Kernel-Based Smoothness Analysis of Residual Networks

Tom Tirer [1] Joan Bruna [2] Raja Giryes [1]

[1] Tel Aviv University

[2] New York University

MSML21: Mathematical and Scientific Machine Learning

Overview

Introduction and Motivation: MLP and ResNet

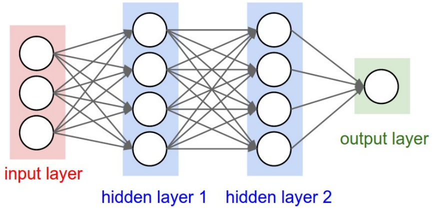
- Deep neural networks have led to a major improvement in various fields. The advance in the network performance is tightly related to the introduction of various novel architectures.
- One classical architectures is the multilayer perceptron (MLP), a plain feed-forward network with fully connected layers.
- Example: an MLP with $L = 2$ hidden layers

Introduction and Motivation: MLP and ResNet

- Deep neural networks have led to a major improvement in various fields. The advance in the network performance is tightly related to the introduction of various novel architectures.
- One classical architectures is the multilayer perceptron (MLP), a plain feed-forward network with fully connected layers.
- Example: an MLP with $L = 2$ hidden layers

Introduction and Motivation: MLP and ResNet

- Deep neural networks have led to a major improvement in various fields. The advance in the network performance is tightly related to the introduction of various novel architectures.
- One classical architectures is the multilayer perceptron (MLP), a plain feed-forward network with fully connected layers.
- Example: an MLP with $L = 2$ hidden layers



Introduction and Motivation: MLP and ResNet

Mathematical model of an MLP with L hidden layers, input $\boldsymbol{x} \in \mathbb{R}^d$, parameter vector $\boldsymbol{\theta} := \mathrm{vec}(\boldsymbol{w}^{(L+1)}, \{\boldsymbol{W}^{(\ell)}\})$, and output $f(\boldsymbol{x}; \boldsymbol{\theta}) \in \mathbb{R}$

$$\begin{aligned} \boldsymbol{g}^{(\ell)} &= \frac{\sigma_w}{\sqrt{n}} \boldsymbol{W}^{(\ell)} \boldsymbol{x}^{(\ell-1)}, \qquad \ell = 1, \ldots, L \\ \boldsymbol{x}^{(\ell)} &= \phi(\boldsymbol{g}^{(\ell)}), \qquad \ell = 1, \ldots, L \\ \boldsymbol{x}^{(0)} &= \boldsymbol{x}, \qquad f(\boldsymbol{x}, \boldsymbol{\theta}) = g^{(L+1)} = \frac{\sigma_w}{\sqrt{n}} \boldsymbol{w}^{(L+1)\top} \boldsymbol{x}^{(L)}, \end{aligned} \tag{1}$$

where $\phi(\cdot)$ is an element-wise activation function, $\sigma_w > 0$ scales the standard deviation of the weights, $\boldsymbol{w}^{(L+1)} \in \mathbb{R}^n$, $\boldsymbol{W}^{(\ell)} \in \mathbb{R}^{n \times n}$, $\boldsymbol{W}^{(0)} \in \mathbb{R}^{n \times d}$, and all the weights are initialized by the standard normal distribution $w_i^{(L+1)}, W_{ij}^{(\ell)} \sim \mathcal{N}(0, 1)$.

