ULPT: Prompt Tuning with Ultra-Low-Dimensional Optimization

Anonymous Authors[1]

Abstract

Large language models achieve state-of-the-art performance but are costly to fine-tune due to their size. Parameter-efficient fine-tuning methods, such as prompt tuning, address this by reducing trainable parameters while maintaining strong performance. However, prior methods tie prompt embeddings to the model's dimensionality, which may not scale well with larger LLMs and more customized LLMs. In this paper, we propose Ultra-Low-dimensional Prompt Tuning (ULPT), which optimizes prompts in a low-dimensional space (e.g., 2D) and use a random but frozen matrix for the up-projection. To enhance alignment, we introduce learnable shift and scale embeddings. ULPT drastically reduces the trainable parameters, e.g., 2D only using 2% parameters compared with vanilla prompt tuning while retaining most of the performance across 21 NLP tasks. Our theoretical analysis shows that random projections can capture high-rank structures effectively, and experimental results demonstrate ULPT's competitive performance over existing parameter-efficient methods.[1]

1. Introduction

Fine-tuning large language models (LLMs) is essential for adapting them to specific tasks and controlling their outputs (Raffel et al., 2020; Wei et al., 2022a). However, the enormous size of LLMs makes full fine-tuning prohibitively resource intensive, as it involves updating millions or even billions of parameters. To address this challenge, parameter-efficient fine-tuning methods have emerged as practical solutions, such as low-rank adaptation (LoRA; Hu et al., 2022) and prompt tuning (Lester et al., 2021; Li & Liang, 2021). These methods drastically reduce the number of tunable parameters, offering an efficient alternative while achieving performance comparable to full fine-tuning.

Prompt tuning introduces learnable prompt embeddings exclusively in the input layer of the model (Lester et al., 2021; Liu et al., 2024), automating prompt engineering by gradient descent to guide the frozen LLM in producing task-specific outputs (Petrov et al., 2024b;a). By contrast, LoRA modifies the model by injecting low-rank weight matrices into its layers, causing the number of trainable parameters to scale with model's depth (Hu et al., 2022). Given that LLMs encode substantial knowledge during pretraining (Brown et al., 2020; Kojima et al., 2022) and that both in-context learning and expertly crafted prompts can achieve remarkable results (Wei et al., 2022b; Dong et al., 2024), prompt tuning offers a more efficient and effective alternative to LoRA in many scenarios (Shi & Lipani, 2024).

Despite its advantages, most existing prompt tuning approaches couple the dimensionality of prompt embeddings with the hidden size of the model (Lester et al., 2021; Li & Liang, 2021; Liu et al., 2022; Choi et al., 2023; Razdaibiedina et al., 2023). As the size of the model increases, the dimensionality of the prompt embedding space also increases (Raffel et al., 2020; Touvron et al., 2023). This scaling leads to unnecessary complexity, as full dimensionality is often not required for task adaptation (Aghajanyan et al., 2021; Qin et al., 2022). Consequently, optimizing in this expanded space becomes inefficient in parameter's usage and may also increase the risk of overfitting, especially for less complex tasks or with limited training data.

In this paper, we propose **U**ltra-**L**ow-Dimensional **P**rompt **T**uning (ULPT), a method that decouples the prompt/model dimensions and enables prompt tuning in an ultra-low-dimensional space (e.g., 2D). A naïve attempt is to jointly optimize the ultra-low-dimensional embeddings with an up-projection matrix (Xiao et al., 2023; Guo et al., 2024), but the learnable up-projection matrix may result in more trainable parameters than vanilla prompt tuning, therefore offsetting the gains in parameter efficiency. As shown in Figure 1a, our ULPT eliminates this overhead by using a *random* but *frozen* matrix for the up-projection. We further introduce learnable *shift* and *scale* embedding vectors to better align the up-projected embeddings and the model's prompt space (Wu et al., 2024c). In addition, we provide

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1]Our code is available anonymously at https://github.com/ULPT-anonymous/code

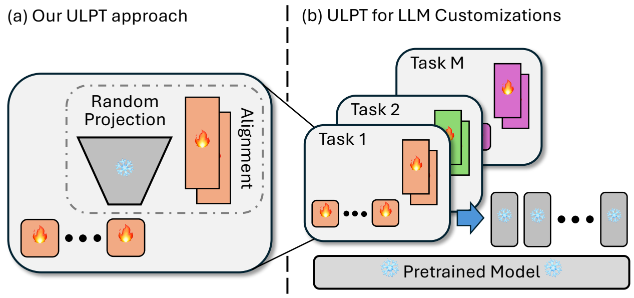

Figure 1. Overview of our approach. (a) ULPT up-projects ultra-low-dimensional embeddings with a random but fixed matrix, followed by a learnable alignment mechanism shared across all up-projected embeddings. (b) ULPT can significantly reduce parameters usage for LLM customizations.

theoretical analysis for our ULPT, which not only proves its convergence but also shows that a low-dimensional space with random projection can effectively approximate high-rank information.

The ultra-low-dimensional nature of our ULPT is particularly suitable for scenarios requiring massive LLM customizations (Mangrulkar et al., 2022) and continual learning (Wang et al., 2022), as shown in Figure 1b. For example, in a typical prompt tuning approach, each task might require 100K real-valued parameters, which can add up significantly when scaling to millions of customized LLMs or tasks. By contrast, our ULPT with 2D prompt embeddings can reduce this to just 2K parameters per task, presenting a dramatic parameter saving to just 2% of the original usage.

We conducted experiments across 21 NLP datasets to evaluate our ULPT. The results demonstrate that ULPT can extend a few-token vanilla prompt tuning setup to a 100-token configuration without increasing the number of trainable parameters, while matching the performance of a fully parameterized 100-token prompt tuning setup. With appropriate settings of the dimension, ULPT outperforms existing prompt tuning-based methods, while requiring much fewer trainable parameters.

In summary, our main contributions include:

- We introduce ULPT (Ultra-Low-Dimensional Prompt Tuning), which optimizes prompts in a low-dimensional space with a random up-projection and learnable shift and scale vectors, drastically reduces trainable parameters while maintaining performance.
- We provide theoretical analysis showing ULPT's ability to capture high-rank structures effectively and ensure convergence.
- Across 21 NLP tasks, ULPT delivers comparable performance to vanilla prompt tuning while reducing trainable parameters 2% of the original usage. When scaling to higher dimensions, it outperforms existing prompt tuning-based methods with much fewer trainable parameters.

2. Related Work

Parameter-efficient fine-tuning. With the rapid growth of pretrained neural networks, researchers have investigated parameter-efficient fine-tuning methods that update only a small set of parameters while maintaining high performance. One straight way is to tune specific components of the model. For example, BitFit (Ben Zaken et al., 2022) updates only the bias terms, and LayerNorm tuning (Zhao et al., 2024) only trains the layer-norm parameters. Another line of work involves introducing and training small, task-specific non-linear modules, such as Adapters (Houlsby et al., 2019) and AdapterDrop (Rücklé et al., 2021). Other methods steer the activation representations either globally or locally (Wu et al., 2024b; Yin et al., 2024).

Two other prominent paradigms are low-rank adaptation (LoRA; Hu et al., 2022) and prompt tuning methods (Lester et al., 2021), which are more related to our work. They will be further elaborated in the subsequent sections.

Low-rank adaptation. Hu et al. (2022) assume that weight updates can be approximated by low-rank matrices and propose a low-rank adaptation (LoRA) method for fine-tuning a model. Building upon this foundational work, many extensions have been developed to enhance LoRA's performance. For example, ReLoRA (Lialin et al., 2024) iteratively trains and merges low-rank adapters to achieve high-rank updates. Hayou et al. (2024) propose learning low-rank matrices with different learning rates. Wu et al. (2024a) explore training a mixture of LoRA modules and leverage dynamic routing mechanisms for different task distributions or domains.

However, for large models, LoRA still requires a considerable number of trainable parameters. To address this limitation, several works have explored the use of random projection (Bingham & Mannila, 2001) to further improve parameter efficiency. For example, FLORA (Hao et al., 2024) updates the pretrained matrices with randomly projected gradients, while VeRA (Kopiczko et al., 2024) uses random projections combined with two trainable scaling vectors to approximate each update matrix.

Prompt tuning. Shin et al. (2020) introduce the concept of learning prompt tokens to elicit knowledge from LLMs. Subsequently, Lester et al. (2021) extend this idea to continuous prompt tuning, where prompt embeddings are optimized through gradient descent while keeping the LLM frozen. Building on this, Li & Liang (2021) further generalize prompt embeddings to a multi-layer setting. Razdaibiedina et al. (2023) re-parameterize prompt tuning by incorporating a feed-forward neural network with residual connections. Shi & Lipani (2024) observe that redistributing parameters to learn offsets for input token embeddings can enhance performance. On the other hand, multi-task prompt tuning has been explored, where the learned prompt parameters are reused across different tasks (Wang et al., 2023). Closely with our work, Xiao et al. (2023) decompose the prompt embedding matrix into two low-rank components: a low-dimensional prompt matrix and a learnable up-projection matrix. By contrast, our ULPT method freezes the up-projection matrix, so that we are able to achieve high performance with much fewer trainable parameters, supported by random-projection theory (Bingham & Mannila, 2001). Overall, our approach is able to function well with an ultra-low dimension, making it practical to customize millions of LLMs and perform continual learning in an ever-changing environment.

3. Methodology

3.1. Problem Formulation

Prompt tuning introduces learnable token embeddings in the input layer of a language model (Lester et al., 2021). These embeddings are optimized via gradient descent based on the task-specific loss signals. During optimization, the model weights remain frozen, while the gradient is backpropagated to the input layer to update the learnable embeddings. Typically, learnable prompt embeddings $\boldsymbol{e}_1, \cdots, \boldsymbol{e}_n \in \mathbb{R}^d$ serve as a prefix (Li & Liang, 2021), followed by the text prompt, which is tokenized and represented by token embeddings $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l \in \mathbb{R}^d$. Overall, the LLM has an input in the form of

$$(\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_n, \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_l) \tag{1}$$

where n is a predefined prompt length and l represents the length of the tokenized text. The objective is to optimize the prompt embedding matrix $\boldsymbol{E} \in \mathbb{R}^{n \times d}$ over a given dataset $\mathcal{D}$ based on the conditional log-likelihood:

$$\arg\max_{\boldsymbol{E}} \sum_{(x,y)\in\mathcal{D}} \log P(y \mid \boldsymbol{E}, x) \tag{2}$$

where $(x, y) \in \mathcal{D}$ represents input–output pairs in a dataset.

