# Stochastic Bridges as Effective Regularizers for Parameter-Efficient Tuning

**Weize Chen**[1], **Xu Han**[1]*, **Yankai Lin**[2], **Zhiyuan Liu**[1]*, **Maosong Sun**[1]* **and Jie Zhou**[3]
[1]NLP Group, DCST, IAI, BNRIST, Tsinghua University, Beijing
[2]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing
[3]Pattern Recognition Center, WeChat AI, Tencent Inc.
chenwz21@mails.tsinghua.edu.cn
hanxu2022@tsinghua.edu.cn

## Abstract

Parameter-efficient tuning methods (PETs) have achieved promising results in tuning large pre-trained language models (PLMs). By formalizing frozen PLMs and additional tunable parameters as systems and controls respectively, PETs can be theoretically grounded to optimal control and further viewed as optimizing the terminal cost and running cost in the optimal control literature. Despite the elegance of this theoretical grounding, in practice, existing PETs often ignore the running cost and only optimize the terminal cost, i.e., focus on optimizing the loss function of the output state, regardless of the running cost that depends on the intermediate states. Since it is non-trivial to directly model the intermediate states and design a running cost function, we propose to use latent stochastic bridges to regularize the intermediate states and use the regularization as the running cost of PETs. As the first work to propose regularized PETs that use stochastic bridges as the regularizers (running costs) for the intermediate states, we show the effectiveness and generality of this regularization across different tasks, PLMs and PETs. In view of the great potential and capacity, we believe more sophisticated regularizers can be designed for PETs and better performance can be achieved in the future. The code is released at https://github.com/thunlp/stochastic-bridge-pet/tree/main.

## 1 Introduction

Recent years have witnessed the dramatic growth of pre-trained language models (PLMs) in various fields (Devlin et al., 2019; Dosovitskiy et al., 2021). As the size of PLMs continues to increase, the number of parameters has now even reached hundreds of billions (Brown et al., 2020; Smith et al., 2022), making fine-tuning the whole PLM both computationally impractical and environmentally unfriendly. In view of this, a variety of Parameter-Efficient Tuning methods (PETs) are proposed (Houlsby et al., 2019; Hu et al., 2022; Zaken et al., 2022; Lester et al., 2021). By only tuning a small number of additional parameters, PETs can be comparable to full-parameter fine-tuning.

Despite the success of PETs, their underlying mechanism remains an open problem. Recently, several works have proposed to interpret PETs with optimal control theory. Yang and Liu (2022) first show that the optimization in Prefix Tuning (Li and Liang, 2021) (a typical method of PETs) can be considered as the search for optimal control variables in the context of optimal control, i.e., the trainable prefixes can be seen as the control variables that drive the PLM (the system) to the desired output. Ding et al. (2022) further show that the optimal control perspective can be applied to almost all PETs. The optimization of PETs' parameters can be seen as minimizing the two cost functions in the optimal control literature: (1) *terminal cost* $\mathcal{L}_T$, which measures the quality of the terminal state, and (2) *running cost* $\mathcal{L}_R$, which measures the feasibility of the controlled intermediate states and the control variables. Although $\mathcal{L}_T$ can well correspond to the loss function of the model output, $\mathcal{L}_R$ is only vaguely described as the regularizers on the parameters of PETs (control variables) in Yang and Liu (2022) and Ding et al. (2022), *ignoring the dependency of* $\mathcal{L}_R$ *on the intermediate states*.

In this work, we show that designing a running cost to regularize intermediate states not only makes the optimal control perspective of PETs more theoretically sound but also empirically leads to better PETs. We begin by assuming that in PLMs, the intermediate hidden states for generating different tokens in a sentence have different dynamics (or trajectories), and the dynamics can be approximated with stochastic processes in a latent space. Specifically, we first freeze the PLM and learn a mapping from the original hidden state space of the PLM to a latent space. In the latent

*Corresponding author.



*Findings of the Association for Computational Linguistics: ACL 2023*, pages 10400–10420
July 9-14, 2023 

space, the dynamics of the intermediate hidden states for generating different target tokens can be approximated with different target-specific *diffusion bridges*. The obtained mapping can then be plugged to the model to regularize the intermediate hidden states when training PETs. Besides, since a diffusion bridge is (1) a Markov process and (2) a solution to a stochastic differential equation (SDE), we correspondingly propose two methods to learn the mapping: (1) fitting the Markov transition probability density function (PDF) and (2) fitting the SDE directly. The two methods act as a trade-off between efficiency and effectiveness: the first method incurs only negligible computational cost and has satisfactory results, while the second one is slower but yields better regularizers.

We conduct experiments on different PLMs of different sizes, and the experimental results on GLUE (Wang et al., 2019) under both full-set and few-shot settings demonstrate the effectiveness of our proposal across four different PETs. Further analyses show that the learned regularizer helps pull apart the hidden states of different label words. We also observe that when we project the intermediate hidden states of PETs without our regularizer into our latent space, the better the PETs perform, the closer the latent states are to our latent bridges. This spontaneous approaching behavior may indicate that stochastic-bridge-like latent dynamics naturally exist in well-trained PETs.

In summary, our work has the following contributions: (1) Guided by the perspective of optimal control for PETs, we design latent stochastic bridge regularizers on the intermediate states during the training of PETs. (2) We propose two methods to construct the latent space according to the two representations of stochastic bridges, offering a trade-off between efficiency and effectiveness. (3) Our regularizers are shown to be effective and general across different PLMs, different PETs, and different tasks. (4) We show that well-trained PETs without any regularization spontaneously exhibit stochastic-bridge-like latent dynamics.

## 2 Background

### 2.1 Definition and Mathematical Notations

Consider using a $L$-layer PLM with the vocabulary $\mathbb{V}$ to handle a text-to-text task $\mathcal{D}$. For each sample $(\boldsymbol{x}, y) \in \mathcal{D}$, $y \in \mathbb{V}$ is the output token and $\boldsymbol{x} \in \mathbb{V}^N$ is the input token sequence [1], where $N$ is the length of $\boldsymbol{x}$. With $\boldsymbol{x}$ as the input, each layer of the PLM will output a sequence of hidden states, and we denote the hidden states of the $i$-th PLM layer as $\boldsymbol{h}^{(i)} = \{\boldsymbol{h}_j^{(i)}\}_{j=1}^N \in \mathbb{R}^{d \times N}$. We denote the position where the model outputs the target $y$ as $o$, i.e., the model should predict $y$ with the hidden states $\boldsymbol{h}_o^{(L)}$.

### 2.2 Optimal Control Perspective of PETs

Conventionally, adapting the PLM to $\mathcal{D}$ requires full-parameter fine-tuning, which is given as:

$$\min_{\Delta\theta} \mathbb{E}_{\boldsymbol{x},y\sim\mathcal{D}}\Big[\mathcal{L}\big(\boldsymbol{h}_o^{(L)}, y\big) + \mathcal{R}\big(\Delta\theta\big)\Big],$$
$$\boldsymbol{h}^{(i)} = \begin{cases} \boldsymbol{h}^{(i-1)} + \mathcal{G}_{\theta+\Delta\theta}^{(i)}\big(\boldsymbol{h}^{(i-1)}\big), & i = 1, \ldots, L, \\ \texttt{Embed}(\boldsymbol{x}), i = 0, \end{cases} \tag{1}$$

where $\theta$ is the parameters, $\Delta\theta$ is the full-parameter update, $\mathcal{L}$ is the loss function, $\mathcal{R}$ is the regularization function, $\mathcal{G}_{\theta+\Delta\theta}^{(i)}$ is the $i$-th layer forward propagation with updated parameters, Embed transforms the input tokens into embeddings.

As $|\theta|$ continues to increase, full-parameter fine-tuning becomes impractical, and various PETs are proposed to mitigate this problem. Let $\phi = \{\phi^{(i)}\}_{i=0}^L$ be PETs' parameters. Ding et al. (2022) give a unified view of PETs from the perspective of optimal control, and Eq. 1 can be re-written as

$$\min_{\phi} \mathbb{E}_{\boldsymbol{x},y\sim\mathcal{D}}\Big[\mathcal{L}_T\big(\boldsymbol{h}_o^{(L)}, y\big) + \sum_{i=0}^{L} \mathcal{L}_R\big(\phi^{(i)}\big)\Big],$$
$$\boldsymbol{h}^{(i)} = \begin{cases} \boldsymbol{h}^{(i-1)} + \tilde{\mathcal{G}}_{\theta}^{(i)}\big(\boldsymbol{h}^{(i-1)}, \phi^{(i)}\big), & i = 1, \ldots, L, \\ \big[\phi^{(0)}; \texttt{Embed}(\boldsymbol{x})\big], & i = 0, \end{cases} \tag{2}$$

where $\tilde{\mathcal{G}}_{\theta}^{(i)}$ represents the $i$-th layer forward propagation intervened by PETs, $[\cdot;\cdot]$ is the concatenation operation, $\mathcal{L}_T$ is the terminal cost and $\mathcal{L}_R$ is the running cost. Since $|\phi| \ll |\theta|$, PETs can greatly reduce the tuning cost (more details in Appx. A). Typically, $\mathcal{L}_T$ corresponds to the prediction loss, and $\mathcal{L}_R$ can be seen as the regularizer on PETs' parameters $\phi$. However, in the optimal control literature, $\mathcal{L}_R$ depends on not only the control variables $\phi$, but also the controlled intermediate states $\{\boldsymbol{h}_o^{(i)}\}_{i=1}^L$. In this paper, we show that additionally introducing dependence on $\{\boldsymbol{h}_o^{(i)}\}_{i=1}^L$ for $\mathcal{L}_R$ makes the optimal control perspective of PETs more theoretically sound, and empirically leads to better PETs.

[1] Here we assume $y \in \mathbb{V}$ since a sample where $\boldsymbol{y} \in \mathbb{V}^M$ can be decomposed to $M$ samples, The $i$-th sample is $([\boldsymbol{x}; \boldsymbol{y}_{<i}], y_i)$ for auto-regressive language modeling or $([\boldsymbol{x}; \boldsymbol{y}_{-i}], y_i)$ for auto-encoding language modeling.

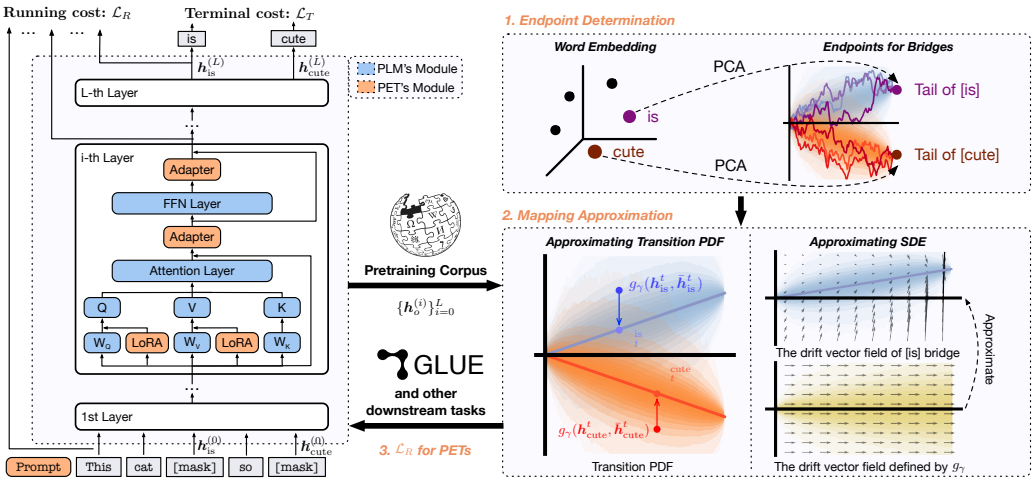



Figure 1: An overview of our proposed latent stochastic bridge regularizer.

### 2.3 Diffusion Bridges

A diffusion process $X = (X_t)_{t\in[0,T]}$ is a continuous-time Markov process. For any $t_a < t_b$, the diffusion process is equipped with a transition Probability Density Function (PDF) $p(t_b, b|t_a, a)$, which gives the probability density of reaching $b$ at time $t_b$ given the history of reaching $a$ at time $t_a$. A diffusion process is also the solution to an Itô SDE $d\tilde{X}_t = \mu(t, \tilde{X}_t)dt + \sigma(t, \tilde{X}_t)dB_t$, where $B_t$ is a standard Brownian motion, $\mu(\cdot,\cdot)$ is called drift function and $\sigma(\cdot,\cdot)$ is called diffusion function.

A diffusion bridge $X^{T;\alpha,\beta}$ is a diffusion process conditioning on the path observations of the two endpoints $(0,\alpha)$ and $(T,\beta)$, i.e., $X_0^{T;\alpha,\beta} = \alpha$ and $X_T^{T;\alpha,\beta} = \beta$. For simplicity, we assume $\alpha$=0 in this work, and omit the superscript $\alpha$. We consider two typical diffusion bridges, the Brownian bridge and the Ornstein-Uhlenbeck bridge (OU bridge). We present here the properties of the Brownian bridge and leave the properties of OU bridge to Appx. B.

**Proposition 2.1** (Properties of Brownian Bridge)**.** A Brownian bridge $X^{T;\beta}$ with $X_0^{T;\beta} = 0$ and $X_T^{T;\beta} = \beta$ is the solution to the following SDE:

$$d\tilde{X}_t = (\beta - \tilde{X}_t)/(T-t)\, dt + dB_t, \quad \tilde{X}_0 = 0. \quad (3)$$

The transition PDF from $X_0^{T;\beta} = 0$ to $X_{t_b}^{T;\beta} = b$ is given as

$$p^{T;\beta}(t_b, b|0,0) = \frac{1}{\sqrt{2\pi t_b(T-t_b)}} \exp\left[-\frac{(b-(t_b/T)\beta)^2}{2t_b(T-t_b)}\right]. \quad (4)$$

Diffusion bridges and SDEs are battle-tested tools to model the stochastic dynamics of complex systems in engineering (Sobczyk, 2013), finance (Wang and Sloan, 2011), etc. Considering the dynamics of PLMs' hidden states are necessarily complex, diffusion bridges and SDEs serve as ideal tools for us to model the dynamics.

