Can the Spectrum of the Neural Tangent Kernel Anticipate Fine-Tuning Performance?

Zahra Rahimi Afzal **Tara Esmaeilbeig** **Mojtaba Soltanalian** **Mesrob I Ohannessian**
Department of Electrical and Computer Engineering
University of Illinois Chicago
{zrahim2,zesmae2,msol,mesrob}@uic.edu

Abstract

Parameter-Efficient Fine-tuning (PEFT) offers a scalable and resource-efficient solution for adaptating large models. Despite its popularity, the mechanisms underlying the performance of PEFT in terms of empirical risk and generalization remain underexplored. In this paper, we provide new insights into fine-tuning by analyzing PEFT through the lens of kernel methods, specifically by examining the relationship between the Neural Tangent Kernel (NTK) spectrum and the effectiveness of fine-tuning. Our findings reveal a strong correlation between the NTK spectrum and the model's adaptation performance, shedding light on both empirical risk and generalization properties. We evaluate our theory with Low Rank Adaptation (LoRA) on large language models. These insights not only deepen our understanding of LoRA but also offer a novel perspective for enhancing other PEFT techniques, paving the way for more robust and efficient adaptation in large language models.

1 Introduction

With the emergence of large language models (LLMs) [1, 2], their application to various NLP tasks has been on the rise. However, due to the enormous number of trainable parameters in these models, full fine-tuning can be expensive in terms of time and other computational costs. To address this issue, several fine-tuning approaches have been proposed [3]. To further reduce the computational burden and improve efficiency, Parameter-Efficient Fine-tuning (PEFT) methods have gained popularity [4–10]. These methods aim to reduce the number of trainable parameters while maintaining the model's performance.

In this paper, we look at adaptation in large models through the lens of Neural Tangent Kernel (NTK) approximations and show that the spectral behaviour of the NTK is crucial in determining the performance of fine-tuning. We used the promising PEFT, Low Rank Adaptation (LoRA), to validate our results [11]. Our contributions are:

- We formulate the fine-tuning problem as a neural tangent kernel regression and use the spectrum of the neural tangent kernel of the pretrained model to derive bounds on the empirical risk of the end result of fine-tuning.
- Through extensive experiments, we validate our theoretical results. We evaluate the condition number of NTK as an at-initialization metric, to anticipate the performance of LoRA before training. Even though our experiments focus on LoRA, the technical tools we introduce could be equally used in the context of other PEFT methods.

38th Conference on Neural Information Processing Systems (NeurIPS 2024) Workshop on Adaptive Foundation Models.

2 Benefit of fine-tuning in the linearized regime

Let the pre-trained model be $f_{\boldsymbol{\theta}}$. We call the model *linearized* or in the *lazy regime*, if during training, the change of the network can be approximated by its first-order Taylor expansion [12, 13] as in

$$f_{\boldsymbol{\theta}_t}(\mathbf{x}) \approx f_{\boldsymbol{\theta}_{t-1}}(\mathbf{x}) + \left\langle \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_{t-1}}(\mathbf{x}), \boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1} \right\rangle, \tag{1}$$

where $\boldsymbol{\theta}_t$ is the collection of all trainable parameters at step t of optimization. For instance, in SGD, the update to parameters at step t is given by

$$\begin{aligned} \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t &= \eta \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_T} \left[\nabla_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}_t}(\mathbf{x}), \mathbf{y}) \right] \\ &= \eta \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_T} \left[\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x}) \nabla_f \mathcal{L}\left(f_{\boldsymbol{\theta}_t}(\mathbf{x}), \mathbf{y}\right) \right]. \end{aligned} \tag{2}$$

Therefore, we have

$$\begin{aligned} & f_{\boldsymbol{\theta}_{t+1}}(\mathbf{x}') - f_{\boldsymbol{\theta}_t}(\mathbf{x}') \approx \left\langle \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x}'), \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t \right\rangle \\ & = \eta \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x}')^\top \cdot \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_T} \left[\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x}) \nabla_f \mathcal{L}\left(f_{\boldsymbol{\theta}_t}(\mathbf{x}), \mathbf{y}\right) \right] \\ & = \eta \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_T} \left[\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x}')^\top \cdot \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x}) \nabla_f \mathcal{L}\left(f_{\boldsymbol{\theta}_t}(\mathbf{x}), \mathbf{y}\right) \right] \\ & = \eta \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_T} \left[\mathbf{k}_t(\mathbf{x}, \mathbf{x}') \nabla_f \mathcal{L}\left(f_{\boldsymbol{\theta}_t}(\mathbf{x}), \mathbf{y}\right) \right]. \end{aligned} \tag{3}$$