3.2. Our Ultra-Low-Dimensional Prompt Tuning

The learnable prompt embeddings do not inherently need to match the model dimension $\mathbb{R}^d$ due to the low intrinsic dimensionality of downstream tasks (Aghajanyan et al., 2021; Qin et al., 2022). Inspired by low-rank adaptation (Hu et al., 2022), the prompt embedding matrix $\boldsymbol{E}$ can be decomposed into the product of two matrices: $\boldsymbol{E} = \boldsymbol{Z}\boldsymbol{P}$, where $\boldsymbol{Z} \in \mathbb{R}^{n \times r}$ represents the prompt embeddings in an ultra-low r-dimensional space, and $\boldsymbol{P} \in \mathbb{R}^{r \times d}$ is a projection

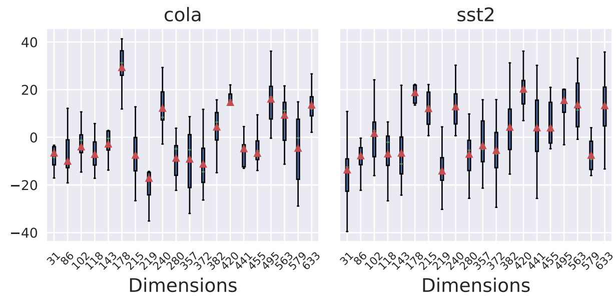

Figure 2. Distribution of prompt embedding values over 100 prompt tokens. We randomly selected 20 dimensions from the original prompt embeddings, which have 768 dimensions as in the T5-base model. The mean, 25/75 percentiles, and min/max are shown for the embedding values learned in the CoLA and SST-2 tasks (details explained in §4.1).

matrix that maps the low-dimensional embeddings back to the model's embedding space.

A naïve implementation of this decomposition treats both $\boldsymbol{Z}$ and $\boldsymbol{P}$ as learnable parameters (Xiao et al., 2023), which reduces the number of trainable parameters to $nr + rd$. This, unfortunately, scales poorly for larger models, as an $r \times d$ up-projection matrix should be learned and stored, which undermines the savings of learnable parameters.

To address this limitation, we propose an ultra-low-dimensional prompt tuning (ULPT) method that only learns r-dimensional prompt embeddings $\boldsymbol{Z}$, while keeping the projection $\boldsymbol{P}$ *randomly initialized* and *frozen* during training, denoted by $\tilde{\boldsymbol{P}} \in \mathbb{R}^{r \times d}$. In implementation, we only need to store one single number—the random seed of a random number generator—to reconstruct $\tilde{\boldsymbol{P}}$ when an LLM is loaded.

In this way, we completely eliminate the need for storing the up-project matrix. This not only largely reduces the learnable parameters from $nr + rd$ to nr (plus one extra random seed), but also combats overfitting especially when the fine-tuning dataset is small.

In our pilot study, we observe that typical prompt embeddings $\boldsymbol{E}$, even without low-rank treatment, exhibit significant variation across different dimensions, as shown in Figure 2. These variations may cause difficulty during training, therefore, we further introduce a learnable *shift* embedding $\boldsymbol{s} \in \mathbb{R}^d$ and a learnable *scale* embedding $\boldsymbol{b} \in \mathbb{R}^d$ to adjust the projected embeddings to ensure better alignment with the varying distributions across dimensions. Notice that the shift and scale embeddings are shared across different prompt token positions, but may vary for different tasks.

Specifically, an entry $\hat{e}_{ij}$ in the up-projected embedding matrix $\hat{\boldsymbol{E}}$ has the following form:

$$\hat{e}_{ij} = \left(\sum_{k=1}^{r} z_{ik}\tilde{p}_{kj}\right) s_j + b_j, \tag{3}$$

where z_{ik} and $\tilde{p}_{kj}$ are an entry in $\boldsymbol{Z}$ and $\tilde{\boldsymbol{P}}$ matrices, respectively; s_j and b_j are an entry in $\boldsymbol{s}$ and $\boldsymbol{j}$ vectors, respectively.

Such a treatment introduces two d-dimensional vectors, resulting in the total number of trainable parameters being $nr + 2d$. This is significantly more parameter-efficient than full-dimension prompt tuning with nd-many parameters (Lester et al., 2021) and vanilla low-rank prompt tuning with $(nr + rd)$-many parameters (Xiao et al., 2023).

3.3. Theoretical Analyses

We first show that an ultra low-dimensional space can capture the structure of the original embeddings (i.e., expressiveness). We then show the convergence of gradient descent with our random projection (i.e., optimization).

Expressiveness. Our low-dimensional parameterization approximately captures high-dimensional structure with high confidence. To show this, we first state the following lemma.

Lemma 1. *Sample a random matrix $\boldsymbol{A} \in \mathbb{R}^{r\times m}$ such that each element follows the standard Gaussian distribution. Let $\epsilon \in (0, 1/2]$ and $r \in \mathbb{N}_+$. There exists a constant c such that*

$$\Pr\left(\left|\frac{(1/\sqrt{r})\|\boldsymbol{A}\boldsymbol{x}\| - \|\boldsymbol{x}\|}{\|\boldsymbol{x}\|}\right| \geq \epsilon\right) \leq \frac{2}{\exp\left(\epsilon^2 r/\, c\right)} \tag{4}$$

for any $\boldsymbol{x} \in \mathbb{R}^d$.

This result is adapted from Indyk & Motwani (1998). Essentially, the lemma characterizes the high-probability bound of the well known Johnson–Lindenstrauss lemma (Dasgupta & Gupta, 2003; Matoušek, 2008). Based on this, we formally show the expressiveness of our ultra low-dimensional embeddings in the following theorem.

Theorem 2. *Let $\boldsymbol{e}_1, \cdots, \boldsymbol{e}_n \in \mathbb{R}^d$ be the embedding vectors in the high-dimensional space. Let $\boldsymbol{P} \in \mathbb{R}^{r\times d}$ be a random projection matrix with each element $p_{i,j} \sim \mathcal{N}(0, 1/r)$. There exists a set of low-dimensional vectors $\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n \in \mathbb{R}^r$ such that with confidence at least $1 - \delta$ we have*

$$(1-\epsilon)\|\boldsymbol{e}_i - \boldsymbol{e}_j\| \leq \|\boldsymbol{z}_i - \boldsymbol{z}_j\| \leq (1+\epsilon)\|\boldsymbol{e}_i - \boldsymbol{e}_j\| \tag{5}$$

for all $i, j \in [n]$, as long as $r \geq 2c\epsilon^{-2}\log(2n/\delta)$.

Proof. See Appendix B.1. □

In essence, our theorem asserts that there exists a set of low-dimensional vectors such that the pair-wise L^2 distances of the original high-dimensional vectors are preserved for all (i, j) pairs. More importantly, the projected dimension r only grows logarithmically with the original dimension n, demonstrating a favorable property of scaling. It should be noted that, although our theorem uses L^2 as the metric, it can be easily extended to the dot-product metric as well by noticing that $\|\boldsymbol{x} - \boldsymbol{y}\|^2 = \|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2 - 2\boldsymbol{x}\cdot\boldsymbol{y}$.

Optimization. The above theorem shows the existence of an expressive low-dimensional space. We assert in the following theorem that, given a random up-projection matrix, the optimal low-dimensional embeddings can be learned by gradient descent under mild assumptions.

Theorem 3. *Assume the original loss function $\mathcal{L}$ is Polyak–Lojasiewic and element-wise Lipschitz on the original d-dimensional embeddings. Let $\boldsymbol{P} \in \mathbb{R}^{r\times d}$ be a given full-rank random Gaussian matrix (i.e., rank r), and our parametrization be $\hat{\boldsymbol{e}}_i = \operatorname{diag}(\boldsymbol{s})\boldsymbol{P}^\top\boldsymbol{z}_i + \boldsymbol{b}$. With a proper learning rate schedule $\eta_1, \eta_2, \ldots$, our parameters $\boldsymbol{x} = [\boldsymbol{b}, \boldsymbol{s}, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n]$ converge to the global optimum with gradient descent if $\boldsymbol{s}$ is always non zero.*

Proof. See Appendix B.2. □

Theorem 3 shows that, even with the naïve gradient descent, the fixed random matrix $\boldsymbol{P}$ does not hinder the optimization procedure. By combining Theorem 2, we theoretically justifies our overall practice of ULPT.

4. Experiments

4.1. Experimental Settings

Datasets. We evaluate the proposed ULPT method across 21 NLP tasks following prior work (Asai et al., 2022; Wang et al., 2023; Shi & Lipani, 2024). Those tasks are grouped into 4 categories: (1) **GLUE** (Wang et al., 2018) is a benchmark suite consisting of various language understanding tasks, such as MNLI (Williams et al., 2018), QQP (Wang et al., 2018), QNLI (Demszky et al., 2018), SST-2 (Socher et al., 2013), STS-B (Cer et al., 2017), MRPC (Dolan & Brockett, 2005), RTE (Giampiccolo et al., 2007) and CoLA (Warstadt et al., 2019). (2) **SuperGLUE** (Wang et al., 2019) extends GLUE with more challenging tasks with limited training data, consisting of MultiRC (Khashabi et al., 2018), BoolQ (Clark et al., 2019), WiC (Pilehvar & Camacho-Collados, 2019), WSC (Levesque et al., 2012), and CB (De Marneffe et al., 2019). (3) The **MRQA** 2019 Shared Tasks (Fisch et al., 2019) are a set of QA tasks to test LLM generation capabilities, consisting of Natural Questions (Kwiatkowski et al., 2019), HotpotQA (Yang et al., 2018), SearchQA (Dunn et al., 2017) and NewsQA (Trischler et al., 2017). (4)

Table 1. Performance on GLUE and SuperGLUE benchmarks based on the T5-base model. We report standard evaluation metrics, namely, Pearson correlation for STS-B, F1 for MultiRC, and accuracy for other tasks. [†]We replicate prompt tuning (PT; Lester et al., 2021) and DPT (Xiao et al., 2023) using their default configurations. Our replicated PT results slightly exceed those reported in previous studies. All other baseline results are directly sourced from Shi & Lipani (2024). [‡]The suggested rank for DPT is r=10 based on Xiao et al. (2023); we replicate DPT with r=64 for controlled comparison with our ULPT. [t]Transfer learning methods. [m]Multi-task learning methods, whose "#param/task" scores are calculated based on the GLUE benchmark.