## 3 Latent Stochastic Bridges Regularizer

### 3.1 The Overall Framework

**Building latent dynamics in the latent space.** Since directly regularizing the intermediate states and constructing the running cost are non-trivial, we introduce a projection from the intermediate state space to a latent space, and leverage diffusion bridges as regularizers to construct the running cost. Specifically, we define a $r$-dimensional latent space $\mathbb{U} \subseteq \mathbb{R}^r (r < d)$ and a learnable mapping $g_\gamma : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{U}$, where $\gamma$ denotes the parameters. $g_\gamma$ projects the hidden state $\boldsymbol{h}_o^{(i)}$ and its context state $\bar{\boldsymbol{h}}^{(i)}$ into the latent space $\mathbb{U}$ at each layer of the PLM. Since $\boldsymbol{h}_o^{(i)}$ is contextualized while latent bridges are not, introducing the dependency on $\bar{\boldsymbol{h}}^{(i)}$ can inform $g_\gamma$ about the context at the $i$-th layer and allow $g_\gamma$ to decontextualize the hidden states. We simply take the averaged states at the $i$-th layer $\bar{\boldsymbol{h}}^{(i)} = \frac{1}{N}\sum_{j=1}^N \boldsymbol{h}_j^{(i)}$ as the context. We define the latent states with discrete time as

$$\begin{aligned} \boldsymbol{u}_D(g_\gamma, \{\boldsymbol{h}_o^{(i)}\}_{i=0}^L) &= \{t_{i+1}, g_\gamma(\boldsymbol{h}_o^{(i)}, \bar{\boldsymbol{h}}^{(i)})\}_{i=0}^L, \\ t_{i+1} &= (i+1)/(L+2), \end{aligned} \quad (5)$$

where $t_{i+1}$ is the normalized layer index. We include the 0-th layer (input layer) because some PETs (e.g., prompt tuning) act on the 0-th layer. We use $t_0 = 0, t_{L+2} = 1$ represent the two endpoints. By using natural cubic spline knotted at $\{\boldsymbol{h}_o^{(i)}\}_{i=0}^L$ to interpolate over $\mathcal{L} = [-1, L+1]$, we further give a continuous representation of the

states in the latent space $\mathbb{U}$ as

$$\begin{aligned} \boldsymbol{u}_C(g_\gamma, \{\boldsymbol{h}_o^{(x)}\}_{x\in\mathcal{L}}) &= \{t_{x+1}, g_\gamma(\boldsymbol{h}_o^{(x)}, \bar{\boldsymbol{h}}^{(x)})\}_{x\in\mathcal{L}}, \\ t_{x+1} &= (x+1)/(L+2) \in [0,1]. \end{aligned} \tag{6}$$

**Learning the mapping from hidden state space to latent space.** Since adapting PLMs to downstream tasks can be seen as transferring the knowledge obtained from pre-training tasks to downstream tasks, we argue that the latent dynamics of intermediate hidden states for generating the same token $y$ should be similar in both the pre-training and downstream tasks. Therefore, we train the mapping $g_\gamma$ on the corpus that is used to pre-train the backbone PLM[2], and then apply the learned mapping to downstream tasks to encourage the latent dynamics to be similar to that in pre-training.

Specifically, we assume that the states to generate the token $y$ in the latent space $\mathbb{U}$ form a trajectory that is a path sampled from $X^{1;\beta_y}$ with high probability, where $X^{1;\beta_y}$ is the diffusion bridge describing the latent dynamics to generate $y$, and $\beta_y$ is the tail endpoint of the diffusion bridge. More details of $X^{1;\beta_y}$ will be discussed in Sec. 3.2.

On the corpus where the PLM is pre-trained, we fix the PLM and use its hidden states $\{\boldsymbol{h}_o^{(i)}\}_{i=1}^L$ to learn $g_\gamma$ by maximizing the goodness of approximation for latent states $\boldsymbol{u}$ under the bridge $X^{1;\beta_y}$:

$$\gamma \leftarrow \arg\max_{\gamma'} \left[\text{goodness}\big(\boldsymbol{u}(g_{\gamma'}, \{\boldsymbol{h}_o^{(\cdot)}\}), X^{1;\beta_y}\big)\right], \tag{7}$$

where $\boldsymbol{u}$ can be $\boldsymbol{u}_D$ (Eq. 5) or $\boldsymbol{u}_C$ (Eq. 6) depending on the fitting method, goodness$(\cdot,\cdot)$ is also a function depends on the choice of the fitting method, measuring how likely $\boldsymbol{u}$ is a sample trajectory of $X^{1;\beta_y}$. In Sec. 3.3, we will define this function alongside the fitting methods.

**Regularizing PETs with latent dynamics.** After learning $g_\gamma$ with Eq. 7, we freeze $\gamma$ and use the goodness function as the running cost in Eq. 2 for PETs on downstream tasks. The objective becomes

$$\mathcal{L} = \mathcal{L}_T(\boldsymbol{h}_o^{(L)}, y) + \alpha \cdot \text{goodness}\big(\boldsymbol{u}(g_\gamma, \{\boldsymbol{h}_o^{(\cdot)}\}), X^{1;\beta_y}\big), \tag{8}$$

where the second term is the running cost and $\alpha$ is a hyper-parameter controlling the regularization intensity. By optimizing Eq. 8, PETs learn to predict $y$ correctly and keep the latent states at the position $o$ conform to the diffusion bridge $X^{1;\beta_y}$. Note that introducing $g_\gamma$ as the regularizer does not increase the number of trainable parameters for PETs during the training stage since $\gamma$ is fixed, and since we only use the pre-training corpus, no extra information of downstream tasks is leaked. Moreover, the regularizer only helps in training better PETs and does not intervene the inference.

[2]A small portion (0.1%) of the pre-training corpus is sufficient, see Appx. F

### 3.2 Determining Endpoints for Bridges

An intuitive approach to determine the endpoints for the diffusion bridges for each target token is to optimize the endpoints together with the mapping $g_\gamma$. However, optimizing endpoints and $g_\gamma$ jointly may admit a trivial solution: endpoints are both $\mathbf{0} \in \mathbb{R}^r$ and $g_\gamma$ always outputs $\mathbf{0}$. Since $\mathbf{0}$ is always a point in the sample path of such a degenerated diffusion bridge, the value of goodness function can be meaninglessly high. Although sophisticated constraints can be imposed here, as the first work that uses diffusion bridges as regularizers, we simply pre-determine the endpoints and keep them fixed. We leave introducing constraints as future work.

Specifically, we apply principal component analysis (PCA) to the output token embedding matrix $\boldsymbol{V} \in \mathbb{R}^{|\mathbb{V}|\times d}$ of the PLM, obtaining a $r$-dimensional embedding matrix, and re-normalize each row to have a norm $\eta$. Let the resulting embedding matrix be $\boldsymbol{\beta} \in \mathbb{R}^{|\mathbb{V}|\times r}$. We then use $\mathbf{0} \in \mathbb{R}^r$ as the heads for all the bridges, and $\boldsymbol{\beta}$ as the tails of the diffusion bridges, i.e., the $r$-dimensional embedding of $y$ in $\boldsymbol{\beta}$ is used as $\beta_y$ in $X^{1;\beta_y}$. The intuition for using $\boldsymbol{\beta}$ as the tails is that the trajectories of the intermediate states for similar target tokens should be close. In $\boldsymbol{V}$, similar tokens are close, and $\boldsymbol{\beta}$ obtained by PCA can well preserve the token similarity after reducing dimensions.

### 3.3 Fitting the Mapping $g_\gamma$

We use the Brownian bridge to illustrate the fitting of $g_\gamma$. It can be analogous to OU bridge easily.

**Method 1: Approximating the Transition PDF.** Generalizing Eq. 4 to high dimension, we can derive the transition PDF from $(0, \mathbf{0})$ to $(t_{i+1}, g_\gamma(\boldsymbol{h}_o^{(i)}, \bar{\boldsymbol{h}}^{(i)}))$ for $X^{1;\beta_y}$:

$$\begin{aligned} &p^{1;\beta_y}(t_{i+1}, g_\gamma(\boldsymbol{h}_o^{(i)}, \bar{\boldsymbol{h}}^{(i)}) \mid 0, \mathbf{0}) \\ &\propto \exp\Big(\frac{\|g_\gamma(\boldsymbol{h}_o^{(i)}, \bar{\boldsymbol{h}}^{(i)}) - t_{i+1}\beta_y\|^2}{2t_{i+1}(1-t_{i+1})}\Big), \quad (i = 0, \ldots, L), \end{aligned}$$

where $t_i$ has the same definition as that in $\boldsymbol{u}_D$ (Eq. 5). To make $g_\gamma$ approximate the transition PDF, we maximize the sum of log-probability of $\boldsymbol{u}_D$ under the Brownian bridge $X^{1;\beta_y}$:

$$\text{goodness} = \sum_{i=0}^{L} \log\left[p^{1;\beta_y}(t_{i+1}, g_\gamma(\boldsymbol{h}_o^{(i)}, \bar{\boldsymbol{h}}^{(i)}) \mid 0, \mathbf{0})\right] + C, \tag{9}$$

where $C$ is a constant. Here, $g_\gamma$ can be seen as a mapping from the hidden state space to the latent space by predicting the expectation of the Brownian bridge $X^{1;\beta_y}$ at $\{t_{i+1}\}_{i=0}^{L}$.

**Method 2: Approximating the SDE.** Since the Brownian bridge is the solution to the SDE in Eq. 3, we let $g_\gamma$ approximate the SDE. Solving the SDE requires continuous latent states, while we only have $L+1$ discrete observations, we thus use the continuous representation $\boldsymbol{u}_C$ introduced in Eq. 6. Generalizing Eq. 3 to high dimension, the SDE approximated by $g_\gamma$ can be defined as:

$$dZ_t = g_\gamma(\boldsymbol{h}_o^{(x)}, \bar{\boldsymbol{h}}^{(x)}, t)dt + dB_t,\ x = (L+2)t - 1, \tag{10}$$

where $x$ is the same as that in Eq. 6, $B : [0,1] \to \mathbb{R}^r$ is a standard $r$-dimensional Brownian motion. Here, we additionally introduce the dependence on $t$ for $g_\gamma$, since time information is shown to be important in previous neural differential equation works (Zhang et al., 2020; Dupont et al., 2019). Following Li et al. (2020), when two SDEs share the same diffusion function, the KL divergence between the probability measures induced by the two SDEs is finite. Since the diffusion function $\sigma \equiv \boldsymbol{I}$ for Eq. 10 and the multi-dimensional generalization of Eq. 3, the KL divergence between the probability measures $\mu_Y$ of Eq. 10 and $\mu_X$ of generalized Eq. 3 can be estimated by:

$$\begin{aligned} D_{\mathrm{KL}}(\mu_X||\mu_Y) &= \mathbb{E}_Z\big[\int_0^T \frac{1}{2}\|u(t,\gamma)\|_2^2\big], \\ u(t,\gamma) &= \sigma^{-1}\big(g_\gamma(\boldsymbol{h}_o^{(x)}, \bar{\boldsymbol{h}}^{(x)}, t) - \mu(t, Z_t)\big) \\ &= g_\gamma(\boldsymbol{h}_o^{(x)}, \bar{\boldsymbol{h}}^{(x)}, t) - (\beta_y - Z_t)/(1-t), \end{aligned}$$

where $\mu(\cdot,\cdot)$ is the drift function of the predetermined Brownian bridge $X^{1;\beta_y}$. We use the KL divergence as the goodness function to optimize the mapping $g_\gamma$. Here, $g_\gamma$ can be seen as a mapping from the hidden state space to the latent state space by approximating the drift vector field of the underlying Brownian bridge $X^{1;\beta_y}$.

## 4 Experiments

To verify the effectiveness and generality of the regularizers built on stochastic bridges, we conduct experiments on (1) different PLMs: $\text{BERT}_{\text{large}}$ (340M) (Devlin et al., 2019) and $\text{Deberta}_{\text{xlarge}}$ (750M) (He et al., 2021); (2) different PETs: Prompt tuning, LoRA, BitFit and Adapter; (3) different diffusion bridges: Brownian bridge and OU bridge. We show that the regularizers effectively improve the performance on GLUE (Wang et al., 2019) under both full-set and few-shot settings.

### 4.1 Experimental Setups

**Datasets.** Since both $\text{BERT}_{\text{large}}$ and $\text{Deberta}_{\text{xlarge}}$ use Wikipedia and BookCorpus (Zhu et al., 2015) for pre-training, we thus use these two corpora to train $g_\gamma$. We report F1 for MRPC and QQP; Matthews correlation for CoLA; and accuracy for other tasks. We report the average performance and the standard deviation on the development set over 3 different runs. We append `[MASK]` to each sequence, and require the PLM to output the label word at `[MASK]` (e.g., *negative* or *positive* for SST-2). We exclude STS-B for it is a regression task.

**Models and PETs.** We use the checkpoint released by Shoeybi et al. (2019) for $\text{BERT}_{\text{large}}$, and the official v1 checkpoint for $\text{Deberta}_{\text{xlarge}}$. We use a simple three-layer MLP to build $g_\gamma$. For Prompt tuning, we use a soft prompt of length 20, and append it to the end of each sequence. For LoRA, we apply it to the query and value of attention modules. For Adapter, we apply it to the output of attention and feed-forward modules. For BitFit, we tune all the bias terms in linear layers and layer normalization modules. Hereafter, we use **PDF regularizer** to refer to using $g_\gamma$ fitted by approximating the transition PDF, and **SDE regularizer** to refer to using $g_\gamma$ fitted by approximating the SDE, **vanilla** $x$ to refer to the PET $x$ without using regularizers.

