Rethinking Influence Functions of Neural Networks in the Over-Parameterized Regime

Rui Zhang,[1,2] **Shihua Zhang**[1,2*]
[1]NCMIS, CEMS, RCSDS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China
[2]School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China
{rayzhang, zsh}@amss.ac.cn

Abstract

Understanding the black-box prediction for neural networks is challenging. To achieve this, early studies have designed influence function (IF) to measure the effect of removing a single training point on neural networks. However, the classic implicit Hessian-vector product (IHVP) method for calculating IF is fragile, and theoretical analysis of IF in the context of neural networks is still lacking. To this end, we utilize the neural tangent kernel (NTK) theory to calculate IF for the neural network trained with regularized mean-square loss, and prove that the approximation error can be arbitrarily small when the width is sufficiently large for two-layer ReLU networks. We analyze the error bound for the classic IHVP method in the over-parameterized regime to understand when and why it fails or not. In detail, our theoretical analysis reveals that (1) the accuracy of IHVP depends on the regularization term, and is pretty low under weak regularization; (2) the accuracy of IHVP has a significant correlation with the probability density of corresponding training points. We further borrow the theory from NTK to understand the IFs better, including quantifying the complexity for influential samples and depicting the variation of IFs during the training dynamics. Numerical experiments on real-world data confirm our theoretical results and demonstrate our findings.

Introduction

Influence function (Hampel 1974) is a classic technique from robust statistics, which measures the effect of changing a single sample point on an estimator. Koh and Liang (2017) transferred the concept of IFs to understanding why neural networks make corresponding predictions. IF is one of the most common approaches in explainable AI and is widely used for boosting model performance (Wang, Huan, and Li 2018), measuring group effects (Koh et al. 2019), investigating model bias (Wang, Ustun, and Calmon 2019), understanding generative models (Kong and Chaudhuri 2021) and so on. Specifically, IF reflects the effect of removing one training point on a neural network's prediction, thus can be used to discover the most influential training points for a given prediction. In their work, the implicit Hessian-vector product (IHVP) was utilized to estimate the IF for neural networks. However, the numerical experiments in (Basu, Pope, and Feizi 2021) pointed out that IFs calculated via IHVPs are often erroneous for neural networks. Furthermore, theoretical understanding for why these phenomena happened still lacks in the neural network regime.

To theoretically understand the IF for neural networks, we need to overcome the non-linearity and over-parameterized properties in neural networks. Fortunately, recent advances of NTK (Jacot, Gabriel, and Hongler 2018; Lee et al. 2019) shed light on the theory of over-parameterized neural networks. The key idea of NTK is that an infinitely wide neural network trained by gradient descent is equivalent to kernel regression with NTK. Remarkably, the theory of NTK builds a bridge between the over-parameterized neural networks and the kernel regression method, which dramatically reduces the difficulty of analyzing the neural networks theoretically. With the help of NTK, the theory for over-parameterized neural networks has achieved rapid progresses (Arora et al. 2019a; Du et al. 2019a; Hu, Li, and Yu 2020; Zhang et al. 2021), which encourage us to deal with the puzzle of calculating and understanding the IFs for neural networks in the NTK regime.

In this work, we utilize the NTK theory to calculate IFs and analyze the behavior theoretically for the over-parameterized neural networks. In summary, we make the following contributions:

- We utilize the NTK theory to calculate IFs for over-parameterized neural networks trained with regularized mean-square loss, and prove that the approximation error can be arbitrarily small when the width is sufficiently large for two-layer ReLU networks. Remarkably, we prove the first rigorous result to build the equivalence between the fully-trained neural network and the kernel predictor in the regularized situation. Numerical experiments confirm that IFs calculated in the NTK regime can approximate the actual IFs with high accuracy.
- We analyze the error bound for the classic IHVP method in the over-parameterized regime to understand when and why it fails or not. On the one hand, our bounds reveal that the accuracy of IHVP depends on the regularization term which was only characterized before by numerical experiments in (Basu, Pope, and Feizi 2021). On the other hand, we theoretically prove that the accuracy of IHVP has a significant correlation with the probability density of corresponding training points, which has not

[*]Shihua Zhang is the corresponding author.

been revealed in previous literature. Numerical experiments verify our bounds and statements.

- Furthermore, we borrow the theory from NTK to understand the behavior of IFs better. On the one hand, we utilize the theory of model complexity in NTK to quantify the complexity for influential samples and reveal that the most influential samples make the model more complicated. On the other hand, we track the dynamic system of the neural networks and depict the variation of IFs during the training dynamics.

Related Works

Influence Functions in Machine Learning

To explain the black-box prediction in neural networks, Koh and Liang (2017) utilized the concept of IF to trace a model's predictions through its learning algorithm and back to the training data. To be specific, they considered the following question: *How would the model's predictions change if we did not have this training point?*

To calculate the IF of a training point for neural networks, Koh and Liang (2017) proposed an approximate method based on IHVP and Chen et al. (2021) considered the training trajectory to avoid the calculation of Hessian matrix. However, Basu, Pope, and Feizi (2021) figured out that the predicting precision via IHVP may become particularly poor under certain conditions for neural networks.

Theory and Applications of NTK

In the last few years, several papers have shown that the infinite-width neural network with the square loss during training can be characterized by a linear differential equation (Jacot, Gabriel, and Hongler 2018; Lee et al. 2019; Arora et al. 2019a). In particular, when the loss function is the mean square loss, the inference performed by an infinite-width neural network is equivalent to the kernel regression with NTK. The progresses about NTK shed light on the theory of over-parameterized neural networks and were utilized to understand the optimization and generalization for shallow and deep neural networks (Arora et al. 2019a; Cao and Gu 2019; Allen-Zhu, Li, and Song 2019; Nguyen and Mondelli 2020), regularization methods (Hu, Li, and Yu 2020), and data augmentation methods (Li et al. 2019; Zhang et al. 2021) in the over-parameterized regime. NTK can be analytically calculated using exact Bayesian inference, which has been implemented in NEURAL TANGENTS for working with infinite-width networks efficiently (Novak et al. 2020).

In this work, we will firstly give rigorous prove to reveal the equivalence between the regularized neural networks and the kernel ridge regression predictor via NTK in the over-parameterized regime, then utilize the tools about NTK to calculate and better understand the properties of IFs.