Method	#Param/ Task	GLUE									SuperGLUE					
		MNLI	QQP	QNLI	SST-2	STS-B	MRPC	RTE	CoLA	Avg.	MultiRC	Bool	WiC	WSC	CB	Avg.
Single-Task Learning																
Fine-tuning	220M	86.8	91.6	93.0	94.6	89.7	90.2	71.9	61.8	84.9	72.8	81.1	70.2	59.6	85.7	73.9
Adapter	1.9M	86.5	90.2	93.2	93.8	90.7	85.3	71.9	64.0	84.5	75.9	82.5	67.1	67.3	85.7	75.7
AdapterDrop	1.1M	86.3	90.2	93.2	93.6	91.4	86.3	71.2	62.7	84.4	72.9	82.3	68.3	67.3	85.7	75.3
BitFit	280K	85.3	90.1	93.0	94.2	90.9	86.8	67.6	58.2	83.3	74.5	79.6	70.0	59.6	78.6	72.5
LoRA	3.8M	86.3	89.0	93.2	94.3	90.0	90.1	75.5	63.3	85.3	72.6	81.3	68.3	67.3	92.9	76.5
LST	3.8M	85.6	88.8	93.3	94.0	90.7	90.4	71.9	58.1	84.1	–	–	–	–	–	–
PT[†]	76.8K	84.6	90.2	93.3	94.4	90.5	88.7	77.7	59.5	84.9	72.3	80.4	67.7	67.3	78.6	73.3
DePT	76.8K	85.0	90.4	93.2	94.2	90.8	90.7	79.1	63.8	85.9	74.3	79.3	68.7	67.3	92.9	76.5
DPT[†](r=10)	9.0K	84.4	90.2	93.3	94.6	91.2	87.7	77.7	57.8	84.6	74.5	78.7	66.8	67.3	71.4	71.7
DPT[‡](r=64)	55.6K	85.2	90.3	92.9	93.6	90.4	88.2	79.1	63.5	85.4	73.2	80.1	63.0	67.3	85.7	73.9
ULPT (r=2)	1.7K	81.9	90.3	92.3	92.9	89.8	89.2	76.3	59.5	84.0	73.4	76.7	67.4	67.3	71.4	71.2
ULPT (r=16)	3.1K	82.9	90.0	93.1	93.8	90.5	89.2	80.6	54.3	84.3	72.6	77.7	66.1	67.3	89.3	74.6
ULPT (r=64)	7.9K	84.9	90.3	93.1	93.5	90.7	90.2	81.3	63.7	**86.0**	73.1	78.2	69.0	67.3	96.4	**76.8**
ULPT (r=256)	27.1K	85.5	90.3	92.8	94.3	90.6	90.7	76.3	63.7	85.5	74.3	79.9	63.3	67.3	89.3	74.8
Multi-Task Learning & Transfer Learning																
Fine-tuning[m]	28M	85.7	91.1	92.0	92.5	88.8	90.2	75.4	54.9	83.8	74.4	81.1	70.0	71.2	85.7	76.1
Adapter[m]	1.8M	86.3	90.5	93.2	93.0	89.9	90.2	70.3	61.5	84.4	72.6	82.3	66.5	67.3	89.3	75.6
HyperFormer[m]	638K	85.7	90.0	93.0	93.0	89.7	87.2	75.4	63.7	84.8	72.9	82.5	69.0	67.3	85.7	75.4
HyperDecoder[m]	1.8M	86.0	90.5	93.4	94.0	90.5	87.7	71.7	55.9	83.7	70.4	78.8	67.1	61.5	82.1	72.0
SPoT[t]	76.8K	85.4	90.1	93.0	93.4	90.0	79.7	69.8	57.1	82.3	74.0	77.2	67.0	50.0	46.4	62.9
ATTEMPT[t]	232K	84.3	90.3	93.0	93.0	89.7	85.7	74.3	57.4	83.4	74.4	78.8	66.8	53.8	78.6	70.5
MPT[t]	77.6K	85.9	90.3	93.1	93.8	90.4	89.1	79.4	62.4	85.6	74.8	79.6	69.0	67.3	79.8	74.1
ATTEMPT[t+m]	96K	83.8	90.0	93.1	93.7	90.8	86.1	79.9	64.3	85.2	74.4	78.5	66.5	69.2	82.1	74.1
MPT[t+m]	10.5K	84.3	90.0	93.0	93.0	90.4	89.2	82.7	63.5	85.8	74.8	79.6	70.2	67.3	89.3	76.1

Other tasks beyond the above test suites are also considered, including WinoGrande (Sakaguchi et al., 2021), Yelp-2 (Zhang et al., 2015), SciTail (Khot et al., 2018), and PAWS-Wiki (Zhang et al., 2019). Further details on these datasets are provided in Table 5 in Appendix A.

Baselines. We compare our ULPT against a wide range of baselines to demonstrate its effectiveness and parameter efficiency. First, we evaluate against full-model fine-tuning, which optimizes all model parameters for downstream task adaptation, serving as a strong but parameter-intensive baseline. Second, we include state-of-the-art parameter-efficient methods such as Adapter (Houlsby et al., 2019), AdapterDrop (Rücklé et al., 2021), BitFit (Ben Zaken et al., 2022), HyperFormer (Karimi Mahabadi et al., 2021), HyperDecoder (Ivison & Peters, 2022), LoRA (Hu et al., 2022), and Ladder Side-Tuning (LST; Sung et al., 2022). Third, we compare ULPT with vanilla prompt tuning (PT) and its variants, including DePT (Shi & Lipani, 2024), which learns offsets to the input token embeddings while using a separate learning rate for the prompt embeddings, and DPT (Xiao et al., 2023), which is closely related to ULPT and decomposes prompt embeddings into low-rank matrices. Finally, we compare ULPT with transfer or multi-task learning methods, including SPoT (Vu et al., 2022), ATTEMPT (Asai et al., 2022), and MPT (Wang et al., 2023).

Implementation details. In our pilot study (Figure 2), we perform vanilla prompt tuning on the T5-base model (Raffel et al., 2020) with CoLA and SST-2, using $n = 100$ prompt embeddings, each having a dimensionality of $d = 768$. We randomly select 20 dimensions and report the mean, 25th/75th percentiles, and the minimum/maximum values for each dimension.

In our main experiment, we use the T5-base model with $d = 768$. Consistent with prior work (Shi & Lipani, 2024; Xiao et al., 2023), we set the number of prompt tokens $n = 100$ for the prompt embeddings $\boldsymbol{Q} \in \mathbb{R}^{n\times r}$ of our ULPT. For the rank r, we evaluate four configurations: $r = 2$, 16, 64, and 256, ranging from an ultra-low-dimensional setup to a more expressive configuration of $1/3$ of the original prompt dimension. All experiments use a batch size of 16 and a default learning rate of $6e{-}1$ with AdamW. The learning rate follows a linear schedule, warming up for 500 steps and then decaying linearly to 0. We set a maximum sequence length of 256 for most tasks, except for SuperGLUE-MultiRC being 348 and MRQA being 512. ULPT is trained on all tasks for up to $100,000$ steps. Performance is evaluated every $1,000$ steps, with the best

Table 2. Performance on MRQA and other benchmarks using the T5-base model. The standard metrics reported are the F1 score for MRQA tasks and accuracy for other datasets. †Results are obtained based on our replication using default configurations. Other baseline results are sourced from Shi & Lipani (2024).

		MRQA					Others				
Method	**#Param**	**NQ**	**HQA**	**SQA**	**NewsQA**	**Avg.**	**WG**	**Yelp**	**SciTail**	**PAWS**	**Avg.**
Fine-tuning	220M	75.1	77.5	81.1	65.2	**74.7**	61.9	96.7	95.8	94.1	**87.1**
Adapter	1.9M	74.2	77.6	81.4	65.6	74.7	59.2	96.9	94.5	94.3	86.2
BitFit	280K	70.7	75.5	77.7	64.1	72.0	57.2	94.7	94.7	92.0	84.7
LoRA	3.8M	72.4	62.3	72.5	56.9	66.0	58.2	97.1	94.7	94.0	86.0
SPoT	76.8K	68.2	74.8	75.3	58.2	69.1	50.4	95.4	91.2	91.1	82.0
ATTEMPT	232K	70.4	75.2	78.5	62.8	71.4	57.6	96.7	93.1	92.1	84.9
PT†	76.8K	70.0	74.7	75.3	63.0	70.8	49.6	95.6	92.0	57.9	73.8
DPT†(r=10)	9.0K	71.3	75.5	76.3	63.5	71.7	49.6	96.1	95.6	92.2	83.4
DPT (r=256)	222K	71.4	76.0	77.6	64.2	72.3	49.6	96.3	95.2	55.8	74.2
DePT	76.8K	73.2$_{0.3}$	76.0$_{0.2}$	77.6$_{0.2}$	64.4$_{0.1}$	73.0	59.0$_{0.2}$	96.8$_{0.1}$	95.6$_{0.2}$	93.7$_{0.1}$	86.3
MPT	77.6K	72.0$_{0.1}$	75.8$_{0.1}$	77.2$_{0.1}$	63.7$_{0.1}$	72.2	56.5$_{0.9}$	96.4$_{0.0}$	95.5$_{0.3}$	93.5$_{0.1}$	85.5
ULPT (r=2)	1.7K	67.2$_{0.2}$	74.0$_{0.1}$	71.7$_{0.2}$	61.4$_{0.1}$	68.6	49.5$_{0.2}$	95.6$_{0.1}$	93.0$_{0.9}$	90.4$_{0.2}$	82.1
ULPT (r=16)	3.1K	68.0$_{0.3}$	74.3$_{0.0}$	72.9$_{0.1}$	61.3$_{0.5}$	69.1	52.3$_{0.9}$	95.6$_{0.2}$	93.1$_{0.7}$	90.5$_{0.3}$	82.9
ULPT (r=64)	7.9K	70.7$_{0.3}$	75.3$_{0.1}$	75.3$_{0.1}$	62.9$_{0.5}$	71.1	56.6$_{0.9}$	96.2$_{0.1}$	94.4$_{0.9}$	91.7$_{0.4}$	84.7
ULPT (r=256)	27.1K	72.6$_{0.2}$	76.5$_{0.1}$	77.9$_{0.1}$	64.2$_{0.2}$	72.8	57.6$_{0.8}$	96.6$_{0.2}$	96.2$_{0.1}$	93.0$_{0.1}$	85.9

