Subspace Selection based Prompt Tuning with Nonconvex Nonsmooth Black-Box Optimization

Haozhen Zhang[*]
School of Artificial Intelligence
Jilin University
Changchun, Jilin, China
haozhen23@mails.jlu.edu.cn

Hualin Zhang[*]
Mohamed bin Zayed University of Artificial Intelligence
Masdar, United Arab Emirates
zhanghualin98@gmail.com

Bin Gu[†]
School of Artificial Intelligence
Jilin University
Changchun, Jilin, China
Mohamed bin Zayed University of Artificial Intelligence
Masdar, United Arab Emirates
jsgubin@gmail.com

Yi Chang[†]
School of Artificial Intelligence
International Center of Future Science
Jilin University
Engineering Research Center of Knowledge-Driven
Human-Machine Intelligence
Ministry of Education
Changchun, Jilin, China
yichang@jlu.edu.cn

ABSTRACT

In this paper, we introduce a novel framework for black-box prompt tuning with a subspace learning and selection strategy, leveraging derivative-free optimization algorithms. This approach is crucial for scenarios where user interaction with language models is restricted to API usage, without direct access to their internal structures or gradients, a situation typical in Language-Model-as-a-Service (LMaaS). Our framework focuses on exploring the low-dimensional subspace of continuous prompts. Previous work on black-box prompt tuning necessitates a substantial number of API calls due to the random choice of the subspace. To tackle this problem, we propose to use a simple zeroth-order optimization algorithm to tackle nonconvex optimization challenges with nonsmooth nonconvex regularizers: the Zeroth-Order Mini-Batch Stochastic Proximal Gradient method (ZO-MB-SPG). A key innovation is the incorporation of nonsmooth nonconvex regularizers, including the indicator function of the ℓ_0 constraint, which enhances our ability to select optimal subspaces for prompt optimization. The experimental results show that our proposed black-box prompt tuning method on a few labeled samples can attain similar performance to the methods applicable to LMaaS with much fewer API calls.

CCS CONCEPTS

• **Theory of computation** → **Continuous optimization**.

[*]Both authors contributed equally to this research
[†]Joint Corresponding Author

KEYWORDS

Gradient-free optimization, black-box prompt tuning

ACM Reference Format:
Haozhen Zhang, Hualin Zhang, Bin Gu, and Yi Chang. 2024. Subspace Selection based Prompt Tuning with Nonconvex Nonsmooth Black-Box Optimization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671986

1 INTRODUCTION

In the rapidly evolving landscape of natural language processing (NLP), the advent of Large Language Models (LLMs) such as GPT-4 has marked a significant milestone. These models, trained on diverse datasets encompassing a broad spectrum of human knowledge and language nuances, have demonstrated unprecedented capabilities in understanding and generating human-like text. This has opened new avenues for applying these models to a variety of downstream tasks with task-specific tuning [5, 8, 9, 12, 20, 28, 41]. However, the traditional approach of fine-tuning LLMs for specific applications poses significant challenges, particularly in scenarios where direct access to the model's parameters is restricted or when the fine-tuning process requires substantial computational resources and memory.

Prompt-based tuning methods [19, 21, 22] have emerged as a potent alternative, offering a more parameter-efficient way to leverage LLMs for specific tasks. By carefully crafting prompts, users can guide the model's responses to fit particular applications, effectively adapting the model without retraining. This method not only preserves the model's general capabilities but also allows for task-specific customizations with a fraction of the computational cost associated with traditional fine-tuning methods. Moreover, the prompt-based approach aligns well with the emerging model-as-a-service (MaaS) paradigm, where users interact with LLMs through cloud-based APIs, avoiding the overhead of hosting and computing the models locally [36].

However, the application of black-box optimization in the context of LLMs, especially within the framework of Language Model as a Service (LMaaS), faces its own set of challenges. The high dimensionality of continuous prompts, while offering a rich space for finding effective prompts, also introduces difficulties for optimization algorithms, which tend to perform less efficiently in high-dimensional spaces. This has led to a search for methods that can effectively navigate these high-dimensional spaces without incurring prohibitive computational costs.

The concept of intrinsic dimensionality offers a ray of hope in this regard. Recent research suggests that despite the apparent high dimensionality of the embedding space, the effective dimensionality—i.e., the intrinsic dimensionality—required to capture the significant variance in prompts may be much lower. This realization opens the door to more efficient optimization strategies that focus on this lower-dimensional subspace, potentially overcoming the challenges posed by the curse of dimensionality. Recent advancements in black-box prompt tuning have shown promise in automating this selection process, using derivative-free optimization to identify optimal prompts in a low-dimensional subspace of the original high-dimensional embedding space [34, 36, 39, 43]. To be specific, instead of optimizing continuous prompts in its original high-dimensional space, these methods optimize a latent representation of the prompt in a space characterized by its low intrinsic dimensionality. Subsequently, a random low-rank matrix is employed to map this optimized latent representation back into the original prompt space. This approach significantly reduces the complexity and computational demands of the tuning process, making it more accessible and practical for a wider range of applications. Nonetheless, despite these advancements in black-box prompt tuning, a significant challenge persists: the necessity for an extensive number of API calls, which could be thousands, primarily attributed to the randomness of the subspace generation process.

To tackle the above problems, our work builds on these insights to propose a novel framework for black-box prompt tuning that leverages the intrinsic dimensionality of LLMs as well as the subspace learning and subspace selection strategy. By focusing on this reduced subspace, we aim to develop more efficient and effective methods for customizing LLMs for specific tasks, making the power of these models more accessible to users and applications in the LMaaS ecosystem. Through experimental validation across various tasks and models, we demonstrate the feasibility and effectiveness of our approach, marking a step forward in the practical application of LLMs in a world increasingly reliant on sophisticated language processing capabilities. The main contributions of this work are summarized as follows:

- We propose a novel black-box prompt tuning framework that exploits the intrinsic low-dimensional subspace of large language models (LLMs). This framework enables efficient optimization of prompts without the need for gradient-based methods, significantly reducing computational complexity and resource requirements.
- We establish the convergence rate of the zeroth-order mini-batch stochastic proximal gradient (ZO-MB-SPG) method for solving general nonconvex optimization problems with nonsmooth nonconvex regularizers. The theoretical result shows that ZO-MB-SPG has a comparable convergence rate with its first-order counterpart.
- The experimental results demonstrate that the proposed black-box prompt tuning method, even with limited label samples, can achieve performance comparable to existing methods. This is particularly noteworthy in the context of Language Model as a Service (LMaaS), where our approach significantly reduces the number of required API calls, offering a more practical solution for real-world applications.

2 RELATED WORKS

2.1 Black-Box Optimization.

Black-box optimization, also known as gradient-free optimization, refers to a class of optimization methods used to optimize objective function without requiring its gradient information. These methods are particularly useful in scenarios where the objective function is not explicitly known, is difficult to differentiate, or when gradients are not readily available or reliable. The basic idea of the gradient-free method is to approximate the true gradient using either a one-point gradient estimator (e.g., Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [15] and Natural Evolution Strategy (NES) are two types of evolution strategy-based black-box optimization algorithms which iteratively maintain the search distribution of parameters) or a two-point gradient estimator (e.g., simultaneous perturbation stochastic approximation [33] and zero-order gradient estimation [1, 11, 13, 25] use the difference of the two function values to approximate the gradient). Many modern machine learning applications are associated with zeroth-order optimization, for example, black-box adversarial attacks on deep neural networks [7], policy search in reinforcement learning [31], neural network training with forward gradient [3, 6, 16, 29] and fine-tuning language models [24].

2.2 Black-Box Prompt Learning.

The exploration of black-box prompt tuning of large language models (LLMs) has garnered significant attention in recent years. [36] propose the black-box tuning framework (BBT) to optimize the continuous prompt prepended to the input text via derivative-free optimization in low-dimensional subspaces. This approach circumvents the need for backpropagation through the LLM, significantly reducing memory consumption and computational overhead. The study demonstrates that optimizing prompts within a carefully chosen affine subspace can achieve performance on par with traditional prompt-tuning methods, underlining the potential of derivative-free optimization (DFO) methods in high-dimensional search spaces. [35] improve the framework above, which prepend continuous prompts to each layer of the LLM and introduce a gradient-free algorithm based on divide-and-conquer principles to alternately optimize prompts at different layers. [43] propose a novel approach, Black-box Prompt Tuning with Subspace Learning (BSL), which leverages meta-learning to identify optimal subspaces for prompt tuning. The study by [34] further extends the application of black-box prompt tuning into the federated learning framework, proposing a method that allows for efficient and privacy-preserving tuning of prompts across decentralized datasets. Moreover, [10] establish the Black-box Discrete Prompt Learning (BDPL) framework to optimize the discrete prompts, which describes the the prompt word choice as a reinforcement learning policy.