Introduction and Motivation: MLP and ResNet

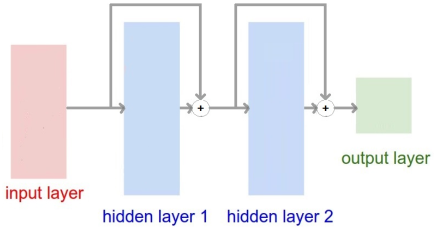
- Deep residual network (ResNet) [He et al., 2016] is a modern alternative to the classical MLP, which has led to a major leap in performance.
- ResNets use skip connections, i.e., identity paths in the network that add to the output features of a given layer its input features.
- Example: a ResNet with $L = 2$ hidden layers

Introduction and Motivation: MLP and ResNet

- Deep residual network (ResNet) [He et al., 2016] is a modern alternative to the classical MLP, which has led to a major leap in performance.
- ResNets use skip connections, i.e., identity paths in the network that add to the output features of a given layer its input features.
- Example: a ResNet with $L = 2$ hidden layers

Introduction and Motivation: MLP and ResNet

- Deep residual network (ResNet) [He et al., 2016] is a modern alternative to the classical MLP, which has led to a major leap in performance.
- ResNets use skip connections, i.e., identity paths in the network that add to the output features of a given layer its input features.
- Example: a ResNet with $L = 2$ hidden layers



Introduction and Motivation: MLP and ResNet

Mathematical model of a ResNet with L non-linear hidden layers, input $\boldsymbol{x} \in \mathbb{R}^d$, parameter vector $\boldsymbol{\theta} := \mathrm{vec}(\boldsymbol{w}^{(L+1)}, \{\boldsymbol{W}^{(\ell)}\}, \{\boldsymbol{V}^{(\ell)}\}, \boldsymbol{U})$, and output $f(\boldsymbol{x};\boldsymbol{\theta}) \in \mathbb{R}$

$$\begin{aligned} \boldsymbol{g}^{(\ell)} &= \frac{\sigma_w}{\sqrt{n}} \boldsymbol{W}^{(\ell)} \boldsymbol{x}^{(\ell-1)}, \qquad \ell = 1, \dots, L \\ \boldsymbol{x}^{(\ell)} &= \boldsymbol{x}^{(\ell-1)} + \alpha \frac{\sigma_v}{\sqrt{n}} \boldsymbol{V}^{(\ell)} \phi(\boldsymbol{g}^{(\ell)}), \qquad \ell = 1, \dots, L \\ \boldsymbol{x}^{(0)} &= \frac{1}{\sqrt{d}} \boldsymbol{U}\boldsymbol{x}, \qquad f(\boldsymbol{x},\boldsymbol{\theta}) = g^{(L+1)} = \frac{\sigma_w}{\sqrt{n}} \boldsymbol{w}^{(L+1)\top} \boldsymbol{x}^{(L)}, \end{aligned} \tag{2}$$

where $\phi(\cdot)$ is an element-wise activation function, $\sigma_w, \sigma_v > 0$ scale the standard deviation of the weights, and α is a positive hyperparameter that weighs the residual block. $\boldsymbol{w}^{(L+1)} \in \mathbb{R}^n$, $\boldsymbol{W}^{(\ell)}$, $\boldsymbol{V}^{(\ell)} \in \mathbb{R}^{n \times n}$, $\boldsymbol{U} \in \mathbb{R}^{n \times d}$, and all the weights are initialized by the standard normal distribution $w_i^{(L+1)}, W_{ij}^{(\ell)}, V_{ij}^{(\ell)}, U_{ij} \sim \mathcal{N}(0,1)$.

Introduction and Motivation: MLP and ResNet

- Different efforts were dedicated to explaining the success of ResNets
- Previous works mainly focused on the optimization advantages of ResNets over MLPs, such as overcoming the problem of vanishing gradients [Veit et al., 2016] and enjoying a better loss surface [Li et al., 2018], just to name a few.
- In different regression experiments, where the optimization of both models is optimal (without any regularization), we observed another distinction between the two ReLU-based models:
  ResNets tend to promote smoother interpolations than MLPs
- Why smoothness of the outputs is interesting?
  Under some smoothness assumptions on the g.t. function (which creates the data), several works connect better smoothness of interpolators with better generalization [Lu et al., 2019; Giryes, 2020; Xie et al., 2020].