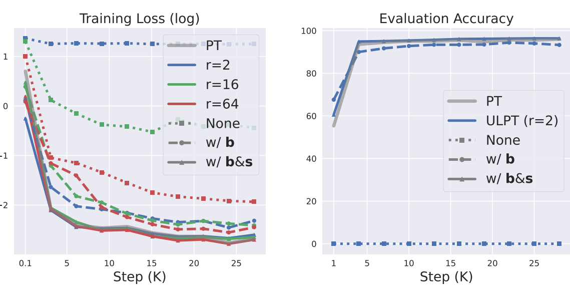

Figure 3. **Left:** Training loss curves on SST2 comparing ULPT with and without learnable shift and scale embeddings across different rank configurations. **Right:** Evaluation accuracy curves on SST2. For clarity, we present the case $r = 2$, where our ULPT is at a disadvantage. The trend for other configurations is similar.

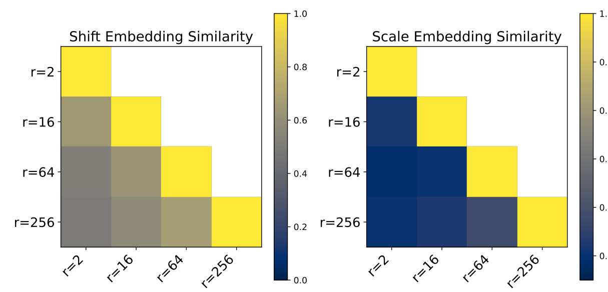

Figure 4. Pairwise similarities of the learned shift (**left**) and scale (**right**) embeddings for various rank configurations on SST-2.

checkpoint selected based on the validation set.

In our analysis, T5-small ($d = 512$) and T5-large model ($d = 1024$) are considered to evaluate the generality of ULPT across different model sizes. We also vary the number of prompt tokens from 10 to 100 under different rank configurations. Further details are provided in §4.3.

4.2. Main Results

Performance on GLUE and SuperGLUE. As shown in Table 1, our ULPT achieves similar or higher performance on GLUE and SuperGLUE benchmark datasets compared with previous methods, while maintaining remarkable parameter efficiency.

Profoundly, the ultra-low-rank configuration of $r = 2$ retains at least 97% performance of vanilla prompt tuning (PT), achieving average accuracy points of 84.0 on GLUE and 71.2 on SuperGLUE with only 2% of the parameters. This highlights the capability of ULPT and its advantage in large-scale LLM customization.

With a moderate rank of $r = 64$, our ULPT outperforms that with $r = 256$ and other state-of-the-art models, showing that our approach not only reduces the number of parameters but also alleviates the overfitting problem. Specifically, the DPT model (Xiao et al., 2023) learns an up-projection matrix, resulting in lower performance and 7x more parameters when the rank is controlled; even with the best rank $r = 10$ suggested by the original paper (Xiao et al., 2023), DPT is inadmissible as it is worse than our ULPT (with $r = 64$) in both parameter efficiency and performance.

Our ULPT also has significant advantages in multi-task setups. A transfer learning method initializes a model by task mixtures and then adapts it to a specific task; therefore, it cannot save parameters. Example studies of transfer learning include SPoT (Vu et al., 2022) and ATTEMPT (Asai et al., 2022). Our ULPT approach outperforms them in terms of accuracy and parameter efficiency, while offering a simpler training pipeline. Multi-task learning, on the other hand, shares certain parameters across different tasks (Karimi Mahabadi et al., 2021; Ivison & Peters, 2022; Wang et al., 2023), and thus, the parameter efficiency is measured on a per-task basis. Despite this, our ULPT still outperforms multi-task prompt tuning methods in both accuracy and per-task parameter efficiency due to its ultra-low-dimensional nature.

Performance on MRQA and other NLP tasks. Table 2 presents the results on the MRQA dataset and four additional tasks in the "Others" category. Following the standard practice on these benchmarks (Wang et al., 2023; Shi & Lipani, 2024), we run ULPT three times with different seeds and report the mean and standard deviation.

Unlike GLUE and SuperGLUE performance, ULPT exhibits consistent improvement when the rank is higher. This is probably because these tasks are more challenging, which aligns with the observation that full-model fine-tuning outperforms parameter-efficient methods on these tasks. Nevertheless, ULPT achieves competitive performance (slightly worse than the best-performing DePT approach), while saving parameters by multiple folds.

4.3. In-Depth Analyses

Ablation study on shift and scale embeddings. We conduct an ablation study on the learnable *shift* embedding $\boldsymbol{b} \in \mathbb{R}^d$ and *scale* embedding $\boldsymbol{s} \in \mathbb{R}^d$, using the SST-2 dataset[2] with the T5-base model as the testbed, where we set the token number to be $n = 100$. The results are shown in Figure 3. As seen, the dotted lines correspond to removing both shift and scale embeddings; their training loss remains high, suggesting that naïvely freezing the projection matrix $\tilde{\boldsymbol{P}}$ hinders the optimization process and consequently lowers the model performance. Introducing a learnable shift embedding $\boldsymbol{b}$ provides a substantial improvement (dashed lines), particularly in the low-dimensional configuration of $r = 2$. A learnable scale embedding *scale* $\boldsymbol{s}$ further improves the training process and performance (solid lines). The ablation study shows that, although shift and scale embeddings are additional $2d$-many parameters, they play an important role in ultra-low-dimensional prompt tuning.

To further investigate the behavior of these embeddings, we analyze the pairwise cosine similarities of the shift $\boldsymbol{s}$ and scale $\boldsymbol{b}$ embeddings under different rank configurations, visualized in Figure 4. We see that they exhibit interesting patterns: the learned shift embeddings have consistently high similarity scores with different rank configurations, indicating their primary role as an alignment mechanism after up-projection. By contrast, the scale embeddings show near-zero pairwise similarities, as they depend on the sampled (and frozen) random projection matrix $\tilde{\boldsymbol{P}}$.

[2] Our preliminary experiments show that prompt tuning on SST-2, a smaller dataset, leads to faster convergence, making it suitable for validating the effectiveness of the ablated models. However, for the rest of the analysis, we use MNLI and Natural Questions, to better test the expressiveness of the ablated models and reduce the risk of overfitting observed in our main experiment.

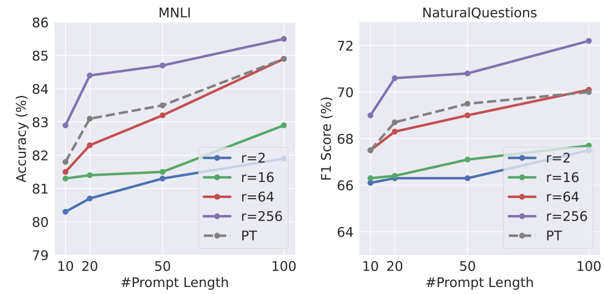

Figure 5. Results on MNLI and Natural Questions with the T5-base model. The number of prompt tokens for both ULPT and naïve prompt tuning varies from 10 to 100.

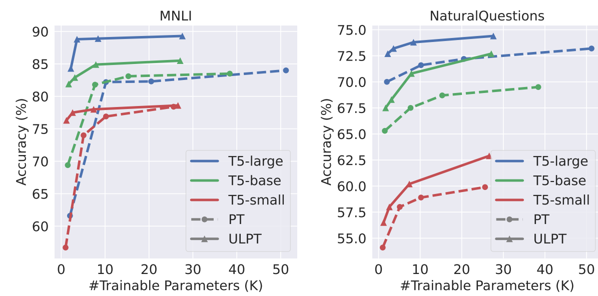

Figure 6. Results on MNLI and Natural Questions with controlled numbers of trainable parameters, comparing ULPT and prompt tuning across three T5 model sizes (small, base, and large).

Table 3. Results on MNLI and Natural Questions for training either $\boldsymbol{P}$ or $\boldsymbol{Z}$ with T5-base. Numbers in the brackets refer to the rank r given the controlled number of parameters.

	Train?		#Trainable Parameters			
Dataset	$\boldsymbol{Z}$	$\boldsymbol{P}$	1.7K	3.1K	7.9K	27.1K
MNLI	✓		**81.9** (2)	**82.9** (16)	**84.9** (64)	**85.5** (256)
		✓	–	**82.9** (2)	84.5 (8)	85.3 (33)
NQ	✓		**67.2**$_{0.2}$ (2)	**68.0**$_{0.4}$ (16)	**70.7**$_{0.3}$ (64)	**72.6**$_{0.2}$ (256)
		✓	–	66.9 (2)	70.0 (8)	72.0 (33)

Analysis of prompt lengths and dimensions. Recall that Table 1 has analyzed our ULPT performance with different ranks. We now vary the number of prompt tokens and plot the trend in Figure 5. We see that our ULPT exhibits a similar trend as vanilla prompting, where the performance increases with a longer prompt. With an appropriate rank configuration, our ULPT consistently outperforms vanilla

Table 4. Results on Bloomz, a decoder model with varying sizes (560M, 1.7B, and 3B) and hidden dimensions (1024, 2048, and 2560). We compare ULPT with prompt tuning by conditioning on the same number of trainable parameters.

