor was killed by his parents . </s>	Entailment	Not Entailment
	got because people during go N or work both support come also <s> one of the dead was a child, **said** a **doctor** at Civil Hospital Karachi. </s> <s> A **doctor** was **killed** by his parents . </s>	Not Entailment	
CI	<s> This appeared to solve the problem, and the results presented later for the average degree of generalisation do not show an over-generalisation compared with those given in Li and Abe (1998) . </s>	Background	CompareOr Contrast
	last ie may man life show F best most state well around <s> This appeared to solve the problem, and the **results** presented later for the **average** degree of generalisation do not show an over-generalisation **compared** with those given in Li and Abe (1998) . </s>	CompareOr Contrast	
RCT	<s> It is not clear whether these patients would benefit from antifungal treatment .</s>	Results	Background
	go such time part event city use found season play news people <s> It is not **clear** whether these patients would **benefit** from antifungal **treatment** .</s>	Background	

Figure 4: Four correctly predicted examples by BDPL. We display the prompts and salience map of the token <s>. The prompts are in green and the input tokens are in red. The salient tokens are highlighted in a blue background, where the darker color denotes the more dominant weights for the prediction.

sample 2,000 instances as the test set. The results are shown in Figure 3 (b). Consistent with the previous observation in Section 4, BDPL outperforms ManualPrompt in the source task by a large margin. Moreover, learned prompts are helpful in two target tasks, demonstrating that our black-box method is robust under transfer learning settings. The experimental results display the expansion potential of *prompt transfer*, which is a promising practical application of BDPL, especially when there are N edge devices sharing a similar task, but they have no training data. We can update the black-box prompts in a general domain and then transfer them to the target domain.

5.2 Prompt Explanation

To intuitively understand the prompts, we visualize the salience maps using the Language Interpretability Tool (LIT) (Tenney et al., 2020). We choose CoLA, RTE, CI, and RCT datasets because the sentences contained in these datasets are easy for human interpretation. The comparisons between models with discrete prompts and without them are shown in Figure 4. By adding discrete prompt tokens, the model is able to find coherence-related clues. For example, CoLA aims to distinguish the acceptability of sentences. The grammar-related word ‘won’t’ and phrase ‘let alone’ dominate its prediction, which is consistent with the decision process of human beings. Similar observations are found in other datasets. Due to space limitations, more visualized examples are not shown here. Based on considerable empirical evidence, we conclude that BDPL can capture helpful information to guide the model.

We notice that most of the optimized prompts are readable but in-comprehensible, useful to model improvement, but semantically confusing to humans. Ilyas et al. (2019b) find that neural networks rely on some *non-robust* features to achieve the highest possible accuracy. These *non-robust* features are usually semantically meaningless to humans, but we can still train a well-performed model only using these features. We argue that while the optimized prompts that our method finds are meaningless to humans, they are useful for the models to make a more accurate prediction. In contrast, forcing the prompts to have semantic meaning may remove the useful information and leads to degraded performance. This observation is consistent with previous discrete prompt learning studies (Shin et al., 2020).

6 Related Work

In this section, we present the review on prompt learning for pre-trained language models and black-box optimization.

6.1 Prompts for Pre-trained Models

Large pre-trained language models are of great importance and a standard paradigm is pre-training a language model on a large unlabeled corpus and then fine-tuning the pre-trained model on different supervised tasks. This approach shows great improvements on lots of downstream tasks but there are several issues: (1) fine-tuning needs to change all the parameters of the model, which causes computational costs in terms of money and time. (2) it has to fine-tune a model for different tasks and save them separately, which is clumsy and resource-intensive. Therefore, prompt-based learning, which does not require tuning the large model, is proposed to solve the problem. Based on the format of prompts, the prompt-based learning can be categorized into two kinds: discrete prompt (Wallace et al., 2019; Shin et al., 2020; Jiang et al., 2020; Yuan et al., 2021; Haviv et al., 2021; Gao et al., 2021; Ben-David et al., 2021) and continuous prompt (Zhong et al., 2021; Qin & Eisner, 2021; Hambardzumyan et al., 2021; Liu et al., 2021c; Han et al., 2021; Li & Liang, 2021). The discrete prompt is usually a sequence of tokens or natural language phrases while the continuous prompt is designed as a sequence of vectors. However, all of these studies are limited to a white-box setting, which requires accessing all the parameters of a pre-trained model so that the gradients could be back-propagated to optimize the prompts. Recently, Sun et al. (2022) proposed black-box tuning methods but they are optimizing continuous prompts, which is impractical in real applications, because most of the commercial APIs do not accept continuous vectors as input. Our method, black-box prompt learning, provides a truly black-box solution with discrete optimization, which optimizes a set of discrete prompts without accessing the pre-trained model.