This indicates that the learning dynamics of SGD is equivalent to NTK regression when the kernel is chosen to be the NTK, i.e., $\mathbf{k}_t(\mathbf{x}, \mathbf{x}') = \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x}')^\top \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}_t}(\mathbf{x})$ [12]. The NTK at $t = 0$ defines a neural tangent space to the pre-trained model $f_{\boldsymbol{\theta}_0}$. Often, during fine-tuning, the evolution of parameters is minimal and therefore the fine-tuned model closely follows regression on the tangent space. In this regime, training follows the linear dynamic in (3). The benefit of linearized models for fine-tuning is twofold. First, due to the laziness of the model we have $\mathbf{k}_t(\mathbf{x}, \mathbf{x}') \approx \mathbf{k}_0(\mathbf{x}, \mathbf{x}')$ [13]. In other words, $\mathbf{k}_0(\mathbf{x}, \mathbf{x}')$ can act as an at-initialization metric to predict the performance of fine-tuning. Second, the kernel that appears in the gradient descent steps in (3) generalizes to values of $\mathbf{x}'$ outside the dataset $\mathcal{D}_T$. This could allow investigating the generalization properties of fine-tuning by looking at the properties of the NTK [14–18].

3 Fine-tuning meets neural tangent kernel regression

We formally define the fine-tuning problem as a regularized function estimation in the reproducing kernel Hilbert space (RKHS), $\mathcal{H}$, generated by the NTK, $\mathbf{k}(\mathbf{x}, \mathbf{x}') = \nabla f_{\boldsymbol{\theta}_0}(\mathbf{x})^\top \nabla f_{\boldsymbol{\theta}_0}(\mathbf{x}')$. In fine-tuning, we are given the pre-trained model $f_{\boldsymbol{\theta}_0}(\cdot) : \mathbb{R}^d \to \mathbb{R}^c$, the target dateset $\mathcal{D}_T = (\mathbf{x}_i, \mathbf{y}_i)_{i=1}^n$ for the downstream task, with $\mathcal{L}(\cdot, \cdot) : \mathbb{R}^c \times \mathbb{R}^c \to \mathbb{R}$ denoting a loss function. The fine-tuned model is denoted by $f_{\boldsymbol{\theta}^*}(\cdot) : \mathbb{R}^d \to \mathbb{R}^c$ which is obtained by minimizing the typical empirical risk minimization problem

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\text{minimize}}\ \mathcal{R}(\boldsymbol{\theta}), \tag{4}$$

where

$$\mathcal{R}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}_i), \mathbf{y}_i). \tag{5}$$

Assuming that the fine-tuning is in the linearized regime, $f_{\boldsymbol{\theta}^*}(\cdot)$ can be approximated by its dual in the tangent space. In particular, the empirical risk minimization (4) is approximated by kernel regression when the kernel is NTK. Moreover, fine-tuning the model using mean square error (MSE) is also equivalent and can be achieved by solving the optimization problem presented in (6). Let $\mathcal{H}$ be the reproducing kernel Hilbert space (RKHS) endowed with a positive definite kernel function $\mathbf{k}(\cdot, \cdot)$, i.e.,

$$\mathcal{H} = \left\{ f(\cdot) = \sum_{i=1}^{n} \alpha_i \mathbf{k}(\cdot, \mathbf{x}_i) \right\}.$$

