Improving Pretrained Language Model Fine-Tuning With Noise Stability Regularization

Hang Hua, Xingjian Li, Dejing Dou, Senior Member, IEEE, Cheng-Zhong Xu, Fellow, IEEE, and Jiebo Luo, Fellow, IEEE

Abstract—The advent of large-scale pretrained language models (PLMs) has contributed greatly to the progress in natural language processing (NLP). Despite its recent success and wide adoption, fine-tuning a PLM often suffers from overfitting, which leads to poor generalizability due to the extremely high complexity of the model and the limited training samples from downstream tasks. To address this problem, we propose a novel and effective fine-tuning framework, named layerwise noise stability regularization (LNSR). Specifically, our method perturbs the input of neural networks with the standard Gaussian or in-manifold noise in the representation space and regularizes each layer's output of the language model. We provide theoretical and experimental analyses to prove the effectiveness of our method. The empirical results show that our proposed method outperforms several state-of-the-art algorithms, such as L2 norm and start point (L2-SP), Mixout, FreeLB, and smoothness inducing adversarial regularization and Bregman proximal point optimization (SMART). In addition to evaluating the proposed method on relatively simple text classification tasks, similar to the prior works, we further evaluate the effectiveness of our method on more challenging question-answering (QA) tasks. These tasks present a higher level of difficulty, and they provide a larger amount of training examples for tuning a well-generalized model. Furthermore, the empirical results indicate that our proposed method can improve the ability of language models to domain generalization.

Index Terms—Domain generalization, fine-tuning, in-domain generalization, pretrained language models (PLMs), regularization.

NOMENCLATURE

| Variables | Description |
|-----------|-------------|
| d         | Dimensionality of the input in the original representation space. |
| (x, y)    | Input point \( x \in \mathbb{R}^d \) and its corresponding label \( y \). |
| \( \varepsilon \) | Small random noise with the same dimension as the input \( x \). |

\( \tilde{x} \) | Perturbed input that \( \tilde{x} = x + \varepsilon \). |
\( \theta \) | Parameter of a model. |
\( L \) | Total number of layers in a bidirectional encoder representations from transformers (BERT) model. |
\( b \) | Index of a layer where the noise is injected on its input. |
\( r \) | Index of a layer where the noise stability is enforced, \( 1 \leq b \leq r \leq L \). |
\( k \) | Number of nearest neighbors of an input \( x \). |
\( N_k(x) \) | Set of the \( k \)-nearest neighbors (KNN) of \( x \). |

Functions and Operators:

\( F \) | BERT model parameterized with \( \theta \). |
\( f \) | Real-valued function in convenience of the theoretical analysis. |
\( \mathcal{L} \) | Loss function. |
\( \mathcal{R} \) | Regularization term. |
\( \| \cdot \|_2 \) | \( L^2 \) norm of a vector, i.e., if \( x \in \mathbb{R}^d \), \( \|x\| = \|x\|_2 = \sqrt{\sum_{i=1}^{d} x_i^2} \). |
\( \| \cdot \|_F \) | Frobenius Norm of a matrix, i.e., if \( A \in \mathbb{R}^{m \times n} \), \( \|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij}} \). |
\( \odot \) | Hadamard product of two matrices \( A \) and \( B \) (with the same dimension) as \( (A \odot B)_{ij} = A_{ij} B_{ij} \). |

Constants:

\( I \) | Identity matrix with ones on the main diagonal and zeros elsewhere. |
\( I \) | All-ones matrix where every element is equal to one. |

I. INTRODUCTION

Large-scale pretrained language models (PLMs) have significantly boosted state-of-the-art performance on natural language processing (NLP) tasks [1], [2], [3], [4], [5], [6]. In particular, some recently emerged powerful language models [5], [7], [8] with impressing performance on natural language understanding (NLU) tasks in popular NLP benchmarks, such as general language understanding evaluation (GLUE) [9], Super GLUE [10], language model analysis (LAMA) [11], [12], and variants of these models have been successfully applied in ever-wide scenarios [6], [13], [14], [15], [16].

Fine-tuning is the prevalent paradigm for utilizing large PLMs to perform downstream tasks. By initializing with the
resilient to a Gaussian noise injected on lower layers, named optimization term that forces higher layers of BERT to be fine-tuning in this work. Specifically, we impose an additional framework of noise stability regularization to improve PLMs’ input and higher layers can be made as follows. From the results shown in Fig. 1, two important observations investigate the sensitivity to the Gaussian noise on transformers. A preliminary experiment is conducted to investigate the behaviors of noise stability, as shown in [21], we find that it is not exactly the case for transformer-based language models, is characterized by noise sensitivity of the network with respect to input. While deep convolutional networks exhibit decent behaviors of noise stability, as shown in [21], we find that it is not exactly the case for transformer-based language models, e.g., BERT, which has more complex multihead self-attention mechanisms. A systematic solution to this challenge, especially in conditions where labeled examples are insufficient, is needed.

Improving generalization is always one of the fundamental goals of machine learning. Adding noise to the input [19], [20] has been proven to have an equivalent effect to training over clean input with an additional regularization term to constrain the solution space. Recent work [21], [22] discovers that the generalization capacity of deep neural networks (DNNs) is theoretically linked with the so-called interlayer cushion, characterized by noise sensitivity of the network with respect to input. While deep convolutional networks exhibit decent behaviors of noise stability, as shown in [21], we find that it is not exactly the case for transformer-based language models, e.g., BERT, which has more complex multihead self-attention architectures. A preliminary experiment is conducted to investigate the sensitivity to the Gaussian noise on transformers. From the results shown in Fig. 1, two important observations can be made as follows.

1) The propagating noise, if injected in lower layers, can be amplified in some higher layers of BERT.
2) Noise stability of a higher layer has a roughly positive correlation with the generalization performance.

Motivated by the above observations, we introduce a new framework of noise stability regularization to improve PLMs’ fine-tuning in this work. Specifically, we impose an additional optimization term that forces higher layers of BERT to be resilient to a Gaussian noise injected on lower layers, named layerwise noise stability regularization (LNSR) [23], [24]. The proposed regularization term has a good theoretical property of smoothing the learned function. We further design an advanced implementation of LNSR that generates a random noise with directions restricted by neighborhoods of the input point. The qualitative analysis demonstrates its equivalence to noise stability with respect to a Gaussian noise injected in the data manifold, thus referred to as in-manifold LNSR.

The main contributions of this article can be summarized as follows.

1) Our work is the first step in the investigation of noise stability properties on transformer-based architectures, which are of great interest in NLP and computer vision applications. We empirically extend observations about noise stability on fully connected networks and deep convolutional networks to transformers.

2) We propose two alternative implementations of noise stability regularization. Different from earlier works that directly use perturbed input examples to fit the labels, our method adopts a novel layerwise regularization that explicitly enforces noise stability of middle layers. Based on this idea, we further present a more effective in-manifold noise stability regularization. Specifically, the sampled noise is restricted in the region formed by interpolations between the input point and its nearest neighborhoods. Under commonly accepted assumptions, the simple method can be regarded as for sampling Gaussian noise in the low-dimensional data manifold.