6.2 Black-Box Optimization

One of the applications of black-box optimization is the score-based black-box adversarial attack (Ilyas et al., 2018; 2019a; Huang & Zhang, 2020; Andriushchenko et al., 2020; Cheng et al., 2019), where the models are also invisible to the attacker. These studies use zeroth-order optimization methods such as natural evolution strategy (NES) (Wierstra et al., 2014) to optimize the input and increase the loss to fool the model. Instead of deteriorating the models' performance in the adversarial attack, our direction is to find the inputs that improve the accuracy. Policy gradient Sutton et al. (1999), which also belongs to the black-box optimization, is widely used in reinforcement learning to find the best policy. In contrast to NES that can only be used to search in the continuous search, policy gradient allows the choice of discrete policy and can be used to find the optimal discrete prompts. BDPL uses black-box optimization methods to find the optimal prompts, which is a novel application direction of these methods.

7 Conclusion

This paper proposes a novel setting for text categorization namely black-box prompt learning, where a large pre-trained model is invisible so that the gradients can not be back-propagated to update the prompts. Compared with the standard pre-training then fine-tuning paradigm, our approach only requires updating very few parameters. Compared with previous prompt-based methods, our approach does not require the visibility of pre-trained models, and thus it provides more flexibility in practical applications. We propose a black-box prompt learning method, BDPL, which employs a variance-reduced policy gradient estimator to approximate the gradients, and then update the prompts. Experimental results demonstrate that our approach outperforms all black-box methods and is comparable with white-box methods, illustrating the effectiveness of black-box optimization. Experiments on the transfer learning settings further show the potential of our approach in realistic scenarios, where the pre-trained model is deployed on the cloud, and the prompt learning can be implemented on each device.

References

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square Attack: a Query-efficient Black-box Adversarial Attack via Random Search. 2020.

Eyal Ben-David, Nadav Oved, and Roi Reichart. PADA: A Prompt-based Autoregressive Approach for Adaptation to Unseen Domains. *ArXiv preprint*, abs/2102.12206, 2021. URL `https://arxiv.org/abs/2102.12206`.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. In *TAC*, 2009.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`.

Shuyu Cheng, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Improving black-box adversarial attacks with a transfer-based prior. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 10932–10942, 2019. URL `https://proceedings.neurips.cc/paper/2019/hash/32508f53f24c46f685870a075eaaa29c-Abstract.html`.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pp. 177–190. Springer, 2005.

Franck Dernoncourt and Ji Young Lee. PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 308–313, Taipei, Taiwan, 2017. Asian Federation of Natural Language Processing. URL `https://aclanthology.org/I17-2052`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Shizhe Diao, Ruijia Xu, Hongjin Su, Yilei Jiang, Yan Song, and Tong Zhang. Taming pre-trained language models with n-gram representations for low-resource domain adaptation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3336–3349, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.259. URL `https://aclanthology.org/2021.acl-long.259`.

William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL `https://aclanthology.org/I05-5002`.

Zhe Dong, Andriy Mnih, and George Tucker. Disarm: An antithetic gradient estimator for binary latent variables. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/d880e783834172e5ebd1868d84463d93-Abstract.html`.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3816–3830, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.295. URL `https://aclanthology.org/2021.acl-long.295`.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pp. 1–9, Prague, 2007. Association for Computational Linguistics. URL `https://aclanthology.org/W07-1401`.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.740. URL `https://aclanthology.org/2020.acl-main.740`.