Assuming the solution lies in or close to this Hilbert space, then as an alternative to (4), we solve

$$\underset{f \in \mathcal{H}}{\text{minimize}}\ \frac{1}{n} \sum_{(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_T} \|f(\mathbf{x}) - \mathbf{y}\|_2^2 + \sigma \|f\|_{\mathcal{H}}^2, \tag{6}$$

where $\sigma > 0$ is a regularization parameter and $\|\cdot\|_{\mathcal{H}}$ is the norm corresponding to the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ defined on the RKHS $\mathcal{H}$. We thus effectively formulate the fine-tuning problem as a regularized function estimation in the RKHS, $\mathcal{H}$, generated by the NTK, $\mathbf{k}(\mathbf{x},\mathbf{x}') = \nabla f_{\boldsymbol{\theta}_0}(\mathbf{x})^\top \nabla f_{\boldsymbol{\theta}_0}(\mathbf{x}')$. According to the representer's theorem [19], the problem (6) on training dataset $\mathcal{D}_T = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$ possesses the closed-form solution

$$f^*(\cdot) = \sum_{i=1}^{n} \alpha_i \mathbf{k}(\cdot, \mathbf{x}_i) = \boldsymbol{\alpha}^\top \mathbf{K}(\cdot, \mathbf{X}), \tag{7}$$

where $\mathbf{K}(\cdot, \mathbf{X}) = [\mathbf{k}(\cdot, \mathbf{x}_1), \ldots, \mathbf{k}(\cdot, \mathbf{x}_n)] \in \mathbb{R}^{1\times n}$. Substituting (7) in (6), we have

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \;\; \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_T} \left\|\boldsymbol{\alpha}^\top \mathbf{K}(\cdot, \mathbf{X}) - \mathbf{y}\right\|^2 + \sigma \|f\|_{\mathcal{H}}^2, \tag{8}$$

which is a convex problem with $\boldsymbol{\alpha}^* = \left[\mathbf{K}\left(\mathbf{X},\mathbf{X}\right) + \sigma\mathbf{I}\right]^{-1}\mathbf{y}$ as the solution. Equivalently,

$$f^*(\cdot) = \mathbf{K}\left(\cdot, \mathbf{X}\right)\left[\mathbf{K}\left(\mathbf{X},\mathbf{X}\right) + \sigma\mathbf{I}\right]^{-1}\mathbf{y}, \tag{9}$$

where $\mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\right]^\top$ is a $n \times d$ matrix, $\mathbf{y} = \left[\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\right]^\top$ is a $n \times 1$ matrix for binary classification, n is the sample size of the training dataset and $[\mathbf{K}(\mathbf{X},\mathbf{X})]_{i,j} = \mathbf{k}(\mathbf{x}_i, \mathbf{x}_j)$. For the sake of brevity, hereafter, we use $\mathbf{K}(\mathbf{X},\mathbf{X})$ and $\mathbf{K}$ interchangeably.

Theorem 1. *The empirical risk is bounded as*

$$\frac{\sigma \left\|\mathbf{y}\right\|_2^2}{\sigma + \lambda_{max}(\mathbf{K})} \leq \mathcal{R}(\boldsymbol{\theta}) \leq \frac{\sigma \left\|\mathbf{y}\right\|_2^2}{\sigma + \lambda_{min}(\mathbf{K})} \tag{10}$$

where $\lambda_{min}(\mathbf{K})$ and $\lambda_{max}(\mathbf{K})$ are the minimum and maximum eigenvalues of $\mathbf{K}(\mathbf{X},\mathbf{X})$, respectively.

Proof. Let $\mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^\top$ denote the eigenvalue decomposition of $\mathbf{K}\left(\mathbf{X},\mathbf{X}\right)$, where $\boldsymbol{\Sigma} = \text{Diag}\left(\lambda_{\text{min}}(\mathbf{K}), \ldots, \lambda_{\text{max}}(\mathbf{K})\right)$ and $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$. Then