**Few-shot Experiments.** We randomly sample $2 \times k$ examples from the original training set $\mathcal{D}_{\text{train}}$ for each class. The sampling is performed 5 times with different seeds to form 5 training sets and development sets $\{\tilde{\mathcal{D}}_{\text{train}}^{(i)}, \tilde{\mathcal{D}}_{\text{dev}}^{(i)}\}_{i=1}^{5}$ with each being $k$-shot. Each time we train PETs on $\tilde{\mathcal{D}}_{\text{train}}^{(i)}$, we select the best model on $\tilde{\mathcal{D}}_{\text{dev}}^{(i)}$, and report its performance on the original development set $\mathcal{D}_{\text{dev}}$.

**Hyper-parameters.** Hyper-parameters are listed in Appx. E. We focus on the difference in performance between vanilla PETs and regularized PETs. Therefore, we set the hyper-parameters to common values from previous works and do not perform much hyper-parameter search. But we ensure the hyper-parameters for vanilla PETs and regularized PETs are the same for a fair comparison.

### 4.2 Full-set Results

The experimental results for $\text{BERT}_{\text{large}}$ and $\text{Deberta}_{\text{xlarge}}$ are reported in Tab. 1 and Appx. C respectively. Due to space limitation, see Tab. 6 for the complete results including OU bridge regularizers. The first line of each block in the table is the performance of vanilla PETs, and the rest of the

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | CoLA | RTE | Average | Δ |
|---|---|---|---|---|---|---|---|---|---|
| PROMPT | $84.4_{0.1}$ | $85.3_{0.3}$ | $91.5_{0.1}$ | $95.5_{0.1}$ | $73.9_{2.4}$ | $55.5_{3.4}$ | $60.8_{1.5}$ | $78.1_{0.6}$ | - |
| +BROWN_PDF | $84.7_{0.2}$ | $\mathbf{85.5}_{0.0}$ | $\mathbf{91.8}_{0.6}$ | $95.7_{0.1}$ | $75.4_{0.5}$ | $56.4_{3.3}$ | $61.5_{2.2}$ | $78.7_{0.4}$ | 0.6 |
| +BROWN_SDE | $\mathbf{84.9}_{0.2}$ | $85.4_{0.1}$ | $\mathbf{91.8}_{0.4}$ | $\mathbf{95.8}_{0.3}$ | $\mathbf{78.8}_{1.2}$ | $\mathbf{61.4}_{2.9}$ | $\mathbf{64.7}_{1.1}$ | $\mathbf{80.4}_{0.2}$ | **2.3** |
| LORA | $88.8_{0.1}$ | $89.2_{0.2}$ | $93.5_{0.2}$ | $95.5_{0.1}$ | $84.6_{0.4}$ | $62.8_{1.6}$ | $78.9_{1.6}$ | $84.8_{0.3}$ | - |
| +BROWN_PDF | $\mathbf{88.9}_{0.1}$ | $\mathbf{89.6}_{0.1}$ | $\mathbf{93.9}_{0.1}$ | $95.6_{0.2}$ | $85.1_{0.7}$ | $63.7_{0.5}$ | $80.0_{0.5}$ | $85.2_{0.1}$ | 0.4 |
| +BROWN_SDE | $\mathbf{88.9}_{0.1}$ | $89.5_{0.1}$ | $93.7_{0.1}$ | $\mathbf{95.7}_{0.1}$ | $\mathbf{86.5}_{1.2}$ | $\mathbf{63.9}_{0.4}$ | $\mathbf{80.9}_{0.8}$ | $\mathbf{85.6}_{0.1}$ | **0.8** |
| BITFIT | $\mathbf{87.9}_{0.2}$ | $87.6_{0.1}$ | $92.7_{0.2}$ | $95.6_{0.1}$ | $79.4_{2.3}$ | $60.2_{0.8}$ | $77.0_{1.5}$ | $82.9_{0.3}$ | - |
| +BROWN_PDF | $\mathbf{87.9}_{0.1}$ | $\mathbf{87.8}_{0.0}$ | $\mathbf{93.0}_{0.2}$ | $\mathbf{95.7}_{0.1}$ | $83.1_{0.8}$ | $60.3_{0.6}$ | $\mathbf{78.3}_{0.9}$ | $83.7_{0.2}$ | 0.8 |
| +BROWN_SDE | $\mathbf{87.9}_{0.2}$ | $87.7_{0.0}$ | $92.8_{0.1}$ | $\mathbf{95.7}_{0.1}$ | $\mathbf{83.3}_{0.8}$ | $\mathbf{61.1}_{1.2}$ | $77.7_{1.5}$ | $\mathbf{83.8}_{0.3}$ | **0.9** |
| ADAPTER | $88.8_{0.1}$ | $89.6_{0.3}$ | $93.7_{0.1}$ | $95.6_{0.1}$ | $83.6_{0.1}$ | $60.4_{1.2}$ | $79.5_{1.2}$ | $84.5_{0.3}$ | - |
| +BROWN_PDF | $\mathbf{89.0}_{0.1}$ | $89.7_{0.2}$ | $93.8_{0.2}$ | $\mathbf{95.8}_{0.1}$ | $\mathbf{86.5}_{1.1}$ | $\mathbf{62.6}_{0.7}$ | $\mathbf{83.2}_{0.2}$ | $\mathbf{85.8}_{0.2}$ | **1.3** |
| +BROWN_SDE | $88.9_{0.1}$ | $\mathbf{89.8}_{0.1}$ | $\mathbf{93.9}_{0.2}$ | $\mathbf{95.8}_{0.1}$ | $85.9_{0.4}$ | $62.3_{1.8}$ | $82.2_{0.2}$ | $85.5_{0.2}$ | 1.0 |

Table 1: The results on GLUE for $\text{BERT}_{\text{large}}$. The values are the average value of the best performances over three different runs, and the subscripts are the standard deviations. The $\Delta$ column shows the difference of the average performance between the vanilla PETs regularized PETs.

lines are the performances of the regularized PETs.

In general, both Brownian and OU bridges, and both PDF and SDE regularizers are able to improve the performance of PETs, showing the effectiveness of our proposed regularizers. Particularly, for Prompt tuning, the SDE regularizer with both diffusion bridges yield an average performance improvement of more than 2%. We assume that it is because Prompt tuning has far less trainable parameters than other PETs, and it only acts at the input layer, which is far from the supervision signals of the terminal cost $\mathcal{L}_T$. Therefore, when provided with the regularization on the hidden states, the prompts receive more guidance and eventually reaching a better local optimal.

Overall, the two diffusion bridges in our experiments do not show much difference. As for the two fitting methods, SDE regularizer is generally more effective, especially for Prompt tuning where the number of trainable parameters is restricted. However, we also observe that SDE regularizer is about 3 times slower than PDF regularizer, which brings the trade-off between performance and efficiency. One can expect a better performance by leveraging more sophisticated underlying stochastic bridges, exploring more reasonable endpoints for bridges and designing better mapping $g_\gamma$. As the first work using latent stochastic bridges as regularizers, we mainly consider the most simple cases and aim to show the potential of the approach.

### 4.3 Few-shot Results

In Tab. 1, the improvements are more substantial on small datasets MRPC, CoLA and RTE. This is probably because in large datasets, the abundant data has provided enough information to train high quality PETs; while in small datasets, the data is insufficient and the regularizer can offer additional supervision. To validate this, we conduct the experiments under the few-shot setting on GLUE.

The 16-shot results are shown in Tab. 2, and the results for the OU bridge, results for 4-, 8- and 32-shot and results for $\text{Deberta}_{\text{xlarge}}$ are placed in Appx. C. For all PETs, the SDE regularizer yields an improvement of more than 3%. Particularly, the SDE regularizer on LoRA brings an improvement of 5.2%. Also, there is now a substantial boost on what was originally a rich-resource dataset, such as MNLI, QQP and QNLI. The PDF regularizer also gives modest improvements. Though slightly inferior to the SDE regularizer, it is still satisfying, considering that the PDF regularizer brings such a performance improvement with little computational cost introduced. We additionally observe that the improvement is more significant on $\text{Deberta}_{\text{xlarge}}$ in Tab. 8, demonstrating the potential of our regularizers on larger models.

## 5 Analyses

To better understand the role played by our regularizers, we analyze the hidden states of the PETs with and without regularizers. We choose Prompt tuning as a representative. By varying the hyper-parameter $\alpha$ in Eq. 8, we show that as the regularization intensity gets stronger, the clusters of hidden states corresponding to different labels become more distinguishable. Also, we show that the hidden states of vanilla PETs spontaneously approach the latent bridges in the latent space without knowing the bridges, indicating that there may exist intrinsically diffusion-bridge-like latent dynamics for PETs.

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | RTE | Average | Δ |
|---|---|---|---|---|---|---|---|---|
| PROMPT | $38.1_{1.5}$ | $53.0_{3.1}$ | $51.6_{1.4}$ | $70.1_{4.9}$ | $50.1_{3.0}$ | $48.0_{1.3}$ | $51.8_{0.9}$ | - |
| +BROWN_PDF | $38.7_{2.3}$ | $54.9_{2.8}$ | $52.1_{1.1}$ | $75.0_{11.0}$ | $\mathbf{52.8}_{2.2}$ | $50.8_{3.3}$ | $54.0_{1.8}$ | 2.2 |
| +BROWN_SDE | $\mathbf{40.6}_{0.8}$ | $\mathbf{55.4}_{2.1}$ | $\mathbf{52.9}_{1.6}$ | $\mathbf{80.0}_{10.9}$ | $51.9_{3.6}$ | $\mathbf{51.7}_{3.1}$ | $\mathbf{55.4}_{1.5}$ | **3.6** |
| LORA | $48.7_{4.5}$ | $59.9_{5.5}$ | $53.2_{1.2}$ | $90.2_{1.1}$ | $53.6_{3.4}$ | $64.2_{0.9}$ | $61.6_{0.6}$ | - |
| +BROWN_PDF | $52.0_{1.3}$ | $62.7_{2.2}$ | $55.1_{3.8}$ | $\mathbf{91.3}_{0.1}$ | $57.4_{5.0}$ | $65.3_{1.5}$ | $64.0_{0.7}$ | 2.4 |
| +BROWN_SDE | $\mathbf{54.1}_{0.9}$ | $\mathbf{65.5}_{1.4}$ | $\mathbf{64.3}_{5.1}$ | $91.2_{0.3}$ | $\mathbf{60.2}_{3.1}$ | $\mathbf{65.8}_{1.4}$ | $\mathbf{66.8}_{0.6}$ | **5.2** |
| BITFIT | $48.4_{1.6}$ | $56.0_{6.1}$ | $51.7_{2.5}$ | $90.8_{0.8}$ | $52.0_{2.3}$ | $61.7_{1.4}$ | $60.1_{1.1}$ | - |
| +BROWN_PDF | $48.5_{1.9}$ | $56.0_{6.0}$ | $53.5_{2.0}$ | $\mathbf{91.0}_{0.4}$ | $53.9_{2.3}$ | $63.2_{1.6}$ | $61.0_{0.8}$ | 0.9 |
| +BROWN_SDE | $\mathbf{52.3}_{0.5}$ | $\mathbf{61.2}_{2.9}$ | $\mathbf{58.8}_{4.7}$ | $90.8_{0.4}$ | $\mathbf{54.8}_{3.0}$ | $\mathbf{63.9}_{2.5}$ | $\mathbf{63.6}_{0.8}$ | **3.5** |
| ADAPTER | $47.4_{3.7}$ | $57.0_{7.2}$ | $55.8_{2.9}$ | $91.0_{0.4}$ | $55.8_{2.5}$ | $62.7_{2.0}$ | $61.6_{1.2}$ | - |
| +BROWN_PDF | $49.0_{4.8}$ | $58.5_{7.4}$ | $56.9_{3.1}$ | $91.4_{0.2}$ | $57.2_{4.9}$ | $63.2_{3.0}$ | $62.7_{1.5}$ | 1.1 |
| +BROWN_SDE | $\mathbf{52.3}_{2.2}$ | $\mathbf{62.4}_{2.9}$ | $\mathbf{64.8}_{4.5}$ | $\mathbf{91.9}_{0.4}$ | $\mathbf{57.3}_{4.1}$ | $\mathbf{63.8}_{1.8}$ | $\mathbf{65.4}_{1.3}$ | **3.8** |

Table 2: The results on GLUE for $\text{BERT}_{\text{large}}$ under the 16-shot setting. We exclude CoLA because all PETs fail to give reasonable results under the few-shot setting.

### 5.1 Distances between Labels are Widen

We use the different prompts obtained with or without regularizers on the full-set GLUE, and record the intermediate hidden states $\{\boldsymbol{h}^{(i)}_{[\text{MASK}]}\}^{L}_{i=1}$. We vary the regularization intensity by adjusting the coefficient $\alpha$ in Eq. 8 to inspect the impact of the regularization intensity on the hidden states. Note that when $\alpha = 0$, it degenerates to the vanilla PET.

We randomly sample 100 samples for each label in MNLI, use UMAP (McInnes et al., 2018) to reduce the dimension of the last layer's hidden states of Prompt tuning and plot them in Fig. 2. It shows clearly that for both regularizers, as the regularization intensity gets stronger, the hidden states of the last layer become more distinguishable among labels. By looking at the axes of these plots, we find that the distances between the clusters generally increase when the regularization intensity is increased. We also notice that the SDE regularizer better helps separate the hidden states of the last layer by substantially enlarging the distance between the centroids of different labels, which could be one of the reasons why the SDE regularizer has better effectiveness in almost all experiments.