3) We provide a detailed theoretical analysis of the noise stability regularization with respect to the Gaussian noise, revealing its connection with the Lipschitz continuity and the Tikhonov regularizer. The proposed noise stability regularization is also shown to have a form with better optimization properties than the conventional method that simply trains the model over the perturbed inputs. For the in-manifold noise stability, we provide a qualitative analysis of its relationship with manifold learning.

4) We conduct extensive experiments on several popular NLP tasks, covering different task types [text classification and question answering (QA)] and a wide range of pre-RoBERTa, the new task reuses most of the well-learned parameters, thus preserving the intrinsic generalizable knowledge while pursuing adaptation to the desired domain. However, despite the simplicity and ubiquity of fine-tuning in modern NLP, this process is brittle [17], i.e., a straightforward fine-tuning process sometimes leads to unstable solutions that generalize poorly to unseen data. Empirical studies have discovered that randomness brought by data order and weight initialization causes unexpected results [18]. Nevertheless, ad hoc strategies, such as seed selection and early stopping [18], provide a neither theoretical nor practical guarantee. A systematic solution to this challenge, especially in conditions where labeled examples are insufficient, is needed.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
dataset scales (from $\sim 10^2$ to $\sim 10^6$). We compare our approach with state-of-the-art methods aimed at improving fine-tuning PLMs, such as L2 norm and start point (L2-SP) [25], Mixout [26], and smoothness inducing adversarial regularization and Bregman proximal point optimization (SMART) [27]. Our approach not only consistently improves the overall performance but also obtains more stable fine-tuning results over multiple random trials. Moreover, our algorithm is also effective in dealing with the risk of domain shift, demonstrated by additional experiments on domain generalization benchmarks.

The remainder of this article’s structure includes preliminaries in Section II, our proposed framework in Section III, method evaluation in Section IV, performance analysis in Section V, and a conclusion in Section VI.

II. NOTATIONS AND PRELIMINARIES

This section covers commonly used notations and provides a brief overview of the preliminary knowledge relevant to our proposed algorithm and its theoretical analysis.

A. Notations

Throughout, we will frequently use the set of notations and terminology listed in Nomenclature.

III. METHODOLOGY

In this section, we systematically introduce our algorithm composed of the following parts. We first present the general framework of our proposed LNSR for BERT in Section III-A. Next, we describe two alternative methods of noise generalization in Section III-B, which are the standard LNSR that directly injects Gaussian noise and in-manifold LNSR that adds random noise constrained on the subspace formed by the input’s nearest neighbors. Then, we provide a theoretical analysis for the two specific choices of noise generation methods. In Section III-C, we prove that the standard LNSR has good properties related to the Lipschitz continuity and Tikhonov regularization. In Section III-D, we demonstrate that the in-manifold LNSR is equivalent to the standard LNSR imposed on the data manifold under certain assumptions.

A. General Framework

Given a pretrained model as initialization, fine-tuning BERT is a general task of supervised learning that aims at minimizing an expected error $\mathcal{L}$ with respect to the model’s parameter $\theta$ over the data distribution. Considering the high memorization capacity of DNNs, a regularization term $\mathcal{R}$ responsible to control the model’s complexity is often employed to improve the generalized performance of the model. Therefore, a general form of the optimization objective can be represented as follows:

$$
\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)} \left[ \mathcal{L}(f(x; \theta), y) \right] + \mathcal{R}(\theta)
$$

where we omit the notation of the data distribution without ambiguity. A most common choice for the regularization $\mathcal{R}$ is the $L^2$ normalization of the parameter $\theta$, that is, $\|\theta\|^2$, also called the weight decay, particularly for DNNs. Despite its ubiquity, there is no theoretical evidence for the effectiveness of such a simple data-independent regularization in DNNs [36], [37].

We are interested in the behavior of noise stability, which serves as a data-dependent regularizer. Specifically, given an input point $x$, we generate a perturbed input $\tilde{x}$ by adding a random noise $\varepsilon$ with a small magnitude to $x$. Noise stability characterizes to what degree the output with respect to $\tilde{x}$ deviates from that with respect to the clean input $x$. For the multilayer transformer architecture, we adopt a layerwise regularization to enforce noise stability at each layer. Formally, we define the input of layer $b$ as $x^b$ and the network between the $bth$ and $rth$ layers as $f^{b,r}$, which is parameterized by $\theta^{b,r}$. Note that when $b = r$, $f^{b,r}$ represents a single layer. If we inject the noise at a fixed layer $b$ and regularize each higher layer $r \geq b$, the noise stability term can be represented as follows:

$$
\mathcal{R}(\theta) = \mathbb{E}_{x, \varepsilon} \sum_{r=b}^{L} \lambda^{b,r} \left\| f^{b,r}(x^b + \varepsilon; \theta^{b,r}) - f^{b,r}(x^b; \theta^{b,r}) \right\|^2
$$

(2)

where $\lambda^{b,r}$ is the coefficient to control the weight of regularizing $f^{b,r}$. Given (2) as the general form of noise stability used in multilayer transformers, a subsequent question is how to generate the noise $\varepsilon$ for a specific input $x^b$.

B. Methods for Noise Generation

In this work, we introduce two progressive methods for noise generation. The first sample noise from the multivariate Gaussian distribution for the noise stability regularization is called standard LNSR. We also propose an improved method called in-manifold LNSR, which samples random noise on the subspace formed by the input’s KNNs. The following subsections provide details on the two methods.

1) Standard LNSR: For the standard LNSR, $\varepsilon$ is randomly sampled from the standard multivariate Gaussian distribution as $\varepsilon \sim \mathcal{N}(0, \sigma^2I)$. That is, each element $\varepsilon_i$ is independently sampled from the zero-mean Gaussian distribution as $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$.

2) In-Manifold LNSR: In this section, we introduce the implementation of in-manifold LNSR in detail with intuitive explanations. More formal discussions are deferred to Section III-D. A widely accepted assumption for a smooth manifold is that, an input point $x$ and its $k$ neighborhoods $N_k(x)$ form a subspace that is approximated linearly. With $x^{(j)} \in N_k(x)$, $j = 1, 2, \ldots, k$ denoting these neighborhoods, we get the set of neighbored differences as $d^{(j)} = x^{(j)} - x$. Under the manifold assumption, vectors in $d^{(j)}$ lie on the same linear subspace around $x$ as the origin. Imagining concatenating all these row vectors $[d^{(j)}]$ to form a matrix, we perform the Gram–Schmidt orthogonalization on the matrix and obtain the transformed set $\tilde{d}^{(j)}$, which are all orthogonal to each other. It is obvious

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
that \(\{\tilde{d}^{(j)}\}\) lies on the same subspace as \(\{d^{(j)}\}\). To generate in-
manifold random noise, we independently sample \(k\) random variables from the univariate Gaussian distribution as \(e^{(j)} \sim N(0, \sigma^2)\). Then, we generate the in-manifold noise \(e\) by a random interpolation in the region formed by \(\{d^{(j)}\}\) as \(e = \sum_{j=1}^{k} e^{(j)} \tilde{d}^{(j)}\). Fig. 2 demonstrates the difference between sampling standard Gaussian noise and sampling in-manifold noise.