Model	Method	SST-2	HQA	WG	**Avg.**	SST-2	HQA	WG	**Avg.**	SST-2	HQA	WG	**Avg.**	SST-2	HQA	WG	**Avg.**
		#Param=2K, ULPT (r=2)				#Param=4K, ULPT (r=16)				#Param=8K, ULPT (r=64)				#Param=28K, ULPT (r=256)			
Bloomz-560M	PT	89.8	42.9	48.6	60.4	91.1	53.0	52.0	65.4	91.9	57.2	50.0	66.4	92.2	60.6	52.2	68.3
	ULPT	90.2	52.4	51.7	**64.8**	92.2	55.7	53.1	**67.0**	91.8	59.3	53.1	**68.1**	92.6	62.5	52.2	**69.1**
		#Param=4K, ULPT (r=2)				#Param=6K, ULPT (r=16)				#Param=10K, ULPT (r=64)				#Param=30K, ULPT (r=256)			
Bloomz-1.7B	PT	93.2	64.6	50.1	69.3	93.5	66.1	51.5	70.4	94.0	67.3	55.3	72.2	94.7	69.1	55.4	73.1
	ULPT	94.4	65.6	54.6	**71.5**	93.9	66.3	55.6	**71.9**	94.3	68.0	55.2	**72.5**	95.1	69.3	57.4	**73.9**
		#Param=5K, ULPT (r=2)				#Param=8K, ULPT (r=16)				#Param=13K, ULPT (r=64)				#Param=31K, ULPT (r=256)			
Bloomz-3B	PT	93.2	66.1	50.5	69.9	94.5	69.0	56.0	73.2	94.9	69.1	58.9	74.3	94.9	71.5	60.0	75.5
	ULPT	94.0	68.1	53.5	**71.9**	94.4	68.9	57.1	**73.5**	94.7	70.9	58.5	**74.7**	95.0	71.8	60.7	**75.8**

prompt tuning under different lengths.

Our low-rank ULPT provides a trade-off between the prompt length and dimension. We compare ULPT with vanilla prompt tuning when the learnable parameters are controlled. For our ULPT, we keep the prompt token number as 100 and vary the rank from 2 to 256; for vanilla full-dimensional prompt tuning, we vary the token number from 2 to 50. This analysis is also conducted with three model sizes: T5-small, T5-base, and T5-large.

Figure 6 illustrates the results, showing that our low-dimensional ULPT with more tokens (solid lines) always outperform vanilla full-dimensional prompt tuning with fewer tokens (dashed lines). The analysis suggests that, when the number of learnable parameters is controlled, a longer prompt with a lower dimension offers more flexibility due to the additional Transformer steps.

Comparison with an alternative method of tuning $\boldsymbol{P}$. The low-rank decomposition $\boldsymbol{E} = \boldsymbol{Z}\boldsymbol{P}$ allows an alternative approach that freezes $\boldsymbol{Z}$ and tunes $\boldsymbol{P}$, which contrasts with our approach that freezes $\boldsymbol{P}$ and tunes $\boldsymbol{Z}$. The comparison is shown in Table 3. The alternative setup (tuning $\boldsymbol{P}$) can be viewed as learning an up-projection from a set of random but frozen low-dimensional vectors. However, a key drawback of making $\boldsymbol{P}$ trainable is the rapid growth in the number of parameters when the rank r increases, since $d \gg n$ in most practical scenarios. To ensure a fair comparison, we control the number of parameters by varying the rank r for both methods.

As seen, tuning $\boldsymbol{P}$ fails to be feasible in the 1.7K-parameter setup. Even if we set $r = 2$, tuning $\boldsymbol{P}$ results in 3.1K parameters, equivalent to our $r = 16$ setup. With a larger budget, tuning $\boldsymbol{P}$ achieves slightly worse performance than our ULPT which tunes $\boldsymbol{Z}$. This analysis verifies the expressiveness of random projections; it also shows that our ULPT is superior to the alternative approach.

4.4. Results on Decoder Models

In our main experiments, we use the encoder–decoder T5 model (Raffel et al., 2020), following most previous work on prompt tuning (Lester et al., 2021; Wang et al., 2023; Shi & Lipani, 2024).

We extend the evaluation of ULPT to Bloomz (Muennighoff et al., 2023), a decoder-only model with three difference sizes: 560M, 1.7B, and 3B, having hidden dimensions of 1024, 2048, and 2560 respectively. For evaluation diversity, we select three mid-sized tasks from each task group: SST-2, HotpotQA, and Winogrande, providing assessment across classification, multi-hop reasoning, and coreference reasoning. Since Bloomz models are larger than the T5 series, we train up to 30K steps with a batch size of 4, while keeping other hyperparamters the same as our main experiment.

We consider comparing ULPT with prompt tuning under different parameter budgets for text generation. Specifically, we vary the rank of ULPT from 2 to 256 while fixing the length $n = 100$. For full-dimensional prompt tuning, the token number is adjusted to match the parameter count.

Results in Table 4 show that ULPT consistently outperforms prompt tuning across all model sizes and tasks given a fixed parameter budget. These findings align with our earlier analysis (§4.3), confirming that ULPT can be applied to different model architectures.

5. Conclusion

In this paper, we propose Ultra-Low-Dimensional Prompt Tuning (ULPT), a novel parameter-efficient prompt tuning method that achieves superior performance across diverse NLP tasks with significantly fewer trainable parameters. ULPT decouples prompt embeddings from the model's dimensionality, optimizing in a low-dimensional space and projecting into the model's embedding space by a frozen random projection. Our research offers future opportunities for large-scale LLM customizations, as efficient storage of task-specific models is increasingly critical.

6. Impact Statements

This paper presents a method aimed at enabling more parameter-efficient fine-tuning for large language models. By significantly improving the storage efficiency of prompt tuning, our approach makes it practical to create millions of customized AI systems, including those for personal use, thereby contributing to the democratization of access to large-scale customized AI solutions. No specific concerns require attention in this context.

References

Aghajanyan, A., Gupta, S., and Zettlemoyer, L. 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*, pp. 7319–7328, 2021. URL https://aclanthology.org/2021.acl-long.568.

Asai, A., Salehi, M., Peters, M., and Hajishirzi, H. ATTEMPT: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 6655–6672, 2022. URL https://aclanthology.org/2022.emnlp-main.446.

Ben Zaken, E., Goldberg, Y., and Ravfogel, S. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pp. 1–9, 2022. URL https://aclanthology.org/2022.acl-short.1.

Bingham, E. and Mannila, H. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 245–250, 2001. URL https://doi.org/10.1145/502512.502546.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, pp. 1877–1901, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., and Specia, L. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pp. 1–14, 2017. URL https://aclanthology.org/S17-2001/.

Choi, J.-Y., Kim, J., Park, J.-H., Mok, W.-L., and Lee, S. SMoP: Towards efficient and effective prompt tuning with sparse mixture-of-prompts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14306–14316, 2023. URL https://aclanthology.org/2023.emnlp-main.884.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2924–2936, 2019. URL https://aclanthology.org/N19-1300.

Dasgupta, S. and Gupta, A. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003. URL https://doi.org/10.1002/rsa.10073.

De Marneffe, M.-C., Simons, M., and Tonhauser, J. The commitmentbank: Investigating projection in naturally occurring discourse. In *Proceedings of Sinn und Bedeutung*, volume 23, pp. 107–124, 2019. URL https://semanticsarchive.net/Archive/Tg3ZGI2M/Marneffe.pdf.

Demszky, D., Guu, K., and Liang, P. Transforming question answering datasets into natural language inference datasets, 2018. URL https://arxiv.org/abs/1809.02922.

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

Dong, Q., Li, L., Dai, D., Zheng, C., Ma, J., Li, R., Xia, H., Xu, J., Wu, Z., Chang, B., Sun, X., Li, L., and Sui, Z. A survey on in-context learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1107–1128, 2024. URL https://aclanthology.org/2024.emnlp-main.64.

Dunn, M., Sagun, L., Higgins, M., Guney, V. U., Cirik, V., and Cho, K. SearchQA: A new Q&A dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*, 2017. URL https://arxiv.org/abs/1704.05179.

Fisch, A., Talmor, A., Jia, R., Seo, M., Choi, E., and Chen, D. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pp. 1–13, 2019. URL https://aclanthology.org/D19-5801.

Giampiccolo, D., Magnini, B., Dagan, I., and Dolan, B. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pp. 1–9, 2007. URL https://aclanthology.org/W07-1401.

Guo, S., Damani, S., and hao Chang, K. LoPT: Low-rank prompt tuning for parameter efficient language models, 2024. URL https://arxiv.org/abs/2406.19486.

Hao, Y., Cao, Y., and Mou, L. Flora: Low-rank adapters are secretly gradient compressors. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/hao24a.html.

Hayou, S., Ghosh, N., and Yu, B. LoRA+: Efficient low rank adaptation of large models. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/hayou24a.html.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2790–2799, 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Indyk, P. and Motwani, R. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pp. 604–613, 1998. URL https://dl.acm.org/doi/10.1145/276698.276876.

Ivison, H. and Peters, M. Hyperdecoders: Instance-specific decoders for multi-task NLP. In *Findings of the Association for Computational Linguistics: EMNLP*, pp. 1715–1730, 2022. URL https://aclanthology.org/2022.findings-emnlp.124.

Karimi, H., Nutini, J., and Schmidt, M. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases*, pp. 795–811, 2016. URL https://link.springer.com/chapter/10.1007/978-3-319-46128-1_50.

Karimi Mahabadi, R., Ruder, S., Dehghani, M., and Henderson, J. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 565–576, 2021. URL https://aclanthology.org/2021.acl-long.47.

Khashabi, D., Chaturvedi, S., Roth, M., Upadhyay, S., and Roth, D. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 252–262, 2018. URL https://aclanthology.org/N18-1023.

Khot, T., Sabharwal, A., and Clark, P. Scitail: A textual entailment dataset from science question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. URL https://ojs.aaai.org/index.php/AAAI/article/view/12022.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, pp. 22199–22213, 2022. URL https://openreview.net/pdf?id=e2TBb5y0yFf.