Preliminaries

Notations

We utilize bold-faced letters for vectors and matrices. For a matrix $\mathbf{A}$, let $[\mathbf{A}]_{ij}$ be its (i,j)-th entry and $\mathrm{vec}(\mathbf{A})$ be its vectorization. For a vector $\mathbf{a}$, let $[\mathbf{a}]_i$ be its i-th entry. We use $\|\cdot\|_2$ to denote the Euclidean norm of a vector or the spectral norm of a matrix, and use $\|\cdot\|_F$ to denote the Frobenius norm of a matrix. We use $\langle\cdot,\cdot\rangle$ to denote the standard Euclidean inner product between two vectors or matrices. Let $\mathbf{I}_n$ be an $n\times n$ identity matrix, and $\mathbf{e}_i$ denote the i-th unit vector and $[n]=\{1,2,\cdots,n\}$. For a set A, we utilize $\mathrm{unif}(A)$ to denote the uniform distribution over A. We utilize $\mathbb{I}(\cdot)$ to denote the indicator function. We utilize $f^{\setminus i}$ to denote the network or kernel predictor trained without the i-th training point. To be clear, we respectively define f_{nn} and f_{ntk} as neural network (nn) and its corresponding NTK predictor. We further denote $\mathbf{W}(t)$ as the parameters of neural networks at time t during the training process.

Network Models and Training Dynamics

In this paper, we consider the two-layer neural networks with rectified linear unit (ReLU) activation:

$$f_{nn}(\mathbf{x})=\frac{1}{\sqrt{m}}\sum_{r=1}^{m}a_r\sigma\left(\mathbf{w}_r^\top\mathbf{x}\right), \tag{1}$$

where $\mathbf{x}\in\mathbb{R}^d$ is the input, $\mathbf{W}=[\mathbf{w}_1,\cdots,\mathbf{w}_m]\in\mathbb{R}^{d\times m}$ is the weight matrix in the first layer, and $\mathbf{a}=[a_1,\cdots,a_m]^\top\in\mathbb{R}^m$ is the weight vector in the second layer. We initialize the parameters randomly as follows:

$$\mathbf{w}_r(0)\sim\mathcal{N}\left(\mathbf{0},\kappa^2\mathbf{I}_d\right),\ a_r(0)\sim\mathrm{unif}(\{-1,1\}),\quad\forall r\in[m],$$

where $0<\kappa\ll 1$ controls the magnitude of initialization, and all randomnesses are independent. For simplicity, we fix the second layer $\mathbf{a}$ and only update the first layer $\mathbf{W}$ during training. The same setting has been used in (Arora et al. 2019a; Du et al. 2019b).

We are given n training points $(\mathcal{X},\mathcal{Y})=\{\mathbf{x}_i,y_i\}_{i=1}^n$ drawn i.i.d. from an underlying data distribution $\mathcal{D}$ over $\mathbb{R}^d\times\mathbb{R}$. For simplicity, we assume that for each $(\mathbf{x},y)$ sampled from $\mathcal{D}$ satisfying $\|\mathbf{x}\|_2=1$ and $|y|\le 1$.

To study the effect of regularizer on calculating the IF, we train the neural networks through gradient descent on the regularized mean square error loss function as follows:

$$\mathcal{L}(\mathbf{W})=\frac{1}{2}\sum_{i=1}^{n}\left(f_{nn}\left(\mathbf{x}_i;\mathbf{W}\right)-y_i\right)^2+\frac{\lambda}{2}\|\mathbf{W}-\mathbf{W}(0)\|_F^2, \tag{2}$$

where the regularizer term restricts the distance between the network parameters to initialization, and has been previously studied in (Hu, Li, and Yu 2020). And we consider minimizing the loss function $\mathcal{L}(\mathbf{W})$ in the gradient flow regime, i.e., gradient descent with infinitesimal step size, then the evolution of $\mathbf{W}(t)$ can be described by the following ordinary differential equation:

$$\begin{aligned}\frac{\mathrm{d}\,\mathbf{W}(t)}{\mathrm{d}\,t}&=-\frac{\partial\mathcal{L}(\mathbf{W}(t))}{\partial\mathbf{W}(t)}\\&=\sum_{i=1}^{n}(y_i-f_{nn}(\mathbf{x}_i;\mathbf{W}(t)))\frac{\partial f_{nn}(\mathbf{x}_i;\mathbf{W}(t))}{\partial\mathbf{W}(t)}\\&\quad-\lambda(\mathbf{W}(t)-\mathbf{W}(0)).\end{aligned} \tag{3}$$

NTK for Two-layer ReLU Neural Networks

Given two data points $\mathbf{x}$ and $\mathbf{x}'$, the NTK associated with two-layer ReLU neural networks has a closed form expression as follows (Xie, Liang, and Song 2017; Du et al. 2019b):

$$\begin{aligned}\mathbf{K}^{\infty}(\mathbf{x},\mathbf{x}') &\triangleq \lim_{m\to\infty}\left\langle \frac{\partial f_{nn}(\mathbf{x};\mathbf{W}(0))}{\partial \mathbf{W}(0)}, \frac{\partial f_{nn}(\mathbf{x}';\mathbf{W}(0))}{\partial \mathbf{W}(0)}\right\rangle \\ &= \mathbb{E}_{\mathbf{w}\sim\mathcal{N}(\mathbf{0},\mathbf{I}_d)}\left[\mathbf{x}^\top\mathbf{x}'\mathbb{I}\left\{\mathbf{w}^\top\mathbf{x}\ge 0, \mathbf{w}^\top\mathbf{x}'\ge 0\right\}\right] \\ &= \frac{\mathbf{x}^\top\mathbf{x}'\left(\pi - \arccos\left(\mathbf{x}^\top\mathbf{x}'\right)\right)}{2\pi}.\end{aligned} \tag{4}$$

Equipped with the NTK function, we consider the following kernel ridge regression problem:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^n} \frac{1}{2}\|\mathcal{Y}-\mathbf{K}_{tr}^{\infty}\boldsymbol{\beta}\|_2^2 + \frac{\lambda}{2}\boldsymbol{\beta}^\top\mathbf{K}_{tr}^{\infty}\boldsymbol{\beta}, \tag{5}$$

where $\mathbf{K}_{tr}^{\infty} \in \mathbb{R}^{n\times n}$ is the NTK matrix evaluated on the training data, i.e., $[\mathbf{K}_{tr}^{\infty}]_{i,j} = \mathbf{K}^{\infty}(\mathbf{x}_i,\mathbf{x}_j)$, and $\boldsymbol{\beta}^* \triangleq (\mathbf{K}_{tr}^{\infty}+\lambda\mathbf{I}_n)^{-1}\mathcal{Y}$ is the optimizer of the problem (5). Hence the prediction of kernel ridge regression using NTK on a test point $\mathbf{x}_{te}$ is:

$$f_{ntk}(\mathbf{x}_{te}) = (\mathbf{K}_{te}^{\infty})^\top(\mathbf{K}_{tr}^{\infty}+\lambda\mathbf{I}_n)^{-1}\mathcal{Y}, \tag{6}$$

where $\mathbf{K}_{te}^{\infty} \in \mathbb{R}^n$ is the NTK evaluated between the test point $\mathbf{x}_{te}$ and the training points $\mathcal{X}$, i.e., $[\mathbf{K}_{te}]_i = \mathbf{K}^{\infty}(\mathbf{x}_{te},\mathbf{x}_i)$.