3 SUBSPACE LEARNING FOR PROMPT LEARNING

In this section, we briefly revisit black-box prompt tuning and then introduce the proposed subspace selection based black-box prompt tuning method. The detailed algorithm pseudocode is shown in Algorithm 1 and an illustration of our method is depicted on the right side of Figure 1.

3.1 Brief Review of Black-Box Prompt Tuning

Previous work Lester et al. [19] emphasized that a necessary number of prompt tokens for prompt tuning to reach competitive performance levels on specific downstream tasks is usually in the dozens. Considering the prompt embedding dimension of pre-trained large language models is usually one thousand. Consequently, the dimensionality of the continuous prompts employed in these models can extend into the tens of thousands. This high-dimensional representation facilitates the prompt-tuning process, allowing the model to effectively adapt to a wide range of tasks with relatively minor adjustments to its prompt embeddings.

Nevertheless, this high-dimensional framework poses challenges for black-box optimization, especially within the context of Language Model as a Service (LMaaS). Black-box optimization techniques often struggle with high-dimensional spaces, encountering what is known as the "curse of dimensionality." This phenomenon can lead to increased complexity and computational demand, making the optimization process less efficient and more resource-intensive.

Fortunately, research by Aghajanyan et al. [2], Qin et al. [27] has revealed that large-scale Pretrained Language Models (PTMs) inherently possess a low intrinsic dimensionality. This refers to a latent, lower-dimensional representation embedded within the original, high-dimensional prompt space. Such a discovery paves the way for employing derivative-free optimization algorithms that can efficiently operate within these lower-dimensional subspaces. By denoting the dimensionality of these subspaces as d (with $d \ll D$, where D represents the original high-dimensional space), we can formulate the objective of black-box prompt tuning as follows,

$$\mathbf{z}^* = \underset{\mathbf{z}\in\mathcal{Z}}{\operatorname{argmin}}\, \mathcal{L}\left(f\left(\mathbf{A}\mathbf{z} + \mathbf{p}_0; \tilde{\mathbf{X}}\right), \tilde{\mathbf{Y}}\right), \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{D\times d}$ is the random projection matrix, $\mathbf{z} \in \mathbb{R}^d$ the learnable low dimensional latent variable, $\mathbf{p}_0 \in \mathbb{R}^D$ is the initial prompt, $\mathcal{Z}$ is the search space of $\mathbf{z}$. This approach seeks to optimize latent representations within the low intrinsic dimension subspaces, offering a more practical and computationally feasible method for tuning large-scale PTMs.

As shown on the left side of Figure 1, one iteration of the approach above includes: (i) Project $\mathbf{z}$ into the prompt space using random matrix $\mathbf{A}$ and then add $\mathbf{Az}$ to $\mathbf{p}_0$. (ii) Concatenate $\mathbf{Az} + \mathbf{p}_0$ with the embedding $\tilde{\mathbf{X}}$ of input texts to obtain the final prompt embeddings. (iii) Acquire the predictions by calling the black-box API, which are used to calculate loss along with ground-truth labels. (iv) The derivative-free optimizer, such as CMA-ES, uses the loss to update $\mathbf{z}$.

3.2 Our Algorithm of Prompt Tuning

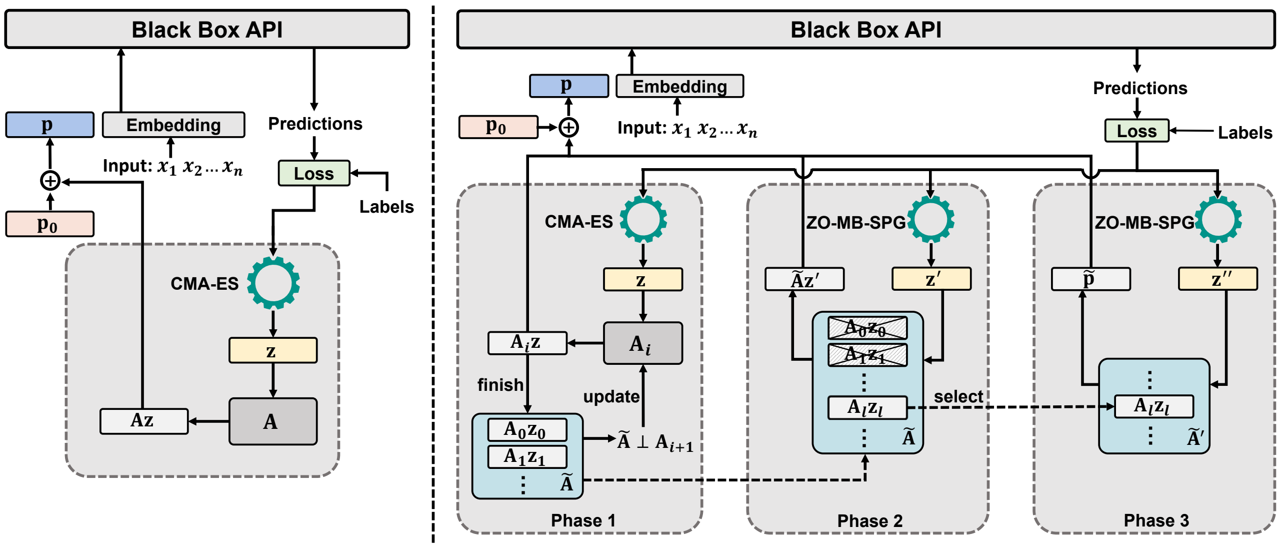
As previously noted, BBT necessitates a substantial number of API calls. To tackle this problem, we introduce our proposed method, which consists of three phases: subspace learning, subspace selection, and prompt tuning. As shown in Figure 1, the process of our method can be summarized as: (i) Learn a d' dimensional subspace represented by a project matrix $\tilde{\mathbf{A}}$. (ii) Select k dimensions from the learned d' dimensional subspace. (iii) Solve problem (1) in the k dimensional space to obtain the final prompt.

3.2.1 *Subspace Learning.* The first phase of the proposed method is subspace learning, whose goal is to learn a d' ($d' < d \ll D$) dimensional subspace spanned by $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{d'}\}$. In the initialization step, we choose a random projection matrix $\mathbf{A}_1 \in \mathbb{R}^{D\times d}$ from normal distribution $\mathcal{N}(\mu_\mathbf{A}, \frac{\alpha\hat{\sigma}}{\sqrt{d}\sigma_z})$ as suggested in [35], where $\hat{\sigma}$ is the observed standard deviation of word embeddings and σ_z is the initial standard deviation of the normal distribution of $\mathbf{z} \sim \mathcal{N}(\mu_z, \sigma_z^2)$. Both $\mu_\mathbf{A}$ and $\mu_\mathbf{z}$ are set to be $\mathbf{0}$ initially. Then, we generate $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{d'}\}$ iteratively by the following way: at i-th iteration, optimization parameter $\mathbf{z}_i$ is obtained by optimizing the loss function $\mathcal{L}(f(\mathbf{A}_i\mathbf{z} + \mathbf{p}_0; \tilde{\mathbf{X}}), \tilde{\mathbf{Y}})$ and compute $\mathbf{p}_i = \mathbf{A}_i\mathbf{z}_i$. The projection matrix $\mathbf{A}_{i+1}$ is generated manually such that $\mathbf{A}_{i+1} \perp \tilde{\mathbf{A}}$, where $\tilde{\mathbf{A}} = [\mathbf{p}_1, \ldots, \mathbf{p}_i]$. After d' iterations, we obtain a new projection matrix $\tilde{\mathbf{A}} \in \mathbb{R}^{D\times d'}$, in which each column is orthogonal to the others.

Algorithm 1 Subspace Learning for Prompt Tuning

```
Initialization: $\tilde{\mathbf{A}} = \emptyset$, a random projection $\mathbf{A}_1 \in \mathbb{R}^{D\times d}$ chosen
    from normal distribution $\mathcal{N}(\mu_\mathbf{A}, \frac{\alpha\hat{\sigma}}{\sqrt{d}\sigma_z})$, initial random prompt
    $\mathbf{p}_0 \in \mathbb{R}^D$
    Phase 1
    for $i = 1$ to $d'$ do
        $\mathbf{z}_i \in \operatorname{argmin}_{\mathbf{z}\in\mathbb{R}^d} \mathcal{L}(f(\mathbf{A}_i\mathbf{z} + \mathbf{p}_0; \tilde{\mathbf{X}}), \tilde{\mathbf{Y}})$
        $\mathbf{p}_i = \mathbf{A}_i\mathbf{z}_i$
        Update $\tilde{\mathbf{A}} = \text{Concat}(\tilde{\mathbf{A}}, \mathbf{p}_i)$
        Construct $\mathbf{A}_{i+1}$ such that $\mathbf{A}_{i+1} \perp \tilde{\mathbf{A}}$
    end for
    Phase 2
    $\mathbf{z}' \in \operatorname{argmin}_{\mathbf{z}\in\mathbb{R}^d} \mathcal{L}(f(\tilde{\mathbf{A}}\mathbf{z} + \mathbf{p}_0; \tilde{\mathbf{X}}), \tilde{\mathbf{Y}}) + r(\mathbf{z})$
    Construct $\tilde{\mathbf{A}}'$ based on $\tilde{\mathbf{A}}$ and the non-zero components in $\mathbf{z}'$
    Phase 3
    $\mathbf{z}'' \in \operatorname{argmin}_{\mathbf{z}\in\mathbb{R}^d} \mathcal{L}(f(\tilde{\mathbf{A}}'\mathbf{z} + \mathbf{p}_0; \tilde{\mathbf{X}}), \tilde{\mathbf{Y}})$
    $\tilde{\mathbf{p}} = \tilde{\mathbf{A}}'\mathbf{z}''$
Output: $\tilde{\mathbf{p}} \in \mathbb{R}^D$
```