Introduction and Motivation: MLP and ResNet

- Different efforts were dedicated to explaining the success of ResNets
- Previous works mainly focused on the optimization advantages of ResNets over MLPs, such as overcoming the problem of vanishing gradients [Veit et al., 2016] and enjoying a better loss surface [Li et al., 2018], just to name a few.
- In different regression experiments, where the optimization of both models is optimal (without any regularization), we observed another distinction between the two ReLU-based models:
  ResNets tend to promote smoother interpolations than MLPs
- Why smoothness of the outputs is interesting?
  Under some smoothness assumptions on the g.t. function (which creates the data), several works connect better smoothness of interpolators with better generalization [Lu et al., 2019; Giryes, 2020; Xie et al., 2020].

Introduction and Motivation: MLP and ResNet

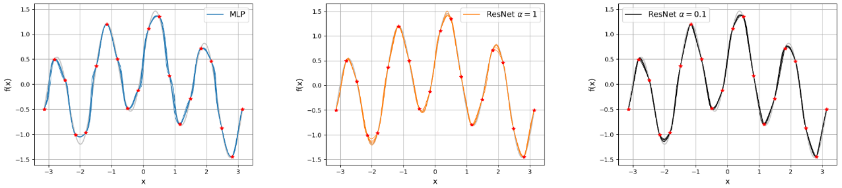
- Different efforts were dedicated to explaining the success of ResNets
- Previous works mainly focused on the optimization advantages of ResNets over MLPs, such as overcoming the problem of vanishing gradients [Veit et al., 2016] and enjoying a better loss surface [Li et al., 2018], just to name a few.
- In different regression experiments, where the optimization of both models is optimal (without any regularization), we observed another distinction between the two ReLU-based models:
  ResNets tend to promote smoother interpolations than MLPs
- Why smoothness of the outputs is interesting?
  Under some smoothness assumptions on the g.t. function (which creates the data), several works connect better smoothness of interpolators with better generalization [Lu et al., 2019; Giryes, 2020; Xie et al., 2020].

Introduction and Motivation: MLP and ResNet

- Different efforts were dedicated to explaining the success of ResNets
- Previous works mainly focused on the optimization advantages of ResNets over MLPs, such as overcoming the problem of vanishing gradients [Veit et al., 2016] and enjoying a better loss surface [Li et al., 2018], just to name a few.
- In different regression experiments, where the optimization of both models is optimal (without any regularization), we observed another distinction between the two ReLU-based models:
  ResNets tend to promote smoother interpolations than MLPs
- Why smoothness of the outputs is interesting?
  Under some smoothness assumptions on the g.t. function (which creates the data), several works connect better smoothness of interpolators with better generalization [Lu et al., 2019; Giryes, 2020; Xie et al., 2020].

Introduction and Motivation: MLP and ResNet

Empirical results:

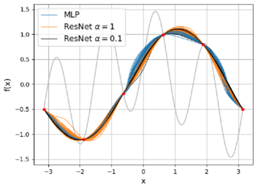


(a) Interpolation by MLP (Adam) (b) Interpolation by ResNet $\alpha = 1$ (Adam) (c) Interpolation by ResNet $\alpha = 0.1$ (Adam)

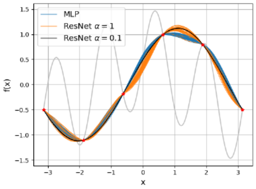
Figure 4: Empirical interpolations of MLP and ResNet (for different values of α) with $L = 5$ nonlinear layers, ReLU nonlinearities, $\sigma_v = \sigma_w = 1$, for inputs on the sphere (circle) in $\mathbb{R}^2$. We use practical models with width of $n = 500$, 5 different Xavier's random Gaussian initializations (instead of normalizations by $1/\sqrt{n}$) and 1K iterations of Adam optimizer.