IFs for Over-parameterized Neural Networks

In this section, our goal is to evaluate the effect of a single training point for the over-parameterized neural network's predictions. In detail, given a training point $\mathbf{x}_i \in \mathcal{X}$, we need to calculate the variation of test loss after removing $\mathbf{x}_i$ from $\mathcal{X}$ for the neural network f_{nn}, which is denoted by $\mathcal{I}_{nn}(\mathbf{x}_i,\mathbf{x}_{te})$ as follows:

$$\mathcal{I}_{nn}(\mathbf{x}_i,\mathbf{x}_{te}) \triangleq \frac{1}{2}(f_{nn}^{\setminus i}(\mathbf{x}_{te})-y_{te})^2 - \frac{1}{2}(f_{nn}(\mathbf{x}_{te})-y_{te})^2. \tag{7}$$

To calculate $\mathcal{I}_{nn}(\mathbf{x}_i,\mathbf{x}_{te})$ for all $\mathbf{x}_i$, it is impossible to retrain the neural network one by one. Furthermore, due to the fact that the neural network is over-parameterized, it is prohibitively slow to calculate the IF via the IHVP method (Koh and Liang 2017). In this work, we utilize the corresponding NTK predictor f_{ntk} to approximate the behavior of f_{nn}, hence we define the IF in the NTK regime as follows:

$$\mathcal{I}_{ntk}(\mathbf{x}_i,\mathbf{x}_{te}) \triangleq \frac{1}{2}(f_{ntk}^{\setminus i}(\mathbf{x}_{te})-y_{te})^2 - \frac{1}{2}(f_{ntk}(\mathbf{x}_{te})-y_{te})^2. \tag{8}$$

To calculate $f_{ntk}^{\setminus i}(\mathbf{x}_{te})$ efficiently, we borrow the technique from the online Gaussian regression to update the inverse of kernel matrix (Csató and Opper 2002; Engel, Mannor, and Meir 2004). Then we have:

$$f_{ntk}^{\setminus i}(\mathbf{x}_{te}) = (\mathbf{K}_{te}^{\infty})^\top\left((\mathbf{K}_{tr}^{\infty}+\lambda\mathbf{I}_n)^{-1} - \frac{\mathbf{k}_{-i}\mathbf{k}_{-i}^\top}{k_{-ii}}\right)\mathcal{Y}, \tag{9}$$

where $\mathbf{k}_{-i}$ and k_{-ii} denote the i-th column and the (i,i)-th entry of $(\mathbf{K}_{tr}^{\infty}+\lambda\mathbf{I}_n)^{-1}$ respectively.

After giving a calculation method for $\mathcal{I}_{ntk}(\mathbf{x}_i,\mathbf{x}_{te})$, we show that $\mathcal{I}_{ntk}(\mathbf{x}_i,\mathbf{x}_{te})$ can be a good approximation of $\mathcal{I}_{nn}(\mathbf{x}_i,\mathbf{x}_{te})$ in the over-parameterized regime. Note that in the ridgeless regime, i.e., $\lambda = 0$, Arora et al. (2019b) rigorously showed the equivalence between a fully-trained neural network and its corresponding kernel predictor. However, in the kernel ridge regime, i.e., $\lambda > 0$, there is no rigorous result so far. In this paper, we propose the following theorem which reveals the equivalence between the two-layer fully-trained wide ReLU neural network f_{nn} and its corresponding NTK predictor f_{ntk} in the ridge regression regime.

Theorem 1 *Suppose* $0 < \lambda < n^{\frac{1}{2}}$, $\kappa = \mathcal{O}\left(\frac{\sqrt{\delta}\lambda\epsilon}{n}\right)$ *and* $m = \Omega\left(\frac{n^8}{\kappa^2\epsilon^2\delta^4\lambda^6}\right)$, *then for any* $\mathbf{x}_{te}\in\mathbb{R}^d$ *with* $\|\mathbf{x}_{te}\|_2 = 1$, *with probability at least* $1-\delta$ *over random initialization, we have:*

$$|f_{nn}(\mathbf{x}_{te}) - f_{ntk}(\mathbf{x}_{te})| = \mathcal{O}(\epsilon). \tag{10}$$

For the proof of Theorem 1, we first analyze the evolution of $\mathbf{W}(t)$ and prove the perturbation of parameters are small during training. Then we follow the ideas from (Du et al. 2019b; Arora et al. 2019a) to prove the perturbation of the kernel matrix can be bounded by the perturbation of parameters during training, and build the equivalence between the wide neural network and its linearization. After that, we borrow the lemma from (Arora et al. 2019a) to establish the equivalence between the linearization of neural network and NTK predictor, then the theorem can be proved via triangle inequality. See Appendix B for the proof.

After proving Theorem 1, the approximation error can be evaluated via simple analysis, which is shown in the following theorem. See Appendix B for the proof.

Theorem 2 *Suppose* $0 < \lambda < n^{\frac{1}{2}}$, $\kappa = \mathcal{O}\left(\frac{\sqrt{\delta}\lambda\epsilon}{n}\right)$, *and* $m = \Omega\left(\frac{n^8}{\kappa^2\epsilon^2\delta^4\lambda^6}\right)$, f_{nn} *and* f_{ntk} *are uniformly bounded over the unit sphere by a constant* C, *i.e.* $|f_{nn}(\mathbf{x})| < C$ *and* $|f_{ntk}(\mathbf{x})| < C$ *for all* $\|\mathbf{x}\|_2 = 1$. *Then for any training point* $\mathbf{x}_i \in \mathcal{X}$ *and test point* $\mathbf{x}_{te}\in\mathbb{R}^d$ *with* $\|\mathbf{x}_{te}\|_2 = 1$, *with probability at least* $1-\delta$ *over random initialization, we have:*

$$|\mathcal{I}_{nn}(\mathbf{x}_i,\mathbf{x}_{te}) - \mathcal{I}_{ntk}(\mathbf{x}_i,\mathbf{x}_{te})| = \mathcal{O}(\epsilon). \tag{11}$$

Theorem 2 reveals that the IF calculated in the NTK regime can be arbitrarily close to the actual ones with high probability as long as the hidden layer is sufficiently wide. We compare the IFs calculated in the NTK regime and the ones obtained via leave-one-out retraining to verify our theory. In particular, we evaluate our method on MNIST (Lecun et al. 1998) and CIFAR-10 (Krizhevsky and Hinton 2009) for two-layer ReLU neural networks with the width from 10^4 to 8×10^4 respectively. Details of experimental settings and more experiments can be seen in the Appendix A. The numerical results are shown in Figure 1 and Table 1 respectively. We find that the predicted IFs are highly close to the actual ones, with the Pearson correlation coefficient (R) and Spearman's rank correlation coefficient (ρ) greater than 0.90

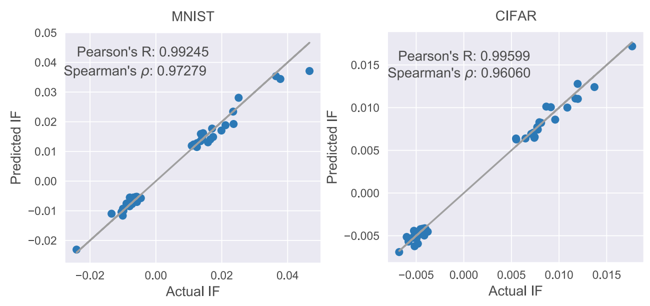


Figure 1: Comparison of predicted IFs and actual ones on the 40 most influential training points for a two-layer ReLU networks with 8×10^4 neurons in the hidden layer.