C. Analysis of the Standard LNSR

Here, we present mathematical analyses that highlight the properties of our proposed approach. Our primary goal is to examine the relationships between LNSR and classical techniques used for quantifying and controlling model complexity. The reduction of model complexity is widely acknowledged as a crucial principle in mitigating overfitting in machine learning. However, due to the intricate nature of DNNs, which encompass a massive number of parameters, traditional metrics for quantification often become computationally challenging. Based on the analyses in this part, LNSR is shown to have implicit connections with Lipschitz continuity and Tikhonov regularization.

Without loss of generality, the analysis is based on a general real-valued function \(f : \mathbb{R}^d \to \mathbb{R}\). For simplicity, we ignore the notation of the parameter \(\theta\) in the following analysis. \(e\) is the standard Gaussian noise as defined in Section III-B1. Given that \(e\) has a small magnitude (i.e., its variance \(\sigma^2\)), we adopt the second-order Taylor approximation to present \(f(x + e)\) as follows:

\[
f(x + e) = f(x) + J(x)e + \frac{1}{2} e^T H(x)e + O(e^3)
\]

where \(J(x)\) and \(H(x)\) are the Jacobian and Hessian of \(f\) with respect to \(x\), respectively.

1) Connection With Lipschitz Continuity: By ignoring the second-order and higher order terms in the Taylor expansion, it is easy to derive that the noise stability regularization equals the \(L^2\) norm of the Jacobian \(J_f\)

\[
\mathbb{E}_{x,e} \|f(x + e) - f(x)\|^2 = \mathbb{E}_{x,e} \|J(x)e\|^2 = \sigma^2 \mathbb{E}_{x,e} \|J(x)e\|^2 \|e\|^2.
\]

As indicated, the Lipschitz constant of \(f\) is bounded by the supremum of spectral norm of the Jacobian \(J_f\). Therefore, injecting input noise and explicitly minimizing the output discrepancy (between the clean and perturbed output) have the same form as inside the operation of calculating the supremum.

Note that the objective of minimizing (4) (with respect to a random noise) is likely to lower the bound but is, of course, not guaranteed to do so. In fact, \(e\) with a direction that maximizes \(\|J(x)e\|_2\), or \(\|f(x + e) - f(x)\|_2\), is so-called an adversarial perturbation, and such a perturbed input \(\tilde{x} = x + e\) is called an adversarial example [38]. Though the adversarial example induces a more accurate indicator of the Lipschitz constant, empirical studies show that training with such extreme inputs, aiming at promoting the robustness of adversarial examples, significantly harms the performance on clean inputs [39]. An intuitive explanation is that the adversarial example focuses on a rare perturbation direction (barely seen in real data) that is the most difficult for the model to be robust. In contrast, our noise stability regularization cares about noise with a uniform direction, which is more diverse and more possible to exist in real data, though not providing the tight Lipschitz continuity bound.

2) Connection With Tikhonov Regularization: Here, we provide a further analysis considering the first- and second-order terms in (3). Let \(\text{Tr}(\cdot)\) be the trace of a matrix. We first present our main claim as follows.

Claim 1: If a real-valued function \(f : \mathbb{R}^d \to \mathbb{R}\) is twice differentiable with respect to its input \(x \in \mathbb{R}^d\), \(e \in \mathbb{R}^d\) conforms to the standard multivariate Gaussian distribution \(e \sim N(0, \sigma^2 I)\). Then, omitting terms of higher order than the second degree, the noise stability regularization \(R\) for a given \(x\) can be represented as follows:

\[
R = \mathbb{E}_{x,e} \|f(x + e) - f(x)\|^2
\]

\[
\approx \frac{\sigma^2}{4} \mathbb{E}_x \{4\|J(x)\|^2 + \|\text{Tr}(H(x))\|^2 + \|(1 - I) \circ H(x)\|_F^2\}.
\]

Analysis: Equation (5) characterizes our connection with the Tikhonov regularizer, where the proposed noise stability regularization has the effect of constraining the first- and second-order input derivatives of the objective function \(f\).

In an analogy with previous works, our method involves common terms of the \(L^2\) norm of Jacobian, the \(L^2\) norm of Hessian trace, and the Frobenius norm of Hessian. Compared with [19] and [20], our method is capable of inheriting their merits and overcoming their flaws. Specifically, the proposed LNSR regularizes the positive guaranteed Hessian trace that avoids undesirable solutions as [19]. Compared with [20], our method is more efficient, as adding noise to the input does not introduce much computation for DNNs. However, Rifai et al. [20] involve the gradient of Jacobian during the backpropagation process, which is considerably more complex for DNNs.

Furthermore, it is important to note that the conclusions drawn in [19] and [20] primarily rely on the assumption of
Algorithm 1 Fine-Tuning With LNSR Regularization

**Require:** Training set $D$, learning rate $\tau$, number of training iterations $N$, number of layers $L$, neural network $f$ and its corresponding parameter $\theta$, the layer index $b$ where noise is injected, and regularization weights $[\lambda^b]_{b \in L}$.

1. Initialize $\theta$ with $\theta_0$ learned from the pre-trained task
2. for iteration=1, 2, ..., $N$ do
3. sample a batch of data $B \sim D$
4. $\mathcal{R} \leftarrow 0$
5. for each $x \in B$ do
6. if Standard LNSR then
7. $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$
8. else if In-manifold LNSR then
9. get $k$-nearest neighborhoods $N_k(x) = \{x^{(j)}\}$
10. get differences $[d^{(j)}][d^{(j)} = x^{(j)} - x]$  
11. orthogonalize $[d^{(j)}]$ and get $[d^{(j)}]$ 
12. perform $k$ iid sampling of $[x^{(j)}][x^{(j)} \sim \mathcal{N}(0, \sigma^2)]$
13. $\epsilon = \sum_{j=1}^{k} e^{(j)}d^{(j)}$
14. end if
15. $\tilde{x} \leftarrow x + \epsilon$
16. feed $x$ and $\tilde{x}$ into the network $f$
17. for $r = b, b + 1, \ldots, L$ do
18. $\mathcal{R} \leftarrow \mathcal{R} + \lambda^b \cdot ||f^b(\tilde{x}) - f^b(x)||^2$
19. end for
20. end for
21. $g \leftarrow \frac{1}{|B|} \sum_{x \in B} \nabla_{\theta} \mathcal{L}(f(x; \theta), y) + \mathcal{R}$
22. $\theta \leftarrow \theta - \tau g$
23. end for

Ensure: $\theta$

---

a regression task with the mean-squared error (MSE) loss. These approaches utilize labels to impose regularization by ensuring the perturbed input aligns with its corresponding label or minimizing the perturbed Jacobian, which also depends on labels. Although our LNSR method shares a similar concept of directly adding noise to the input, it introduces a regularizer in an unsupervised manner, leveraging the clean output as virtual supervision. As a result, our framework offers significantly greater flexibility by reducing the reliance on labels and specific forms of the loss function.

### D. Analysis of In-Manifold LNSR

In this part, we shall analyze the effect of in-manifold LNSR. The main conclusion is summarized as follows.

**Claim 2:** Suppose that the input data lie in a $k$-dimensional smooth manifold, with each data point $x \in \mathbb{R}^d$ and its neighbors lying on a locally linear patch. Provided there are sufficient neighbors with uniformly distributed directions around $x$, noise $\epsilon$ sampled according to Algorithm 1 is approximate standard multivariate Gaussian in the manifold space around $x$.