Introduction and Motivation: MLP and ResNet

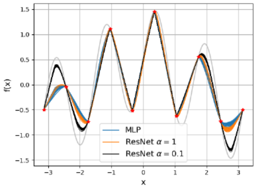
Empirical results (the difference is more visible when the number of samples decreases):



(a) Interpolation with 6 samples (Adam)



(b) Interpolation with 6 samples (SGD)



(c) Interpolation with 10 samples (SGD)

Figure 3: Empirical interpolations of MLP and ResNet (for different values of α) with $L = 5$ nonlinear layers, ReLU nonlinearities, $\sigma_v = \sigma_w = 1$, for inputs on the sphere (circle) in $\mathbb{R}^2$. We use practical models with width of $n = 500$, 30 different Xavier's Gaussian initializations (instead of normalizations by $1/\sqrt{n}$) and 1K iterations of SGD/Adam optimizers.

Background on NTK

- We choose to explore this difference between MLP and ResNet via the Neural Tangent Kernel (NTK) approach [Jacot et al., 2018].
- At initialization, when $n \to \infty$ each pre-activation $g_i^{(\ell)}(\boldsymbol{x})$, is a stochastic Gaussian Process (GP) with zero mean [Neal, 2012]. Denote the GP kernel (covariance) of this process by $K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left[g_i^{(L+1)}(\boldsymbol{x}) g_i^{(L+1)}(\tilde{\boldsymbol{x}})\right]$, we have
$$f(\boldsymbol{x}) f(\tilde{\boldsymbol{x}}) = g^{(L+1)}(\boldsymbol{x}) g^{(L+1)}(\tilde{\boldsymbol{x}}) \xrightarrow[a.s.]{n \to \infty} K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$
- Similarly, define the NTK as $\Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle$ where $n \to \infty$. Under mild conditions, we have at initialization
$$\left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle \xrightarrow[a.s.]{n \to \infty} \Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$
- The significant impact of NTK: under appropriate conditions it can characterize DNN training with gradient descent (GD).

Background on NTK

- We choose to explore this difference between MLP and ResNet via the Neural Tangent Kernel (NTK) approach [Jacot et al., 2018].
- At initialization, when $n \to \infty$ each pre-activation $g_i^{(\ell)}(\boldsymbol{x})$, is a stochastic Gaussian Process (GP) with zero mean [Neal, 2012]. Denote the GP kernel (covariance) of this process by $K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left[g_i^{(L+1)}(\boldsymbol{x}) g_i^{(L+1)}(\tilde{\boldsymbol{x}})\right]$, we have

$$f(\boldsymbol{x})f(\tilde{\boldsymbol{x}}) = g^{(L+1)}(\boldsymbol{x})g^{(L+1)}(\tilde{\boldsymbol{x}}) \xrightarrow[a.s.]{n\to\infty} K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$

- Similarly, define the NTK as $\Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle$ where $n \to \infty$. Under mild conditions, we have at initialization

$$\left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle \xrightarrow[a.s.]{n\to\infty} \Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$

- The significant impact of NTK: under appropriate conditions it can characterize DNN training with gradient descent (GD).

Background on NTK

- We choose to explore this difference between MLP and ResNet via the Neural Tangent Kernel (NTK) approach [Jacot et al., 2018].
- At initialization, when $n \to \infty$ each pre-activation $g_i^{(\ell)}(\boldsymbol{x})$, is a stochastic Gaussian Process (GP) with zero mean [Neal, 2012]. Denote the GP kernel (covariance) of this process by $K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}} \left[g_i^{(L+1)}(\boldsymbol{x}) g_i^{(L+1)}(\tilde{\boldsymbol{x}}) \right]$, we have

$$f(\boldsymbol{x})f(\tilde{\boldsymbol{x}}) = g^{(L+1)}(\boldsymbol{x})g^{(L+1)}(\tilde{\boldsymbol{x}}) \xrightarrow[a.s.]{n \to \infty} K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$

- Similarly, define the NTK as $\Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}} \left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle$ where $n \to \infty$. Under mild conditions, we have at initialization

$$\left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle \xrightarrow[a.s.]{n \to \infty} \Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$

- The significant impact of NTK: under appropriate conditions it can characterize DNN training with gradient descent (GD).