MNIST				
Width (10^4)	8	4	2	1
Pearson's R	0.9925	0.9866	0.9802	0.9766
Spearman's ρ	0.9728	0.9565	0.9531	0.9341
CIFAR				
Width (10^4)	8	4	2	1
Pearson's R	0.9960	0.9904	0.9808	0.9683
Spearman's ρ	0.9606	0.9146	0.8672	0.8583

Table 1: The correlation coefficients between actual and predicted IFs on MNIST and CIFAR respectively.

in general, and the approximate accuracy is increasing with the width of neural networks, which is consistent with Theorem 2.

Error Analysis for the IHVP Method

In this section, we aim to analyze the approximation error of the IHVP method (Koh and Liang 2017) when calculating the IF in the over-parameterized regime. Our analysis also reveals when and why IHVP can be accurate or not, which theoretically explains the phenomenon that the regularization term controls the estimation accuracy of IFs proposed in (Basu, Pope, and Feizi 2021), and also brings new insights into the understanding of IHVP.

The IHVP Method in Over-parameterized Regime

Formally, we utilize $\hat{\mathcal{I}}_{nn}(\mathbf{x}_i, \mathbf{x}_{te})$ to denote the IF calculated by IHVP, which is defined as follows:

$$\hat{\mathcal{I}}_{nn}(\mathbf{x}_i, \mathbf{x}_{te}) = \text{vec}\left(\frac{\partial \ell(\mathbf{x}_{te})}{\partial \mathbf{W}}\right)^\top H(\infty)^{-1} \text{vec}\left(\frac{\partial \ell(\mathbf{x}_i)}{\partial \mathbf{W}}\right), \tag{12}$$

where $\ell(\mathbf{x}) \triangleq \frac{1}{2}(f_{nn}(\mathbf{x}) - y)^2$ and $H(\infty) \triangleq \sum_{i=1}^n \nabla^2_{\mathbf{W}} \ell(\mathbf{x}_i; \mathbf{W}(\infty)) + \lambda \mathbf{I}_{md}$ denote the mean-square loss and the Hessian matrix at the end of training respectively. Remarkably, the calculation of Equation (12) is infeasible due to the size of modern neural networks. To solve this problem, Koh and Liang (2017) and Chen et al. (2021) utilized stochastic estimation or hypergradient-based optimization to calculate Equation (12) respectively. However, the gap between $\hat{\mathcal{I}}_{nn}(\mathbf{x}_i, \mathbf{x}_{te})$ and the actual IF is inevitable in general, and cannot be avoided by these methods. Thus in this work, we only analyze the error between $\hat{\mathcal{I}}_{nn}(\mathbf{x}_i, \mathbf{x}_{te})$ and the actual IF in the over-parameterized regime. The following proposition shows that $\hat{\mathcal{I}}_{nn}(\mathbf{x}_i, \mathbf{x}_{te})$ tends to $\hat{\mathcal{I}}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})$ when the width m goes to infinity, where $\hat{\mathcal{I}}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})$ denotes the IF calculated via IHVP method in the NTK regime.

Proposition 1 *For any $\mathbf{x}_{te} \in \mathbb{R}^d$ with $\|\mathbf{x}_{te}\|_2 = 1$, let the width $m \to \infty$, then with probability arbitrarily close to 1 over random initialization, we have:*

$$\begin{aligned}\hat{\mathcal{I}}_{nn}(\mathbf{x}_i, \mathbf{x}_{te}) \to& \alpha(\mathbf{x}_i, \mathbf{x}_{te})(f_{ntk}(\mathbf{x}_{te}) - y_{te})(f_{ntk}(\mathbf{x}_i) - y_i) \\ \triangleq& \hat{\mathcal{I}}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te}),\end{aligned} \tag{13}$$

where $\alpha(\mathbf{x}_i, \mathbf{x}_{te}) \triangleq (\mathbf{K}^\infty_{te})^\top (\mathbf{K}^\infty_{tr} + \lambda \mathbf{I}_n)^{-1} \mathbf{e}_i$.

For simplicity of analysis, we rewrite $\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})$ as follows:

Proposition 2 *For any $\mathbf{x}_{te} \in \mathbb{R}^d$ with $\|\mathbf{x}_{te}\|_2 = 1$ and $\mathbf{x}_i \in \mathcal{X}$, we have:*

$$\begin{aligned}\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te}) =& \underbrace{\alpha(\mathbf{x}_i, \mathbf{x}_{te})(f_{ntk}(\mathbf{x}_{te}) - y_{te})(f^{\setminus i}_{ntk}(\mathbf{x}_i) - y_i)}_{I} \\ &+ \underbrace{\frac{1}{2}\alpha(\mathbf{x}_i, \mathbf{x}_{te})^2 (f^{\setminus i}_{ntk}(\mathbf{x}_i) - y_i)^2}_{II}.\end{aligned} \tag{14}$$

The proof of these two propositions can be seen in the Appendix C. It is worth mentioning that the expectation of $|f_{ntk}(\mathbf{x}_{te}) - y_{te}|$ and $\left|f^{\setminus i}_{ntk}(\mathbf{x}_i) - y_i\right|$ both represent the generalization error of the model (Elisseeff, Pontil et al. 2003). And $\alpha(\mathbf{x}_i, \mathbf{x}_{te})$ represents the coefficient corresponding to $\mathbf{x}_i$ when projecting $\mathbf{J}(\mathbf{x}_{te})$ onto $\mathbf{J}(\mathcal{X})$, where $\mathbf{J}(\cdot)$ denotes the feature map in NTK. In general, $\alpha(\mathbf{x}_i, \mathbf{x}_{te})$ is distinctly smaller than 1. Thus we have $|\text{term I}| \gg |\text{term II}|$, which means it is reasonable to approximate $\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})$ as follows:

$$\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te}) \approx \alpha(\mathbf{x}_i, \mathbf{x}_{te})(f_{ntk}(\mathbf{x}_{te}) - y_{te})(f^{\setminus i}_{ntk}(\mathbf{x}_i) - y_i). \tag{15}$$

By Comparing Equation (15) with Equation (13), we can obtain that the main approximation error is caused by the variance between $f^{\setminus i}_{ntk}(\mathbf{x}_i)$ and $f_{ntk}(\mathbf{x}_i)$, which plays a key role when bounding the approximation error of IHVP. In the following two subsections, we reveal why and how the regularization term and the probability density of the corresponding training points control the approximation error.