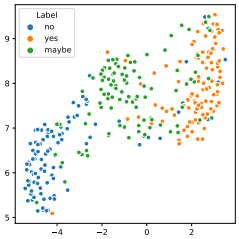
We also calculate the Pearson's correlation between the $\alpha$ and the average distance between the centroids of different clusters. The results are shown in Tab. h. On all the datasets, the $\alpha$ has a positive correlation to the average centroid distance, and on most of the datasets, the correlations are significant ($p$-value <.05). This indicates that as the regularization intensity gets stronger, the centroids of different label clusters become more distant, which is a desired effect because the regularizer encourages the hidden states for different labels to conform to different latent bridges.

### 5.2 Hidden States Spontaneously Approach the Latent Bridges

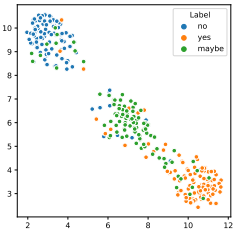
An interesting phenomenon we observe is that the vanilla PETs' intermediate hidden states spontaneously approach our latent bridges when they are projected by our mapping $g_\gamma$. That is, applying our mapping $g_\gamma$ to the hidden states of vanilla PETs, we find that when the performance of vanilla PETs becomes better, the average distance from $g_\gamma(\{\boldsymbol{h}^{(\cdot)}_o, \bar{\boldsymbol{h}}^{(\cdot)}\})$ to our latent bridge gets closer. Here, similar to Wang et al. (2022), we define the distance from $g_\gamma(\boldsymbol{h}^{(\cdot)}_o, \bar{\boldsymbol{h}}^{(\cdot)})$ to its corresponding latent bridge $X_y$ using Eq. 9 without the constant. Note that the vanilla PETs have no access to $g_\gamma$ and the latent bridges during the training process, and $g_\gamma$ also has no access to the PETs during its fitting.

We show the above phenomenon by conducting analyses in few-shot scenarios with PDF regularizer, and reporting in Tab. 3 the correlation between (1) the number of shots and the average distance from latent hidden states to latent bridges (2) the performance and the average distance from latent hidden states to latent bridges. We report Kendall's rank correlation for (1), and Pearson's correlation for (2). See Appx. D for the detailed setup.

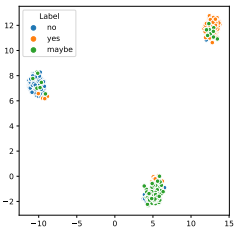
From Tab. 3, the number of shots has a negative correlation to the distance, and the correlation is significant on 4 out of 6 datasets. This indicates that as the amount of available data increases for vanilla PETs, its intermediate hidden states in latent space spontaneously approach latent bridges even without knowing the mapping $g_\gamma$ and the bridges. Additionally, the results in Tab. 3 show the negative correlation between the performance of vanilla PETs and the distance to the latent bridges, and it is significant on 3 out of 6 datasets.



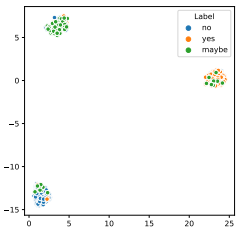

(a) w/o regularizer



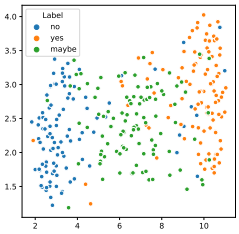

(b) SDE $\alpha = 0.001$



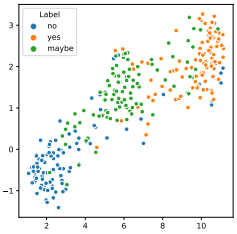

(c) SDE $\alpha = 0.01$



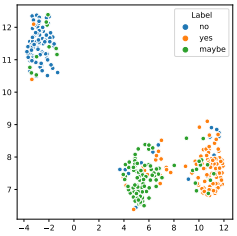

(d) SDE $\alpha = 0.1$




(e) PDF $\alpha = 0.01$




(f) PDF $\alpha = 0.1$




(g) PDF $\alpha = 1.0$

| Dataset | PDF | SDE |
|---|---|---|
| MNLI | 0.928*** | 0.894*** |
| QQP | 0.715* | 0.897*** |
| QNLI | 0.971*** | 0.837** |
| SST-2 | 0.966*** | 0.867*** |
| MRPC | 0.226 | 0.433 |
| CoLA | 0.633* | 0.807** |
| RTE | 0.440 | 0.589* |

(h) Pearson's correlation

Figure 2: The visualization of the last layer's hidden states on MNLI using the prompt that is trained (a) without regularizer (b-d) with the SDE regularizer (e-g) with the PDF regularizer. And the table in (h) is the Pearson's correlation between the regularization strength $\alpha$ and the average distance between the centroids of different label clusters. ***: $p < .001$, **: $p < .01$, *: $p < .05$.

| | Dist-Shot | | Dist-Perf | |
|---|---|---|---|---|
| Dataset | Coef. | $p$-value | Coef. | $p$-value |
| MNLI | -0.39 | .026* | -0.09 | .715 |
| QQP | -0.36 | .037* | -0.42 | .062 |
| QNLI | -0.32 | .069 | -0.77 | <.001*** |
| SST-2 | -0.50 | .005** | -0.69 | <.001*** |
| MRPC | -0.21 | .225 | -0.13 | .591 |
| RTE | -0.34 | .049* | -0.87 | <.001*** |

Table 3: Kendall's rank correlation between the number of shots and the distance to the latent bridges (Dist-Shot), and Pearson's correlation between the performance and the distance to the latent bridges (Dist-Perf).

Altogether, the two findings on correlation show that as the PETs' performance improve, their intermediate hidden states projected by $g_\gamma$ spontaneously approach our bridges in the latent space. This implies that there exists intrinsically diffusion-bridge-like latent dynamics for PETs, and also justifies our use of diffusion bridges as regularizers.

## 6 Related Works

Recent years have witnessed the success of PLMs (Raffel et al., 2020; Brown et al., 2020). However, as the sizes of PLMs continue to grow, it becomes increasingly impractical to perform fine-tuning on downstream tasks. Many efforts have been devoted to PETs, aiming to tune only a few parameters rather than the whole PLM on downstream tasks. For example, Prompt tuning (Lester et al., 2021) prepends tunable embeddings to the input, Adapter (Houlsby et al., 2019) inserts small modules into each layer, BitFit (Zaken et al., 2022) tunes only the bias terms, and LoRA (Hu et al., 2022) decomposes the weight updates into low-rank matrices. In this paper, based on the theoretical grounding of PETs on optimal control (Yang and Liu, 2022; Ding et al., 2022), we propose stochastic bridges as the regularizer on intermediate hidden states and introduce regularized PETs.

Our work also closely relates to continuous-time neural differential equations (NDEs) (Chen et al., 2018; Rubanova et al., 2019; Li et al., 2019; Kidger et al., 2021). Continuous-time NDEs model the dynamics of the hidden states with ODEs or SDEs parameterized by neural networks. Inspired by these works, we use SDEs to represent the latent dynamics of PETs in the latent space. Our work differs from them in that we focus on using neural SDEs as regularizers for intermediate hidden states, rather than feature extractors. We also notice that Wang et al. (2022) explore the use of Brownian bridge in regularizing the model dynamics across time. Our work differs from theirs in that we regularize the dynamics of intermediate hidden states across model layers. We additionally show that other diffusion bridges can be easily applied as the regularizer. As far as we know, we are the first to show the diffusion-bridge-like dynamics for hidden states across PLM layers and use diffusion bridges as regularizers on intermediate hidden states.

## 7 Conclusion

Starting from the optimal control perspective of PETs, we notice that the existing PETs lack a running cost that regularizes the intermediate hidden states. We thus propose to use stochastic bridges in a latent space as the regularizers for PETs. Experimental results on different models, tasks and PETs show that the proposed regularizers effectively improve the PETs' performance. Our analyses further show that the hidden states of the vanilla PETs spontaneously approach our diffusion bridges, indicating that there may exists intrinsically diffusion-bridge-like dynamics in PETs. As the first work using stochastic bridges as regularizers, we show its effectiveness and generality. We believe it will be a promising direction.

## 8 Limitations

Introducing the regularizers inevitably incurs additional computational cost in the training of PETs. To show their impact on the training speed, we plot the time-performance curves for both PDF and SDE regularizers on full-set GLUE in Figures 5, 6, 7 and 8.

On different PETs, the regularized PETs with PDF regularizer has similar running time to the vanilla PETs. On the two large datasets, QQP and MNLI, regularized PETs with SDE regularizer take about 2 to 3 times longer to achieve the best performance than vanilla PETs. However, on medium-sized (QNLI, SST-2) and small datasets (CoLA, MRPC, RTE), the time to achieve the best results with SDE regularizer is comparable to vanilla PETs.

Overall, the PDF regularizer can effectively improve the performance of PETs without introducing much computational cost. In scenarios where there is relatively more focus on the inference performance of PETs and less concern about the slightly longer training time, or when the dataset is small, SDE regularizer should be a good choice.

Our method does not introduce additional risk to the original risks of PETs.

## Acknowledgement

This work is supported by the National Natural Science Foundation of China (No. 62236004, No. 62236011), Institute Guo Qiang at Tsinghua University.

## References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2018. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6572–6583.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *CoRR*, abs/2203.06904.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. 2019. Augmented neural odes. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3134–3144.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot

learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3816–3830. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Patrick Kidger, James Foster, Xuechen Li, and Terry J. Lyons. 2021. Neural sdes as infinite-dimensional gans. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 5453–5463. PMLR.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. 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, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. 2019. Scalable gradients and variational inference for stochastic differential equations. In *Symposium on Advances in Approximate Bayesian Inference, AABI 2019, Vancouver, BC, Canada, December 8, 2019*, volume 118 of *Proceedings of Machine Learning Research*, pages 1–28. PMLR.

Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. 2020. Scalable gradients for stochastic differential equations. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 3870–3882. PMLR.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. UMAP: uniform manifold approximation and projection. *J. Open Source Softw.*, 3(29):861.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Yulia Rubanova, Tian Qi Chen, and David Duvenaud. 2019. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5321–5331.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zheng, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. 2022. Using deepspeed and megatron to train megatron-turing NLG 530b, A large-scale generative language model. *CoRR*, abs/2201.11990.

K Sobczyk. 2013. *Stochastic Differential Equations: With Applications to Physics and Engineering*, volume 40. Springer Science & Business Media.

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

Rose E. Wang, Esin Durmus, Noah D. Goodman, and Tatsunori Hashimoto. 2022. Language modeling via stochastic processes. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Xiaoqun Wang and Ian H. Sloan. 2011. Quasi-monte carlo methods in financial engineering: An equivalence principle and dimension reduction. *Oper. Res.*, 59(1):80–95.

Zonghan Yang and Yang Liu. 2022. On robust prefix-tuning for text classification. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. 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 (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1–9. Association for Computational Linguistics.

Han Zhang, Xi Gao, Jacob Unterman, and Tom Arodz. 2020. Approximation capabilities of neural odes and invertible residual networks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 11086–11095. PMLR.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

## A Background for Parameter-Efficient Tuning Methods

The large number of parameters in the PLMs makes fine-tuning impractical, therefore different PETs are proposed to mitigate the problem. The current PETs can be categorized into three groups: addition-based, specification-based and reparameterization-based (Ding et al., 2022). To verify the generality of our method, we include one or two PETs from each category in this work, and we give a brief review to these PETs.

**Prompt Tuning** is an addition-based PET. It prepends or appends trainable virtual tokens $\boldsymbol{P} \in \mathbb{R}^{m\times d}$ to each sequence $\boldsymbol{x} \in \mathbb{R}^{n\times d}$ to form a new input sequence $[\boldsymbol{P};\boldsymbol{x}]$ or $[\boldsymbol{x};\boldsymbol{P}] \in \mathbb{R}^{(n+m)\times d}$, where $n, m$ are length of original sequence and virtual tokens respectively, $d$ is the embedding dimension. The virtual tokens $\boldsymbol{P}$ can be either continuous (Lester et al., 2021) or be restricted to be embeddings of discrete tokens in vocabulary (Gao et al., 2021).

**Adapter** (Houlsby et al., 2019) is an addition-based PET. It inserts two-layer MLPs after the attention module and feed-forward module at each layer. Denote $\boldsymbol{h} \in \mathbb{R}^d$ as the input of Adapter, $r$ as the intermediate dimension of Adapter's MLP, $\boldsymbol{W}_d \in \mathbb{R}^{r\times d}$, $\boldsymbol{W}_u \in \mathbb{R}^{d\times r}$ as the down-projection and up-projection of Adapter, and $\sigma$ as the activation function. Then the computation of Adapter can be formulated as

$$\boldsymbol{h} \leftarrow \boldsymbol{W}_u\sigma(\boldsymbol{W}_d\boldsymbol{x}) + \boldsymbol{h}$$

| | PDF | SDE |
|---|---|---|
| Learning rate | 1e-3 | 1e-3 |
| Weight decay | 0 | 0 |
| Batch size | 128 | 128 |
| Grad norm | 1.0 | 1.0 |
| Max steps | 100k | 500k |
| Warmup ratio | 0.01 | 0.01 |

Table 4: Hyper-parameters for training $g_\gamma$

**BitFit** (Zaken et al., 2022) is a specification-based PET. It specifies the bias terms in layer normalization modules and linear transformation modules as trainable.