Next, we provide a detailed analysis to verify the claim. First, in Section III-D1, we explain why the generated noise lies in the manifold, adopting the manifold assumption of locally linear embedding (LLE) [31]. Then, in Section III-D2, we prove that the generated noise follows the standard multivariate Gaussian distribution in the manifold space. Finally, in Section III-D3, we discuss the assumptions and approximations involved in the claim.

1) Noise on the Locally Linear Patch: We first give a brief overview of LLE [31] for manifold learning. Note that our purpose is not designing or applying a manifold learning algorithm. Instead, we intend to borrow the assumptions and understandings about the manifold to generate in-manifold noise.

**LLE:** The intuition behind LLE [31] is to regard a smooth manifold as a collection of overlapping linear patches, provided these patches are small enough. Then, the local geometry can be characterized by a weight matrix $W$, which is to be solved by minimizing the reconstruction error

$$R(W) = \sum_{i} ||x^{(i)} - \sum_{j} W_{ij}x^{(j)}||^2.$$  \hspace{1cm} (6)

We ignore the constraints used for computing $W$, as it is not directly related to our work. Ideally, there exists an appropriate $W$ that makes $R(W)$ near zero. In such cases, each $x^{(i)}$ can be approximately represented as a linear combination of its neighbors, i.e., they lie on a linear subspace.

**In-Manifold Noise:** Here, we show that the noise with respect to a data point $x$ generated according to Algorithm 1 lies on the linear patch expanded by $x$ and its neighbors. It is obvious that the difference vectors $[d^{(j)}][d^{(j)} = x^{(j)} - x]$ lie on the same linear patch with $x$ and its neighbors. Linear transformations of these $[d^{(j)}]$, e.g., the orthogonal variants $[d^{(j)}]$, should be still in the same subspace and so are the linear combination of these $[d^{(j)}]$. Therefore, the perturbed input $x + \epsilon$ with the noise $\epsilon = \sum_{j=1}^{k} e^{(j)}d^{(j)}$ lies on the linear space formed by $x$ and its neighbors. Note that, this does not mean a guarantee that $x + \epsilon$ lies within the linear patch unless both the noise magnitude and direction are properly constrained. However, by suggesting that there exist sufficient neighbors, which have diverse directions, $x + \epsilon$ should be on the linear patch with a high probability. See Fig. 3 for an intuitive illustration of the locally linear patch.

2) In-Manifold Standard Multivariate Gaussian Noise: Recall that the noise is generated based on orthogonal difference vectors as $\epsilon = \sum_{j=1}^{k} e^{(j)}d^{(j)}$. When being projected to the manifold space by $M : \mathbb{R}^d \rightarrow \mathbb{R}^k$, their angles will also be preserved according to the locally linear patch assumption [31]. So, the projected difference vectors $M(d^{(j)}) \in \mathbb{R}^k$ are still orthogonal.
Imagine that we change the coordinate system in order to let \( M(d^{(j)}) \) be the one-hot vector where only the \( j \)th element is 1 and all remaining are 0. As a result, the \( j \)th element of \( M(e) = \sum_{j=1}^{k} \epsilon^{(j)}M(d^{(j)}) \) is just \( \epsilon^{(j)} \), which follows the standard Gaussian distribution. Thus, the projected noise \( M(e) \) conforms the standard multivariate Gaussian distribution.

E. Choices of Hyperparameters

Our method involves additional hyperparameters. Here, we describe how we choose a reasonable hyperparameter configuration to ensure the performance of our method.

1) Noise Magnitude: We adopt an adaptive scheme to determine the magnitude of injected noise. Specifically, a noise is first sampled from a standard multivariate Gaussian distribution and then rescaled by a scalar coefficient \( \eta \). \( \eta \) is set to be a fixed proportion between the magnitude of the original feature vector \( x \) and that of the noise vector \( \epsilon \) as \( \eta = 0.05 \|x\|_2^2/\|\epsilon\|_2^2 \).

2) Noise Injected Position: In this instance, the noise is always injected at the lowest layer, i.e., the word embedding layer, for the regularization of noise stability. In this way, we achieve the effect of stabilizing all transformer layers. Moreover, for in-manifold LNSR, obtaining hidden-layer representations of KNNs is much more laborious, since it needs additional feedforward computations.

3) Number of Nearest Neighbors for In-Manifold LNSR: Under the manifold assumption, the number of nearest neighbors \( b \) depends on the underlying manifold dimension. However, estimating such a dimension is often intractable for real-world data [32]. In this work, we empirically find that \( b = 10 \) usually performs well for in-manifold LNSR. While further increasing \( b \) tends to violate the manifold assumption, a too-small \( b \) induces an overconstrained noise space and would reduce the effectiveness of the regularization.

F. Computational Complexity Analysis

1) Standard LNSR: Although the regularization term of noise stability is calculated for every layer, noise generation is performed only once at the noise input layer, for which we choose the first intermediate layer. Denoting the length of input tokens as \( M \), and the dimensionality of the embeddings as \( d \), generating standard Gaussian noise in a training iteration requires the complexity of \( O(Md) \). Note that we do not employ expensive high-dimensional multivariate Gaussian sampling.

2) In-Manifold LNSR: This advanced method involves additional calculations on searching for \( k \)-nearest neighborhoods and forming a manifold space for each input token. Through the utilization of widely adopted space partitioning algorithms, the computational complexity of the first component can be expressed as \( O(Mkd + \log(N)) \), where \( N \) is the size of the vocabulary used in KNN searching. As for the latter component, only linear operations are needed with the complexity of \( O(Mkd) \).

3) Adversarial Perturbations: Since adversarial-based methods aim to calculate the worst-case perturbation for a given input instance, it typically requires several training iterations over the entire network to guarantee an optimal solution regarding the adversarial objective. Denoting the number of layers as \( L \), and the number of iterations by \( T \), the complexity of generating adversarial perturbations will be dominated by \( O(TLM^2d) \). Note that this term only accounts for the forward computation while disregarding the gradient calculation on the input.

As all the complexity approximations involve the same \( d \), we ignore it for easier comparison. Given the approximate values as \( M = 10, k = 10, N = 10^5, T = 10, \) and \( L = 10 \), the complexity approximations for standard LNSR, in-manifold LNSR, and adversarial perturbations are \( O(10,d) \), \( O(10^2,d) \), and \( O(10^3d) \), respectively. We can observe that the standard LNSR is much more efficient than the other two. Moreover, the LNSR approaches exhibit better scalability regarding the input sequence length.

IV. EXPERIMENTS

A. Datasets

To verify the effectiveness of our method for improving the generalizability of language models, we conduct experiments on text classification and QA tasks, respectively.

1) Text Classification Tasks: We adopt four few-sample (less than 10k training samples) text classification tasks of GLUE,\(^2\) and we present a brief description below and refer readers to Table I for more details.