The Effects of Regularization

Although previous literature (Basu, Pope, and Feizi 2021) has pointed out that the regularization term is essential to get high-quality IF estimates via numerical experiments, this phenomenon is not well-understood for neural networks in theory. In this subsection, we prove that the lower bound of the approximation error is controlled by the regularization parameter λ, and the least eigenvalue of NTK also plays a

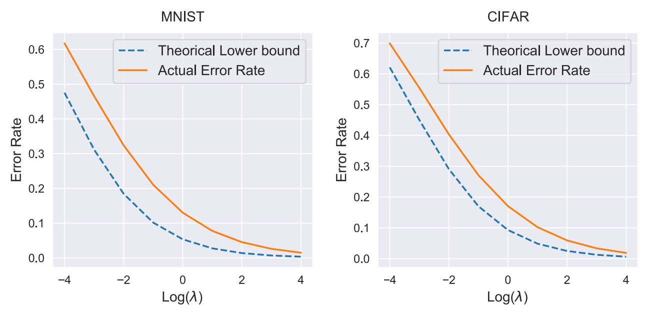


Figure 2: Comparison of the theoretical lower bound of error rates and the actual error rates for different λ on MNIST and CIFAR respectively.

key role. The following theorem reveals the relationship between the approximation error and the regularization term for over-parameterized neural networks. See Appendix C for the proof.

Theorem 3 *Given $\mathbf{x}_i \in \mathcal{X}$ and $\mathbf{x}_{te} \in \mathbb{R}^d$ with $\|\mathbf{x}_{te}\|_2 = 1$, we have:*

$$\left|\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te}) - \hat{\mathcal{I}}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})\right| \geq \frac{\lambda_{min}}{\lambda_{min} + \lambda}|\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})| - \frac{1}{2}\alpha(\mathbf{x}_i, \mathbf{x}_{te})^2(f_{ntk}^{\setminus i}(\mathbf{x}_i) - y_i)^2, \tag{16}$$

where λ_{min} is the least eigenvalue of $\mathbf{K}_{tr}^{\infty}$. Furthermore, if $|\alpha(\mathbf{x}_i, \mathbf{x}_{te})| \leq \sqrt{\frac{\gamma}{n}}\|\boldsymbol{\alpha}(\mathbf{x}_{te})\|_2$ for some $\gamma > 0$, where $\boldsymbol{\alpha}(\mathbf{x}_{te}) \triangleq (\mathbf{K}_{te}^{\infty})^\top(\mathbf{K}_{tr}^{\infty} + \lambda\mathbf{I}_n)^{-1}$, and $\mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[(f_{ntk}(\mathbf{x}) - y)^2] = \mathcal{O}(\sqrt{1/n})$, then with probability at least $1 - \delta$ we have:

$$\left|\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te}) - \hat{\mathcal{I}}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})\right| \geq \frac{\lambda_{min}}{\lambda_{min} + \lambda}|\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})| - \mathcal{O}(\frac{\gamma}{\delta\lambda n^{3/2}}). \tag{17}$$

Remark: Similar to Theorem 3, we can give a upper bound of the error rate controlled by $\frac{\lambda_{max}}{\lambda_{max}+\lambda}$, where λ_{max} is the largest eigenvalue of $\mathbf{K}_{tr}^{\infty}$. However, λ_{max} is of order $\mathcal{O}(n)$ (Li, Soltanolkotabi, and Oymak 2020), hence the term $\frac{\lambda_{max}}{\lambda_{max}+\lambda}$ will close to 1 and be meaningless in general. Arora et al. (2019a) proved that the generalization error $\mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[(f_{ntk}(\mathbf{x}) - y)^2]$ is in the order of $\mathcal{O}(\sqrt{1/n})$ for two-layer ReLU networks when the data is generated by certain functions in Theorem 5.1. Hence this assumption is reasonable.

Theorem 3 reveals that the error rate can be lower bounded by $\frac{\lambda_{min}}{\lambda_{min}+\lambda}$. To verify our lower bound, we compare the theoretical lower bound of error rates with the mean actual error rates for different λ, ranging from 2^{-4} to 2^4 respectively. Numerical experiment results reveal our bound can reflect the performance of IHVP well in the NTK regime (Figure 2).

The Effects of Training Points

In this subsection, we show that the approximation error is highly related to the probability density of corresponding training point, which has not been clearly revealed in previous literature. To model this phenomenon, we firstly assume the data sampled from the following finite mixture distribution.

Definition 1 *Consider the data is generated from a finite mixture distribution $\mathcal{D}$ as follows. There are K unknown distributions $\{\mathcal{D}_k\}_{k=1}^K$ over $\mathbb{R}^d$ with probabilities $\{p_k\}_{k=1}^K$ respectively, such that $\sum_{k=1}^K p_k = 1$. Assume we sample n_k data points from $\mathcal{D}_k$ respectively such that $\sum_{k=1}^K n_k = n$, and let $\mathcal{X}_k$ be the training data sampled from D_k. Let us define the support of a distribution $\mathcal{D}$ with density p over $\mathbb{R}^d$ as $\operatorname{supp}(\mathcal{D}) \triangleq \{x : p(x) > 0\}$, and the radius of a distribution $\mathcal{D}$ over $\mathbb{R}^d$ as $r(\mathcal{D}) \triangleq \max\{\|\mathbf{x} - \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[\mathbf{z}]\|_2 : \mathbf{x} \in \operatorname{supp}(\mathcal{D})\}$. Then we make the following assumptions about the data.*

***(A1)** (Sample Proportion) We have $n_k = p_k \cdot n$ for all $k \in [K]$.*

***(A2)** (Uniformly Bounded) There exists a constant $r > 0$, such that $r(\mathcal{D}_k) < r$ for all $k \in [K]$.*

The above assumptions of data follow that of the previous works in (Li and Liang 2018; Dong et al. 2019; Li, Soltanolkotabi, and Oymak 2020). We prove the following theorem which reveals the relationship between the approximation error for IFs of $\mathbf{x}_i \in \mathcal{X}_k$ and its corresponding probability density p_k. See Appendix C for the proof.