**LoRA** (Hu et al., 2022) is a reparameterization-based PET. It assumes that when training the model, the updates $\Delta\boldsymbol{W}$ for model's pre-trained parameters $\boldsymbol{W} \in \mathbb{R}^{d\times k}$ are low-rank, and thus reparameterize the $\Delta\boldsymbol{W}$ of each matrix in attention module with a low-rank decomposition $\Delta\boldsymbol{W} = \boldsymbol{B}\boldsymbol{A}$, where $\boldsymbol{B} \in \mathbb{R}^{d\times r}$, $\boldsymbol{A} \in \mathbb{R}^{r\times k}$. For a forward pass $\boldsymbol{h} = \boldsymbol{W}\boldsymbol{x}$, the computation of LoRA can be written as

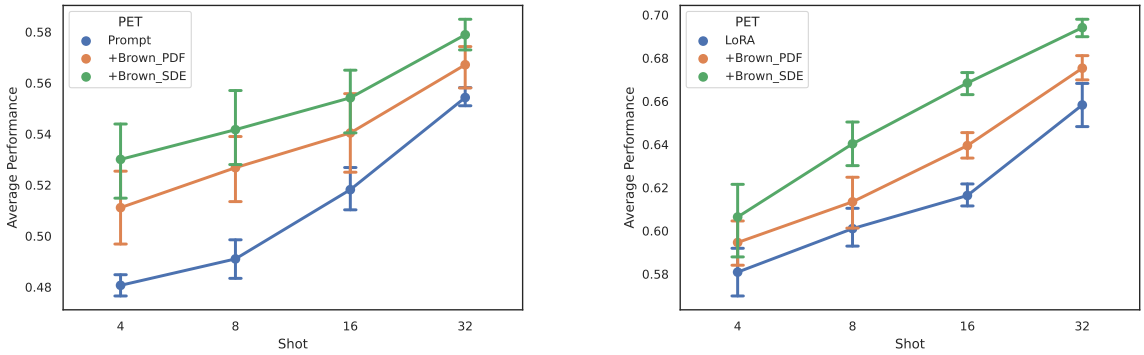
$$\boldsymbol{h} = (\boldsymbol{W} + \Delta\boldsymbol{W})\boldsymbol{x} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{A}\boldsymbol{x}$$

## B Properties for Ornstein-Uhlenbeck Bridge

**Proposition B.1** (Properties of Ornstein-Uhlenbeck Bridge)**.** A Ornstein-Uhlenbeck $X^{T;\beta}$ pinned at $X_0^{T;\beta} = 0$ and $X_T^{T;\beta} = \beta$ is the solution to the following SDE:

$$\begin{aligned} d\tilde{X}_t &= q\mu_t dt + \sigma dB_t, \\ \mu_t &= -\coth\left[q(T-t)\right]\tilde{X}_t + \frac{\beta}{\sinh\left[q(T-t)\right]}, \\ \tilde{X}_0 &= 0, \end{aligned} \tag{11}$$

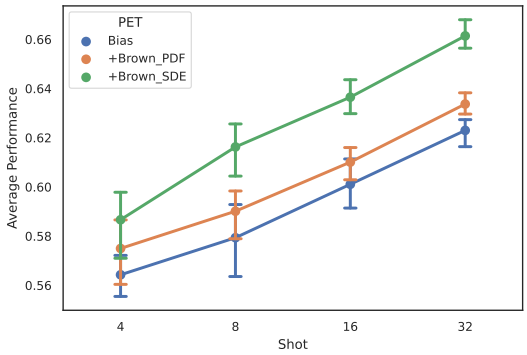
where $q$ is the diffusion coefficient and $\sigma$ is the diffusion for the OU process. The transition proba-



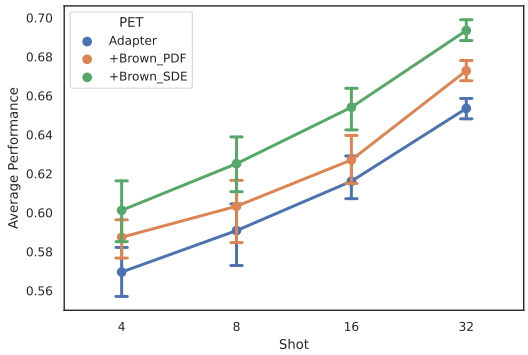

(a) PROMPT TUNING




(b) LORA




(c) BIAS TUNING




(d) ADAPTER

Figure 3: The average $\text{BERT}_{\text{large}}$ few-shot GLUE results trained with different PETs under different shots. The results are averaged across 5 different seeds and the error bars indicate the 95% confidence. SDE regularizer consistently outperforms the baseline PDF regularizer.

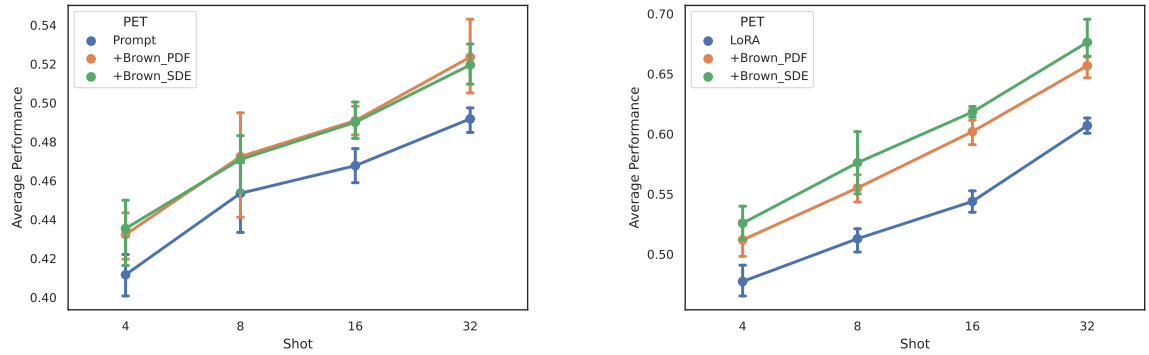
bility density function reads as:

$$
\begin{aligned}
p^{T;\beta}(t, y \mid 0, 0) &= \frac{1}{\sqrt{2\pi\sigma(s,t)}} \exp\left\{-\frac{(y-\mu(t))^2}{2\sigma(s,t)}\right\}, \\
\mu(t) &= \frac{\sinh(q(T-t))}{\sinh(q(T-s))}\beta \\
\sigma(s,t) &= \frac{\sigma^2}{q}\,\frac{\sinh(q(T-t))\sinh(q(t-s))}{\sinh(q(T-s))}.
\end{aligned} \tag{12}
$$

## C Other Results for GLUE Experiments

In this section, we present the complete results including OU bridge regularizer for Tab. 1 and Tab. 2. We also report the results for $\text{Deberta}_{\text{xlarge}}$, and the results on few-shot GLUE for both $\text{BERT}_{\text{large}}$ and $\text{Deberta}_{\text{xlarge}}$ under 4-, 8-, 16-, and 32-shot. We observe that $\text{BERT}_{\text{large}}$ cannot give reasonable answers on CoLA and the Matthews correlations are around 0 for all the PETs and all the shots we have experienced with. However, the situation gets better for the larger model $\text{Deberta}_{\text{xlarge}}$. Therefore, we exclude CoLA for $\text{BERT}_{\text{large}}$ and keep it for $\text{Deberta}_{\text{xlarge}}$. We only select the Brownian bridge as the representative in this section, since the Brownian bridge and Ornstein-Uhlenbeck bridge have no significant difference in Tab. 1.

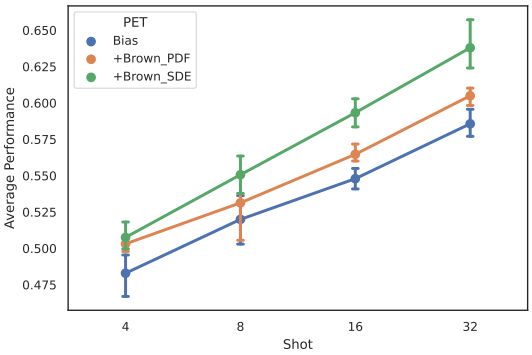
In Tab. 6 and Tab. 5, we report the performance of OU bridge regularizers. The experimental setups are the same as Tab. 1 and Tab. 2 respectively. The performances between OU bridge and Brownian bridge do not have a significant difference. In Tab. 7, we report the performance of $\text{Deberta}_{\text{xlarge}}$ on full GLUE datasets. On all four PETs, the SDE regularizer outperforms the PDF regularizer, this is consistent with the results we see in Tab. 1. The results for 4-, 8-, and 32-shot for $\text{BERT}_{\text{large}}$ and $\text{Deberta}_{\text{xlarge}}$ are plotted respectively in Fig. 3 and Fig. 4. For simplicity, we only plot the average performance for each PET. We report the results for 16-shot experiments for $\text{Deberta}_{\text{xlarge}}$ in Tab. 8. The setup for the experiment is almost the same as the experiment in Sec. 4.3, and the hyper-parameters are listed in Appx. E. The SDE regularizer outperforms the PDF regularizer on most of the PETs except Prompt tuning. We no-



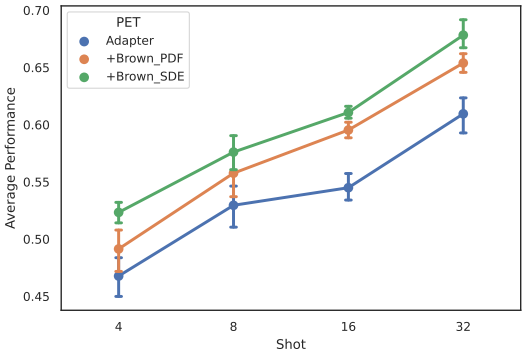

(a) PROMPT TUNING




(b) LORA




(c) BITFIT




(d) ADAPTER

Figure 4: The average $\text{Deberta}_{\text{xlarge}}$ few-shot GLUE results trained with different PETs under different shots. The results are averaged across 5 different seeds and the error bars indicate the 95% confidence.

tice that the SDE regularizer helps $\text{Deberta}_{\text{xlarge}}$ substantially on CoLA for most of the PETs, indicating the SDE regularizer can effectively provide useful guidance when the data is scarce and the task is hard.

## D Calculation of Correlation in Sec. 5

In this section, we elaborate on how we calculate the correlations reported in Tab. 3.

### D.1 Correlation between Number of Shots and Distance to Bridge

**Definition D.1** (Tie)**.** A pair of observation $\{(x_i, y_i), (x_j, y_j)\}$ is defined as tied if $x_i = x_j$ or $y_i = y_j$.

Since we generate the few-shot datasets using 5 random seeds for each shot, each PET has 5 results for each shot. This results in observations with ties, e.g., the two observations for distances on the first seed and second seed for 8-shot $\{(8, d_1), (8, d_2)\}$ are tied. To calculate the correlation for data with ties, the Tau-b of Kendall's rank correlation is more suitable than Pearson's correlation. We therefore report the Kendall's rank correlation for the correlation between the number of shots and the hidden states distance to the latent bridges.

### D.2 Correlation between Performance and Distance to Bridge

We mix all the few-shot results for different shots and different seeds to form observations of performance and the hidden states distances to bridges, and then calculate the Pearson's correlation.

## E Hyper-parameters

### E.1 Training $g_\gamma$

we use simple 3-layer MLP for $g_\gamma$ in all of our experiments. For PDF regularizer, the output dimensions of each layer are 1024, 256, 128, and for SDE regularizer, the output dimensions of each layer are 1024, 256, 32. We observe that the running time increases noticeably with the final output dimension for SDE regularizer, we thus choose a smaller one for SDE regularizer. The hyper-parameters for training $g_\gamma$ are listed in Tab. 4.