R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, volume 7, 2006.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. WARP: Word-level Adversarial ReProgramming. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4921–4933, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.381. URL `https://aclanthology.org/2021.acl-long.381`.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. PTR: Prompt Tuning with Rules for Text Classification. *ArXiv preprint*, abs/2105.11259, 2021. URL `https://arxiv.org/abs/2105.11259`.

Adi Haviv, Jonathan Berant, and Amir Globerson. BERTese: Learning to speak to BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 3618–3623, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.316. URL `https://aclanthology.org/2021.eacl-main.316`.

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

Zhichao Huang and Tong Zhang. Black-box adversarial attack with transferable model-based embedding. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=SJxhNTNYwB`.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2142–2151. PMLR, 2018. URL `http://proceedings.mlr.press/v80/ilyas18a.html`.

Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019a. URL `https://openreview.net/forum?id=BkMiWhR5K7`.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 125–136, 2019b. URL `https://proceedings.neurips.cc/paper/2019/hash/e2c420d928d4bf8ce0ff2ec19b371514-Abstract.html`.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL `https://openreview.net/forum?id=rkE3y85ee`.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020. doi: 10.1162/tacl_a_00324. URL `https://aclanthology.org/2020.tacl-1.28`.

David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406, 2018. doi: 10.1162/tacl_a_00028. URL `https://aclanthology.org/Q18-1028`.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 829–839, Minneapolis, Minnesota, USA, 2019. Association for Computational Linguistics. doi: 10.18653/v1/S19-2145. URL `https://aclanthology.org/S19-2145`.

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. Thieves on sesame street! model extraction of bert-based apis. *arXiv preprint arXiv:1910.12366*, 2019.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL `https://aclanthology.org/2021.emnlp-main.243`.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL `https://aclanthology.org/2020.acl-main.703`.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL `https://aclanthology.org/2021.acl-long.353`.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ArXiv preprint*, abs/2107.13586, 2021a. URL `https://arxiv.org/abs/2107.13586`.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *CoRR*, abs/2110.07602, 2021b. URL `https://arxiv.org/abs/2110.07602`.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. GPT Understands, Too. *ArXiv preprint*, abs/2103.10385, 2021c. URL `https://arxiv.org/abs/2103.10385`.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv preprint*, abs/1907.11692, 2019. URL `https://arxiv.org/abs/1907.11692`.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=Bkg6RiCqY7`.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. *ArXiv preprint*, abs/2104.08786, 2021. URL `https://arxiv.org/abs/2104.08786`.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3219–3232, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1360. URL `https://aclanthology.org/D18-1360`.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, 2011. Association for Computational Linguistics. URL `https://aclanthology.org/P11-1015`.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. True few-shot learning with language models. *Advances in Neural Information Processing Systems*, 34:11054–11070, 2021.

Matthew E Peters, Sebastian Ruder, and Noah A Smith. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pp. 7–14, 2019.

Guanghui Qin and Jason Eisner. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5203–5212, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.410. URL `https://aclanthology.org/2021.naacl-main.410`.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pp. 1278–1286. JMLR.org, 2014. URL `http://proceedings.mlr.press/v32/rezende14.html`.

Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2339–2352, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.185. URL `https://aclanthology.org/2021.naacl-main.185`.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4222–4235, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.346. URL `https://aclanthology.org/2020.emnlp-main.346`.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, 2013. Association for Computational Linguistics. URL `https://aclanthology.org/D13-1170`.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. Black-box tuning for language-model-as-a-service. *ArXiv preprint*, abs/2201.03514, 2022. URL `https://arxiv.org/abs/2201.03514`.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 107–118, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.15. URL `https://aclanthology.org/2020.emnlp-demos.15`.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. Spot: Better Frozen Model Adaptation Through Soft Prompt Transfer. *ArXiv preprint*, abs/2110.07904, 2021. URL `https://arxiv.org/abs/2110.07904`.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2153–2162, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1221. URL `https://aclanthology.org/D19-1221`.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=rJ4km2R5t7`.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. Entailment as Few-Shot Learner. *ArXiv preprint*, abs/2104.14690, 2021. URL `https://arxiv.org/abs/2104.14690`.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019. doi: 10.1162/tacl_a_00290. URL `https://aclanthology.org/Q19-1040`.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. URL `https://openreview.net/forum?id=yzkSU5zdwD`. Survey Certification.

Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural Evolution Strategies. *The Journal of Machine Learning Research*, 15(1):949–980, 2014.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Ze Yang, Wei Wu, Can Xu, Xinnian Liang, Jiaqi Bai, Liran Wang, Wei Wang, and Zhoujun Li. StyleDGPT: Stylized response generation with pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1548–1559, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.140. URL `https://aclanthology.org/2020.findings-emnlp.140`.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. BARTScore: Evaluating Generated Text as Text Generation. *ArXiv preprint*, abs/2106.11520, 2021. URL `https://arxiv.org/abs/2106.11520`.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 270–278, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.30. URL `https://aclanthology.org/2020.acl-demos.30`.

Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5017–5033, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.398. URL `https://aclanthology.org/2021.naacl-main.398`.

Xiao Zhou, Weizhong Zhang, Zonghao Chen, Shizhe Diao, and Tong Zhang. Efficient neural network training via forward and backward propagation sparsification. *Advances in Neural Information Processing Systems*, 34: 15216–15229, 2021.

A Appendix

A.1 Computing Infrastructure

For experiments on GPT-3, we directly call its APIs without any GPU for computation. For experiments on RoBERTa, they are conducted with NVIDIA 2080Ti GPUs with 11GB memory.

A.2 Evaluation Measures

For tasks from the GLUE Benchmark, we adopt Matthews correlation coefficient for CoLA, F1 for MRPC, and accuracy for RTE and QNLI following their original metric choices. We adopt macro-F1 for CitationIntent, SciERC, RCT, and HyperPartisan as evaluation metrics.

A.3 Bounds of Hyper-parameters

Hyper-parameter	**GPT-3**	**RoBERTa**
number of epochs	30	30
train batch size	4	32
eval and test batch size	4	16
prompt length	{10, 25, 50, 75}	
learning rate	[1e-5, 1e-3]	
dropout	0.1	
learning rate optimizer	AdamW	
loss type	{hinge loss, cross-entropy loss}	

Table 5: Bounds of hyper-parameters.

A.4 Configuration of Best Model

The configuration of the best model for each dataset is shown in Table 6.

Dataset	MRPC	CoLA	QNLI	RTE	CI	SE	RCT	HP
prompt length.	50	50	50	50	50	50	50	50
prompt learning rate	1e-4	3e-4	2e-4	1e-4	1e-4	1e-4	1e-4	1e-4

Table 6: Configuration of the best model for each dataset. The rest hyper-parameters are the default value in Table 5.

A.5 Manual Templates

The manual templates are shown in Table 7.

B Projection Calculation

The projection from $\boldsymbol{z}$ to $\mathcal{C}$ can be calculated by:

Algorithm 2 Projection from $\boldsymbol{z}$ to $\mathcal{C}$

```
Require: a vector $\boldsymbol{z}$.
Solve $v_1$ from $\mathbf{1}^\top[\min(1, \max(0, \boldsymbol{z} - v_1^*\mathbf{1}))] - 1 = 0$.
$\boldsymbol{p} \leftarrow \min(1, \max(0, \boldsymbol{p} - v_1^*\mathbf{1}))$.
output $\boldsymbol{p}$
```