Theorem 4 *Consider a dataset $(\mathcal{X}, \mathcal{Y}) = \{\mathbf{x}_i, y_i\}_{i=1}^n$ generated from the finite mixture model described in Definition 1. Let $\boldsymbol{\alpha}(\mathbf{x}_i) \triangleq (\mathbf{K}_{tr}^{\infty})^\top(\mathbf{K}_{tr}^{\infty} + \lambda\mathbf{I}_n)^{-1}\mathbf{e}_i$. Suppose $|[\boldsymbol{\alpha}(\mathbf{x}_i)]_i| \leq \sqrt{\frac{\gamma}{n}}\|\boldsymbol{\alpha}(\mathbf{x}_i)\|_2$ for some $\gamma > 0$, and $\lambda > \frac{\sqrt{2}\lambda_{max}\epsilon_r}{1-\sqrt{2}\epsilon_r}$, where $\epsilon_r^2 \triangleq 2r^2 + \arccos(1 - 2r^2)$ is a small constant. Then for $\mathbf{x}_i \in \mathcal{X}_k$ and $\mathbf{x}_{te} \sim \mathcal{D}$, we have:*

$$\left|\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te}) - \hat{\mathcal{I}}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})\right| \leq \underbrace{\sqrt{\frac{\gamma}{n^2 p_k}}|\mathcal{I}_{ntk}(\mathbf{x}_i, \mathbf{x}_{te})|}_{I} + \underbrace{\frac{1}{2}\alpha(\mathbf{x}_i, \mathbf{x}_{te})^2(f_{ntk}^{\setminus i}(\mathbf{x}_i) - y_i)^2}_{II}. \tag{18}$$

Remark: Notice that term I decreases with p_k, and term II represents the leave-one-out error which also decreases with p_k in general. In the theorem, we require $\lambda > \frac{\sqrt{2}\lambda_{max}\epsilon_r}{1-\sqrt{2}\epsilon_r}$, which can be well controlled when r is small, and our numerical experiments show the phenomenon is founded whether λ is large or small.

We can see that the approximation error rate of the classic IHVP method has a significant correlation with the probability density of corresponding training points no matter λ is large or not, which confirms our conclusion in the NTK regime (Figure 3). More experiments on simulated data can be seen in the Appendix A.

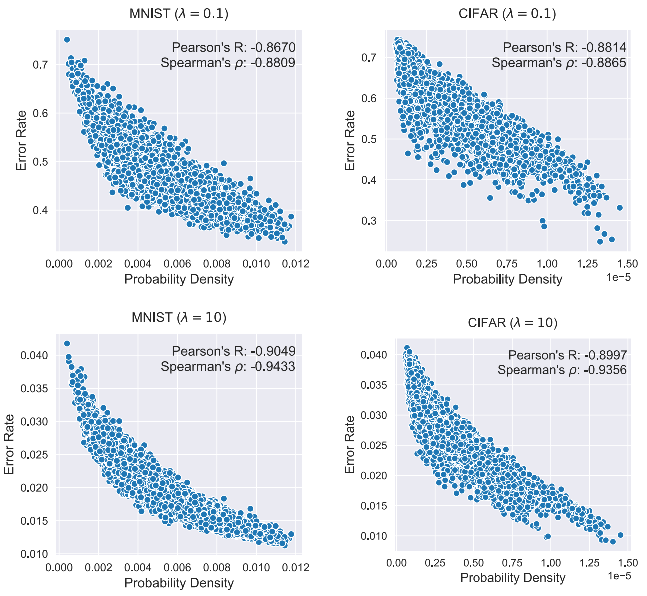
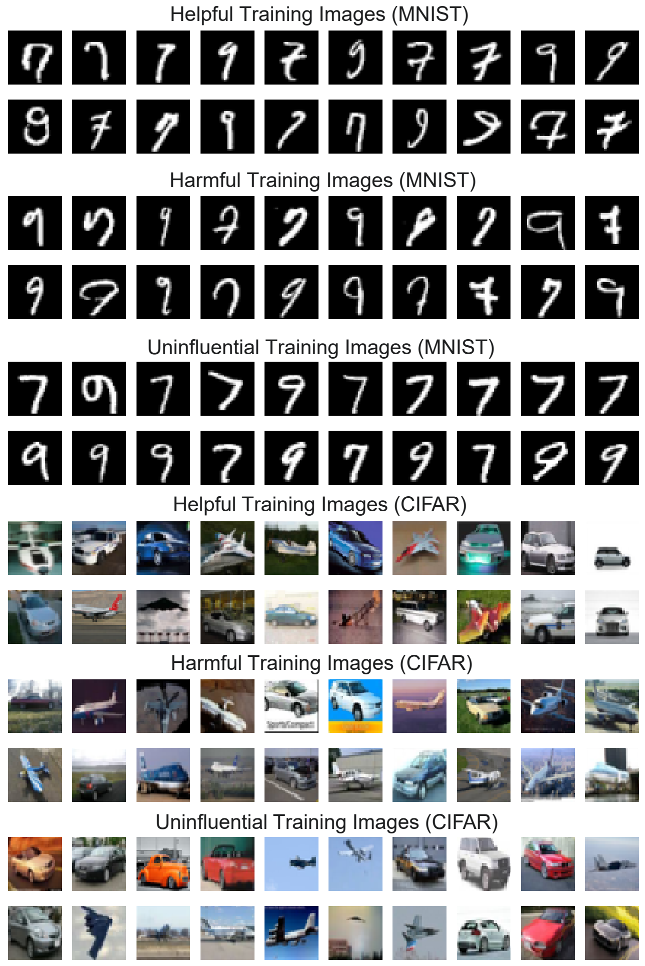


Figure 3: The probability density of training data and its corresponding error rate of estimating the IFs via IHVP on MNIST and CIFAR, respectively. Density for training data is calculated through the Gaussian kernel density estimate method.



Helpful Training Images (MNIST)
Harmful Training Images (MNIST)
Uninfluential Training Images (MNIST)
Helpful Training Images (CIFAR)
Harmful Training Images (CIFAR)
Uninfluential Training Images (CIFAR)

Figure 4: The most helpful, harmful and uninfluential images in MNIST and CIFAR respectively.

Towards Understanding of IFs

The NTK theory can not only be utilized to calculate the IFs and estimate the approximation error, but also help to understand IFs better from two new views. On the one hand, we can quantify the complexity for the influential samples through the complexity metric in the NTK theory. On the other hand, we can trace the influential samples during the training process through the linear ODE depicting the training dynamics.