3.2.2 *Subspace Selection.* The goal of the subspace selection phase is to select k dimensions from d' dimensional subspace learned in the first phase, to further reduce the dimensionality of the optimization variables. We consider the objective of black-box prompt tuning, then the selection subspace can be viewed as sparsifying $\mathbf{z}$. The most commonly used approach for sparsification is to add a



(a) BBT

(b) Our method

Figure 1: Overview of BBT (left) and our method (right). Phase 1 (Subspace Learning): learn a projection matrix $\tilde{\mathbf{A}}$ by generating $\mathbf{A}_i\mathbf{z}_i$ iteratively. Phase 2 (Subspace Selection): select k columns from $\tilde{\mathbf{A}}$ through a sparsity-inducing regularizer. Phase 3 (Prompt Tuning): optimize $\mathbf{z}''$ in the k dimensional subspace to obtain the prompt.

sparsity-inducing regularizer to the objective function. As a consequence, subspace selection becomes the task of solving problem (2) in the black-box setting.

$$\mathbf{z}' = \underset{\mathbf{z}\in\mathbb{R}^{d'}}{\operatorname{argmin}}\, \mathcal{L}\left(f\left(\tilde{\mathbf{A}}\mathbf{z} + \mathbf{p}_0; \tilde{\mathbf{X}}\right), \tilde{\mathbf{Y}}\right) + r(\mathbf{z}) \tag{2}$$

Each component in $\mathbf{z}$ corresponds to a $\mathbf{p}_i$ in $\tilde{\mathbf{A}}$. We propose a zeroth-order optimization algorithm named Zeroth-Order Mini-Batch Stochastic Proximal Gradient method (ZO-MB-SPG) to solve the problem above in section 4. After solving problem (2), we obtain the sparse vector $\mathbf{z}'$, where the non-zero components correspond to the selected dimensions. We retain the dimensions corresponding to non-zero components and discard the rest, i.e., we select the corresponding $\mathbf{p}_i$ from $\tilde{\mathbf{A}}$ based on the non-zero components in $\mathbf{z}'$ as the result of subspace selection, which are concatenated to construct $\tilde{\mathbf{A}}'$ and will be further utilized in the prompt tuning phase.

3.2.3 Prompt Tuning. The final phase of our method is prompt tuning, and its goal is to obtain the final prompt $\tilde{\mathbf{p}}$. We aim to solve the same objective as black-box prompt tuning in this phase, which is problem (1), but the difference is that we optimize $\mathbf{z}''$ to obtain the prompt in the k dimensional subspace provided by subspace selection phase and the projected matrix is $\tilde{\mathbf{A}}'$. The ultra-low dimensional subspace significantly reduces the number of API calls in this phase, representing a substantial advantage compared to BBT. The optimization problem in this phase can be viewed as the scenario where $r(\mathbf{z}) \equiv 0$ in problem (2). Therefore, we continue to utilize ZO-MB-SPG to solve it.

Subspace learning and selection phases are part of the pre-training in our method and the prompt tuning phase can be viewed as fine-tuning based on the results of the first two phases in practical applications. Therefore, we report the results of the prompt tuning phase by default in section 5.

4 OPTIMIZATION ALGORITHM

In this section, we first define some common notations that will be used throughout the section. Then we describe the formulation of the problem of interest in this work. Finally, we propose a zeroth-order algorithm to solve the nonconvex optimization problem with a nonsmooth nonconvex regularizer.

4.1 Notations and Definitions

Throughout this paper, we use $\|\mathbf{x}\|$ to denote the Euclidean norm of a vector $\mathbf{x} \in \mathbb{R}^d$. Denote by $\mathcal{S} = \{\xi_1, \dots, \xi_m\}$ a set of random variables, let $|\mathcal{S}|$ be the number of elements in set $\mathcal{S}$ and $f_{\mathcal{S}}(\mathbf{x}) = \frac{1}{|\mathcal{S}|}\sum_{\xi_i\in\mathcal{S}} f(\mathbf{x}; \xi_i)$. We denote by $\operatorname{dist}(\mathbf{x}, \mathcal{S})$ the distance between the vector $\mathbf{x}$ and a set $\mathcal{S}$. Denote by $\hat{\partial}h(\mathbf{x})$ the Fréchet subgradient and $\partial h(\mathbf{x})$ the limiting subgradient of a nonconvex function $h(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$, i.e.,

$$\hat{\partial}h(\bar{\mathbf{x}}) = \left\{\mathbf{v} \in \mathbb{R}^d : \liminf_{\mathbf{x}\to\bar{\mathbf{x}}} \frac{h(\mathbf{x}) - h(\bar{\mathbf{x}}) - \mathbf{v}^\top(\mathbf{x} - \bar{\mathbf{x}})}{\|\mathbf{x} - \bar{\mathbf{x}}\|} \geq 0\right\}$$

$$\partial h(\bar{\mathbf{x}}) = \left\{\mathbf{v} \in \mathbb{R}^d : \exists \mathbf{x}_k \xrightarrow{h} \bar{\mathbf{x}}, v_k \in \hat{\partial}h(\mathbf{x}_k), \mathbf{v}_k \to \mathbf{v}\right\}$$

where $\mathbf{x} \xrightarrow{h} \bar{\mathbf{x}}$ means $\mathbf{x} \to \bar{\mathbf{x}}$ and $h(\mathbf{x}) \to h(\bar{\mathbf{x}})$.

We aim to find an ϵ-stationary point of the problem (1), i.e., to find a solution $\mathbf{x}$ such that $\operatorname{dist}(0, \hat{\partial}F(\mathbf{x})) \leq \epsilon$. Since f is differentiable, then we have $\hat{\partial}F(\mathbf{x}) = \hat{\partial}(f+r)(\mathbf{x}) = \nabla f(\mathbf{x}) + \hat{\partial}r(\mathbf{x})$. (see Exercise 8.8, Rockafellar and Wets [30]). Thus, it is equivalent to find a solution x satisfying

$$\mathrm{dist}(0, \nabla f(\mathbf{x}) + \hat{\partial} r(\mathbf{x})) \leq \epsilon. \tag{3}$$

4.2 Problem Formulation and Zero-Order Gradient Estimation

In this subsection, we consider the following stochastic nonsmooth nonconvex optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}) := \underbrace{\mathbb{E}_{\xi}[f(\mathbf{x};\xi)]}_{f(\mathbf{x})} + r(\mathbf{x}) \tag{4}$$

where ξ is a random variable, $f(x)$ is a smooth nonconvex function, and $r(x)$ is a proper nonsmooth nonconvex lower-semicontinuous function. This formulation is particularly focused on the exploration of sparsity-induced nonsmooth nonconvex regularizers, which play a pivotal role in subspace selection. For example, $\ell_p (0 \leq p \leq 1)$ norm, indicator function of a nonconvex constraint (e.g., $\|\mathbf{x}\|_0 \leq k$).

Given a black-box model parameterized by $\mathbf{x} \in \mathbb{R}^d$ and a loss function f. A random gradient estimator with mini-batch $\mathcal{S}$ is defined as follows,

$$\hat{\nabla} f_{\mathcal{S}}(\mathbf{x}) = \frac{f_{\mathcal{S}}(\mathbf{x} + \mu\mathbf{u}) - f_{\mathcal{S}}(\mathbf{x} - \mu\mathbf{u})}{2\mu}\mathbf{u}, \tag{5}$$

where $\mathbf{u} \in \mathbb{R}^d$ is distributed in $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

4.3 Zeroth-Order Mini-Batch Stochastic Proximal Gradient Method

The proposed algorithm for solving problem (4) is the zeroth-order mini-batch stochastic proximal gradient method (ZO-MB-SPG), which is presented in Algorithm 2. At the t-th iteration, ZO-MB-SPG first computes a mini-batch stochastic gradient of $f(\mathbf{x})$ using gradient estimator (5). Then the parameters are updated by performing standard proximal gradient descent.