$$\begin{aligned}
\mathcal{R}(\boldsymbol{\theta}) &= \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x}_i), \mathbf{x}_i) \\
&= \frac{1}{n}\sum_{i=1}^{n} \left\| y_i - \mathbf{K}\left(\mathbf{x}_i, \mathbf{X}\right)\left[\mathbf{K}\left(\mathbf{X},\mathbf{X}\right) + \sigma\mathbf{I}\right]^{-1}\mathbf{y}\right\|_2^2 \\
&= \frac{1}{n}\left\| \mathbf{y} - \mathbf{K}\left(\mathbf{X},\mathbf{X}\right)\left(\mathbf{K}\left(\mathbf{X},\mathbf{X}\right) + \sigma\mathbf{I}\right)^{-1}\mathbf{y}\right\|_2^2 \\
&= \frac{1}{n}\left\| \left(\mathbf{I} - \mathbf{K}\left(\mathbf{X},\mathbf{X}\right)\left(\mathbf{K}\left(\mathbf{X},\mathbf{X}\right) + \sigma\mathbf{I}\right)^{-1}\right)\mathbf{y}\right\|_2^2 \\
&= \frac{1}{n}\left\| \left(\mathbf{I} - \mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^\top(\mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^\top + \sigma\mathbf{I})^{-1}\right)\mathbf{y}\right\|_2^2 \\
&= \frac{1}{n}\left\| \left(\mathbf{I} - \mathbf{U}\boldsymbol{\Sigma}(\boldsymbol{\Sigma} + \sigma\mathbf{I})^{-1}\mathbf{U}^\top\right)\mathbf{y}\right\|_2^2 \\
&= \frac{1}{n}\left\| \mathbf{U}\left(\mathbf{I} - \boldsymbol{\Sigma}(\boldsymbol{\Sigma} + \sigma\mathbf{I})^{-1}\right)\mathbf{U}^\top\mathbf{y}\right\|_2^2 \\
&= \frac{1}{n}\left\| \left(\mathbf{I} - \boldsymbol{\Sigma}(\boldsymbol{\Sigma} + \sigma\mathbf{I})^{-1}\right)\mathbf{U}^\top\mathbf{y}\right\|_2^2 .
\end{aligned} \tag{11}$$

Since $\mathbf{I} - \boldsymbol{\Sigma}(\boldsymbol{\Sigma} + \sigma\mathbf{I})^{-1}$ is a diagonal matrix, we have

$$\frac{\sigma \left\|\mathbf{y}\right\|_2^2}{\sigma + \lambda_{\text{max}}(\mathbf{K})} \leq \mathcal{R}(\boldsymbol{\theta}) \leq \frac{\sigma \left\|\mathbf{y}\right\|_2^2}{\sigma + \lambda_{\text{min}}(\mathbf{K})}. \tag{12}$$

□

Theorem 1 conveys that the spectrum of the NTK directly affects the empirical risk. We study the regularized condition number $\kappa(\mathbf{K} + \sigma\mathbf{I}) = \frac{\lambda_{\text{max}}(\mathbf{K}) + \sigma}{\lambda_{\text{min}}(\mathbf{K}) + \sigma}$ as an at-initialization metric for the performance of fine-tuning.

Dataset	Selected Layers	Selected Parameters	Condition Number	Train Loss	Evaluation Loss	Evaluation Accuracy
CoLA	{0}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	11,618	0.5271	0.5265	73.15
	{0,11}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	9,490	0.5174	0.5293	73.15
	{0,5,11}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	7,503	0.5093	0.5272	73.44
	{0,5,11}	$\{\mathbf{W}_k\}$	2,320	0.5128	0.5357	73.25
SST-2	{0}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	5,195	0.4794	0.3871	83.48
	{0,11}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	6,413	0.4677	0.3916	82.79
	{0,5,11}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	6,792	0.5078	0.3913	82.68
	{0,5,11}	$\{\mathbf{W}_k\}$	450.64	0.4717	0.3893	83.60
Yelp	{0}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	274	0.29	0.2597	88.20
	{0,11}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	4,167	0.2882	0.2596	88.24
	$\{0, 5, 11\}$	$\{\mathbf{W}_q, \mathbf{W}_v\}$	1,336	0.2885	0.2596	88.21
	{0,5,11}	$\{\mathbf{W}_k\}$	39.33	0.2865	0.2596	88.23
IMDb	{0}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	179	0.3512	0.2702	89.56
	{0,11}	$\{\mathbf{W}_q, \mathbf{W}_v\}$	5,899	0.3597	0.2717	89.50
	$\{0, 5, 11\}$	$\{\mathbf{W}_q, \mathbf{W}_v\}$	1,277	0.3709	0.2727	89.49
	{0,5,11}	$\{\mathbf{W}_k\}$	9.605	0.3642	0.2719	89.49

Table 1: RoBERTa-base models performance on GLUE tasks, condition number of the NTK, train loss, and evaluation loss at one snapshot of the training at 10-th epoch. LoRA with $r = 8$ is used for fine-tuning. Condition number is calculated as $\kappa(\mathbf{K} + \sigma\mathbf{I}) = \frac{\lambda_{\max}(\mathbf{K})+\sigma}{\lambda_{\min}(\mathbf{K})+\sigma}$, and $\sigma = 1e^{-4}$ is fixed among all tasks.