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | RTE | Average | Δ |
|---|---|---|---|---|---|---|---|---|
| PROMPT | $38.1_{1.5}$ | $53.0_{3.1}$ | $51.6_{1.4}$ | $70.1_{4.9}$ | $50.1_{3.0}$ | $48.0_{1.3}$ | $51.8_{0.9}$ | - |
| +BROWN_PDF | $38.7_{2.3}$ | $54.9_{2.8}$ | $52.1_{1.1}$ | $75.0_{11.0}$ | $52.8_{2.2}$ | $50.8_{3.3}$ | $54.0_{1.8}$ | 2.2 |
| +BROWN_SDE | $\mathbf{40.6}_{0.8}$ | $55.4_{2.1}$ | $52.9_{1.6}$ | $80.0_{10.9}$ | $51.9_{3.6}$ | $51.7_{3.1}$ | $55.4_{1.5}$ | 3.6 |
| +OU_PDF | $38.9_{1.6}$ | $54.9_{2.5}$ | $51.8_{2.0}$ | $71.4_{9.3}$ | $\mathbf{54.3}_{2.6}$ | $51.5_{3.5}$ | $53.8_{1.9}$ | 2.0 |
| +OU_SDE | $40.0_{1.7}$ | $\mathbf{56.8}_{1.5}$ | $\mathbf{53.5}_{2.2}$ | $\mathbf{82.7}_{3.2}$ | $52.7_{2.1}$ | $\mathbf{52.0}_{1.9}$ | $\mathbf{56.3}_{0.5}$ | **4.5** |
| LORA | $48.7_{4.5}$ | $59.9_{5.5}$ | $53.2_{1.2}$ | $90.2_{1.1}$ | $53.6_{3.4}$ | $64.2_{0.9}$ | $61.6_{0.6}$ | - |
| +BROWN_PDF | $52.0_{1.3}$ | $62.7_{2.2}$ | $55.1_{3.8}$ | $91.3_{0.1}$ | $57.4_{5.0}$ | $65.3_{1.5}$ | $64.0_{0.7}$ | 2.4 |
| +BROWN_SDE | $\mathbf{54.1}_{0.9}$ | $\mathbf{65.5}_{1.4}$ | $64.3_{5.1}$ | $91.2_{0.3}$ | $60.2_{3.1}$ | $65.8_{1.4}$ | $\mathbf{66.8}_{0.6}$ | **5.2** |
| +OU_PDF | $51.6_{2.4}$ | $62.5_{2.3}$ | $55.3_{3.4}$ | $\mathbf{91.4}_{0.5}$ | $56.9_{5.2}$ | $65.6_{0.4}$ | $63.9_{1.1}$ | 2.3 |
| +OU_SDE | $52.9_{2.6}$ | $63.1_{1.5}$ | $\mathbf{64.4}_{5.1}$ | $91.3_{0.3}$ | $\mathbf{62.2}_{2.7}$ | $\mathbf{66.2}_{1.1}$ | $66.7_{0.7}$ | 5.1 |
| BITFIT | $48.4_{1.6}$ | $56.0_{6.1}$ | $51.7_{2.5}$ | $90.8_{0.8}$ | $52.0_{2.3}$ | $61.7_{1.4}$ | $60.1_{1.1}$ | - |
| +BROWN_PDF | $48.5_{1.9}$ | $56.0_{6.0}$ | $53.5_{2.0}$ | $91.0_{0.4}$ | $53.9_{2.3}$ | $63.2_{1.6}$ | $61.0_{0.8}$ | 0.9 |
| +BROWN_SDE | $\mathbf{52.3}_{0.5}$ | $\mathbf{61.2}_{2.9}$ | $58.8_{4.7}$ | $90.8_{0.4}$ | $54.8_{3.0}$ | $\mathbf{63.9}_{2.5}$ | $\mathbf{63.6}_{0.8}$ | **3.5** |
| +OU_PDF | $49.0_{1.8}$ | $56.1_{6.1}$ | $52.8_{2.2}$ | $90.8_{0.7}$ | $53.5_{2.2}$ | $62.5_{1.3}$ | $60.8_{0.8}$ | 0.7 |
| +OU_SDE | $51.8_{1.2}$ | $58.4_{2.2}$ | $\mathbf{58.9}_{4.1}$ | $\mathbf{91.1}_{0.4}$ | $\mathbf{56.4}_{3.6}$ | $63.5_{1.4}$ | $63.4_{0.8}$ | 3.3 |
| ADAPTER | $47.4_{3.7}$ | $57.0_{7.2}$ | $55.8_{2.9}$ | $91.0_{0.4}$ | $55.8_{2.5}$ | $62.7_{2.0}$ | $61.6_{1.2}$ | - |
| +BROWN_PDF | $49.0_{4.8}$ | $58.5_{7.4}$ | $56.9_{3.1}$ | $91.4_{0.2}$ | $57.2_{4.9}$ | $63.2_{3.0}$ | $62.7_{1.5}$ | 1.1 |
| +BROWN_SDE | $\mathbf{52.3}_{2.2}$ | $62.4_{2.9}$ | $\mathbf{64.8}_{4.5}$ | $\mathbf{91.9}_{0.4}$ | $57.3_{4.1}$ | $\mathbf{63.8}_{1.8}$ | $\mathbf{65.4}_{1.3}$ | **3.8** |
| +OU_PDF | $48.4_{4.3}$ | $58.4_{7.4}$ | $57.7_{3.7}$ | $91.3_{0.5}$ | $57.1_{3.3}$ | $63.0_{1.8}$ | $62.6_{1.3}$ | 1.0 |
| +OU_SDE | $48.3_{4.5}$ | $\mathbf{62.8}_{1.8}$ | $63.8_{4.7}$ | $91.1_{0.4}$ | $\mathbf{59.8}_{2.3}$ | $63.6_{1.9}$ | $64.9_{0.6}$ | 3.3 |

Table 5: The complete results on GLUE for $\text{BERT}_{\text{large}}$ under 16-shot setting. We exclude CoLA because all PETs fail to give reasonable result in few-shot setting.

### E.2 Training PETs on full-set GLUE

We run all the experiments for 50k steps, and evaluate on the development set every 1k steps. For $\text{BERT}_{\text{large}}$, we use 32 as the batch size while for $\text{Deberta}_{\text{xlarge}}$, we use 16 as the batch size. We choose learning rate 1e-3 for Prompt tuning for both PLMs, and 1e-4 for other PETs for both PLMs. We use 0.01 weight decay, 1.0 maximum gradient norm and no learning rate warm-up for all the experiments. We search the best regularization strength $\alpha$ in {0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0} for PDF regularizer, and in {0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.5, 1.0} for SDE regularizer. The best $\alpha$ for PDF regularizer are listed in Tab. 9, and best $\alpha$ for SDE regularizer are listed in Tab. 10.

### E.3 Training PETs on few-shot GLUE

We run all the experiments for 1k steps, and evaluate on the development set every 50 steps. For all the shots for both regularizers and both models, we use a batch size of 2. Other hyper-parameters are kept the same as in the experiments on full-set GLUE.

## F Performance of Regularizers Trained with Tiny Corpus

Although we use the pre-training corpus to train our mapping $g_\gamma$, the training is actually fast and data-efficient. We show that when using only 10,000 documents in the pre-training corpus (about 0.1% of the corpus), the obtained regularizers still perform great and are comparable to the regularizers trained on the whole pre-training corpus. We train the mapping $g_\gamma$ for 5,000 iterations with 128 batch size. On a single NVIDIA A100 GPU, the training can be done in 1 hour for PDF regularizer, and 3 hours for SDE regularizer. The cost of training our regularizer is quite small compared to the resources required for pre-training.

We conduct the same experiments as Sec. 4.2 and Sec. 4.3 with the regularizers trained with tiny corpus. The results are presented in Tab. 11 and Tab. 12 respectively.

On full-set GLUE, the PDF regularizer performs even better on three out of four PETs, and although its performance is affected on Adapter, it still outperforms the vanilla Adapter. The SDE regularizer is slightly affected on three out of four PETs, but

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | CoLA | RTE | Average | Δ |
|---|---|---|---|---|---|---|---|---|---|
| PROMPT | $84.4_{0.1}$ | $85.3_{0.3}$ | $91.5_{0.1}$ | $95.5_{0.1}$ | $73.9_{2.4}$ | $55.5_{3.4}$ | $60.8_{1.5}$ | $78.1_{0.6}$ | - |
| +BROWN_PDF | $84.7_{0.2}$ | $\mathbf{85.5}_{0.0}$ | $91.8_{0.6}$ | $95.7_{0.1}$ | $75.4_{0.5}$ | $56.4_{3.3}$ | $61.5_{2.2}$ | $78.7_{0.4}$ | 0.6 |
| +OU_PDF | $84.7_{0.1}$ | $85.4_{0.1}$ | $91.8_{0.3}$ | $95.6_{0.2}$ | $76.9_{1.0}$ | $57.1_{1.2}$ | $60.5_{3.0}$ | $78.9_{0.2}$ | 0.8 |
| +BROWN_SDE | $\mathbf{84.9}_{0.2}$ | $85.4_{0.1}$ | $91.8_{0.4}$ | $\mathbf{95.8}_{0.3}$ | $78.8_{1.2}$ | $61.4_{2.9}$ | $64.7_{1.1}$ | $80.4_{0.2}$ | 2.3 |
| +OU_SDE | $84.7_{0.2}$ | $85.3_{0.1}$ | $\mathbf{92.1}_{0.3}$ | $95.5_{0.2}$ | $\mathbf{80.2}_{0.5}$ | $\mathbf{61.5}_{3.5}$ | $\mathbf{65.9}_{2.7}$ | $\mathbf{80.7}_{0.8}$ | **2.6** |
| LORA | $88.8_{0.1}$ | $89.2_{0.2}$ | $93.5_{0.2}$ | $95.5_{0.1}$ | $84.6_{0.4}$ | $62.8_{1.6}$ | $78.9_{1.6}$ | $84.8_{0.3}$ | - |
| +BROWN_PDF | $\mathbf{88.9}_{0.1}$ | $\mathbf{89.6}_{0.1}$ | $\mathbf{93.9}_{0.1}$ | $95.6_{0.2}$ | $85.1_{0.7}$ | $63.7_{0.5}$ | $80.0_{0.5}$ | $85.2_{0.1}$ | 0.4 |
| +OU_PDF | $\mathbf{88.9}_{0.2}$ | $89.4_{0.1}$ | $93.7_{0.1}$ | $\mathbf{95.7}_{0.3}$ | $86.0_{0.5}$ | $63.6_{0.6}$ | $80.5_{0.8}$ | $85.4_{0.1}$ | 0.6 |
| +BROWN_SDE | $\mathbf{88.9}_{0.1}$ | $89.5_{0.1}$ | $93.7_{0.1}$ | $\mathbf{95.7}_{0.1}$ | $\mathbf{86.5}_{1.2}$ | $\mathbf{63.9}_{0.4}$ | $\mathbf{80.9}_{0.8}$ | $\mathbf{85.6}_{0.1}$ | **0.8** |
| +OU_SDE | $88.8_{0.0}$ | $89.5_{0.1}$ | $93.7_{0.2}$ | $\mathbf{95.7}_{0.3}$ | $86.3_{1.2}$ | $63.7_{0.6}$ | $80.1_{0.9}$ | $85.4_{0.1}$ | 0.6 |
| BITFIT | $\mathbf{87.9}_{0.2}$ | $87.6_{0.1}$ | $92.7_{0.2}$ | $95.6_{0.1}$ | $79.4_{2.3}$ | $60.2_{0.8}$ | $77.0_{1.5}$ | $82.9_{0.3}$ | - |
| +BROWN_PDF | $\mathbf{87.9}_{0.1}$ | $\mathbf{87.8}_{0.0}$ | $\mathbf{93.0}_{0.2}$ | $95.7_{0.1}$ | $83.1_{0.8}$ | $60.3_{0.6}$ | $78.3_{0.9}$ | $83.7_{0.2}$ | 0.8 |
| +OU_PDF | $\mathbf{87.9}_{0.1}$ | $\mathbf{87.8}_{0.1}$ | $\mathbf{93.0}_{0.1}$ | $95.7_{0.1}$ | $82.6_{0.8}$ | $59.8_{1.0}$ | $\mathbf{78.8}_{1.4}$ | $83.6_{0.4}$ | 0.7 |
| +BROWN_SDE | $\mathbf{87.9}_{0.2}$ | $87.7_{0.0}$ | $92.8_{0.1}$ | $95.7_{0.1}$ | $83.3_{0.8}$ | $\mathbf{61.1}_{1.2}$ | $77.7_{1.5}$ | $\mathbf{83.8}_{0.3}$ | **0.9** |
| +OU_SDE | $\mathbf{87.9}_{0.1}$ | $87.6_{0.1}$ | $92.7_{0.1}$ | $\mathbf{95.8}_{0.1}$ | $\mathbf{83.7}_{0.8}$ | $60.8_{1.6}$ | $77.1_{0.9}$ | $83.7_{0.4}$ | 0.8 |
| ADAPTER | $88.8_{0.1}$ | $89.6_{0.3}$ | $93.7_{0.1}$ | $95.6_{0.1}$ | $83.6_{0.1}$ | $60.4_{1.2}$ | $79.5_{1.2}$ | $84.5_{0.3}$ | - |
| +BROWN_PDF | $\mathbf{89.0}_{0.1}$ | $89.7_{0.2}$ | $93.8_{0.2}$ | $\mathbf{95.8}_{0.1}$ | $86.5_{1.1}$ | $\mathbf{62.6}_{0.7}$ | $\mathbf{83.2}_{0.2}$ | $\mathbf{85.8}_{0.2}$ | **1.3** |
| +OU_PDF | $88.9_{0.1}$ | $89.7_{0.0}$ | $93.8_{0.1}$ | $\mathbf{95.8}_{0.1}$ | $\mathbf{86.8}_{0.6}$ | $61.9_{0.2}$ | $82.0_{0.6}$ | $85.6_{0.1}$ | 1.1 |
| +BROWN_SDE | $88.9_{0.1}$ | $\mathbf{89.8}_{0.1}$ | $\mathbf{93.9}_{0.2}$ | $\mathbf{95.8}_{0.1}$ | $85.9_{0.4}$ | $62.3_{1.8}$ | $82.2_{0.2}$ | $85.5_{0.2}$ | 1.0 |
| +OU_SDE | $88.9_{0.1}$ | $\mathbf{89.8}_{0.1}$ | $93.7_{0.1}$ | $95.7_{0.1}$ | $85.9_{0.7}$ | $62.5_{1.2}$ | $82.7_{0.5}$ | $85.6_{0.2}$ | 1.1 |

Table 6: The complete results on GLUE for $\text{BERT}_{\text{large}}$. The values are the average value of the best performances over three different runs, and the subscripts are the standard deviations. The Δ column shows the difference of the average performance between the PETs with and without our regularizers.