Algorithm 2 Zeroth-Order Mini-Batch Stochastic Proximal Gradient

```
Input: initial point $\mathbf{x}_0 \in \mathbb{R}^d$, step size $\eta$, mini-batch size $m$, maximum number of iterations $T$
  for $t = 0$ to $T-1$ do
    Draw samples $\mathcal{S}_t = \{\xi_1, \dots, \xi_m\}$
    for $i = 1$ to $m$ do
      Sample a random direction $\boldsymbol{u}_i$ from a Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$
      Compute $\hat{\nabla} f_{\xi_i}(\mathbf{x}_t)$ through (5)
    end for
    Compute $\hat{\mathbf{g}}_t = \frac{1}{m}\sum_{i=1}^{m} \hat{\nabla} f_{\xi_i}(\mathbf{x}_t)$
    $\mathbf{x}_{t+1} \in \mathrm{prox}_{\eta r}[\mathbf{x}_t - \eta\hat{\mathbf{g}}_t]$
  end for
Output: $\mathbf{x}_R$ chosen uniformly randomly from $\{\mathbf{x}_t\}_{t=1}^{T}$.
```

Before delving into the theoretical result, we first make the following basic assumptions, which are standard in the literature on stochastic gradient methods for nonconvex optimization [14, 38].

Assumption 1. *Assume that the following conditions hold:*

(1) $\mathbb{E}_{\xi}[\nabla_{\xi} f(\mathbf{x})] = \nabla f(\mathbf{x})$ *, and there exists a constant* $\sigma > 0$*, such that* $\mathbb{E}_{\xi}[\|\nabla_{\xi} f(\mathbf{x}) - \nabla f(\mathbf{x})\|^2] \leq \sigma^2, \forall \mathbf{x} \in \mathbb{R}^d$.
(2) *Given an initial point* $\mathbf{x}_0$*, there exists* $\Delta < \infty$ *such that* $F(\mathbf{x}_0) - F(\mathbf{x}_*) \leq \Delta$*, where* $\mathbf{x}_*$ *denotes the global minimum of* (4).
(3) $f(\mathbf{x})$ *is smooth with a L-Lipschitz continuous gradient, i.e., it is differentiable and there exists a constant* $L > 0$ *such that* $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|, \forall \mathbf{x}, \mathbf{y}$.
(4) *There exists a constant* $M > 0$ *such that* $\|\nabla f(\mathbf{x}_t)\| \leq M$.

Assumption 2. *Assume that* $r(\mathbf{x})$ *is a proximal-friendly function such that the proximal operator can be obtained in closed-form, i.e.,*

$$\mathrm{prox}_{\eta r}[\mathbf{x}] = \operatorname*{argmin}_{\mathbf{y}\in\mathbb{R}^d} \left\{ \frac{1}{2\eta}\|\mathbf{y} - \mathbf{x}\|^2 + r(\mathbf{y}) \right\}.$$

We then establish the following general convergence theorem of Algorithm 2.

Theorem 1. *Suppose that Assumption 1 and 2, run Algorithm 1 with* $\eta = \frac{c}{L}\left(0 < c < \frac{1}{2}\right)$*, then the output* $\mathbf{x}_R$ *of Algorithm 1 satisfies*

$$\begin{aligned} &\mathbb{E}_R[\mathrm{dist}(\mathbf{0}, \hat{\partial} F(\mathbf{x}_R))^2] \\ \leq & c_1 \frac{4(d+4)(\sigma^2 + M^2)}{m} + (d+6)^3 L^2 \mu^2 + c_2 \frac{\Delta}{\eta T}, \end{aligned} \tag{6}$$

where $c_1 = \frac{2c(1-2c)+2}{c(1-2c)}, c_2 = \frac{6-4c}{1-2c}$.

Corollary 1. *Under the same assumptions of Theorem 1, run Algorithm 2 with* $\eta = \frac{c}{L}(0 < c < \frac{1}{2})$, $T = \frac{2c_2\Delta}{\eta\epsilon^2}$, $m = \frac{16c_1(d+4)(\sigma^2+M^2)}{\epsilon^2}$, $\mu = \frac{\epsilon}{2\sqrt{c_1(d+6)^3}L}$*, then the output of Algorithm 2 satisfies*

$$\mathbb{E}_R[\mathrm{dist}(\mathbf{0}, \hat{\partial} F(\mathbf{x}_R))^2] \leq \epsilon^2. \tag{7}$$

Remark 1. *According to the corollary above, the total oracle complexity of Algorithm 2 for solving problem 4 is* $O(d/\epsilon^4)$*. The dimensionality penalty term, d, is a recurrent theme in zeroth-order optimization literature [13, 25]. This highlights the challenges associated with employing zeroth-order algorithms for high-dimensional continuous optimization tasks, underscoring the significance of subspace learning strategies.*

5 EXPERIMENTS

5.1 Setup

Dataset and Task. For a fair comparison, we conduct experiments on various typical language understanding tasks as same as BBT [36], including sentiment analysis, topic classification, and natural language inference (NLI). In particular, we use SST-2 [32] and Yelp polarity [42] in sentiment analysis tasks, AG's News and DBPedia [42] in topic classification, RTE [37] and SNLI [4] in natural language inference(NLI). The input template and output label words utilized in the experiment for all datasets are shown in Table 1.
Backbone Model. For comparison with BBT, we conduct experiments using RoBERTa$_{\text{Large}}$ [23], which has approximately 355M parameters with relatively small intrinsic dimensions [2].
Baseline. We evaluate the proposed method in the black-box setting and compare it with gradient-free baselines. We choose the following methods as baselines: **(1) Manual Prompt**, which is a

Table 1: Datasets, input templates, and output label words used in our experiments.

Task	Dataset	Input Template	Output Label Words
sentiment	SST-2	$\langle S \rangle$. It was $[MASK]$.	great, bad
	Yelp P.	$\langle S \rangle$. It was $[MASK]$.	great, bad
topic	AG's News	$[MASK]$ News: $\langle S \rangle$	World, Sports, Business, Tech
	DBPedia	[Category: $[MASK]$] $\langle S \rangle$	Company, Education, Artist, Athlete, Office, Transportation, Building, Natural, Village, Animal, Plant, Album, Film, Written
NLI	RTE	$\langle S_1 \rangle$? $[MASK]$, $\langle S_2 \rangle$	Yes, No
	SNLI	$\langle S_1 \rangle$? $[MASK]$, $\langle S_2 \rangle$	Yes, Maybe, No

prompt-based approach without learning. We directly evaluate performance using human-written templates in Table 1. **(2) Feature-MLP** and **(3) Feature-BiLSTM** [26], which are two feature-based methods by utilizing features extracted from PTM to train MLP and BiLSTM classifiers. MLP utilizes only the $[CLS]$ representation, while LSTM leverages the representation of all tokens in the sequence. **(4) BBT** [36], which optimizes prompts in a low-dimensional subspace and obtains continuous prompts in the original prompt space through a projection matrix.

Evaluation Metrics. For all tasks, we measure performance using **test accuracy** and further provide **F1** and **macro-F1** scores. We conduct experiments with 3 different random seeds provided by BBT and report the average performance and standard deviation for all tasks. Additionally, the count of queries to the black-box model is an important metric in evaluating black-box optimization methods. Therefore, we also provide the mean **number of API calls** in training or optimization under an early stopping strategy.

Few-shot learning settings. Few-shot learning has drawn interest across various applications [40]. We construct the training and development sets by randomly sampling m samples for each class from the original training set. The original development sets are used as test sets. For datasets lacking the development set, the original test sets are used directly. Both the subspace learning phase and the subspace selection phase use the same training and development set, while the third phase uses different ones. The training, development, and test sets used for all baselines are consistent with the third phase.

Implementation Setting. Our method and all baselines are implemented using pytorch and experimented on an NVIDIA GTX 3090 GPU in 16-shot (per class) setting. Specifically, we train Feature-MLP and Feature-LSTM for 1000 epochs using an Adam optimizer [18] with learning rate of 3e-4 and batch size of 16 [36]. The experimental setup of BBT and the subspace learning phase in our method follows the implementation details in [36] and we set up to learn 10 dimensional subspace by default. We consider the indicator function of ℓ_0 constraint $I_{\{\|\mathbf{x}\|_0 \leq \kappa\}}(\mathbf{x})$ as the nonsmooth nonconvex regularizer in subspace selection phase to select 2-3 dimensions, and further provide experimental results for ℓ_0 regularizer $\lambda\|\mathbf{x}\|_0$ and $\ell_{0.5}$ regularizer $\lambda\|\mathbf{x}\|_{0.5}$. The maximum iteration of the second and third phases is 500 and the learning rate is 0.1. All methods uniformly employ the cross-entropy loss function, which is defined as $\mathcal{L}_{\text{CE}}(\hat{\mathbf{y}}, \tilde{y}) = -\log \text{Softmax}_{\tilde{y}}(\hat{\mathbf{y}})$ given the output logits $\hat{\mathbf{y}}$ and the ground-truth labels $\tilde{y}$. Our code is available at https://github.com/ZHZ-JLU/Subspace-Selection-based-Prompt-Tuning.