Kopiczko, D. J., Blankevoort, T., and Asano, Y. M. VeRA: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NjNfLdxr3A.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q., and Petrov, S. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, pp. 452–466, 2019. URL https://aclanthology.org/Q19-1026/.

Lester, B., Al-Rfou, R., and Constant, N. 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,

2021. URL https://aclanthology.org/2021.emnlp-main.243.

Levesque, H., Davis, E., and Morgenstern, L. The winograd schema challenge. In *Preceddings of the 13th International Conference on the Principles of Knowledge Representation and Reasoning*, 2012. URL https://cdn.aaai.org/ocs/4492/4492-21843-1-PB.pdf.

Li, X. L. and Liang, P. 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*, pp. 4582–4597, 2021. URL https://aclanthology.org/2021.acl-long.353.

Lialin, V., Muckatira, S., Shivagunde, N., and Rumshisky, A. ReloRA: High-rank training through low-rank updates. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=DLJznSp6X3.

Liu, X., Ji, K., Fu, Y., Tam, W., Du, Z., Yang, Z., and Tang, J. P-Tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pp. 61–68, 2022. URL https://aclanthology.org/2022.acl-short.8.

Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., and Tang, J. GPT understands, too. *AI Open*, 5:208–215, 2024. URL https://www.sciencedirect.com/science/article/pii/S2666651023000141.

Mangrulkar, S., Gugger, S., Debut, L., Belkada, Y., Paul, S., and Bossan, B. PEFT: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft, 2022.

Matoušek, J. On variants of the Johnson–Lindenstrauss lemma. *Random Structures & Algorithms*, 33(2):142–156, 2008. URL https://doi.org/10.1002/rsa.20218.

Mei, J., Xiao, C., Szepesvari, C., and Schuurmans, D. On the global convergence rates of Softmax policy gradient methods. In *ICML*, volume 119, pp. 6820–6829, 2020. URL https://proceedings.mlr.press/v119/mei20b.html.

Muennighoff, N., Wang, T., Sutawika, L., Roberts, A., Biderman, S., Le Scao, T., Bari, M. S., Shen, S., Yong, Z. X., Schoelkopf, H., Tang, X., Radev, D., Aji, A. F., Almubarak, K., Albanie, S., Alyafeai, Z., Webson, A., Raff, E., and Raffel, C. Crosslingual generalization through multitask finetuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pp. 15991–16111, 2023. URL https://aclanthology.org/2023.acl-long.891/.

Petrov, A., Torr, P., and Bibi, A. Prompting a pretrained transformer can be a universal approximator. In *Proceedings of the 41st International Conference on Machine Learning*, 2024a. URL https://proceedings.mlr.press/v235/petrov24a.html.

Petrov, A., Torr, P., and Bibi, A. When do prompting and prefix-tuning work? a theory of capabilities and limitations. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=JewzobRhay.

Pilehvar, M. T. and Camacho-Collados, J. WiC: The word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019. URL https://aclanthology.org/N19-1128.

Qin, Y., Wang, X., Su, Y., Lin, Y., Ding, N., Yi, J., Chen, W., Liu, Z., Li, J., Hou, L., Li, P., Sun, M., and Zhou, J. Exploring universal intrinsic task subspace via prompt tuning, 2022. URL https://arxiv.org/abs/2110.07867.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, pp. 1–67, 2020. URL https://jmlr.org/papers/v21/20-074.html.

Razdaibiedina, A., Mao, Y., Khabsa, M., Lewis, M., Hou, R., Ba, J., and Almahairi, A. Residual Prompt Tuning: improving prompt tuning with residual reparameterization. In *Findings of the Association for Computational Linguistics: ACL*, pp. 6740–6757, 2023. URL https://aclanthology.org/2023.findings-acl.421.

Rücklé, A., Geigle, G., Glockner, M., Beck, T., Pfeiffer, J., Reimers, N., and Gurevych, I. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021. URL https://aclanthology.org/2021.emnlp-main.626.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. WinoGrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, pp. 99–106, 2021. URL https://doi.org/10.1145/3474381.

Shi, Z. and Lipani, A. DePT: Decomposed prompt tuning for parameter-efficient fine-tuning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=KjegfPGRde.

Shin, T., Razeghi, Y., Logan IV, R. L., Wallace, E., and Singh, S. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 4222–4235, 2020. URL https://aclanthology.org/2020.emnlp-main.346.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. 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, 2013. URL https://aclanthology.org/D13-1170/.

Sung, Y.-L., Cho, J., and Bansal, M. LST: Ladder side-tuning for parameter and memory efficient transfer learning. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=isPnnaTZaP5.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. URL https://arxiv.org/abs/2302.13971.

Trischler, A., Wang, T., Yuan, X., Harris, J., Sordoni, A., Bachman, P., and Suleman, K. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 191–200, 2017. URL https://aclanthology.org/W17-2623.

Vu, T., Lester, B., Constant, N., Al-Rfou', R., and Cer, D. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pp. 5039–5059, 2022. URL https://aclanthology.org/2022.acl-long.346.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, 2018. URL https://aclanthology.org/W18-5446.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *arxiv*, 2019. URL http://arxiv.org/abs/1905.00537.

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., and Pfister, T. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, June 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Wang_Learning_To_Prompt_for_Continual_Learning_CVPR_2022_paper.html.

Wang, Z., Panda, R., Karlinsky, L., Feris, R., Sun, H., and Kim, Y. Multitask prompt tuning enables parameter-efficient transfer learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Nk2pDtuhTq.

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 2019. URL https://aclanthology.org/Q19-1040.

Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=gEZrGCozdqR.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., ichter, b., Xia, F., Chi, E., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, pp. 24824–24837, 2022b. URL https://openreview.net/pdf?id=_VjQlMeSB_J.

Williams, A., Nangia, N., and Bowman, S. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1112–1122, 2018. URL https://aclanthology.org/N18-1101/.

Wu, X., Huang, S., and Wei, F. Mixture of LoRA experts. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=uWvKBCYh4S.

Wu, Z., Arora, A., Wang, Z., Geiger, A., Jurafsky, D., Manning, C. D., and Potts, C. ReFT: Representation finetuning for language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=fykjplMc0V.

Wu, Z., Wu, Y., and Mou, L. Zero-shot continuous prompt transfer: Generalizing task semantics across language models. In *The Twelfth International Conference on Learning Representations*, 2024c. URL https://openreview.net/forum?id=26XphugOcS.

Xiao, Y., Xu, L., Li, J., Lu, W., and Li, X. Decomposed prompt tuning via low-rank reparameterization. In *Findings of the Association for Computational Linguistics: EMNLP*, pp. 13335–13347, 2023. URL https://aclanthology.org/2023.findings-emnlp.890.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., and Manning, C. D. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018. URL https://aclanthology.org/D18-1259/.

Yin, F., Ye, X., and Durrett, G. LoFiT: Localized fine-tuning on LLM representations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=dfiXFbECSZ.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pp. 649–657, 2015. URL https://proceedings.neurips.cc/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

Zhang, Y., Baldridge, J., and He, L. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1298–1308, 2019. URL https://aclanthology.org/N19-1131.

Zhao, B., Tu, H., Wei, C., Mei, J., and Xie, C. Tuning layernorm in attention: Towards efficient multi-modal LLM finetuning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=YR3ETaElNK.

Table 5. Dataset information and statistics.

Dataset	Source Length	Target Length	#Train	#Valid	#Test	Type	Size
GLUE Benchmark							
MNLI	31.8	1.0	392,702	9,832	9,815	Natural language inference	Large
QQP	24.1	1.0	362,846	1,000	40,431	Paraphrasing	Large
QNLI	38.4	1.0	103,743	1,000	5,463	Natural language inference	Large
SST-2	10.4	1.0	66,349	1,000	872	Sentiment analysis	Medium
STS-B	21.9	1.0	5,749	750	750	Sentence similarity	Small
MRPC	45.9	1.0	3,668	204	204	Paraphrasing	Small
RTE	54.4	1.0	2,490	138	139	Natural language inference	Small
CoLA	8.7	1.0	8,551	521	522	Acceptability	Small
SuperGLUE Benchmark							
MultiRC	286.1	1.0	27,243	2,424	2,424	Question answering	Medium
BoolQ	108.3	1.0	9,427	1,635	1,635	Question answering	Small
WiC	18.4	1.0	5,428	319	319	Word sense disambiguation	Small
WSC	28.1	1.0	554	52	52	Commonsense reasoning	Small
CB	64.6	1.0	250	28	28	Natural language inference	Small
MRQA 2019 Shared Task							
NaturalQuestions	242.7	4.5	103,071	1,000	12,836	Question answering	Large
HotpotQA	225.7	2.6	71,928	1,000	5,901	Question answering	Medium
SearchQA	942.8	2.0	116,384	1,000	16,980	Question answering	Large
NewsQA	615.5	5.1	73,160	1,000	4,212	Question answering	Medium
Other Datasets							
WinoGrande	23.8	1.0	39,398	1,000	1,267	Commonsense reasoning	Medium
YelpPolarity	134.0	1.0	100,000	1,000	38,000	Sentiment analysis	Large
SciTail	30.8	1.0	23,596	652	652	Natural language inference	Medium
PAWS	44.7	1.0	49,401	8,000	8,000	Sentence Similarity	Medium

A. Dataset Details

We present detailed information for the 21 NLP tasks in Table 5. Following previous work (Wang et al., 2023; Shi & Lipani, 2024), we preprocess the labels for classification and multiple-choice tasks into a single-token label (e.g., 0, 1, 2, . . .) to simplify evaluation. For MRQA, the model generates an answer containing a sequence of tokens.

Based on the training set size, the tasks can be roughly categorized into three scales: small (<10K samples), medium (10–100K samples), and large (>100K samples). Notably, SuperGLUE contains small training sets, and is generally considered more challenging than GLUE, making it more susceptible to overfitting due to its limited samples. By contrast, MRQA and the tasks in the “Others” category consist of more complex tasks, likely requiring more parameters to capture their difficulty.