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | CoLA | RTE | Average | Δ |
|---|---|---|---|---|---|---|---|---|---|
| PROMPT | $87.2_{0.1}$ | $86.5_{0.1}$ | $93.8_{0.1}$ | $\mathbf{96.8}_{0.1}$ | $75.4_{2.5}$ | $64.2_{3.8}$ | $78.8_{3.5}$ | $83.2_{0.5}$ | - |
| +BROWN_PDF | $\mathbf{87.6}_{0.1}$ | $\mathbf{86.8}_{0.3}$ | $\mathbf{94.2}_{0.1}$ | $\mathbf{96.8}_{0.1}$ | $80.3_{2.9}$ | $\mathbf{65.5}_{1.3}$ | $\mathbf{79.8}_{0.5}$ | $84.5_{0.6}$ | 1.3 |
| +BROWN_SDE | $\mathbf{87.6}_{0.1}$ | $\mathbf{86.8}_{0.1}$ | $94.0_{0.1}$ | $\mathbf{96.8}_{0.1}$ | $\mathbf{84.4}_{0.6}$ | $64.8_{0.5}$ | $79.5_{1.3}$ | $\mathbf{84.8}_{0.2}$ | **1.6** |
| LORA | $\mathbf{91.1}_{0.1}$ | $90.3_{0.1}$ | $95.1_{0.1}$ | $96.8_{0.1}$ | $88.7_{0.7}$ | $68.0_{1.3}$ | $83.4_{1.1}$ | $87.6_{0.3}$ | - |
| +BROWN_PDF | $\mathbf{91.1}_{0.0}$ | $\mathbf{90.5}_{0.0}$ | $\mathbf{95.2}_{0.0}$ | $\mathbf{97.0}_{0.1}$ | $90.1_{0.8}$ | $68.6_{0.8}$ | $\mathbf{85.9}_{1.3}$ | $88.3_{0.2}$ | 0.7 |
| +BROWN_SDE | $\mathbf{91.1}_{0.1}$ | $90.4_{0.0}$ | $95.1_{0.0}$ | $96.9_{0.2}$ | $\mathbf{90.5}_{0.6}$ | $\mathbf{69.6}_{1.1}$ | $85.6_{0.9}$ | $\mathbf{88.5}_{0.1}$ | **0.9** |
| BITFIT | $90.0_{0.1}$ | $\mathbf{88.4}_{0.0}$ | $\mathbf{95.0}_{0.0}$ | $\mathbf{96.6}_{0.1}$ | $87.3_{0.6}$ | $66.9_{0.2}$ | $82.4_{0.6}$ | $86.7_{0.1}$ | - |
| +BROWN_PDF | $\mathbf{90.2}_{0.0}$ | $88.3_{0.1}$ | $\mathbf{95.0}_{0.1}$ | $\mathbf{96.6}_{0.1}$ | $89.8_{0.5}$ | $\mathbf{67.9}_{0.8}$ | $82.9_{0.6}$ | $87.2_{0.1}$ | 0.5 |
| +BROWN_SDE | $90.1_{0.1}$ | $88.3_{0.0}$ | $94.8_{0.0}$ | $\mathbf{96.6}_{0.1}$ | $\mathbf{90.4}_{0.5}$ | $\mathbf{67.9}_{0.4}$ | $\mathbf{83.8}_{0.5}$ | $\mathbf{87.4}_{0.1}$ | **0.7** |
| ADAPTER | $91.1_{0.1}$ | $90.0_{0.1}$ | $95.2_{0.0}$ | $96.8_{0.2}$ | $87.9_{0.5}$ | $68.8_{1.8}$ | $85.0_{0.6}$ | $87.8_{0.2}$ | - |
| +BROWN_PDF | $\mathbf{91.2}_{0.1}$ | $90.0_{0.0}$ | $\mathbf{95.3}_{0.1}$ | $\mathbf{96.9}_{0.2}$ | $89.2_{0.8}$ | $70.1_{1.0}$ | $\mathbf{86.9}_{1.5}$ | $88.5_{0.5}$ | 0.7 |
| +BROWN_SDE | $\mathbf{91.2}_{0.2}$ | $\mathbf{90.1}_{0.1}$ | $95.2_{0.0}$ | $\mathbf{96.9}_{0.2}$ | $\mathbf{90.3}_{0.9}$ | $\mathbf{70.8}_{1.1}$ | $86.3_{1.4}$ | $\mathbf{88.7}_{0.4}$ | **0.9** |

Table 7: The results on GLUE for $\text{Deberta}_{\text{xlarge}}$.

it still brings substantial improvements on all the PETs.

On few-shot GLUE, the impact of the shrinkage of the corpus is relatively obvious. But overall, the regularizers still performs great on all the PETs. The drop in performances are relatively small compared to the boost they bring to vanilla PETs.

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | CoLA | RTE | Average | Δ |
|---|---|---|---|---|---|---|---|---|---|
| PROMPT | $34.4_{1.2}$ | $53.2_{5.1}$ | $51.7_{1.7}$ | $73.3_{8.7}$ | $50.2_{3.2}$ | $2.5_{2.5}$ | $52.2_{4.9}$ | $45.4_{2.1}$ | - |
| +BROWN_PDF | $35.8_{0.9}$ | $57.7_{2.4}$ | $\mathbf{53.5}_{1.5}$ | $\mathbf{87.5}_{3.0}$ | $52.2_{1.9}$ | $2.8_{2.9}$ | $\mathbf{54.1}_{1.8}$ | $\mathbf{49.1}_{0.8}$ | **3.7** |
| +BROWN_SDE | $\mathbf{35.9}_{1.6}$ | $\mathbf{59.6}_{6.4}$ | $53.1_{1.9}$ | $82.1_{4.2}$ | $\mathbf{55.4}_{1.4}$ | $\mathbf{2.9}_{3.0}$ | $\mathbf{54.1}_{1.3}$ | $49.0_{1.1}$ | 3.6 |
| LORA | $43.1_{3.6}$ | $68.4_{2.9}$ | $60.1_{5.3}$ | $\mathbf{91.8}_{1.1}$ | $57.6_{1.5}$ | $3.1_{3.8}$ | $56.6_{2.3}$ | $54.4_{1.0}$ | - |
| +BROWN_PDF | $\mathbf{52.1}_{3.4}$ | $70.2_{2.9}$ | $\mathbf{73.3}_{6.3}$ | $91.7_{1.1}$ | $59.5_{4.5}$ | $14.2_{5.6}$ | $60.4_{4.2}$ | $60.2_{1.2}$ | 5.8 |
| +BROWN_SDE | $49.6_{4.3}$ | $\mathbf{70.6}_{1.5}$ | $72.4_{6.0}$ | $90.7_{1.2}$ | $\mathbf{59.8}_{4.5}$ | $\mathbf{28.9}_{1.5}$ | $\mathbf{60.6}_{3.1}$ | $\mathbf{61.8}_{0.5}$ | **7.4** |
| BITFIT | $41.9_{3.8}$ | $67.7_{2.6}$ | $60.3_{4.2}$ | $\mathbf{91.8}_{0.8}$ | $54.9_{2.5}$ | $9.4_{2.4}$ | $57.6_{2.0}$ | $54.8_{0.8}$ | - |
| +BROWN_PDF | $45.2_{3.7}$ | $\mathbf{70.3}_{1.2}$ | $65.4_{6.7}$ | $90.9_{0.8}$ | $55.6_{2.1}$ | $8.2_{2.5}$ | $\mathbf{59.6}_{2.3}$ | $56.5_{0.7}$ | 1.7 |
| +BROWN_SDE | $\mathbf{45.7}_{3.8}$ | $69.2_{2.3}$ | $\mathbf{69.2}_{6.3}$ | $89.7_{1.3}$ | $\mathbf{57.8}_{4.0}$ | $\mathbf{24.2}_{4.6}$ | $59.4_{2.8}$ | $\mathbf{59.3}_{1.2}$ | **4.5** |
| ADAPTER | $43.1_{2.9}$ | $67.7_{2.7}$ | $55.9_{5.3}$ | $\mathbf{91.1}_{0.9}$ | $56.1_{2.1}$ | $8.6_{5.6}$ | $59.0_{2.3}$ | $54.5_{1.4}$ | - |
| +BROWN_PDF | $\mathbf{50.7}_{3.0}$ | $70.1_{1.6}$ | $70.9_{5.2}$ | $90.6_{1.7}$ | $57.6_{4.3}$ | $16.1_{7.4}$ | $\mathbf{60.6}_{3.9}$ | $59.5_{0.8}$ | 5.0 |
| +BROWN_SDE | $47.1_{1.6}$ | $\mathbf{72.0}_{1.1}$ | $\mathbf{71.3}_{4.3}$ | $91.0_{1.0}$ | $\mathbf{59.7}_{4.5}$ | $\mathbf{26.4}_{5.5}$ | $60.0_{5.7}$ | $\mathbf{61.1}_{0.6}$ | **6.6** |

Table 8: The results on GLUE for $\text{Deberta}_{\text{xlarge}}$ under 16-shot setting.

| | MNLI | QQP | QNLI | SST-2 | MRPC | CoLA | RTE |
|---|---|---|---|---|---|---|---|
| *BERT$_{large}$* | | | | | | | |
| PROMPT | 0.05 | 0.3 | 0.01 | 0.2 | 0.8 | 0.4 | 0.4 |
| LORA | 0.2 | 0.8 | 0.5 | 0.1 | 0.1 | 0.5 | 0.3 |
| BIAS | 0.05 | 0.3 | 0.6 | 0.05 | 0.8 | 0.05 | 0.7 |
| ADAPTER | 0.8 | 0.01 | 0.7 | 0.1 | 0.6 | 0.4 | 0.3 |
| *Deberta$_{xlarge}$* | | | | | | | |
| PROMPT | 0.05 | 0.1 | 0.2 | 0.8 | 0.7 | 0.3 | 0.7 |
| LORA | 0.1 | 0.1 | 0.05 | 0.01 | 0.1 | 0.01 | 0.2 |
| BIAS | 0.05 | 0.01 | 0.01 | 0.01 | 0.8 | 0.01 | 0.8 |
| ADAPTER | 0.3 | 0.01 | 0.01 | 0.01 | 0.01 | 0.6 | 0.1 |

Table 9: Best $\alpha$ for PDF regularizer on full-set GLUE

| | MNLI | QQP | QNLI | SST-2 | MRPC | CoLA | RTE |
|---|---|---|---|---|---|---|---|
| *BERT$_{large}$* | | | | | | | |
| PROMPT | 0.005 | 0.005 | 0.0005 | 0.0001 | 1.0 | 0.2 | 0.05 |
| LORA | 0.0005 | 0.01 | 0.01 | 0.01 | 0.005 | 0.0005 | 0.0005 |
| BIAS | 0.001 | 0.001 | 0.005 | 0.0001 | 1.0 | 1.0 | 0.005 |
| ADAPTER | 0.001 | 0.001 | 0.0005 | 0.001 | 0.005 | 0.5 | 0.005 |
| *Deberta$_{xlarge}$* | | | | | | | |
| PROMPT | 0.0001 | 0.001 | 0.0001 | 0.0001 | 0.2 | 0.0001 | 0.0001 |
| LORA | 0.0005 | 0.05 | 0.0001 | 0.0001 | 0.1 | 0.005 | 0.0001 |
| BIAS | 0.0001 | 0.0005 | 0.0001 | 0.0001 | 0.05 | 0.2 | 0.001 |
| ADAPTER | 0.001 | 0.0005 | 0.0001 | 0.0005 | 0.1 | 0.001 | 0.0005 |

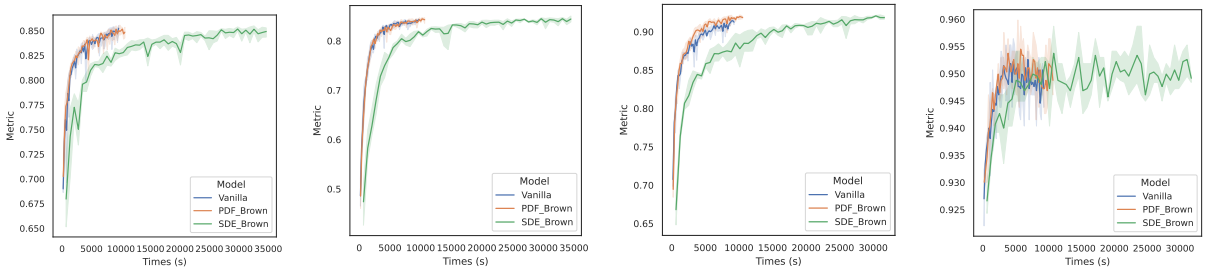
Table 10: Best $\alpha$ for SDE regularizer on full-set GLUE

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | CoLA | RTE | Average | Δ | $\Delta_{\text{whole}}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| PROMPT | $84.4_{0.1}$ | $85.3_{0.3}$ | $91.5_{0.1}$ | $95.5_{0.1}$ | $73.9_{2.4}$ | $55.5_{3.4}$ | $60.8_{1.5}$ | $78.1_{0.6}$ | - | - |
| +BROWN_PDF | $84.7_{0.0}$ | $\mathbf{85.4}_{0.1}$ | $\mathbf{92.1}_{0.1}$ | $\mathbf{95.8}_{0.2}$ | $75.8_{1.4}$ | $56.6_{0.9}$ | $61.1_{3.0}$ | $78.8_{0.5}$ | 0.7 | +0.1 |
| +BROWN_SDE | $\mathbf{84.8}_{0.2}$ | $\mathbf{85.4}_{0.1}$ | $\mathbf{92.1}_{0.1}$ | $\mathbf{95.8}_{0.2}$ | $\mathbf{79.0}_{1.3}$ | $\mathbf{59.8}_{8.2}$ | $\mathbf{65.5}_{1.2}$ | $\mathbf{80.3}_{1.2}$ | **2.2** | -0.1 |
| LORA | $88.8_{0.1}$ | $89.2_{0.2}$ | $93.5_{0.2}$ | $95.5_{0.1}$ | $84.6_{0.4}$ | $62.8_{1.6}$ | $78.9_{1.6}$ | $84.8_{0.3}$ | - | - |
| +BROWN_PDF | $\mathbf{88.9}_{0.1}$ | $\mathbf{89.5}_{0.1}$ | $\mathbf{93.8}_{0.1}$ | $\mathbf{95.8}_{0.2}$ | $86.0_{0.6}$ | $\mathbf{64.4}_{0.6}$ | $80.4_{0.3}$ | $\mathbf{85.5}_{0.1}$ | **0.7** | +0.3 |
| +BROWN_SDE | $\mathbf{88.9}_{0.0}$ | $89.4_{0.1}$ | $\mathbf{93.8}_{0.1}$ | $95.6_{0.2}$ | $\mathbf{86.8}_{0.5}$ | $63.5_{0.9}$ | $\mathbf{80.9}_{0.8}$ | $\mathbf{85.5}_{0.2}$ | **0.7** | -0.1 |
| BITFIT | $\mathbf{87.9}_{0.2}$ | $87.6_{0.1}$ | $92.7_{0.2}$ | $95.6_{0.1}$ | $79.4_{2.3}$ | $60.2_{0.8}$ | $77.0_{1.5}$ | $82.9_{0.3}$ | - | - |
| +BROWN_PDF | $\mathbf{87.9}_{0.1}$ | $\mathbf{87.7}_{0.1}$ | $\mathbf{92.8}_{0.1}$ | $\mathbf{95.7}_{0.2}$ | $\mathbf{83.6}_{0.7}$ | $\mathbf{61.2}_{0.2}$ | $\mathbf{78.7}_{1.1}$ | $\mathbf{84.0}_{0.1}$ | **1.1** | +0.3 |
| +BROWN_SDE | $\mathbf{87.9}_{0.1}$ | $\mathbf{87.7}_{0.1}$ | $\mathbf{92.8}_{0.1}$ | $\mathbf{95.7}_{0.1}$ | $82.8_{0.7}$ | $61.1_{0.8}$ | $77.4_{0.6}$ | $83.6_{0.3}$ | 0.7 | -0.2 |
| ADAPTER | $88.8_{0.1}$ | $89.6_{0.3}$ | $93.7_{0.1}$ | $95.6_{0.1}$ | $83.6_{0.1}$ | $60.4_{1.2}$ | $79.5_{1.2}$ | $84.5_{0.3}$ | - | - |
| +BROWN_PDF | $\mathbf{88.9}_{0.1}$ | $\mathbf{89.8}_{0.1}$ | $\mathbf{93.8}_{0.2}$ | $\mathbf{95.9}_{0.2}$ | $85.4_{0.8}$ | $60.9_{0.5}$ | $\mathbf{82.7}_{1.8}$ | $85.3_{0.2}$ | 0.8 | -0.5 |
| +BROWN_SDE | $\mathbf{88.9}_{0.0}$ | $\mathbf{89.8}_{0.1}$ | $93.7_{0.1}$ | $95.8_{0.3}$ | $\mathbf{85.7}_{0.5}$ | $\mathbf{61.9}_{0.9}$ | $82.4_{0.3}$ | $\mathbf{85.5}_{0.2}$ | **1.0** | 0.0 |