5.2 Results

5.2.1 Comparison with Baseline. We report the results of our proposed method along with all baselines on the 6 datasets in Table 2. It is worth noting that our method achieves the highest average performance and the tunable parameters of our method are significantly fewer compared to other methods, excluding Manual Prompt. Furthermore, we have observed that our proposed method consistently achieves performance comparable to that of BBT across all datasets, and slightly outperforms it in the 5 datasets. The standard deviation of our method is smaller in most cases compared to BBT, demonstrating the higher stability of our method. In particular, our method increases the test accuracy by 0.50%, 0.37%, 0.38%, 2.34%, 0.91% compared to BBT on the datasets of SST-2, AG's News, DBPedia, RTE, SNLI, respectively. The standard deviation of our method drops by 2.02, and 2.40 when solving NLI tasks like RTE and SNLI. The above results indicate that, while our method is based on BBT's framework for subspace learning, it outperforms BBT, especially in addressing the RTE task where BBT performs less optimally and demonstrates the most significant performance enhancement in this task. We attribute this phenomenon to the high standard deviation of BBT in NLI tasks, indicating that the learned $\mathbf{p}_i$ have larger differences than other tasks. In such cases, the advantages of our subspace selection are more effectively showcased. However, on tasks that can be effectively addressed with Manual Prompt, which provides initial points of our method, and has a small standard deviation in BBT, such as Yelp polarity, our method is extremely limited, due to the smaller differences between learned $\mathbf{p}_i$. In comparison to the feature-based method, our method falls short of Feature-LSTM only on the DBPedia, while outperforming Feature-LSTM on all other tasks. This phenomenon may suggest the feature-based method has advantages in solving multi-classification tasks. Finally, we show F1 and macro-F1 scores across various benchmarks in table 3, which exhibit similar trends to those in Table 2.

5.2.2 Comparison on the Number of API Calls. We perform early stopping for all methods and report the number of API calls for all methods in Table 4 with the same experimental settings as the experiments in Table 2. Since BBT sets 20,000 budgets for DBPedia (8,000 for the rest of the datasets), we calculated two averages based on whether DBPedia was included. In either case, the mean number of API calls for our method is significantly lower than BBT, at only 3.8% and 6.3%, and on the DBPedia task is only 1.3% of BBT's. Compared to the Feature-based method, our method still has slightly fewer API calls. Although the mean number of API calls for Feature-LSTM is similar to that of our method when DBPedia is

Table 2: Comparison of test accuracy on various language understanding tasks. We report the mean and standard deviation of test accuracy over 3 different seeds. All results are obtained using a pre-trained RoBERTa$_{\text{Large}}$ model in 16-shot (per class) setting. The last column shows the average accuracy across 6 datasets.

Method	Tunable Params	SST-2 acc	Yelp P. acc	AG's News acc	DBPedia acc	RTE acc	SNLI acc	Avg.
Manual Prompt	0	79.70	89.65	76.96	41.33	51.63	31.09	61.73
Feature-MLP	1M	61.73 ±2.42	72.23 ±6.41	69.91 ±4.43	86.16 ±1.61	48.50 ±3.25	36.07 ±4.77	62.43
Feature-LSTM	17M	64.53 ±1.02	70.97 ±1.36	75.61 ±0.49	87.85 ±2.25	50.06 ±2.40	37.31 ±1.17	64.39
BBT	500	88.45 ±0.81	91.74 ±0.77	83.31 ±1.83	85.83 ±1.98	48.56 ±4.36	44.36 ±4.25	73.71
Ours	<10	88.95 ±1.10	91.36 ±0.71	83.68 ±0.72	86.21 ±1.34	50.90 ±2.37	45.27 ±1.85	**74.40**

Table 3: Comparison of F1 and macro-F1 scores across various benchmarks under the same settings as experiments in Table 2.

Method	Tunable Params	SST-2 f1	Yelp P. f1	AG's News macro-f1	DBPedia macro-f1	RTE f1	SNLI macro-f1	Avg.
Manual Prompt	0	77.68	90.42	77.06	33.90	66.50	24.79	61.73
Feature-MLP	1M	62.94 ±8.77	73.38 ±6.21	70.24 ±4.19	86.20 ±1.42	44.08 ±15.10	32.21 ±8.41	61.51
Feature-LSTM	17M	64.18 ±6.22	72.23 ±0.91	74.91 ±0.63	87.76 ±2.28	49.10 ±13.10	35.98 ±2.77	64.03
BBT	500	89.15 ±0.54	92.25 ±0.66	83.26 ±1.79	85.66 ±2.04	50.82 ±16.50	43.30 ±3.04	74.07
Ours	<10	89.74 ±0.93	91.91 ±0.46	83.58 ±0.72	86.08 ±1.39	64.30 ±3.17	44.99 ±2.21	**76.77**

Table 4: Comparison of the number of API calls during training or optimization under early stopping strategy. w/o DBPedia: excluding the DBPedia when calculating the mean.

Method	SST-2	Yelp P.	AG's News	DBPedia	RTE	SNLI	Avg.	Avg.(-w/o DBPedia)
Feature-MLP	693	677	570	767	43	80	471	413
Feature-LSTM	103	457	460	700	157	20	316	239
BBT	3567	2767	4867	18133	3567	3467	6061	3647
Ours	366	340	207	233	20	214	230	229

not included, our method far exceeds its average performance. The overall experimental results demonstrate that our proposed method can effectively reduce the number of API calls while maintaining performance.

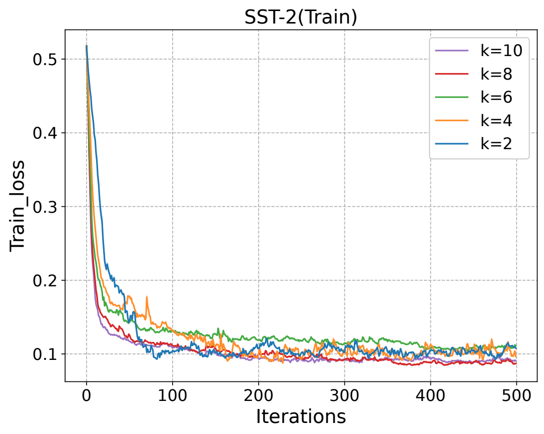


Figure 2: Train loss in subspace selection phase on SST-2. k: The number of preserved dimensions

5.3 Ablations

In this section, we perform ablation experiments to investigate the effect of the various components of our method on the performance. All experimental settings, except for the parameters explicitly specified, remain consistent with the experiments in Table 2. For each ablation, we conduct experiments on 3 different seeds and report the mean and standard deviation of the results.

5.3.1 Effect of Subspace Selection. First, we compare the effect of the number of preserved dimensions on the convergence of the subspace selection algorithm, as shown in Figure 2, when the number of preserved dimensions $k = 2, 4, 6, 8, 10$, the subspace selection algorithms can all converge to approximately 0.1 on the train loss, which proves that our subspace selection algorithm is capable of selecting a specified number of dimensions and continuously optimize the corresponding components in $\mathbf{z}''$. We report the test accuracy on all datasets in Table 5, both with or without using the subspace selection phase. Most of the results show improvement due to subspace selection, demonstrating that our subspace selection algorithm is capable of identifying superior dimensions provided by the subspace learning phase. It is worth noting that the standard deviation of performance decreases on all 6 datasets after subspace selection, indicating that subspace selection not only enhances performance but also contributes to improved stability.