- Corpus of Linguistic Acceptability (CoLA) [40]: It is an English acceptability judgments dataset consisting of 10 657 sentences from 23 linguistics publications. Each sentence is annotated with a binary label, indicating whether this sentence is grammatical in English. The task uses Matthews correlation coefficient (MCC) [41] as the evaluation metric.
- Microsoft Research Paraphrase Corpus (MRPC) [42]: It is a corpus for the paraphrase detection task. Each example is a sentence pair, whose label is 1 if the two sentences in this pair are equivalent in semantics. We evaluate the performance with the commonly adopted accuracy and average F1 score.
- Recognizing Textual Entailment (RTE) [9], [43], [44], [45]: It is a corpus for the textual entailment task. Each example is a sentence pair whose label is whether the first entails the second. The evaluation metric is accuracy.
- Semantic Textual Similarity Benchmark (STS-B) [46]: It is a task for determining the semantic similarity of a sentence pair. The similarity is represented by integral numbers \( \{1,2,3,4,5\} \), which the model is learned to predict. Common metrics for evaluation are the Pearson and Spearman correlation coefficients, and we report the average of them.

\(^2\)https://gluebenchmark.com/
2) QA Tasks: We also evaluate our method using stanford question answering dataset (SQuAD) [47] for in-domain QA tasks and machine reading for question answering (MRQA) 2019 [48] for out-of-domain QA tasks to test the domain generalizability of our approach to more complex NLP challenges. SQuAD is an machine reading comprehension (MRC) dataset with QA pairs from Wikipedia articles, where answers are text segments from the articles or may be unanswerable. We experiment with SQuAD v1.1.

MRQA is a task for testing how well extractive QA models generalize to out-of-domain data. The MRQA 2019 unified various QA dataset formats to test in-domain training against out-of-domain generalization. More details about the different QA datasets are summarized in Appendix A.

B. Baseline Models

1) Fine-Tuning: Fine-tuning refers to the vanilla language model fine-tuning method. We adopt the standard fine-tuning strategy of BERT and robustly optimized BERT pretraining approach (RoBERTa) described in [17] and [49].

L²-SP [25] is a regularization scheme that is used for constraining the extent of parameters to update while fine-tuning a pretrained model. The goal of introducing this regularization item is to preserve the general knowledge contained in the pretrained model. The form of the regularizer is \( \Omega(w) = (\alpha/2)||w_i - w_{0i}|| + (\beta/2)||w_i|| \), where the values of \( w_i \) are parameters shared by the pretrained and fine-tuned models and the values of \( w_i \) are those specific to the target task.

Mixout [26] is a regularization method motivated by Dropout [50] and DropConnect [51]. At each training iteration, instead of replacing parameters with 0, Mixout replaces parameters with their pretrained value with a probability \( p \).

SMART [27] is a noise-based regularization technique that uses adversarial training for model smoothness and a Bregman proximal point method to temper updates during fine-tuning.

FreeLB [52] is a simple adversarial noise regularization method that is applied on input embeddings.

Componentwise Gradient Norm Clipping (CWGNC) [53]: It clips the gradient norms of the key–query–value parameters in transformers that individually help balance gradient distribution and convergence rates, enhancing language model fine-tuning.

C. Experimental Setup and Implementation Details

Our model is implemented using PyTorch based on transformers framework\(^3\) [54], and the backbone language models are BERT [17] and RoBERTa [49]. We adopt settings of learning strategies and hyperparameters recommended by Devlin et al. [17]. We use the HuggingFace edition AdamW [55] optimizer with a learning rate \( \in (2 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}) \) and a batch size \( \in \{16, 32, 64\} \), and \( \beta_1 = 0.9 \) and \( \beta_2 = 0.999 \). We adopt the regularization weight in the range of \( \{1.0, 0.8, 0.6, 0.4, 0.2\} \) for different tasks. The warm-up ratio for the classification task and QA task is set to 6\% and 10\%, respectively. For a fair comparison, we set the maximum numbers of epochs to 3 and 2 for classification and QA tasks, respectively, which is the same as baseline modes’

\(^3\)https://huggingface.co/transformers/index.html

\(^4\)Due to the high training cost, we adopt five random seeds for comparison.

The performance gains of our methods are significant, we calculate the \( p \) values between the performance distributions of the fine-tuning baseline and our proposed LNSR methods. We get very small \( p \) values on all tasks that, 9.7 \times 10^{-7} on RTE, 2.3 \times 10^{-4} on MRPC, 4.7 \times 10^{-8} on CoLA, and 3.3 \times 10^{-8} on STS-2.

Standard deviation can be used to reflect the stability of a learning procedure. In this study, a high standard deviation indicates increased model sensitivity to random seeds. Experimental results show that models utilizing LNSR exhibit lower standard deviations across tasks, indicating reduced sensitivity to randomness from data ordering and initialization. In addition, Fig. 4 provides a clearer illustration. Although the in-manifold LNSR methods have a higher standard deviation compared with LNSR, it gains more improvement on mean and max values. In summary, the two proposed LNSR methods can not only improve the average performance but also reduce the instability of BERT fine-tuning.

For the QA task, Table III shows the results of all the methods on the SQuAD dataset. We can see that on the more challenging QA task with a larger dataset, our proposed LNSR and in-manifold LNSR methods can still improve the models’ fine-tuning performance compared with other methods. Specifically, the BERT large model fine-tuning with in-manifold LNSR achieves an average and max dev-set exact match (EM)/F1 of 86.88/93.07 and 87.48/93.40. The RoBERTa large model fine-tuning with in-manifold LNSR achieves an average and max dev-set EM/F1 of 88.71/94.43 and 88.93/93.40. Fig. 5 shows the mean and the range of EM/F1 scores’ changes during the fine-tuning process.\(^4\) The \( p \) values between the performance distributions of the fine-tuning baseline and our proposed LNSR are 3.2 \times 10^{-5}/1.9 \times 10^{-5}, and for the in-manifold LNSR, the \( p \) values are 4.1 \times 10^{-8}/2.7 \times 10^{-8}.

V. A. Resilience to Domain Shift

We investigate the domain generalization performance of different methods on the MRQA 2019 benchmark. We first
fine-tune a language model on SQuAD and then evaluate the fine-tuned model on the out-of-domain datasets.

As shown in Table IV, language models fine-tuned with our LNSR methods outperform the vanilla fine-tuning method overall on the mean and max $F_1$ scores on each of the MRQA out-of-domain datasets. In addition, compared with SMART, our methods obtain a better $F_1$ score on most datasets, which shows the better generalizability of our method. In-manifold LNSR performs better than the vanilla LNSR on most datasets. The overall results demonstrate our LNSR methods are more robust to domain shift problems, and models fine-tuned with our methods show a better ability for zero-shot domain transfer. For this observation, we argue that LNSR can promote the generalization capacity of fine-tuned language models, so that the models show higher resilience to domain shift.

B. Ablation Study

We conduct ablation experiments on text classification tasks to further validate the mechanism of both the standard and in-manifold LNSR methods. We compare our method with fine-tuning more epochs of the BERT model and injecting noise without an explicit regularization (we add Gaussian/in-
manifold noise to the intermediate representation of a BERT layer in the forward propagation process without explicitly regularizing the noise stability), respectively. The results are in Table V. We observe that the performance gains brought by more training epochs are less significant. Meanwhile, fine-tuning by only imposing perturbations cannot bring satisfying improvement in performance too. However, fine-tuning with our proposed LNSR methods achieves satisfying results on every task, which adds to the mounting evidence that injecting noise can effectively improve the generalizability of language models.