4 Experiments

In our experiments, we implement LoRA on RoBERTa base and evaluate its performance on the GLUE benchmark [20] (including CoLA [21] and SST-2 [22] tasks), IMDb [23], and Yelp [24] datasets. The Yelp dataset originally contains reviews with ratings from 1 to 5. To convert it into a binary classification task, we consider reviews with ratings less than 3 as label 0 (negative sentiment) and those with ratings greater than or equal to 3 as label 1 (positive sentiment). For all experiments, we use LoRA on RoBERTa base from the HuggingFace Transformers library [25], and report its performance on different tasks using NVIDIA Tesla V100 GPUs.

In the LoRA framework, for a pre-trained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{m\times p}$, the update is

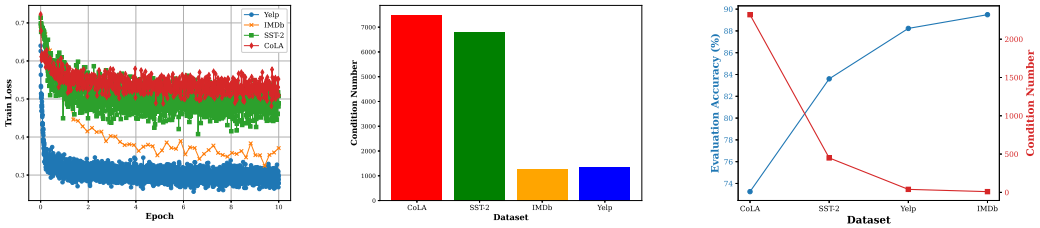
$$\mathbf{W} = \mathbf{W}_0 + \mathbf{BA}, \tag{13}$$

where $\mathbf{B} \in \mathbb{R}^{m\times r}$ and $\mathbf{A} \in \mathbb{R}^{r\times p}$ are the learnable low-rank matrices with r representing the rank of the adaptation and is much smaller than both m and p. During the fine-tuning process, the pre-trained weight matrix $\mathbf{W}_0$ is kept fixed, while the optimization focuses solely on updating the low-rank matrices $\mathbf{B}$ and $\mathbf{A}$. The total number of trainable parameters for each of the query ($\mathbf{W}_q$), key ($\mathbf{W}_k$), and value ($\mathbf{W}_v$) matrices per selected layer is $(m + p) \times r$. In our experiments, we apply LoRA with $r = 8$ which has $(m + p) \times r = 2 \times 768 \times 8$ trainable parameters per selected layer for each of the query, key, and value projection matrices in the self-attention mechanism in the RoBERTa base model.

4.1 NTK evaluation

The hypothesis of this paper is that in the linearized lazy regime of large models, the NTK is assumed constant during training and fine-tuning searches the tangent space during SGD. We verify our proposition that by calculating the condition number of the NTK matrix for the LoRA model at initialization, we can predict the generalization error, including evaluation loss and accuracy.

Table 1 presents train loss, evaluation loss, accuracy, condition number of the NTK before fine-tuning, based on the snapshot at epoch 10. We vary LoRA parameters across different layers ({0}, {11}, {0,11}, {0,5,11}) for query and value parameters, and layers {0,5,11} for key parameters, across various tasks and datasets. We collected $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^\top$ using $n = 32$ samples, randomly selected from the training datasets and computed $\mathbf{k}(\mathbf{x}_i, \mathbf{x}_j)$ with respect to trainable parameters, $\mathbf{A}$ and $\mathbf{B}$ of LoRA. The final empirical NTK matrix is $\mathbf{K}(\mathbf{X}, \mathbf{X}) \in \mathbb{R}^{32\times 32}$. Note that the number of samples used for calculation of the empirical NTK is orders of magnitude smaller than the training dataset for sampling. This shows that the results remain valid even with a sketch of the full kernel.