Background on NTK

- We choose to explore this difference between MLP and ResNet via the Neural Tangent Kernel (NTK) approach [Jacot et al., 2018].
- At initialization, when $n \to \infty$ each pre-activation $g_i^{(\ell)}(\boldsymbol{x})$, is a stochastic Gaussian Process (GP) with zero mean [Neal, 2012]. Denote the GP kernel (covariance) of this process by $K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left[g_i^{(L+1)}(\boldsymbol{x}) g_i^{(L+1)}(\tilde{\boldsymbol{x}})\right]$, we have

$$f(\boldsymbol{x})f(\tilde{\boldsymbol{x}}) = g^{(L+1)}(\boldsymbol{x}) g^{(L+1)}(\tilde{\boldsymbol{x}}) \xrightarrow[a.s.]{n\to\infty} K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$

- Similarly, define the NTK as $\Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle$ where $n \to \infty$. Under mild conditions, we have at initialization

$$\left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle \xrightarrow[a.s.]{n\to\infty} \Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$$

- The significant impact of NTK: under appropriate conditions it can characterize DNN training with gradient descent (GD).

Background on NTK

- Given the training data $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ and a loss function $\ell(\cdot,\cdot)$, consider learning $\boldsymbol{\theta}$ by GD minimization of

$$\mathcal{L} = \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{D}} \ell(f(\boldsymbol{x}_i; \boldsymbol{\theta}), y_i).$$

- In continuous time (for simplicity):

$$\dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^{\top} \nabla_{\boldsymbol{f}(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}$$

where $\boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t) = \mathrm{vec}(\{f(\boldsymbol{x}_i; \boldsymbol{\theta}_t)\}_{\boldsymbol{x}_i \in \mathcal{X}}) \in \mathbb{R}^{|\mathcal{X}| \times 1}$.
- In the function space, we have

$$\dot{\boldsymbol{f}}(\mathcal{X}; \boldsymbol{\theta}_t) = \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^{\top} \nabla_{\boldsymbol{f}(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}.$$

- Under appropriate NTK conditions: $\frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^{\top} \xrightarrow[a.s.]{n \longrightarrow \infty} \boldsymbol{\Theta}$ (NTK Gram matrix), where $\boldsymbol{\Theta} \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ with $\Theta_{ij} = \Theta^{(L+1)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Background on NTK

- Given the training data $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ and a loss function $\ell(\cdot,\cdot)$, consider learning $\boldsymbol{\theta}$ by GD minimization of

$$\mathcal{L} = \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{D}} \ell(f(\boldsymbol{x}_i; \boldsymbol{\theta}), y_i).$$

- In continuous time (for simplicity):

$$\dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \nabla_{\boldsymbol{f}(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}$$

  where $\boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t) = \mathrm{vec}(\{f(\boldsymbol{x}_i; \boldsymbol{\theta}_t)\}_{\boldsymbol{x}_i \in \mathcal{X}}) \in \mathbb{R}^{|\mathcal{X}| \times 1}$.

- In the function space, we have

$$\dot{\boldsymbol{f}}(\mathcal{X}; \boldsymbol{\theta}_t) = \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \nabla_{\boldsymbol{f}(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}.$$

- Under appropriate NTK conditions: $\frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \xrightarrow[a.s.]{n \longrightarrow \infty} \boldsymbol{\Theta}$ (NTK Gram matrix), where $\boldsymbol{\Theta} \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ with $\Theta_{ij} = \Theta^{(L+1)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Background on NTK

- Given the training data $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ and a loss function $\ell(\cdot, \cdot)$, consider learning $\boldsymbol{\theta}$ by GD minimization of

$$\mathcal{L} = \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{D}} \ell(f(\boldsymbol{x}_i; \boldsymbol{\theta}), y_i).$$

- In continuous time (for simplicity):

$$\dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \nabla_{\boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)} \mathcal{L}$$

where $\boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t) = \mathrm{vec}(\{f(\boldsymbol{x}_i; \boldsymbol{\theta}_t)\}_{\boldsymbol{x}_i \in \mathcal{X}}) \in \mathbb{R}^{|\mathcal{X}| \times 1}$.