Quantify the Complexity for Influential Samples

After calculating the IF for every training point, one natural question is that what is the difference between influential samples and uninfluential ones. It is interesting to see in Figure 4 that the uninfluential samples seem to be homogenized, while the most influential ones tend to be more complicated. For instance, planes in the uninfluential groups have the backgrounds of daylight or airport in general, while the backgrounds in the influential groups tend to be more diverse, such as grassland, dusk, midnight. To quantify this phenomenon, we borrow the complexity metric in NTK theory and define the complexity of $\mathcal{X}_{\mathcal{I}} \subset \mathcal{X}$ as follows:

$$\mathcal{C}(\mathcal{X}_{\mathcal{I}}) \triangleq \sqrt{\mathcal{Y}^\top (\mathbf{K}_{tr}^{\infty})^{-1}\mathcal{Y}} - \sqrt{(\mathcal{Y}^{\setminus\mathcal{I}})^\top (\mathbf{K}_{tr}^{\setminus\mathcal{I}})^{-1}\mathcal{Y}^{\setminus\mathcal{I}}}, \tag{19}$$

where $\mathcal{Y}^{\setminus\mathcal{I}}$ and $\mathbf{K}_{tr}^{\setminus\mathcal{I}}$ denote the label sets and kernel matrix constituted without $\mathcal{X}_{\mathcal{I}}$ respectively. On the one hand, for a function $f(\mathbf{x}) = \sum_{i=1}^{n} \beta_i k(\mathbf{x}, \mathbf{x}_i)$ in the reproducing kernel Hilbert space (RKHS) $\mathbb{H}$, its RKHS norm is $\|f\|_{\mathbb{H}} = \sqrt{\boldsymbol{\beta}^T \mathbf{K}\boldsymbol{\beta}}$ (Mohri, Rostamizadeh, and Talwalkar 2018). Hence for $f_{ntk}(\mathbf{x})$ in the NTK regime, we have $\|f_{ntk}\|_{\mathbb{H}} = \sqrt{\mathcal{Y}^\top (\mathbf{K}_{tr}^{\infty})^{-1}\mathcal{Y}}$, and $\mathcal{C}(\mathcal{X}_{\mathcal{I}})$ actually denotes the increment of RKHS norm contributed from the training data $\mathcal{X}_{\mathcal{I}}$. On the other hand, $\sqrt{\mathcal{Y}^\top (\mathbf{K}_{tr}^{\infty})^{-1}\mathcal{Y}}$ controls the upper bound of Rademacher complexity for a class of neural networks, and thus controls the test error (Arora et al. 2019a).

Next, we explore the relationship between the complexity and the IF for each training point. In detail, we divide the training set into ten groups sorted by their IFs, from the most harmful groups to the most helpful groups. Then we calculate $\mathcal{C}(\mathcal{X}_{\mathcal{I}})$ for each group, and it is interesting to see that the more influential data make the model more complicated. Two most influential groups (Group 0 and 9) contribute the most to the model complexity (Figure 5). The results indicate that the most helpful data increase the complexity and the generalization ability of the model. In contrast, the most harmful data increase the complexity of the model but hurts the generalization.

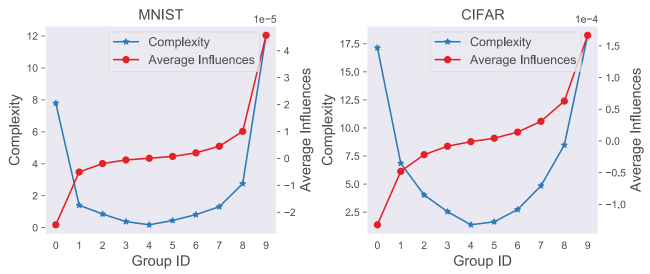


Figure 5: The relationship between complexity and influence of each group divided by their IFs on MNIST and CIFAR respectively. Notice that Group 0 and 9 denote the most harmful and helpful groups respectively, which contribute the most to the model complexity.

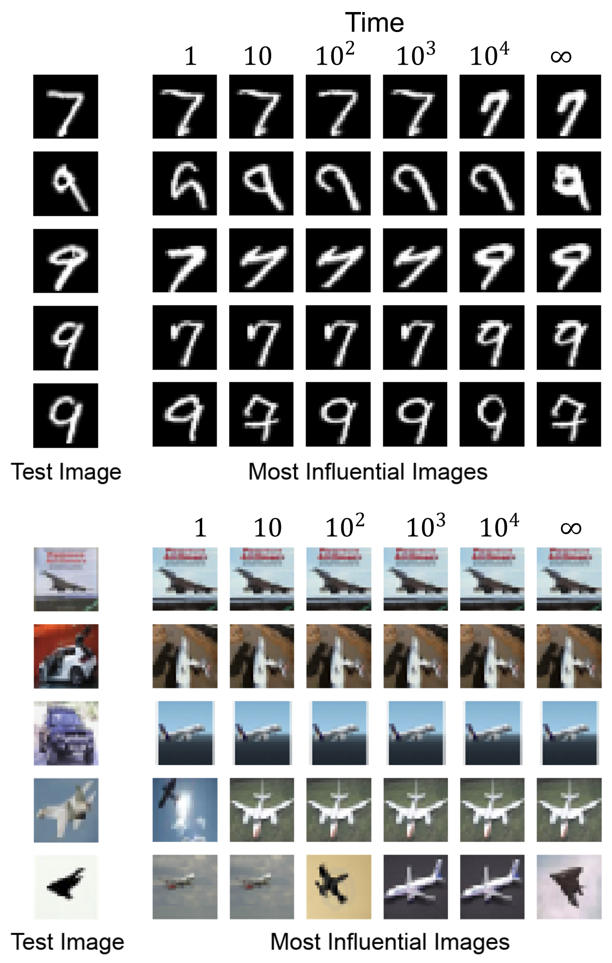


Figure 6: Variation of the most influential images for some test data on MNIST and CIFAR during the training dynamics.

Track Influential Samples During Training

Different with IHVP, which can only calculate the IF at the end of training, the NTK theory depicts the training dynamics via the following linear ODE:

$$f_{ntk}(\mathbf{x}_{te};t) \\ =(\mathbf{K}_{te}^{\infty})^{\top}(\mathbf{K}_{tr}^{\infty})^{-1}\left(\mathbf{I}_n-\exp\left(-\frac{2t}{n}\mathbf{K}_{tr}^{\infty}\right)\right)\mathcal{Y}, \tag{20}$$

hence we can track influential samples efficiently during the training process. The most influential images for certain test points do not keep constant in general (Figure 6). Considering the variation of IFs in the different processes can help to understand the model behavior better. For instance, we track the most influential images for every test point in the presence of label noise during the training process and record the proportion of the clean and noise data in the most influential data in Figure 7 respectively.

At the beginning of the training process, most test images are affected by the clean samples (Figure 7). However, as the training progresses, the influence of noise samples begin to dominate the training, which means that the model begins to learn the noise data. After that, the model has learned the label noise, thus the influence of the noise samples begin to decrease gradually. Numerical experiments reveal the variation of the most influential samples in the presence of label noise and help us understand why early-stopping can prevent over-fitting from the perspective of IFs (Figure 7).