Dataset	**Template**
MRPC	sentence$_1$? `[MASK]`, sentence$_2$. (yes / no)
CoLA	sentence$_1$. It was `[MASK]`. (correct / incorrect)
QNLI	sentence$_1$? `[MASK]`, sentence$_2$. (yes / no)
RTE	sentence$_1$? `[MASK]`, sentence$_2$. (yes / no)
CI	sentence$_1$. What is the intent? `[MASK]`. (background / compare / extends / future / motivation / uses)
SE	sentence$_1$. What is the relation? `[MASK]`. (compare / conjunction / evaluate / feature / hyponym / part / used)
RCT	sentence$_1$. It was `[MASK]`. (background / conclusion / method / objective / result)
HP	sentence$_1$. It was `[MASK]`. (yes / no)
SST-2	sentence$_1$. It was `[MASK]`. (great / terrible)
IMDB	sentence$_1$. It was `[MASK]`. (great / terrible)
CR	sentence$_1$. It was `[MASK]`. (great / terrible)

Table 7: Prompts and label descriptions of ManualPrompt method. Most of them are from Gao et al. (2021).

Proof. The projection from $\boldsymbol{z}$ to set $\mathcal{C}$ can be formulated in the following optimization problem:

$$\min_{\boldsymbol{p}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{p}-\boldsymbol{z}\|^2,$$
$$s.t. \mathbf{1}\top\boldsymbol{p}=1 \text{ and } 0\le p_i\le 1.$$

Then we solve the problem with the Lagrangian multiplier method.

$$\begin{aligned} L(\boldsymbol{p},v) &= \frac{1}{2}\|\boldsymbol{p}-\boldsymbol{z}\|^2 + v(\mathbf{1}^\top\boldsymbol{p}-1) && (8)\\ &= \frac{1}{2}\|\boldsymbol{p}-(\boldsymbol{z}-v\mathbf{1})\|^2 + v(\mathbf{1}^\top\boldsymbol{z}-1) - \frac{n}{2}v^2. && (9)\end{aligned}$$

with $0\le p_i\le 1$. Minimize the problem with respect to $\boldsymbol{p}$, we have

$$\tilde{\boldsymbol{p}} = \mathbf{1}_{\boldsymbol{z}-v\mathbf{1}\ge 1} + (\boldsymbol{z}-v\mathbf{1})_{1>\boldsymbol{z}-v\mathbf{1}>0} \tag{10}$$

Then we have

$$\begin{aligned} g(v) =& L(\tilde{\boldsymbol{p}},v)\\ =& \frac{1}{2}\|[\boldsymbol{z}-v\mathbf{1}]_- + [\boldsymbol{z}-(v+1)\mathbf{1}]_+\|^2\\ &+ v(\mathbf{1}^\top\boldsymbol{z}-1) - \frac{n}{2}v^2\\ =& \frac{1}{2}\|[\boldsymbol{z}-v\mathbf{1}]_-\|^2 + \frac{1}{2}\|[\boldsymbol{z}-(v+1)\mathbf{1}]_+\|^2\\ &+ v(\mathbf{1}^\top\boldsymbol{z}-1) - \frac{n}{2}v^2.\\ g'(v) =& \mathbf{1}^\top[v\mathbf{1}-\boldsymbol{z}]_+ + \mathbf{1}^\top[(v+1)\mathbf{1}-\boldsymbol{z}]_-\\ &+ (1^T\boldsymbol{z}-1) - nv\\ =& \mathbf{1}^\top\min(1,\max(0,\boldsymbol{z}-v\mathbf{1}))-1.\end{aligned}$$

It is easy to verify that $g'(v)$ is a monotone decreasing function with respect to v and we can use a bisection method solve the equation $g'(v)=0$ with solution v_1^*. Finally we have

$$\begin{aligned} \boldsymbol{p}^* =& \mathbf{1}_{\boldsymbol{z}-v_1^*\mathbf{1}\ge 1} + (\boldsymbol{z}-v_1^*\mathbf{1})_{1>\boldsymbol{z}-v_1^*\mathbf{1}>0} && (11)\\ =& \min(1,\max(0,\boldsymbol{z}-v_1^*\mathbf{1})). && (12)\end{aligned}$$

□