- In the function space, we have

$$\dot{\boldsymbol{f}}(\mathcal{X}; \boldsymbol{\theta}_t) = \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \nabla_{\boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)} \mathcal{L}.$$

- Under appropriate NTK conditions: $\frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \xrightarrow[a.s.]{n \to \infty} \Theta$ (NTK Gram matrix), where $\Theta \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ with $\Theta_{ij} = \Theta^{(L+1)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Background on NTK

- Given the training data $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ and a loss function $\ell(\cdot,\cdot)$, consider learning $\boldsymbol{\theta}$ by GD minimization of

$$\mathcal{L} = \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}} \ell(f(\mathbf{x}_i; \boldsymbol{\theta}), y_i).$$

- In continuous time (for simplicity):

$$\dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \mathbf{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \nabla_{\mathbf{f}(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}$$

where $\mathbf{f}(\mathcal{X}; \boldsymbol{\theta}_t) = \mathrm{vec}(\{f(\mathbf{x}_i; \boldsymbol{\theta}_t)\}_{\mathbf{x}_i \in \mathcal{X}}) \in \mathbb{R}^{|\mathcal{X}| \times 1}$.

- In the function space, we have

$$\dot{\mathbf{f}}(\mathcal{X}; \boldsymbol{\theta}_t) = \frac{\partial \mathbf{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \dot{\boldsymbol{\theta}}_t = -\eta \frac{\partial \mathbf{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \mathbf{f}(\mathcal{X}; \boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \nabla_{\mathbf{f}(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}.$$

- Under appropriate NTK conditions: $\frac{\partial \mathbf{f}(\mathcal{X};\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \frac{\partial \mathbf{f}(\mathcal{X};\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}}^\top \xrightarrow[a.s.]{n \longrightarrow \infty} \boldsymbol{\Theta}$ (NTK Gram matrix), where $\boldsymbol{\Theta} \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ with $\Theta_{ij} = \Theta^{(L+1)}(\mathbf{x}_i, \mathbf{x}_j)$.

Background on NTK

- Recursive GP kernel expression for the MLP model in (1)

$$K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \sigma_w^2 T\left(\begin{bmatrix} K^{(L)}(\boldsymbol{x},\boldsymbol{x}) & K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) & K^{(L)}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}) \end{bmatrix}\right), \quad (3)$$

$$K^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{\sigma_w^2}{d}\boldsymbol{x}^\top\tilde{\boldsymbol{x}},$$

where $T(\boldsymbol{\Sigma}) := \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\boldsymbol{\Sigma})}\left[\phi(u)\phi(v)\right]$ (closed-form expression for ReLU).

- Recursive NTK expression for the MLP model in (1) [Jacot et al., 2018]

$$\Theta^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + \Theta^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \cdot \sigma_w^2 \dot{T}\left(\begin{bmatrix} K^{(L)}(\boldsymbol{x},\boldsymbol{x}) & K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) & K^{(L)}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}) \end{bmatrix}\right), \quad (4)$$

$$\Theta^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = K^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}),$$

where $\dot{T}(\boldsymbol{\Sigma}) := \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\boldsymbol{\Sigma})}\left[\phi'(u)\phi'(v)\right]$ (closed-form expression for ReLU).