(a) Train loss over 10 epochs

(b) Condition number

(c) Evaluation accuracy and Condition number

Figure 1: (a)-(b) Illustrate the positive correlation between the convergence rate of optimization steps of LoRA over 10 epochs and $\kappa(\mathbf{K} + \sigma\mathbf{I})$ of NTK at initialization. $\{\mathbf{W}_q, \mathbf{W}_v\}$ of layers $\{0, 5, 11\}$ are fine-tuned. (c) Illustrates the negative correlation between evaluation accuracy after 10 epochs of training and condition number of NTK. LoRA with $r = 8$ is used to fine-tune $\{\mathbf{W}_k\}$ of the layers $\{0, 5, 11\}$.

This finding highlights the great potential of kernel methods for large language models (LLMs), particularly in terms of efficiency.

Training time and NTK calculation time are reported in Table 2 for fine-tuning $\{\mathbf{W}_k\}$ in layers $\{0, 5, 11\}$. In this scenario, the total number of trainable parameters (TTPs) is 0.628M. It is worth mentioning that the classifier layer contains the majority of these parameters. From TTPs, we selected only the LoRA parameters for the NTK calculation, resulting in 36.8K parameters. As shown in the table, fine-tuning, even with just 10 epochs, has significantly higher computational overhead than computing the NTK. This finding supports the advantage of the present approach in terms of time complexity when comparing the risks of different datasets without training. Additionally, since Yelp and IMDb are larger datasets, it is evident that fine-tuning on them requires more time compared to the others.

Dataset	Fine-tuning Time	NTK Calculation Time
CoLA	187	33
SST-2	794	63
Yelp	46,096	245
IMDb	1,541	55

Table 2: Fine-tuning time(s), NTK calculation time(s), $\{\mathbf{W}_k\}$ of layers $\{0, 5, 11\}$ are fine-tuned. In all datasets, only 32 random samples from the training set are used calculating the NTK.

Figure 1(a)-(b), illustrates the positive correlation between condition number of the NTK matrix at initialization and training loss for different tasks. In all datasets, the attention parameters of layers $\{0, 5, 11\}$ are fine-tuned and evaluation accuracy was reported. Although for CoLA, it is customary to report Matthew's correlation coefficients [26], we adhere to reporting the evaluation accuracy for all tasks in Figure 1(c), to maintain consistency in the evaluation metric across different datasets. Figure 1(c) starkly illustrates an inverse relationship between the condition number of the NTK and the model's evaluation accuracy. In our experiments we observed that $\lambda_{\text{min}}(\mathbf{K})$ is almost always close to zero and the regularized condition number, $\kappa(\mathbf{K} + \sigma\mathbf{I})$, is tracing the spectral norm or $\lambda_{\text{max}}(\mathbf{K})$. For instance, the CoLA task, which exhibits highest training loss, also shows the largest condition number. This suggests that by computing the NTK matrix before training, we can identify which tasks are well-conditioned, i.e., lower conditional number indicates lower training and evaluation loss.

5 Conclusion

Our theoretical analysis demonstrates that the empirical risk is bounded by the condition number of NTK. This gives an at-initialization anticipation of fine-tuning performance, at a fraction of the computational cost. More precisely, to achieve this, we propose to calculate the condition number of the NTK using only the LoRA parameters, which can be done significantly quicker than fine-tuning. By comparing these condition numbers, we can predict which tasks will have smaller empirical loss without actually performing the fine-tuning process. This approach provides a quick and efficient way to assess the potential performance of the model on different tasks, saving valuable time and computational resources.

References

[1] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "LLaMA: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[2] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "PaLM: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[3] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *Association for Computational Linguistics*, pp. 328–339, 2018.

[4] E. Ben-Zaken, S. Ravfogel, and Y. Goldberg, "Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models," *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1–9, 2022.

[5] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," *International conference on machine learning*, pp. 2790–2799, 2019.

[6] N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen *et al.*, "Parameter-efficient fine-tuning of large-scale pre-trained language models," *Nature Machine Intelligence*, vol. 5, no. 3, pp. 220–235, 2023.

[7] N. J. Prottasha, A. Mahmud, M. S. I. Sobuj, P. Bhat, M. Kowsher, N. Yousefi, and O. O. Garibay, "Parameter-efficient fine-tuning of large language models using semantic knowledge tuning," *arXiv preprint arXiv:2410.08598*, 2024.