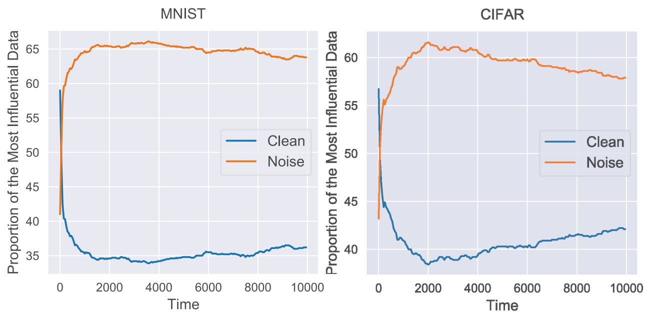


Figure 7: The proportion of the most influential images for 1000 test images on MNIST and CIFAR with the presence of label noise during the training dynamics. We flip the labels of 40% training data to simulate the label noise.

Conclusion

In this paper, we calculate the IFs for over-parameterized neural networks and utilize the NTK theory to understand when and why the IHVP method fails or not. At last, we quantify the complexity and track the variation of influential samples. Our research can help understand IFs better in the regime of neural networks and bring new insights for explaining the training process through IFs. Some future works are mentioned as follows: (1) Explore the approximation error caused by stochastic estimation in IHVP; (2) Generalize our results in Theorem 2 to broad scenarios, such as deep neural networks, convolutional neural networks and non-convex loss functions; (3) Track the IFs to reveal what does neural networks learn during training dynamics.

Acknowledgments

We thank reviewers for their helpful advice. Rui Zhang would like to thank Xingbo Du for his valuable and detailed feedback to improve the writing. This work has been supported by the National Key Research and Development Program of China [2019YFA0709501]; the Strategic Priority Research Program of the Chinese Academy of Sciences (CAS) [XDA16021400, XDPB17]; the National Natural Science Foundation of China [61621003]; the National Ten Thousand Talent Program for Young Top-notch Talents.

References

Allen-Zhu, Z.; Li, Y.; and Song, Z. 2019. A Convergence Theory for Deep Learning via Over-Parameterization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 242–252. PMLR.

Arora, S.; Du, S.; Hu, W.; Li, Z.; and Wang, R. 2019a. Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 322–332. PMLR.

Arora, S.; Du, S. S.; Hu, W.; Li, Z.; Salakhutdinov, R. R.; and Wang, R. 2019b. On Exact Computation with an Infinitely Wide Neural Net. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Basu, S.; Pope, P.; and Feizi, S. 2021. Influence Functions in Deep Learning Are Fragile. In *International Conference on Learning Representations*.

Cao, Y.; and Gu, Q. 2019. Generalization Bounds of Stochastic Gradient Descent for Wide and Deep Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Chen, Y.; Li, B.; Yu, H.; Wu, P.; and Miao, C. 2021. HyDRA: Hypergradient Data Relevance Analysis for Interpreting Deep Neural Networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, 7081–7089. AAAI Press.

Csató, L.; and Opper, M. 2002. Sparse On-Line Gaussian Processes. *Neural Computation*, 14(3): 641–668.

Dong, B.; Hou, J.; Lu, Y.; and Zhang, Z. 2019. Distillation $\approx$ Early Stopping? Harvesting Dark Knowledge Utilizing Anisotropic Information Retrieval for Overparameterized Neural Network. *arXiv preprint arXiv:1910.01255*.

Du, S. S.; Hou, K.; Salakhutdinov, R. R.; Poczos, B.; Wang, R.; and Xu, K. 2019a. Graph Neural Tangent Kernel: Fusing Graph Neural Networks with Graph Kernels. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Du, S. S.; Zhai, X.; Poczos, B.; and Singh, A. 2019b. Gradient Descent Provably Optimizes Over-parameterized Neural Networks. In *International Conference on Learning Representations*.

Elisseeff, A.; Pontil, M.; et al. 2003. Leave-One-Out Error and Stability of Learning Algorithms with Applications. *NATO Science Series III: Computer and Systems Sciences*, 190: 111–130.

Engel, Y.; Mannor, S.; and Meir, R. 2004. The Kernel Recursive Least-Squares Algorithm. *IEEE Transactions on Signal Processing*, 52(8): 2275–2285.

Hampel, F. R. 1974. The Influence Curve and its Role in Robust Estimation. *Journal of the American Statistical Association*, 69(346): 383–393.

Hu, W.; Li, Z.; and Yu, D. 2020. Simple and Effective Regularization Methods for Training on Noisily Labeled Data with Generalization Guarantee. In *International Conference on Learning Representations*.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Koh, P. W.; and Liang, P. 2017. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 1885–1894. PMLR.

Koh, P. W. W.; Ang, K.-S.; Teo, H.; and Liang, P. S. 2019. On the Accuracy of Influence Functions for Measuring Group Effects. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Kong, Z.; and Chaudhuri, K. 2021. Understanding Instance-Based Interpretability of Variational Auto-Encoders. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Krizhevsky, A.; and Hinton, G. 2009. Learning Multiple Layers of Features from Tiny Images. *Master's thesis, Dept. of Comp. Sci., University of Toronto*.

Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Lee, J.; Xiao, L.; Schoenholz, S.; Bahri, Y.; Novak, R.; Sohl-Dickstein, J.; and Pennington, J. 2019. Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Li, M.; Soltanolkotabi, M.; and Oymak, S. 2020. Gradient Descent with Early Stopping is Provably Robust to Label Noise for Overparameterized Neural Networks. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, 4313–4324. PMLR.

Li, Y.; and Liang, Y. 2018. Learning Overparameterized Neural Networks via Stochastic Gradient Descent on Structured Data. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Li, Z.; Wang, R.; Yu, D.; Du, S. S.; Hu, W.; Salakhutdinov, R.; and Arora, S. 2019. Enhanced Convolutional Neural Tangent Kernels. *arXiv preprint arXiv:1911.00809*.

Mohri, M.; Rostamizadeh, A.; and Talwalkar, A. 2018. *Foundations of Machine Learning*. MIT Press.

Nguyen, Q. N.; and Mondelli, M. 2020. Global Convergence of Deep Networks with One Wide Layer Followed by Pyramidal Topology. In *Advances in Neural Information Processing Systems*, volume 33, 11961–11972. Curran Associates, Inc.

Novak, R.; Xiao, L.; Hron, J.; Lee, J.; Alemi, A. A.; Sohl-Dickstein, J.; and Schoenholz, S. S. 2020. Neural Tangents: Fast and Easy Infinite Neural Networks in Python. In *International Conference on Learning Representations*.

Wang, H.; Ustun, B.; and Calmon, F. 2019. Repairing without Retraining: Avoiding Disparate Impact with Counterfactual Distributions. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 6618–6627. PMLR.

Wang, T.; Huan, J.; and Li, B. 2018. Data Dropout: Optimizing Training Data for Convolutional Neural Networks. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, 39–46. IEEE.

Xie, B.; Liang, Y.; and Song, L. 2017. Diverse Neural Network Learns True Target Functions. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, 1216–1224. PMLR.

Zhang, L.; Deng, Z.; Kawaguchi, K.; Ghorbani, A.; and Zou, J. 2021. How Does Mixup Help With Robustness and Generalization? In *International Conference on Learning Representations*.