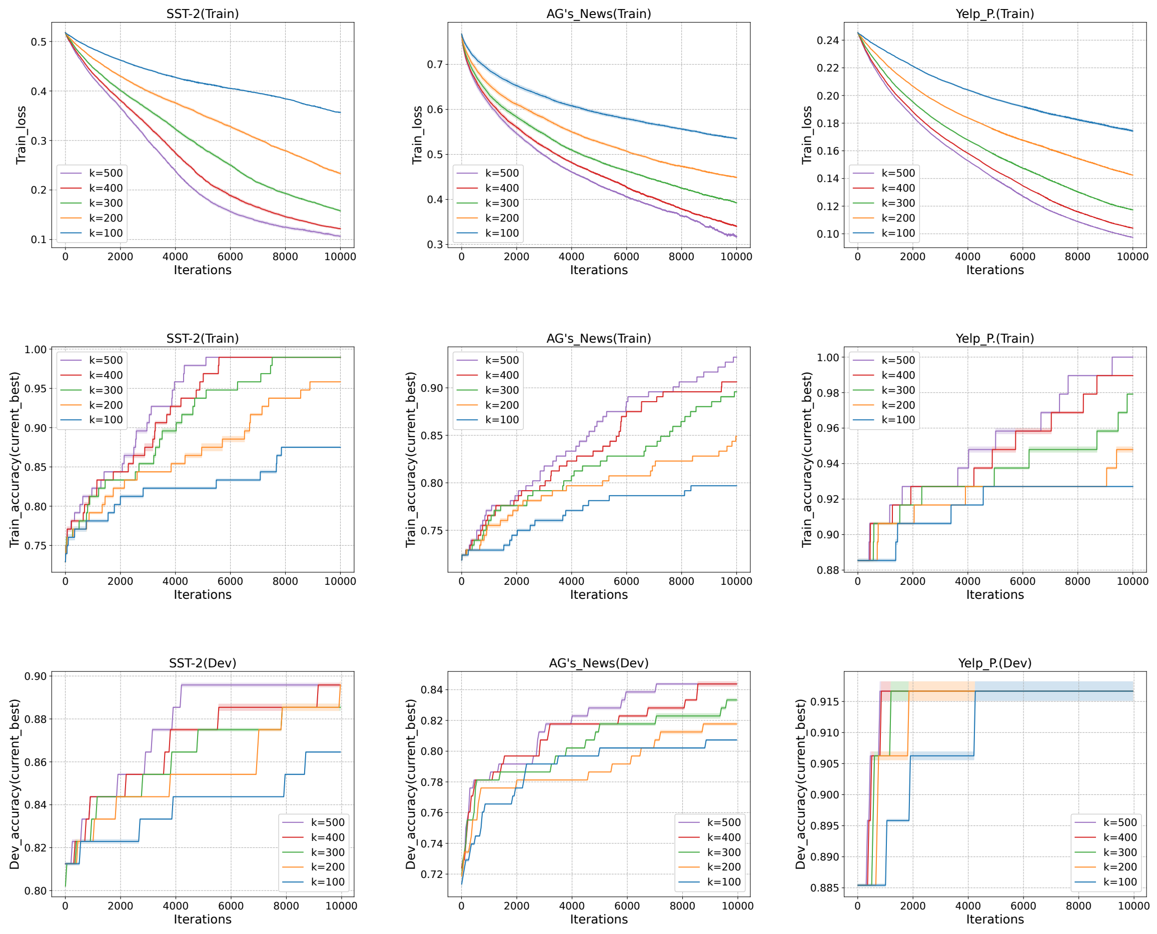


Figure 3: Results after removing subspace learning phase on SST-2, AG's News and Yelp P.

Table 5: Ablations of the subspace selection phase. We show the mean and standard deviation of test accuracy over 3 different seeds across 6 datasets.

Datasets	Subspace Selection	
	Y	N
SST-2	88.95 ±1.10	88.38±1.23
Yelp P.	91.36 ±0.71	91.98 ±1.01
AG's News	83.68 ±0.72	83.56 ±0.86
DBPedia	86.21 ±1.34	77.84 ±4.08
RTE	50.90 ±2.37	47.89 ±3.07
SNLI	45.27 ±1.85	45.21 ±2.96

5.3.2 Effect of Subspace Learning. We remove the subspace learning phase and instead use randomly generated projection matrix $\tilde{\mathbf{A}}$, in which each column is orthogonal to the others. We set d' to 500 and the maximum number of iterations to 10,000. The results of the experiments when the number of preserved dimensions $k = 100, 200, 300, 400, 500$ are shown in Table 6 and Figure 3. In contrast to the original method that optimizes fewer than 10 parameters, the method excluding subspace learning requires expanding the subspace dimension to 500 and increasing the number of iterations to 10,000 to attain comparable performance. This result illustrates the irreplaceable role of the subspace learning phase in improving performance and reducing subsequent training costs. Moreover, from Table 6 and Figure 3, it can be observed that the performance of the method excluding subspace learning decreases with the reduction of k, which indicates a failure in the subspace selection phase. We attribute this phenomenon to the absence of distinction in superiority or inferiority among dimensions because the projected matrix $\tilde{\mathbf{A}}$ is randomly generated rather than learned, in which case a higher dimensionality means a higher probability of containing good solutions. So it can be concluded that the subspace learning phase is the prerequisite for subspace selection in our method.

5.3.3 Effect of Subspace Dimensionality in Subspace Learning. In the subspace learning phase, the actual optimization is performed on a d dimensional subspace. We keep the budget of API calls at

8000 and vary the d in the subspace learning phase to conduct experiments to compare our method’s performance on SST-2, AG’s News at $d = 100, 300, 500$ and the results are shown in the table 7. When $d = 100, 300$, the performance of the method experienced varying degrees of decline compared to $d = 500$, which is because it is challenging for a small subspace to contain a good solution. On the other hand, the influence of dimensionality changes on performance varies across datasets. For the SST-2, decreasing from 500 to 300 dimensions has little impact on the performance, while decreasing from 300 to 100 dimensions reduces the performance by 2.72%. For the AG’s News, the decline trend is relatively smooth.

Table 6: Test accuracy after removing subspace learning phase on SST-2, AG’s News and Yelp P.

k	SST-2	AG’s News	Yelp P.
	acc	acc	acc
500	88.68 ±0.48	82.71 ±1.29	91.36 ±0.75
400	88.27 ±0.74	83.32 ±1.52	91.33 ±0.74
300	87.77 ±0.63	82.62 ±0.76	91.24 ±0.71
200	87.08 ±0.54	82.39 ±1.03	91.04 ±0.66
100	85.89 ±0.35	81.30 ±0.39	90.73 ±0.72

Table 7: Ablations of subspace dimensionality in subspace learning phase on SST-2 and AG’s News.

Dimensionality (d)	SST-2	AG’s News
	acc	acc
500	88.95 ±1.10	83.68 ±0.72
300	88.80 ±1.53	82.62 ±2.51
100	86.08 ±1.68	81.60 ±1.01

5.3.4 *Effect of Nonsmooth Nonconvex Regularizer.*

To investigate the sensitivity of prompt performance to regularizer, we provide experimental results for two other nonsmooth nonconvex regularizers, i.e., ℓ_0 regularizer, $\ell_{0.5}$ regularizer, and compare them with results of the indicator function of ℓ_0 constraint. As shown in table 8, the results on the 3 regularizers show minimal differences, indicating that the prompt performance is relatively insensitive to the choice of regularizer. In the context of such similar results, the advantage of the indicator function of ℓ_0 constraint in explicitly specifying the number of preserved dimensions becomes apparent.

Table 8: Ablations of the 3 different nonsmooth nonconvex regularizers, i.e., indicator function of ℓ_0 constraint, ℓ_0 regularizer, and $\ell_{0.5}$ regularizer.

Datasets	ℓ_0 constraint	ℓ_0 regularizer	$\ell_{0.5}$ regularizer
SST-2	88.95 ±1.10	88.88±1.21	89.03±0.47
Yelp P.	91.36 ±0.71	91.00 ±1.41	91.44±1.13
AG’s News	83.68 ±0.72	83.44 ±0.57	83.32±1.48
DBPedia	86.21 ±1.34	86.71 ±1.19	86.66±1.41
RTE	50.90 ±2.37	50.30 ±4.70	51.02±1.04
SNLI	45.27 ±1.85	44.78 ±2.83	45.59±0.56

6 CONCLUSIONS

It is challenging to obtain continuous prompts through black-box optimization within the framework of Language Model as a Service (LMaaS) due to the high dimensionality of continuous prompts. To overcome this limitation, we propose a novel framework for black-box prompt tuning, which is based on the fact that LLMs possess intrinsic dimensions significantly lower than the embedding space. Our method acquires an ultra-low dimensional subspace through subspace learning and subspace selection and then optimizes prompts within this subspace. Simultaneously, we introduce a zeroth-order optimization algorithm (ZO-MB-SPG) to achieve subspace selection through sparsity-inducing nonsmooth nonconvex regularizers. Extensive experiment results on few-shot learning demonstrate that our method attains performance comparable to existing methods with much fewer API calls.

7 ACKNOWLEDGEMENTS

This work was supported by the National Key R&D Program of China under Grant (No.2023YFF0905400), National Natural Science Foundation of China through grants (No.U2341229, No.62076138)

REFERENCES

[1] Alekh Agarwal, Ofer Dekel, and Lin Xiao. 2010. Optimal Algorithms for Online Convex Optimization with Multi-Point Bandit Feedback.. In *Colt*. Citeseer, 28–40.
[2] Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning. 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)*. Association for Computational Linguistics, Online, 7319–7328.
[3] Atılım Güneş Baydin, Barak A Pearlmutter, Don Syme, Frank Wood, and Philip Torr. 2022. Gradients without backpropagation. *arXiv preprint arXiv:2202.08587* (2022).
[4] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326* (2015).
[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.
[6] Aochuan Chen, Yimeng Zhang, Jinghan Jia, James Diffenderfer, Jiancheng Liu, Konstantinos Parasyris, Yihua Zhang, Zheng Zhang, Bhavya Kailkhura, and Sijia Liu. 2023. DeepZero: Scaling up Zeroth-Order Optimization for Deep Model Training. *arXiv preprint arXiv:2310.02025* (2023).
[7] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. 2017. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*. 15–26.
[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* 24, 240 (2023), 1–113.
[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186.
[10] Shizhe Diao, Zhichao Huang, Ruijia Xu, Xuechun Li, Yong Lin, Xiao Zhou, and Tong Zhang. 2022. Black-box prompt learning for pre-trained language models. *arXiv preprint arXiv:2201.08531* (2022).
[11] John C Duchi, Michael I Jordan, Martin J Wainwright, and Andre Wibisono. 2015. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory* 61, 5 (2015), 2788–2806.
[12] William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research* 23, 1 (2022), 5232–5270.
[13] Saeed Ghadimi and Guanghui Lan. 2013. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization* 23, 4 (2013), 2341–2368.
[14] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. 2016. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization.