C. Sensitivity to the Position of Noise Injection in LNSR

As illustrated in Fig. 1, the performance of BERT fine-tuning is sensitive to the layer of noise injection. So, we investigate the impact of different positions of noise injection on BERT fine-tuning. For comparison, we inject noise into different layers of the BERT model and regularize the noise stability item. According to the experimental results shown in Fig. 7, we can conclude that all injection positions bring significant improvements over vanilla fine-tuning. In particular, noise injected into the lower layers usually brings more performance gains of the BERT fine-tuning, which indicates that lower layer LNSR may be more effective, as it influences more layers (i.e., parameters).

D. Sensitivity to the Mix Ratio of In-Manifold LNSR

To investigate the influence of the mix ratio of in-manifold LNSR on language model fine-tuning, we conduct experiments on RoBERTa with different scales of in-manifold noise on different text classification tasks. We adopt noise scale with \( \{0.10, 0.12, 0.15, 0.20\} \) to avoid catastrophic changes in the embedding layer. As we can see from Table VI, the performance of in-manifold LNSR fine-tuning is affected by both the scale of datasets and the scale of noise mix ratio.

For the RTE dataset with 2.5k training samples, we obtain the best mean/max value of 82.10/84.11 under the mix ratio of 0.1, while larger mix ratios can lead to an unstable and even collapsed fine-tuning process. But, on larger datasets, such as MRPC, CoLA, and STS-B (which have 3.7k, 8.5k, and 7k training examples, respectively), larger mix ratios can bring more absolute performance gains.

In conclusion, the optimal noise magnitude correlates with the size of the training data; larger datasets warrant increased noise magnitudes for improved performance, whereas for smaller datasets, reduced noise magnitudes are recommended to avert potential collapse of the representation.

E. Relationship to Existing Relevant Work

Our algorithm is relevant to the noise-based methods, including SMART [27], FreeLB [52], and R3F [57], most of which focus on the robustness of few-sample fine-tuning through a fashion of adversarial training. Specifically, SMART uses the gradient ascent method to learn a noise constrained within an \( \epsilon \) ball and then minimize the distributional difference between the original and perturbed representations. FreeLB proposes to directly minimize the adversarial loss \( \mathcal{L}_{\text{FreeLB}}(\theta) = \sup_{\Delta \theta : |\Delta \theta| \leq \epsilon} \mathcal{L}(\theta + \Delta \theta) \), implemented by iterative gradient updates. R3F improves the efficiency by removing the procedure of adversarial optimizing in SMART and proposes to directly improve the smoothness.

Compared with this type of adversarial-training-based algorithm, our LNSR not only simplifies the process of noise perturbation and reduces the computing complexity as analyzed in Section III-F, but also enjoys additional properties, which are essential to model generalization.

First, while adversarial-example-based methods are motivated by worst-case robustness with respect to small perturbations on input data, our approach is more directly associated with statistical learning principles, i.e., the generalization bound. Specifically, Arora et al. [21] figured out noise stability as a computation-tractable metric to bound the generalization error. Moreover, our approach has implicit equivalence to the Tikhonov regularizer, which is widely applied to shallow models for controlling model complexity.

Second, our method is expected to better simulate real-world data noise compared with adversarial noise. Previous research [58] has confirmed these two kinds of robustness, i.e., to natural noise and adversarial noise, are fundamentally conflicting. Given that our work aims at natural scenarios with random data noise rather than artificial adversarial noise, the proposed LNSR is a more reasonable solution.
F. Comparison of Gaussian Noise and In-Manifold Noise

To examine in-manifold noise’s effect, we analyzed noise vectors through PCA, comparing standard Gaussian noise with in-manifold noise. As shown in Fig. 2, informally, the sampled in-manifold noise has very limited freedom of direction, if data points actually lie on a low-dimensional manifold. Therefore, such a batch of noise vectors could be characterized by only a few major directions. The PCA eigenvalue distribution in Fig. 6 shows a remarkable difference between the standard and in-manifold Gaussian noise. For the in-manifold noise, almost all information is compressed on the top of a few eigenvectors, indicating its low actual dimensionality. In contrast, the standard Gaussian noise has a relatively smooth distribution of eigenvalues. Our result is consistent with recent studies about the embedding space of BERT, e.g., Cai et al. [59] point out that the word embedding in BERT usually has a local intrinsic dimension of less than 10.

VI. CONCLUSION

In this article, we investigate the problem of fine-tuning PLMs from the perspective of noise stability. We introduce a lightweight and effective framework, named LNSR, to improve generalizability and stability when fine-tuning PLMs on a few training samples. In the LNSR framework, two alternative noise sampling strategies are employed, which are the standard Gaussian noise and in-manifold noise. Our proposed LNSR methods are general techniques that promote the smoothness of language models and, thus, improve the model’s performance. Furthermore, we theoretically analyze the properties of our proposed model connected to the Lipschitz continuity and Tikhonov regularizer. In addition, the experimental results in this article also reflect the effectiveness of our proposed method to improve the generalizability and stability of PLMs.
generalization gaps. Fine-tuning with LNSR and in-manifold LNSR is shown to reduce the generalization gap and enhance performance.

ACKNOWLEDGMENT

The authors would like to thank the Jeffries Data Science Fellowship for supporting Hang Hua’s research.