B. Theoretical Results

B.1. Proof of Theorem 2

Theorem 2. *Let $\boldsymbol{e}_1, \cdots, \boldsymbol{e}_n \in \mathbb{R}^d$ be the embedding vectors in the high-dimensional space. Let $\boldsymbol{P} \in \mathbb{R}^{r\times d}$ be a random projection matrix with each element $p_{i,j} \sim \mathcal{N}(0, 1/r)$. There exists a set of low-dimensional vectors $\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n \in \mathbb{R}^r$ such that with confidence at least $1-\delta$ we have*

$$(1-\epsilon)\|\boldsymbol{e}_i - \boldsymbol{e}_j\| \le \|\boldsymbol{z}_i - \boldsymbol{z}_j\| \le (1+\epsilon)\|\boldsymbol{e}_i - \boldsymbol{e}_j\| \tag{5}$$

for all $i, j \in [n]$, as long as $r \ge 2c\epsilon^{-2}\log(2n/\delta)$.

Proof. Setting $\boldsymbol{z}_i = \boldsymbol{P}\boldsymbol{e}_i$, we have

$$\Pr\left(\left|\frac{\|\boldsymbol{z}_i - \boldsymbol{z}_j\| - \|\boldsymbol{e}_i - \boldsymbol{e}_j\|}{\|\boldsymbol{e}_i - \boldsymbol{e}_j\|}\right| \ge \epsilon\right) = \Pr\left(\left|\frac{\|\boldsymbol{P}\,(\boldsymbol{e}_i - \boldsymbol{e}_j)\,\| - \|\boldsymbol{e}_i - \boldsymbol{e}_j\|}{\|\boldsymbol{e}_i - \boldsymbol{e}_j\|}\right| \ge \epsilon\right) \tag{6}$$

$$\leq \frac{2}{\exp\left(\epsilon^2 r/c\right)}, \tag{7}$$

for any $i, j \in [n]$. The last inequality is a direction application of Lemma 1. Further, Boole's inequality suggests

$$\Pr\left(\text{any } i, j \in [n] : \left|\frac{\|\boldsymbol{z}_i - \boldsymbol{z}_j\| - \|\boldsymbol{e}_i - \boldsymbol{e}_j\|}{\|\boldsymbol{e}_i - \boldsymbol{e}_j\|}\right| \geq \epsilon\right) \leq n^2 \frac{2}{\exp\left(\epsilon^2 r/c\right)}, \tag{8}$$

where n^2 comes from counting all (i, j) pairs. By setting $\delta > 0$ to any value smaller than $\frac{2n^2}{\exp(\epsilon^2 r/c)}$, we have $r \geq 2c\epsilon^{-2} \cdot \log(2n/\delta)$. Therefore, Eqn. (8) can be rewritten as follows: with confidence at least $1 - \delta$, we have

$$(1 - \epsilon)\|\boldsymbol{e}_i - \boldsymbol{e}_j\| \leq \|\boldsymbol{z}_i - \boldsymbol{z}_j\| \leq (1 + \epsilon)\|\boldsymbol{e}_i - \boldsymbol{e}_j\| \tag{9}$$

for all $i, j \in [n]$, as long as $r \geq 2c\epsilon^{-2}\log(2n/\delta)$. □

B.2. Proof of Theorem 3

We first formally explain our assumptions.

Assumption 4. The loss function $\mathcal{L}$ is β element-wise Lipschitz w.r.t. embeddings. Specifically, we have

$$|\nabla\mathcal{L}(x_i) - \nabla\mathcal{L}(y_i)| \leq \beta|x_i - y_i| \tag{10}$$

for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^{nd}$ being unrolled from $n \times d$ embedding matrices. x_i and y_i are elements in the vectors.

Assumption 5. The loss function $\mathcal{L}$ is μ-PL (Polyak–Lojasiewic) w.r.t. embeddings, meaning that

$$\frac{1}{2}\|\nabla\mathcal{L}(\boldsymbol{x})\|_2^2 \geq \mu\left(\mathcal{L}(\boldsymbol{x}) - \mathcal{L}(\boldsymbol{x}^*)\right) \tag{11}$$

for any $\boldsymbol{x} \in \mathbb{R}^{nd}$, where $\boldsymbol{x}$ is embedding parameters and $\boldsymbol{x}^*$ is any finite minimizer of $\mathcal{L}$.

These are the common assumptions used to show the optimization process in deep learning (Karimi et al., 2016; Mei et al., 2020). In addition, we also impose an assumptions on the projection matrix and the scaling vector $\boldsymbol{s}$.

Assumption 6. The projection matrices $\boldsymbol{P} \in \mathbb{R}^{r\times d}$ has a rank of r. In addition, we assume $\boldsymbol{s}$ is not a zero vector during optimization.

Based on these assumptions, we first provide the essential lemmas for our proof.

Lemma 7. *If $\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$ is β-Lipschitz in each element, then $\mathcal{L}$ is β-Lipschitz.*

Proof. Let $\nabla\mathcal{L}(x_i)$ be the partial derivative of $\mathcal{L}$ w.r.t. x_i. We have

$$|\nabla\mathcal{L}(x_i) - \nabla\mathcal{L}(y_i)| \leq \beta|x_i - y_i| \tag{12}$$

for every $x_i, y_i \in \mathbb{R}$. Therefore,

$$\|\nabla\mathcal{L}(\boldsymbol{x}) - \nabla\mathcal{L}(\boldsymbol{y})\|^2 = \sum_{i=1}^{d} |\nabla\mathcal{L}(x_i) - \nabla\mathcal{L}(y_i)|^2 \tag{13}$$

$$\leq \sum_{i=1}^{d} \beta^2 |x_i - y_i|^2 \tag{14}$$

$$= \beta^2\|\boldsymbol{x} - \boldsymbol{y}\|^2. \tag{15}$$

We complete the proof by taking the square root on both sides. □

Lemma 8. *Let $\hat{\mathcal{L}}(\hat{\boldsymbol{x}})$ be the loss function with our ULPT approach, where $\hat{\boldsymbol{x}} \in \mathbb{R}^{nr+2d}$ is the concatenation of shift/scale embeddings and the ultra-low-dimensional prompt embeddings. $\hat{\mathcal{L}}(\hat{\boldsymbol{x}})$ is β'-Lipschitz w.r.t. $\hat{\boldsymbol{x}}$ for some $\beta' > 0$.*

Proof. We prove the Lipschitz condition of $\mathcal{L}$ w.r.t. the ultra-low-dimensional prompt embeddings, scale embedding, and shift embedding separately. Then, Lemma 7 suggests the Lipschitz condition of $\mathcal{L}$ w.r.t to $\hat{\boldsymbol{x}}$. Without loss of generality, we assume the layout of parameters is $\hat{\boldsymbol{x}} = [\boldsymbol{b}, \boldsymbol{s}, \boldsymbol{z}_1, \boldsymbol{z}_2, \dots, \boldsymbol{z}_n]$, where n is the number of prompt tokens.

We first calculate partial derivatives as follows

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{b}} = \sum_{i=1}^{n} \left(\frac{\partial \hat{\boldsymbol{e}}_i}{\partial \boldsymbol{b}}\right)^\top \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{e}}_i} = \sum_{i=1}^{n} \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{e}}_i}, \tag{16}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{s}} = \left(\frac{\partial \hat{\boldsymbol{e}}_i}{\partial \boldsymbol{s}}\right)^\top \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{e}}_i} = \sum_{i=1}^{n} \operatorname{diag}(\boldsymbol{P}^\top \boldsymbol{z}_i) \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{e}}_i}, \text{and} \tag{17}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}_i} = \left(\frac{\partial \hat{\boldsymbol{e}}_i}{\partial \boldsymbol{z}_i}\right)^\top \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{e}}_i} = \boldsymbol{P} \operatorname{diag}(\boldsymbol{s}) \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{e}}_i}. \tag{18}$$

Our proof of the Lipschitz condition starts with checking $\boldsymbol{b}$. For any element b_k, where $k = 1, \cdots, d$, we have

$$\begin{aligned}
\left|\nabla \hat{\mathcal{L}}(b_k^{(1)}) - \nabla \hat{\mathcal{L}}(b_k^{(2)})\right| &= \left|\sum_{i=1}^{n} \left(\nabla \mathcal{L}(\hat{e}_{i,k}^{(1)}) - \nabla \mathcal{L}(\hat{e}_{i,k}^{(2)})\right)\right| && (19)\\
&\le \sum_{i=1}^{n} \left|\nabla \mathcal{L}(\hat{e}_{i,k}^{(1)}) - \nabla \mathcal{L}(\hat{e}_{i,k}^{(2)})\right| && (20)\\
&\le L \sum_{i=1}^{n} |\hat{e}_{i,k}^{(1)} - \hat{e}_{i,k}^{(2)}| && (21)\\
&= nL|b_k^{(1)} - b_k^{(2)}| && (22)
\end{aligned}$$

where superscripts (1) and (2) indicate two values in the Lipschitz condition. $\hat{e}_{i,k}$ refers to the ith prompt token and its kth dimension. Here, the first equation is due to Eqn. (16).