Mathematical Programming 155, 1-2 (2016), 267–305.

[15] Nikolaus Hansen and Andreas Ostermeier. 2001. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation* 9, 2 (2001), 159–195.

[16] Geoffrey Hinton. 2022. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345* (2022).

[17] Feihu Huang, Bin Gu, Zhouyuan Huo, Songcan Chen, and Heng Huang. 2019. Faster gradient-free proximal stochastic methods for nonconvex nonsmooth optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 1503–1510.

[18] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[19] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691* (2021).

[20] Patrick Lewis, Myle Ott, Jingfei Du, and Veselin Stoyanov. 2020. Pretrained language models for biomedical and clinical tasks: understanding and extending the state-of-the-art. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*. 146–157.

[21] Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190* (2021).

[22] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. GPT understands, too. *AI Open* (2023).

[23] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[24] Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev Arora. 2023. Fine-Tuning Language Models with Just Forward Passes. *arXiv preprint arXiv:2305.17333* (2023).

[25] Yurii Nesterov and Vladimir Spokoiny. 2017. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics* 17 (2017), 527–566.

[26] Matthew E Peters, Sebastian Ruder, and Noah A Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987* (2019).

[27] Yujia Qin, Xiaozhi Wang, Yusheng Su, Yankai Lin, Ning Ding, Zhiyuan Liu, Juanzi Li, Lei Hou, Peng Li, Maosong Sun, et al. 2021. Exploring lowdimensional intrinsic task subspace via prompt tuning. *arXiv preprint arXiv:2110.07867* (2021).

[28] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[29] Mengye Ren, Simon Kornblith, Renjie Liao, and Geoffrey Hinton. 2023. Scaling Forward Gradient With Local Losses. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=JxpBP1JM15-

[30] R Tyrrell Rockafellar and Roger J-B Wets. 2009. *Variational analysis*. Vol. 317. Springer Science & Business Media, Springer-Verlag Berlin Heidelberg 1998.

[31] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864* (2017).

[32] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. 1631–1642.

[33] James C Spall. 1992. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE transactions on automatic control* 37, 3 (1992), 332–341.

[34] Jingwei Sun, Ziyue Xu, Hongxu Yin, Dong Yang, Daguang Xu, Yiran Chen, and Holger R Roth. 2023. FedBPT: Efficient Federated Black-box Prompt Tuning for Large Language Models. *arXiv preprint arXiv:2310.01467* (2023).

[35] Tianxiang Sun, Zhengfu He, Hong Qian, Yunhua Zhou, Xuanjing Huang, and Xipeng Qiu. 2022. BBTv2: Towards a Gradient-Free Future with Large Language Models. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 3916–3930.

[36] Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning*. PMLR, 20841–20855.

[37] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461* (2018).

[38] Yi Xu, Rong Jin, and Tianbao Yang. 2019. Non-asymptotic analysis of stochastic methods for non-smooth non-convex regularized problems. *Advances in Neural Information Processing Systems* 32 (2019).

[39] Lang Yu, Qin Chen, Jiaju Lin, and Liang He. 2023. Black-box prompt tuning for vision-language model as a service. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*. 1686–1694.

[40] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. *arXiv preprint arXiv:1805.07513* (2018).

[41] Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, et al. 2021. Pangu-α: Large-scale autoregressive pretrained Chinese language models with auto-parallel computation. *arXiv preprint arXiv:2104.12369* (2021).

[42] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems* 28 (2015).

[43] Yuanhang Zheng, Zhixing Tan, Peng Li, and Yang Liu. 2023. Black-box Prompt Tuning with Subspace Learning. *arXiv preprint arXiv:2305.03518* (2023).

APPENDIX

A PROOF OF THEOREM 1

Proof. Recall that the update of $\mathbf{x}_{t+1}$ is

$$\begin{aligned}\mathbf{x}_{t+1} &\in \arg\min_{\mathbf{x}\in\mathbb{R}^d}\left\{r(\mathbf{x}) + \frac{1}{2\eta}\|\mathbf{x} - (\mathbf{x}_t - \eta\hat{\mathbf{g}}_t)\|^2\right\} \\ &= \arg\min_{\mathbf{x}\in\mathbb{R}^d}\left\{r(\mathbf{x}) + \langle\hat{\mathbf{g}}_t, \mathbf{x}-\mathbf{x}_t\rangle + \frac{1}{2\eta}\|\mathbf{x}-\mathbf{x}_t\|^2\right\},\end{aligned} \tag{8}$$

then by Exercise 8.8 and Theorem 10.1 of [30] we know

$$-\hat{\mathbf{g}}_t - \frac{1}{\eta}(\mathbf{x}_{t+1}-\mathbf{x}_t) \in \hat{\partial}r(\mathbf{x}_{t+1}), \tag{9}$$

which implies that

$$\nabla f(\mathbf{x}_{t+1}) - \hat{\mathbf{g}}_t - \frac{1}{\eta}(\mathbf{x}_{t+1}-\mathbf{x}_t) \in \nabla f(\mathbf{x}_{t+1}) + \hat{\partial}r(\mathbf{x}_{t+1}) = \hat{\partial}F(\mathbf{x}_{t+1}). \tag{10}$$

By the update of $\mathbf{x}_{t+1}$ in Algorithm 1, we also have

$$r(\mathbf{x}_{t+1}) + \langle\hat{\mathbf{g}}_t, \mathbf{x}_{t+1}-\mathbf{x}_t\rangle + \frac{1}{2\eta}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \le r(\mathbf{x}_t). \tag{11}$$

Since $f(\mathbf{x})$ is smooth with parameter L , then

$$f(\mathbf{x}_{t+1}) \le f(\mathbf{x}_t) + \langle\nabla f(\mathbf{x}_t), \mathbf{x}_{t+1}-\mathbf{x}_t\rangle + \frac{L}{2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2. \tag{12}$$

Combining these two inequalities (11) and (12) we get

$$\langle\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1}-\mathbf{x}_t\rangle + \frac{1}{2}(1/\eta - L)\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \le F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}). \tag{13}$$

That is

$$\begin{aligned}&\frac{1}{2}(1/\eta - L)\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \\ \le& F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}) - \langle\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1}-\mathbf{x}_t\rangle \\ \le& F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}) + \frac{1}{2L}\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 + \frac{L}{2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2,\end{aligned}$$

where the last inequality uses Young's inequality $\langle\mathbf{a},\mathbf{b}\rangle \le \frac{1}{2}\|\mathbf{a}\|^2 + \frac{1}{2}\|\mathbf{b}\|^2$. Then by rearranging above inequality and summing it across $t = 0, \ldots, T-1$, we have

$$\begin{aligned}\frac{1-2\eta L}{2\eta}\sum_{t=0}^{T-1}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 &\le F(\mathbf{x}_0) - F(\mathbf{x}_T) + \frac{1}{2L}\sum_{t=0}^{T-1}\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 \\ &\le F(\mathbf{x}_0) - F(\mathbf{x}_*) + \frac{1}{2L}\sum_{t=0}^{T-1}\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 \\ &\le \Delta + \frac{1}{2L}\sum_{t=0}^{T-1}\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2,\end{aligned} \tag{14}$$

where the second inequality uses the fact that $F(\mathbf{x}_*) \le F(\mathbf{x})$ for any $\mathbf{x} \in \mathbb{R}^d$ and the last inequality uses the Assumption of $F(\mathbf{x}_0) - F(\mathbf{x}_*) \le \Delta$.