REFERENCES

[1] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang, “REALM: Retrieval-augmented language model pre-training,” 2020, arXiv:2002.08909.
[2] Y. Li, “Fine-tune BERT for extractive summarization,” 2019, arXiv:1903.10318.
[3] D. Wadden, U. Wennberg, Y. Luan, and H. Hajishirzi, “Entity, relation, and event extraction with contextualized span representations,” in Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 5784–5789.
[4] J. Zhu et al., “Incorporating BERT into neural machine translation,” 2020, arXiv:2002.06823.
[5] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, “ELECTRA: Pre-training text encoders as discriminators rather than generators,” 2020, arXiv:2003.10555.
[6] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, “SpanBERT: Improving pre-training by representing and predicting spans,” Trans. Assoc. Comput. Linguistics, vol. 8, pp. 64–77, Dec. 2020.
[7] C. Raffel et al., “Exploring the limits of transfer learning with a unified text-to-text transformer,” J. Mach. Learn. Res., vol. 21, no. 140, pp. 1–67, 2020.
[8] M. Lewis et al., “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” 2019, arXiv:1910.13461.
[9] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “GLUE: A multi-task benchmark and analysis platform for natural language understanding,” 2018, arXiv:1804.07461.
[10] A. Wang et al., “SuperGLUE: A stickier benchmark for general-purpose language understanding systems,” 2019, arXiv:1905.00537.
[11] F. Petroni et al., “Language models as knowledge bases?” 2019, arXiv:1909.01066.
[12] F. Petroni et al., “How context affects language models’ factual predictions,” 2020, arXiv:2005.04611.
[13] G. Lample and A. Conneau, “Cross-lingual language model pretraining,” 2019, arXiv:1901.07291.
[14] Y. Hu, H. Hua, Z. Yang, W. Shi, N. A. Smith, and J. Luo, “PromptCap: Prompt-guided task-aware image captioning,” 2022, arXiv:2211.09699.
[15] J. Lin et al., “VideoXum: Cross-modal visual and textural summarization of videos,” 2023, arXiv:2303.12060.
[16] J. Zhang, H. Zhang, C. Xia, and L. Sun, “GRAPH-BERT: Only attention is needed for learning graph representations,” 2020, arXiv:2001.05140.
[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in Proc. NAACL-HLT, 2019, pp. 4171–4186.
[18] J. Dodge, G. Ilharco, R. Schwartz, A. Farhadi, H. Hajishirzi, and N. Smith, “Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping,” 2020, arXiv:2002.06305.
[19] C. M. Bishop, “Training with noise is equivalent to Tikhonov regularization,” Neural Comput., vol. 7, no. 1, pp. 108–116, Jan. 1995.
[20] S. Rifai, X. Glorot, Y. Bengio, and P. Vincent, “Adding noise to the input of a model trained with a regularized objective,” 2011, arXiv:1104.3250.
[21] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang, “Stronger generalization bounds for deep nets via a compression approach,” in Proc. 35th Int. Conf. Mach. Learn., 2018, pp. 254–263.
[22] X. Dong, A. T. Liu, M. Lin, S. Yan, and H. Zhang, “How should pretrained language models be fine-tuned towards adversarial robustness?” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, 2021, pp. 4356–4369.
[23] H. Hua, X. Li, D. Dou, C.-Z. Xu, and J. Luo, “Noise stability regularization for improving BERT fine-tuning,” 2021, arXiv:2107.04835.
[24] X. Li, H. Hang, C. Xu, and D. Dou, “Method and apparatus for transfer learning,” U.S. Patent 17820321, Dec. 15, 2022.
[26] C. Lee, K. Cho, and W. Kang, “Mixout: Effective regularization to fine-tune large-scale pretrained language models,” 2019, arXiv:1909.11299.

[27] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and T. Zhao, “SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization,” in Proc. 58th Annu. Meeting Assoc. Comput. Linguistics, 2020, pp. 2177–2190.

[28] H. Federer, Geometric measure theory. Berlin, Germany: Springer, 1996.

[29] J. Sietsema and R. J. F. Dow, “Creating artificial neural networks that generalize,” Neural Netw., vol. 4, no. 1, pp. 67–79, Jan. 1991.

[30] A. N. Tikhonov and V. Y. Arsenin, Solutions of Ill-posed Problems, 1977, p. 487.

[31] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” Science, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.

[32] T. Lin and H. Zha, “Riemannian manifold learning,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 5, pp. 796–809, May 2008.

[33] T. van Laarhoven, “L2 regularization versus batch and weight normalization,” 2018, pp. 1–10.

[34] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE,” J. Mach. Learn. Res., vol. 9, no. 11, pp. 2579–2605, 2008.

[35] L. Cayton, “DuoRC: Towards complex language understanding with paraphrased reading comprehension,” 2018, arXiv:1804.07927.

[36] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in Adv. Neural Inf. Process. Syst., 2017, pp. 1–9.

[37] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and A. Sutskever, “Language models are unsupervised multitask learners,” OpenAI Blog, vol. 1, no. 8, p. 9, 2019.

[38] A. Warstadt, A. Singh, and S. R. Bowman, “Neural network acceptability judgments,” Trans. Assoc. Comput. Linguistics, vol. 7, pp. 625–641, Nov. 2019.

[39] Y. Liu et al., “RoBERTa: A robustly optimized BERT pretraining procedure,” 2019, arXiv:1907.11692.

[40] A. Saha, R. Aralikatte, M. M. Khapra, and K. Sankaranarayanan, “DuoRC: Towards complex language understanding with paraphrased reading comprehension,” 2018, arXiv:1804.07927.

[41] B. W. Matthews, “Comparison of the predicted and observed secondary structure of T4 phage lysozyme,” Biochimica Biophysica Acta (BBA) Protein Struct., vol. 405, no. 2, pp. 442–451, Oct. 1975.

[42] W. Dolan and C. Brockett, “Automatically constructing a corpus of sentential paraphrases,” in Proc. 3rd Int. Workshop Paraphrasing (IWP), 2005, pp. 9–16.

[43] I. Dagan, O. Glickman, and B. Magnini, “The PASCAL recognising textual entailment challenge,” in Proc. Mach. Learn. Challenges Workshop, 2005, pp. 1–16.

[44] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized autoregressive pretraining for language understanding,” 2019, arXiv:1906.08237.

[45] J. Pennington, R. Socher, and C. Manning, “GloVe: Global vectors for word representation,” in Proc. Empirical Methods Natural Lang. Process. (EMNLP), Oct. 2014, pp. 1532–1543.

[46] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” 2018, arXiv:1907.01827.

[47] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “SQuAD: 100,000 questions for machine comprehension,” Adv. Neural Inf. Process. Syst., 2016, pp. 4000–4008.

[48] Q. V. Le, M. Auli, and D. Yarats, “GLUE: A multi-task benchmark and analysis platform for natural language understanding,” arXiv:1804.07461, 2018, pp. 1–17.

[49] Y. Liu, Z. Chen, J. Zhang, W. Feng, X. Wang, and Y. Cao, “Robust yet effective fine-tuning for pre-trained language models,” Adv. Neural Inf. Process. Syst., 2019, pp. 15367–15379.

[50] J. Pennington, R. Socher, and C. Manning, “GloVe: Global vectors for word representation,” in Proc. Empirical Methods Natural Lang. Process. (EMNLP), Oct. 2014, pp. 1532–1543.

[51] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” 2018, arXiv:1907.01827.

[52] Q. V. Le, M. Auli, and D. Yarats, “GLUE: A multi-task benchmark and analysis platform for natural language understanding,” Adv. Neural Inf. Process. Syst., 2019, pp. 15367–15379.

[53] Y. Zhou, L. Liao, Y. Gao, R. Wang, and H. Huang, “TopicBERT: A topic-enhanced neural language model fine-tuned for sentiment classification,” IEEE Trans. Neural Netw. Learn. Syst., vol. 34, no. 1, pp. 380–393, Jan. 2023.

[54] T. Wolf et al., “HuggingFace’s Transformers: State-of-the-art natural language processing,” 2019, arXiv:1910.03771.

[55] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.

[56] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with GPUs,” 2017, arXiv:1702.08734.

[57] A. Aghajanyan, A. Shirivastava, A. Gupta, N. Goyal, L. Zettlemoyer, and S. Gupta, “Better fine-tuning by reducing representational collapse,” 2020, arXiv:2008.01316.

[58] J. Pennington, R. Socher, and C. Manning, “GloVe: Global vectors for word representation,” in Proc. Empirical Methods Natural Lang. Process. (EMNLP), Oct. 2014, pp. 1532–1543.

[59] J. Pennington, R. Socher, and C. Manning, “GloVe: Global vectors for word representation,” in Proc. Empirical Methods Natural Lang. Process. (EMNLP), Oct. 2014, pp. 1532–1543.

[60] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized autoregressive pretraining for language understanding,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 1–11.

[61] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, “The difficulty of training deep architectures and the effect of unsupervised pre-training,” in Proc. 12th Int. Conf. Artif. Intell. Statist., 2009, pp. 153–160.