For the scale embedding $\boldsymbol{s}$, we also consider the kth dimension for $k = 1, \cdots, d$:

$$\begin{aligned}
\left|\nabla \hat{\mathcal{L}}(s_k^{(1)}) - \nabla \hat{\mathcal{L}}(s_k^{(2)})\right| &= \left|\sum_{i} \left(\boldsymbol{z}_i^\top \boldsymbol{P}_{:,k} \nabla \mathcal{L}(\hat{e}_{i,k}^{(1)}) - \boldsymbol{z}_i^\top \boldsymbol{P}_{:,k} \nabla \mathcal{L}(\hat{e}_{i,k}^{(2)})\right)\right| && (23)\\
&= \left|\sum_{i} \left(\boldsymbol{z}_i^\top \boldsymbol{P}_{:,k}\right) \left(\nabla \mathcal{L}(\hat{e}_{i,k}^{(1)}) - \nabla \mathcal{L}(\hat{e}_{i,k}^{(2)})\right)\right| && (24)\\
&\le \sqrt{\sum_{i} \left(\boldsymbol{z}_i^\top \boldsymbol{P}_{:,k}\right)^2} \sqrt{\sum_{i} \left(\nabla \mathcal{L}(\hat{e}_{i,k}^{(1)}) - \nabla \mathcal{L}(\hat{e}_{i,k}^{(2)})\right)^2} && (25)\\
&\le \sum_{i} \|\boldsymbol{z}_i\| \|\boldsymbol{P}_{:,k}\| L \sqrt{\sum_{i} \left(\hat{e}_{i,k}^{(1)} - \hat{e}_{i,k}^{(2)}\right)^2} && (26)\\
&\le L n \sigma_{\max}(\boldsymbol{Z}) \sigma_{\max}(\boldsymbol{P}) \sqrt{\sum_{i} (\boldsymbol{z}_i^\top \boldsymbol{P}_{:,k})^2 (\hat{s}_k^{(1)} - \hat{s}_k^{(2)})^2} && (27)\\
&\le L n \sigma_{\max}(\boldsymbol{Z}) \sigma_{\max}(\boldsymbol{P}) \sqrt{\sum_{i} (\boldsymbol{z}_i^\top \boldsymbol{P}_{:,k})^2} |\hat{s}_k^{(1)} - \hat{s}_k^{(2)}| && (28)\\
&\le L n \sigma_{\max}^2(\boldsymbol{Z}) \sigma_{\max}^2(\boldsymbol{P}) \left|\hat{s}_k^{(1)} - \hat{s}_k^{(2)}\right|, && (29)
\end{aligned}$$

where $\boldsymbol{P}_{:,k}$ is the kth column of the $\boldsymbol{P}$ matrix (as a column vector), and $\sigma_{\max}(\cdot)$ is the maximum singular value of a matrix. Here, Line (25) is obtained by applying the Cauchy–Schwartz inequality. Line (27) is based on matrix norm inequalities.

Finally, we examine $z_{i,k}$, which is the kth dimension ($k = 1, \cdots, r$) of the ith token of our ultra-low-dimensional embeddings:

$$\left|\nabla\hat{\mathcal{L}}(z_{i,k}^{(1)}) - \nabla\hat{\mathcal{L}}(z_{i,k}^{(2)})\right| = \left|\boldsymbol{P}_{k,:}\operatorname{diag}(\boldsymbol{s})\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i^{(1)}} - \boldsymbol{P}_{k,:}\operatorname{diag}(\boldsymbol{s})\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i^{(2)}}\right| \tag{30}$$

$$= \left|\sum_j p_{k,j}s_j\left(\nabla\mathcal{L}(\hat{e}_{ij}^{(1)}) - \nabla\mathcal{L}(\hat{e}_{ij}^{(2)})\right)\right| \tag{31}$$

$$\leq \|\boldsymbol{P}_{k,:}\operatorname{diag}(\boldsymbol{s})\|\left\|\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i^{(1)}} - \frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i^{(2)}}\right\| \tag{32}$$

$$\leq \sigma_{\max}(\boldsymbol{P})\sigma_{\max}(\boldsymbol{s})L\|\hat{\boldsymbol{e}}_i^{(1)} - \hat{\boldsymbol{e}}_i^{(2)}\| \tag{33}$$

$$\leq \sigma_{\max}(\boldsymbol{P})\sigma_{\max}(\boldsymbol{s})L\left\|\operatorname{diag}(\boldsymbol{s})\boldsymbol{P}^\top\left(\boldsymbol{z}_i^{(1)} - \boldsymbol{z}_i^{(2)}\right)\right\| \tag{34}$$

$$\leq L\sigma_{\max}^2(\boldsymbol{P})\sigma_{\max}^2(\boldsymbol{s})\|\boldsymbol{z}_i^{(1)} - \boldsymbol{z}_i^{(2)}\| \tag{35}$$

$$= L\sigma_{\max}^2(\boldsymbol{P})\sigma_{\max}^2(\boldsymbol{s})|z_{i,k}^{(1)} - z_{i,k}^{(2)}|. \tag{36}$$

where Eqn. (36) holds because we examine one element $z_{i,k}$ at a time, so $z_{i,k'}^{(1)} = z_{i,k'}^{(2)}$ for $k' \neq k$.

With these element-wise properties, we can have the full-parameter Lipschitz condition by using Lemma 7. □

Lemma 9. *The loss function $\hat{\mathcal{L}}$ is μ'-PL (Polyak–Lojasiewic) w.r.t. $\hat{\boldsymbol{x}} \in \mathbb{R}^{nr+d}$ for some μ'.*

Proof.

$$\frac{1}{2}\|\nabla\hat{\mathcal{L}}(\hat{\boldsymbol{x}})\|^2 = \frac{1}{2}\left\|\frac{\partial\mathcal{L}}{\partial\boldsymbol{b}}\right\|^2 + \frac{1}{2}\left\|\frac{\partial\mathcal{L}}{\partial\boldsymbol{s}}\right\|^2 + \frac{1}{2}\sum_{i=1}^n\left\|\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}_i}\right\|^2 \tag{37}$$

$$= \frac{1}{2}\left\|\sum_{i=1}^n\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i}\right\|^2 + \frac{1}{2}\left\|\sum_{i=1}^n\operatorname{diag}(\boldsymbol{P}^\top\boldsymbol{z}_i)\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i}\right\|^2 + \frac{1}{2}\sum_{i=1}^n\left\|\boldsymbol{P}\operatorname{diag}(\boldsymbol{s})\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i}\right\|^2 \tag{38}$$

$$\geq \frac{1}{2}\sum_{i=1}^n\left\|\boldsymbol{P}\operatorname{diag}(\boldsymbol{s})\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i}\right\|^2 \tag{39}$$

$$\geq \frac{1}{2}\sigma_{\min}^2(\boldsymbol{P})\sigma_{\min}^2(\boldsymbol{s})\sum_{i=1}^n\left\|\frac{\partial\mathcal{L}}{\partial\hat{\boldsymbol{e}}_i}\right\|^2 \tag{40}$$

$$= \frac{1}{2}\sigma_{\min}^2(\boldsymbol{P})\sigma_{\min}^2(\boldsymbol{s})\|\nabla\mathcal{L}(\hat{\boldsymbol{x}})\|^2 \tag{41}$$

$$\geq \sigma_{\min}^2(\boldsymbol{P})\sigma_{\min}^2(\boldsymbol{s})\mu\left(\mathcal{L}(\hat{\boldsymbol{x}}) - \mathcal{L}(\boldsymbol{x}^*)\right) \tag{42}$$

$$\geq \sigma_{\min}^2(\boldsymbol{P})\sigma_{\min}^2(\boldsymbol{s})\mu\left(\mathcal{L}(\hat{\boldsymbol{x}}) - \mathcal{L}(\hat{\boldsymbol{x}}^*)\right), \tag{43}$$

where $\hat{\boldsymbol{x}}^*$ is the minimizer under our parameterization. This suggests that $\mathcal{L}$ is μ'-PL for some μ'. □

Theorem 3. *Assume the original loss function $\mathcal{L}$ is Polyak–Lojasiewic and element-wise Lipschitz on the original d-dimensional embeddings. Let $\boldsymbol{P} \in \mathbb{R}^{r\times d}$ be a given full-rank random Gaussian matrix (i.e., rank r), and our parametrization be $\hat{\boldsymbol{e}}_i = \operatorname{diag}(\boldsymbol{s})\boldsymbol{P}^\top\boldsymbol{z}_i + \boldsymbol{b}$. With a proper learning rate schedule $\eta_1, \eta_2, \ldots$, our parameters $\boldsymbol{x} = [\boldsymbol{b}, \boldsymbol{s}, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n]$ converge to the global optimum with gradient descent if $\boldsymbol{s}$ is always non zero.*

Proof. At each iteration t, gradient descent produces

$$\boldsymbol{x}_{t+1} \leftarrow \boldsymbol{x}_t - \eta_t\nabla\mathcal{L}(\boldsymbol{x}_t), \tag{44}$$

where $\mathcal{L}$ is the loss function under our parametrization. For each iteration, we choose $\eta_t = 1/\beta'(\boldsymbol{x}_t)$, where $\beta'(\boldsymbol{x}_t)$ is the Lipschitz coefficient in Lemma 8 depending on $\boldsymbol{x}_t$:

$$\mathcal{L}(\boldsymbol{x}_{t+1}) \leq \mathcal{L}(\boldsymbol{x}_t) + \left(\nabla\mathcal{L}(\boldsymbol{x}_t)\right)^\top\left(\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\right) + \frac{\beta'(\boldsymbol{x}_t)}{2}\|\boldsymbol{x}_{t+1} - \boldsymbol{x}_t\|^2 \tag{45}$$

$$
\begin{aligned}
&= \mathcal{L}(\boldsymbol{x}_t) - \frac{1}{2\beta'(\boldsymbol{x}_t)}\|\nabla\mathcal{L}(\boldsymbol{x}_t)\|^2 && (46)\\
&\le \mathcal{L}(\boldsymbol{x}_t) - \frac{\mu'(\boldsymbol{x}_t)}{\beta'(\boldsymbol{x}_t)}(\mathcal{L}(\boldsymbol{x}_t) - \mathcal{L}(\boldsymbol{x}^*)). && (47)
\end{aligned}
$$

where $\mu'(\boldsymbol{x}_t)$ is the PL coefficient in Lemma 9, which also depends on $\boldsymbol{x}_t$. By rearranging the terms, we obtain

$$
\mathcal{L}(\boldsymbol{x}_{t+1}) - \mathcal{L}(\boldsymbol{x}^*) \le \left(1 - \frac{\mu'(\boldsymbol{x}_t)}{\beta'(\boldsymbol{x}_t)}\right)(\mathcal{L}(\boldsymbol{x}_t) - \mathcal{L}(\boldsymbol{x}^*)), \tag{48}
$$

suggesting that the excessive loss $\mathcal{L}(\boldsymbol{x}) - \mathcal{L}(\boldsymbol{x}^*)$ converges to 0. $\square$

Note that our Lipschitz and PL conditions are non-uniform (i.e., depending on the parameters according to the lemmas above). Therefore, a proper learning schedule $\eta_t = 1/\beta(\boldsymbol{x}_t)$ is needed in the theoretical analysis.