On the other hand, multiplying by $\frac{1}{\eta}$ on both sides of (13), we have

$$\begin{aligned}&\frac{2}{\eta}\langle\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_{t+1}), \mathbf{x}_{t+1}-\mathbf{x}_t\rangle + \frac{1-\eta L}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \\ \le& \frac{2(F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}))}{\eta} - \frac{2}{\eta}\langle\nabla f(\mathbf{x}_{t+1}) - \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1}-\mathbf{x}_t\rangle.\end{aligned} \tag{15}$$

Since

$$\begin{aligned}&2\left\langle\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_{t+1}), \frac{1}{\eta}(\mathbf{x}_{t+1}-\mathbf{x}_t)\right\rangle \\ =& \|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_{t+1}) + \frac{1}{\eta}(\mathbf{x}_{t+1}-\mathbf{x}_t)\|^2 - \|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_{t+1})\|^2 \\ &- \frac{1}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2,\end{aligned}$$

then plugging above inequality into (15) and rearranging it we have

$$\begin{aligned}&\left\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_{t+1}) + \frac{1}{\eta}(\mathbf{x}_{t+1}-\mathbf{x}_t)\right\|^2 \\ \le& \|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_{t+1})\|^2 + \frac{1}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 - \frac{1-\eta L}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \\ &+ \frac{2(F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}))}{\eta} - \frac{2}{\eta}\langle\nabla f(\mathbf{x}_{t+1}) - \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1}-\mathbf{x}_t\rangle \\ \le& 2\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 + 2\|\nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t+1})\|^2 + \frac{1}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \\ &- \frac{1-\eta L}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 + \frac{2(F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}))}{\eta} \\ &- \frac{2}{\eta}\langle\nabla f(\mathbf{x}_{t+1}) - \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1}-\mathbf{x}_t\rangle \\ \le& 2\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 + 2L^2\|\mathbf{x}_t - \mathbf{x}_{t+1}\|^2 + \frac{1}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \\ &- \frac{1-\eta L}{\eta^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 + \frac{2(F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}))}{\eta} + \frac{2L}{\eta}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2 \\ =& 2\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 + \frac{2(F(\mathbf{x}_t) - F(\mathbf{x}_{t+1}))}{\eta} + \left(2L^2 + \frac{3L}{\eta}\right)\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2,\end{aligned}$$

where the second inequality is due to Young's inequality $\|\mathbf{a}\pm\mathbf{b}\|^2 \le 2\|\mathbf{a}\|^2 + 2\|\mathbf{b}\|^2$; the last inequality is due to the Assumption of $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \le L\|\mathbf{x}-\mathbf{y}\|$ for any $\mathbf{x},\mathbf{y} \in \mathbb{R}^d$ and Cauchy-Schwartz inequality. By summing up $t = 0, 1, \ldots, T-1$, we have

$$\begin{aligned}&\sum_{t=0}^{T-1}\left\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_{t+1}) + \frac{1}{\eta}(\mathbf{x}_{t+1}-\mathbf{x}_t)\right\|^2 \\ \le& 2\sum_{t=0}^{T-1}\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 + \frac{2(F(\mathbf{x}_0) - F(\mathbf{x}_T))}{\eta} \\ &+ \left(2L^2 + \frac{3L}{\eta}\right)\sum_{t=0}^{T-1}\|\mathbf{x}_t - \mathbf{x}_{t+1}\|^2 \\ \le& 2\sum_{t=0}^{T-1}\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 + \frac{2(F(\mathbf{x}_0) - F(\mathbf{x}_*))}{\eta} \\ &+ \left(2L^2 + \frac{3L}{\eta}\right)\sum_{t=0}^{T-1}\|\mathbf{x}_t - \mathbf{x}_{t+1}\|^2 \\ \le& 2\sum_{t=0}^{T-1}\|\hat{\mathbf{g}}_t - \nabla f(\mathbf{x}_t)\|^2 + \frac{2\Delta}{\eta} + \frac{2}{\eta^2}\sum_{t=0}^{T-1}\|\mathbf{x}_t - \mathbf{x}_{t+1}\|^2,\end{aligned}$$

where the second inequality is due to $F(\mathbf{x}_*) \le F(\mathbf{x}_T)$; the last inequality holds by setting $\eta = \frac{c}{L} < \frac{1}{2L}$ and Assumption of $F(\mathbf{x}_0) - F(\mathbf{x}_*) \le \Delta$. Combining the above inequality with (10) and (14) and taking the expectation, we have

$$\begin{aligned}
&\mathbb{E}_R\left[\operatorname{dist}\left(0,\hat{\partial}F\left(\mathbf{x}_R\right)\right)^2\right]\\
\leq&\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\left\|\hat{\mathbf{g}}_t-\nabla f\left(\mathbf{x}_{t+1}\right)+\frac{1}{\eta}\left(\mathbf{x}_{t+1}-\mathbf{x}_t\right)\right\|^2\right]\\
\leq&\frac{2}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\hat{\mathbf{g}}_t-\nabla f\left(\mathbf{x}_t\right)\|^2\right]+\frac{2\Delta}{\eta T}\\
&+\frac{2}{\eta^2 T}\left(\frac{2}{1/\eta-2L}\Delta+\frac{1}{L/\eta-2L^2}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\hat{\mathbf{g}}_t-\nabla f\left(\mathbf{x}_t\right)\|^2\right]\right)\\
=&\frac{2c(1-2c)+2}{c(1-2c)}\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\hat{\mathbf{g}}_t-\nabla f\left(\mathbf{x}_t\right)\|^2\right]+\frac{6-4c}{1-2c}\frac{\Delta}{\eta T},
\end{aligned}$$

where $0<c<\frac{1}{2}$. And then we can get

$$\begin{aligned}
&\mathbb{E}\|\hat{\mathbf{g}}_t-\nabla f\left(\mathbf{x}_t\right)\|^2\\
=&\mathbb{E}\|\hat{\mathbf{g}}_t-\nabla f_\mu(\mathbf{x}_t)+\nabla f_\mu(\mathbf{x}_t)-\nabla f\left(\mathbf{x}_t\right)\|^2\\
=&\mathbb{E}\|\hat{\mathbf{g}}_t-\nabla f_\mu(\mathbf{x}_t)\|^2+\mathbb{E}\|\nabla f_\mu(\mathbf{x}_t)-\nabla f\left(\mathbf{x}_t\right)\|^2\\
&+\mathbb{E}\left\langle\hat{\mathbf{g}}_t-\nabla f_\mu(\mathbf{x}_t),\nabla f_\mu(\mathbf{x}_t)-\nabla f\left(\mathbf{x}_t\right)\right\rangle\\
=&\mathbb{E}\|\hat{\mathbf{g}}_t-\nabla f_\mu(\mathbf{x}_t)\|^2+\mathbb{E}\|\nabla f_\mu(\mathbf{x}_t)-\nabla f\left(\mathbf{x}_t\right)\|^2\\
\leq&\mathbb{E}\|\hat{\mathbf{g}}_t-\nabla f_\mu(\mathbf{x}_t)\|^2+\frac{L^2\mu^2(d+3)^3}{4}\\
=&\mathbb{E}\|\frac{1}{m}\sum_{i=1}^{m}\hat{\nabla}f_{\xi_i}(\mathbf{x}_t)-\nabla f_\mu(\mathbf{x}_t)\|^2+\frac{L^2\mu^2(d+3)^3}{4}\\
=&\frac{1}{m}\mathbb{E}\|\hat{\nabla}f_\xi(\mathbf{x}_t)-\nabla f_\mu(\mathbf{x}_t)\|^2+\frac{L^2\mu^2(d+3)^3}{4}\\
\leq&\frac{1}{m}\mathbb{E}\|\hat{\nabla}f_\xi(\mathbf{x}_t)\|^2+\frac{L^2\mu^2(d+3)^3}{4}\\
\leq&\frac{1}{m}\mathbb{E}_\xi\left[2(d+4)\|\nabla f_\xi(\mathbf{x}_t)\|^2+\frac{\mu^2L^2(d+6)^3}{2}\right]+\frac{L^2\mu^2(d+3)^3}{4}\\
\leq&\frac{1}{m}\left[2(d+4)\left(2\mathbb{E}_\xi\|\nabla f_\xi(\mathbf{x}_t)-\nabla f(\mathbf{x}_t)\|^2+2\|\nabla f(\mathbf{x}_t)\|^2\right)\right.\\
&\left.+\frac{\mu^2L^2(d+6)^3}{2}\right]+\frac{L^2\mu^2(d+3)^3}{4}\\
\leq&\frac{4(d+4)(\sigma^2+M^2)}{m}+(d+6)^3L^2\mu^2
\end{aligned}$$

where the first, the third and the last inequality use Lemma 6 in [17]. Finally, we have

$$\begin{aligned}
&\mathbb{E}_R\left[\operatorname{dist}\left(0,\hat{\partial}F\left(\mathbf{x}_R\right)\right)^2\right]\\
\leq&\frac{2c(1-2c)+2}{c(1-2c)}\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\hat{\mathbf{g}}_t-\nabla f\left(\mathbf{x}_t\right)\|^2\right]+\frac{6-4c}{1-2c}\frac{\Delta}{\eta T}\\
\leq&\frac{2c(1-2c)+2}{c(1-2c)}\hat{\sigma}^2+\frac{6-4c}{1-2c}\frac{\Delta}{\eta T}
\end{aligned}$$

where $\hat{\sigma}^2=\frac{4(d+4)(\sigma^2+M^2)}{m}+(d+6)^3L^2\mu^2$.

□