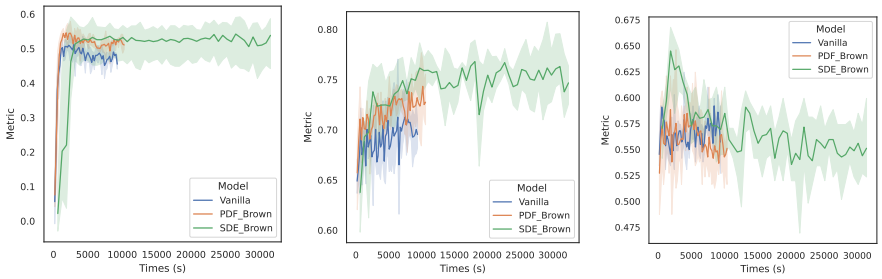
Table 11: The results on GLUE for $\text{BERT}_{\text{large}}$ *with regularizers trained on 0.1% of the pre-training corpus*. $\Delta_{\text{whole}}$ is the difference between the the average performance in this table and the average performance in Tab. 1.

| PET | MNLI | QQP | QNLI | SST-2 | MRPC | RTE | Average | Δ | $\Delta_{\text{whole}}$ |
|---|---|---|---|---|---|---|---|---|---|
| PROMPT | $38.1_{1.5}$ | $53.0_{3.1}$ | $51.6_{1.4}$ | $70.1_{4.9}$ | $50.1_{3.0}$ | $48.0_{1.3}$ | $51.8_{0.9}$ | - | - |
| +BROWN_PDF | $38.4_{1.5}$ | $54.6_{2.5}$ | $52.4_{1.4}$ | $73.1_{6.0}$ | $\mathbf{54.0}_{1.4}$ | $\mathbf{51.1}_{1.7}$ | $53.9_{1.1}$ | 2.1 | -0.1 |
| +BROWN_SDE | $\mathbf{39.3}_{1.1}$ | $\mathbf{56.2}_{3.0}$ | $\mathbf{52.8}_{1.1}$ | $\mathbf{80.8}_{7.1}$ | $53.0_{6.0}$ | $51.0_{2.5}$ | $\mathbf{55.5}_{0.9}$ | **3.7** | +0.1 |
| LORA | $48.7_{4.5}$ | $59.9_{5.5}$ | $53.2_{1.2}$ | $90.2_{1.1}$ | $53.6_{3.4}$ | $64.2_{0.9}$ | $61.6_{0.6}$ | - | - |
| +BROWN_PDF | $51.4_{2.2}$ | $62.0_{1.8}$ | $54.8_{3.1}$ | $\mathbf{91.2}_{0.3}$ | $57.4_{4.0}$ | $65.7_{0.7}$ | $63.7_{0.9}$ | 2.1 | -0.3 |
| +BROWN_SDE | $\mathbf{53.2}_{1.9}$ | $\mathbf{65.4}_{1.7}$ | $\mathbf{64.2}_{5.0}$ | $\mathbf{91.2}_{0.3}$ | $\mathbf{61.4}_{3.6}$ | $\mathbf{66.3}_{0.7}$ | $\mathbf{66.9}_{0.7}$ | **5.3** | -0.1 |
| BITFIT | $48.4_{1.6}$ | $56.0_{6.1}$ | $51.7_{2.5}$ | $90.8_{0.8}$ | $52.0_{2.3}$ | $61.7_{1.4}$ | $60.1_{1.1}$ | - | - |
| +BROWN_PDF | $49.1_{1.8}$ | $56.0_{6.0}$ | $53.1_{2.4}$ | $\mathbf{91.1}_{0.2}$ | $54.0_{2.0}$ | $62.8_{1.0}$ | $61.0_{0.7}$ | 0.9 | 0.0 |
| +BROWN_SDE | $\mathbf{51.4}_{1.4}$ | $\mathbf{60.1}_{1.7}$ | $\mathbf{56.7}_{1.8}$ | $\mathbf{91.1}_{0.3}$ | $\mathbf{57.3}_{4.5}$ | $\mathbf{63.0}_{1.2}$ | $\mathbf{63.2}_{0.7}$ | **3.1** | -0.4 |
| ADAPTER | $47.4_{3.7}$ | $57.0_{7.2}$ | $55.8_{2.9}$ | $91.0_{0.4}$ | $55.8_{2.5}$ | $62.7_{2.0}$ | $61.6_{1.2}$ | - | - |
| +BROWN_PDF | $48.1_{4.0}$ | $58.3_{6.9}$ | $57.9_{3.5}$ | $91.4_{0.4}$ | $57.7_{3.6}$ | $63.0_{3.0}$ | $62.7_{1.0}$ | 1.1 | 0.0 |
| +BROWN_SDE | $\mathbf{50.3}_{2.2}$ | $\mathbf{61.0}_{6.1}$ | $\mathbf{63.4}_{4.4}$ | $\mathbf{91.5}_{0.3}$ | $\mathbf{58.4}_{2.2}$ | $\mathbf{64.3}_{1.9}$ | $\mathbf{64.8}_{1.1}$ | **3.2** | -0.6 |

Table 12: The results on GLUE for $\text{BERT}_{\text{large}}$ under 16-shot setting *with regularizers trained on 0.1% of the pre-training corpus*. $\Delta_{\text{whole}}$ is the difference between the the average performance in this table and the average performance in Tab. 2.

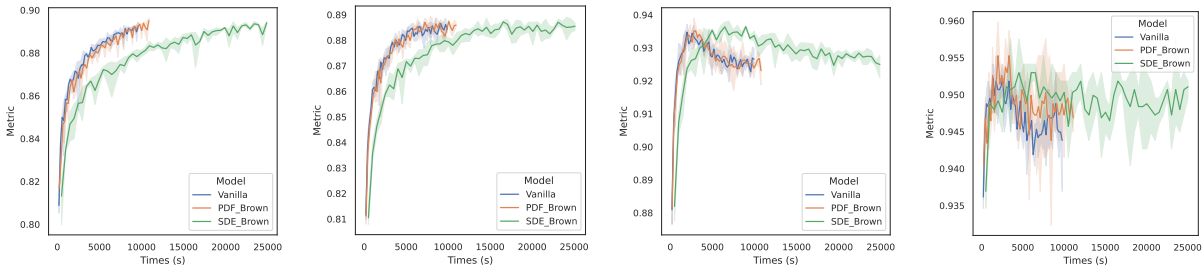



(a) QQP (b) MNLI (c) QNLI (d) SST-2

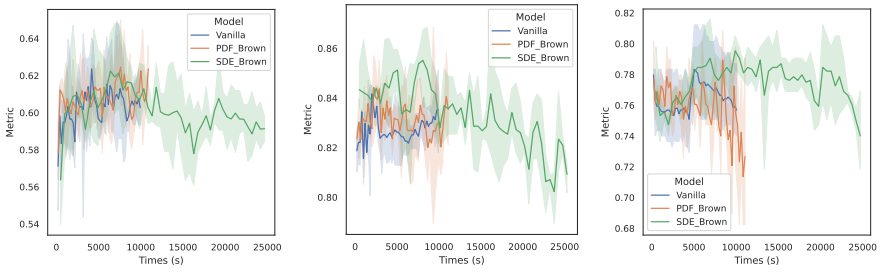



(e) CoLA (f) MRPC (g) RTE

Figure 5: Time-Metric curve for regularizers on *prompt tuning*.

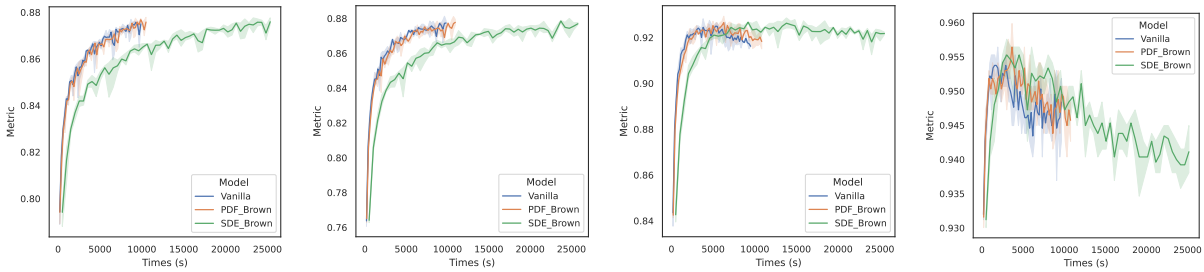



(a) QQP (b) MNLI (c) QNLI (d) SST-2

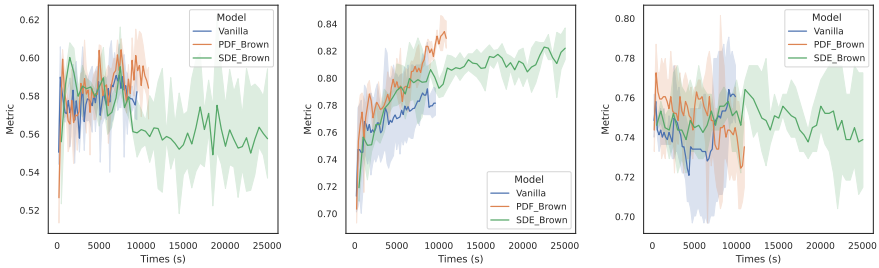



(e) CoLA (f) MRPC (g) RTE

Figure 6: Time-Metric curve for regularizers on *LoRA*.

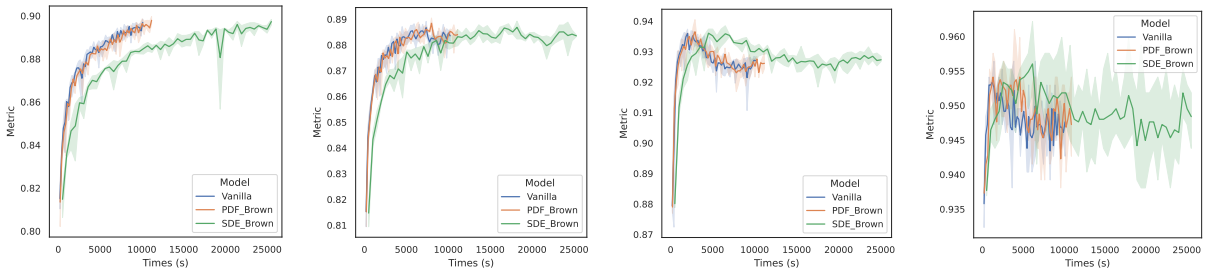



(a) QQP (b) MNLI (c) QNLI (d) SST-2

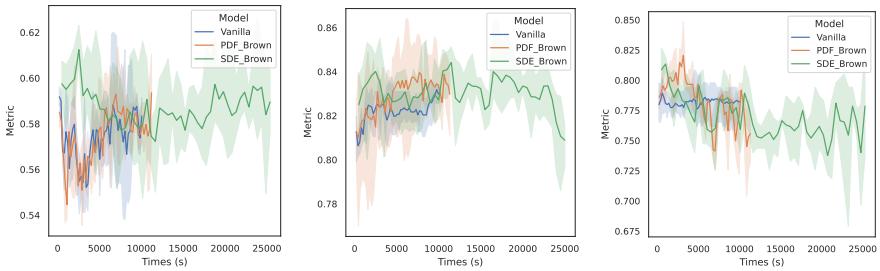



(e) CoLA (f) MRPC (g) RTE

Figure 7: Time-Metric curve for regularizers on *BitFit*.




(a) QQP (b) MNLI (c) QNLI (d) SST-2




(e) CoLA (f) MRPC (g) RTE

Figure 8: Time-Metric curve for regularizers on *Adapter*.

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*Section 8*

☑ A2. Did you discuss any potential risks of your work?
*Section 8*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*Left blank.*

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

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

*Left blank.*

☐ B1. Did you cite the creators of artifacts you used?
*No response.*

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*No response.*

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*No response.*

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*No response.*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*No response.*

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*No response.*

### C ☑ Did you run computational experiments?

*Section 4, 5*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Section 4*

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

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*Section 4 and Appendix*

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
*Section 4*

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
*Section 4*

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

*Left blank.*

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

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

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

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

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