[8] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, 2021.

[9] A. Tomihari and I. Sato, "Understanding linear probing then fine-tuning language models from ntk perspective," *arXiv preprint arXiv:2405.16747*, 2024.

[10] M. Kowsher, T. Esmaeilbeig, C.-N. Yu, M. Soltanalian, and N. Yousefi, "RoCoFT: Efficient fine-tuning of large language models with Row-Column updates," *arXiv preprint arXiv:2410.10075*, 2024.

[11] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank adaptation of large language models," *International Conference on Learning Representations*, 2022.

[12] A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: Convergence and generalization in neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[13] S. Malladi, A. Wettig, D. Yu, D. Chen, and S. Arora, "A kernel-based view of language model fine-tuning," *International Conference on Machine Learning*, pp. 23 610–23 641, 2023.

[14] B. Bordelon, A. Canatar, and C. Pehlevan, "Spectrum dependent learning curves in kernel regression and wide neural networks," in *International Conference on Machine Learning*. PMLR, 2020, pp. 1024–1034.

[15] G. Ortiz-Jiménez, S.-M. Moosavi-Dezfooli, and P. Frossard, "What can linearized neural networks actually say about generalization?" *Advances in Neural Information Processing Systems*, vol. 34, pp. 8998–9010, 2021.

[16] M.-Y. Chung, S.-Y. Chou, C.-M. Yu, P.-Y. Chen, S.-Y. Kuo, and T.-Y. Ho, "Rethinking backdoor attacks on dataset distillation: A kernel method perspective," *The Twelfth International Conference on Learning Representations*, 2024.

[17] U. Jang, J. D. Lee, and E. K. Ryu, "LoRA training in the NTK regime has no spurious local minima," *Forty-first International Conference on Machine Learning*, 2024.

[18] A. Atanasov, A. Meterez, J. B. Simon, and C. Pehlevan, "The optimization landscape of SGD across the feature learning strength," *arXiv preprint arXiv:2410.04642*, 2024.

[19] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, "Reproducing kernel Hilbert space, Mercer's theorem, eigenfunctions, Nyström method, and use of kernels in machine learning: Tutorial and survey," *CoRR*, vol. abs/2106.08443, 2021.

[20] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," *Association for Computational Linguistics*, pp. 353–355, 2018.

[21] A. Warstadt, A. Singh, and S. R. Bowman, "Neural network acceptability judgments," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 625–641, 2019.

[22] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

[23] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

[24] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," *Advances in neural information processing systems*, vol. 28, pp. 649–657, 2015.

[25] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Transformers: State-of-the-art natural language processing," *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

[26] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Association for Computational Linguistics*, 2019.

A Model and Dataset

RoBERTa base incorporates several key modifications to the pre-training process, such as using larger batch sizes, longer sequences, and more diverse data than its antecedents like BERT [27]. Despite its relatively compact size of 125 million parameters, RoBERTa base has proven to be one of the most powerful models for various NLP tasks, including text classification, question answering, and named entity recognition, especially on the GLUE benchmark.

The GLUE benchmark provides a comprehensive evaluation of a model's performance across various NLP challenges, assessing its ability to understand and reason about language in different contexts. The IMDb dataset is a large dataset for binary sentiment classification, containing 50,000 highly polar movie reviews from the Internet Movie Database (IMDb). The Yelp dataset contains customer reviews from Yelp, a popular platform for crowd-sourced reviews about businesses, primarily restaurants. Table 3 shows specific hyperparameters for RoBERTa base across various benchmarks, including GLUE tasks (CoLA, SST-2), Yelp, and IMDb.

Dataset	CoLA	SST-2	Yelp	IMDb
Optimizer	AdamW			
Warmup Ratio	0.06			
LR Schedule	Linear			
Max Sequence Length	512			
LoRA Rank r	8			
LoRA α	8			
Number of Epochs	10			
Batch Size	32	16	32	16
Learning Rate	4e-4	5e-4	4e-4	4e-4

Table 3: Hyperparameters used for RoBERTa base on various benchmarks, including GLUE (CoLA, SST-2), Yelp, and IMDb