[62] D. Erhan, A. Courville, Y. Bengio, P.-A. Manzagol, and P. Vincent, “Why does unsupervised pre-training help deep learning?” J. Mach. Learn. Res., vol. 11, no. 19, pp. 625–660, 2010.

[63] D. Erhan, A. Courville, Y. Bengio, P.-A. Manzagol, and P. Vincent, “Why does unsupervised pre-training help deep learning?” J. Mach. Learn. Res., vol. 11, no. 19, pp. 625–660, 2010.

[64] D. Erhan, A. Courville, Y. Bengio, P.-A. Manzagol, and P. Vincent, “Why does unsupervised pre-training help deep learning?” J. Mach. Learn. Res., vol. 11, no. 19, pp. 625–660, 2010.
[81] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy, “RACE: Large-scale reading comprehension dataset from examinations,” 2017, arXiv:1704.04683.
[82] O. Levy, M. Seo, E. Choi, and L. Zettlemoyer, “Zero-shot relation extraction via reading comprehension,” 2017, arXiv:1706.04115.
[83] A. Kembhavi, M. Seo, D. Schwenk, J. Choi, A. Farhadi, and H. Hajishirzi, “Are you smarter than a sixth grader? Textbook question answering for multimodal machine comprehension,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5376–5384.

Hang Hua received the bachelor’s degree from the South China University of Technology, Guangzhou, China, in 2016, and the master’s degree from Peking University, Beijing, China, in 2020. He is currently pursuing the Ph.D. degree in computer science with the University of Rochester, Rochester, NY, USA. He is a member of the VISIA Group, Auckland, New Zealand, advised by Prof. Jiebo Luo. His research interests are vision and language, machine learning, and social media analysis.

Xingjian Li received the B.S. degree in microelectronics from Tsinghua University, Beijing, China, in 2008, the M.S. degree in computer science and technology from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2011, and the Ph.D. degree in computer science from the University of Macau, Macau, China, in 2023. He is currently a Post-Doctoral Associate at Carnegie Mellon University, Pittsburgh, PA, USA. His research interests include transfer learning, semi-supervised learning, and interpretable deep learning.

Dejing Dou (Senior Member, IEEE) was the Head of the Big Data Lab (BDL) and the Business Intelligence Lab (BIL), Baidu Research, Beijing, China. He was also a Full Professor with the Computer and Information Science Department, University of Oregon, Eugene, OR, USA. He is currently the Chief Data Scientist at BCG in Greater China, Beijing. He has published more than 100 research papers, some of which appear in prestigious conferences and journals, such as Association for the Advancement of Artificial Intelligence (AAAI), International Joint Conference on Artificial Intelligence (IJCAI), International Conference on Machine Learning (ICML), Conference on Neural Information Processing Systems (NeurIPS), International Conference on Learning Representations (ICLR), Knowledge Discovery and Data Mining (KDD), International Conference on Data Mining (ICDM), Association for Computational Linguistics (ACL), Empirical Methods in Natural Language Processing (EMNLP), Conference on Computer Vision and Pattern Recognition (CVPR), International Conference on Computer Vision (ICCV), Conference on Information and Knowledge Management (CIKM), International Semantic Web Conference (ISWC), ACM Transactions on Knowledge Discovery from Data (TKDD), Journal of Intelligent Information Systems (JIIS), and Journal of Data Semantics (JoDS), with more than 4100 Google Scholar citations. His research areas include artificial intelligence, data mining, data integration, natural language processing (NLP), and health informatics.

Dr. Dou is a Senior Member of Association for Computing Machinery (ACM). He has been serving as the program committee members for major international conferences and as the program co-chairs for five of them. His DEXA’15 paper received the Best Paper Award. His KDD’07 paper was nominated for the Best Research Paper Award. He has received over U.S. $5 million PI research grants from the National Science Foundation (NSF) and the National Institutes of Health (NIH). He is on the editorial boards of Journal on Data Semantics, Journal of Intelligent Information Systems, and FLOS One. He is an Editor-in-Chief of Electronic Research Archive (ARMS).

Cheng-Zhong Xu (Fellow, IEEE) received the B.Sc. and M.Sc. degrees from Nanjing University, Nanjing, China, in 1986 and 1989, respectively, and the Ph.D. degree from The University of Hong Kong, Hong Kong, in 1993, all in computer science and engineering.

He was on the faculty of Wayne State University, Detroit, MI, USA, and the Shenzhen Institutes of Advanced Technology, CAS, Shenzhen, China. He was a co-inventor of more than 120 patents and a co-founder of Shenzhen Baidu Applied Technology Company, Shenzhen, with a dedication to smart city technologies. He is currently a Chair Professor of computer science and the Dean of the Faculty of Science and Technology, University of Macau, Macau, SAR, China. He published two research monographs and more than 400 journal and conference papers and received more than 15k citations with an H-index of 59. His recent research interests are in cloud and distributed computing, systems support for artificial intelligence (AI), smart city, and autonomous driving.

Dr. Xu was a Best Paper Awardee or a Nominee of conferences, including HPCA’2013, HPDC’2013, Cluster’2015, ICPP’2015, GPC’2018, UIC’2018, and HPBD&IS’2019. He received the President’s Award for Excellence in Teaching from Wayne State University in 2002. He was the Chair of the IEEE Technical Committee on Distributed Processing from 2015 to 2020. He serves or served for a number of journal editorial boards, including IEEE TRANSACTIONS ON COMPUTERS (TC), IEEE TRANSACTIONS ON CLOUD COMPUTING (TCC), IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS (TPDS), Journal of Parallel and Distributed Computing (JPDC), Science (China), and ZTE Communication.

Jiebo Luo (Fellow, IEEE) is currently an Albert Arendt Hopeman Professor of engineering and a Professor of computer science with the University of Rochester, Rochester, NY, USA, which he joined in 2011 after a prolific career of 15 years with the Kodak Research Laboratories. He has authored nearly 600 technical papers and holds more than 90 U.S. patents. His research interests include computer vision, natural language processing (NLP), machine learning, data mining, computational social science, and digital health.

Dr. Luo is a fellow of National Academy of Inventors (NAI), Association for Computing Machinery (ACM), Association for the Advancement of Artificial Intelligence (AAAI), Society of Photographic Instrumentation Engineers (SPIE), and International Association for Pattern Recognition (IAPR). He has been involved in numerous technical conferences, including serving as the Program Co-Chair for ACM Multimedia 2010, IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2012, ACM International Conference on Multimedia Retrieval (ICMR) 2016, and IEEE International Conference on Image Processing (ICIP) 2017, and the General Co-Chair for ACM Multimedia 2018 and IEEE International Conference on Multimedia & Expo (ICMIE) 2024. He has served on the editorial boards of IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI), IEEE TRANSACTIONS ON MULTIMEDIA (TMM), IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (TCSTV), IEEE TRANSACTIONS ON BIG DATA (TBD), ACM Transactions on Intelligent Systems and Technology (TIST), Pattern Recognition, Knowledge and Information Systems (KAIS), Machine Vision and Applications, and Intelligent Medicine. He was an Editor-in-Chief of IEEE TRANSACTIONS ON MULTIMEDIA from 2020 to 2022.