Background on NTK

- Recursive GP kernel expression for the MLP model in (1)

$$K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \sigma_w^2 T\left(\begin{bmatrix} K^{(L)}(\boldsymbol{x},\boldsymbol{x}) & K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) & K^{(L)}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}) \end{bmatrix}\right), \tag{3}$$

$$K^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{\sigma_w^2}{d}\boldsymbol{x}^\top\tilde{\boldsymbol{x}},$$

where $T(\boldsymbol{\Sigma}) := \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\boldsymbol{\Sigma})}\left[\phi(u)\phi(v)\right]$ (closed-form expression for ReLU).

- Recursive NTK expression for the MLP model in (1) [Jacot et al., 2018]

$$\Theta^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + \Theta^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \cdot \sigma_w^2 \dot{T}\left(\begin{bmatrix} K^{(L)}(\boldsymbol{x},\boldsymbol{x}) & K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) & K^{(L)}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}) \end{bmatrix}\right), \tag{4}$$

$$\Theta^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = K^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}),$$

where $\dot{T}(\boldsymbol{\Sigma}) := \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\boldsymbol{\Sigma})}\left[\phi'(u)\phi'(v)\right]$ (closed-form expression for ReLU).

NTK for ResNet

- We obtain the limiting GP kernel and NTK for the ResNet model in (2). The model's structure allows us to use general results for the existence of the limits [Yang, 2019]. The results resemble those in [Huang et al., 2020], which considered a similar model but with fixed weights at the first and last layers.
- We also provide a theorem on the stability of the NTK during training (extending the technique that was used in [Lee et al., 2019] for MLP). No stability result appears in [Huang et al., 2020].

Theorem (GP kernel at initialization)

Consider the ResNet model in (2).
$\hat{K}_0^{(L+1)}(\mathbf{x}, \tilde{\mathbf{x}}) := f(\mathbf{x}; \boldsymbol{\theta}_0) f(\tilde{\mathbf{x}}; \boldsymbol{\theta}_0) \xrightarrow{n \to \infty} K^{(L+1)}(\mathbf{x}, \tilde{\mathbf{x}}) := \mathbb{E}_{\boldsymbol{\theta}} \left[f(\mathbf{x}; \boldsymbol{\theta}) f(\tilde{\mathbf{x}}; \boldsymbol{\theta}) \right]$,
where $K^{(L+1)}(\mathbf{x}, \tilde{\mathbf{x}})$ can be computed recursively as following:

$$K^{(L+1)}(\mathbf{x}, \tilde{\mathbf{x}}) = K^{(L)}(\mathbf{x}, \tilde{\mathbf{x}}) + \alpha^2 \sigma_v^2 \sigma_w^2 T \left(\begin{bmatrix} K^{(L)}(\mathbf{x}, \mathbf{x}) & K^{(L)}(\mathbf{x}, \tilde{\mathbf{x}}) \\ K^{(L)}(\mathbf{x}, \tilde{\mathbf{x}}) & K^{(L)}(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}) \end{bmatrix} \right), \quad (5)$$

$$K^{(1)}(\mathbf{x}, \tilde{\mathbf{x}}) = \frac{\sigma_w^2}{d} \mathbf{x}^\top \tilde{\mathbf{x}}.$$

NTK for ResNet

- We obtain the limiting GP kernel and NTK for the ResNet model in (2). The model's structure allows us to use general results for the existence of the limits [Yang, 2019]. The results resemble those in [Huang et al., 2020], which considered a similar model but with fixed weights at the first and last layers.
- We also provide a theorem on the stability of the NTK during training (extending the technique that was used in [Lee et al., 2019] for MLP). No stability result appears in [Huang et al., 2020].

Theorem (GP kernel at initialization)

Consider the ResNet model in (2).
$\hat{K}_0^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) := f(\boldsymbol{x};\boldsymbol{\theta}_0)f(\tilde{\boldsymbol{x}};\boldsymbol{\theta}_0) \xrightarrow{n\to\infty} K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left[f(\boldsymbol{x};\boldsymbol{\theta})f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})\right]$,
where $K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}})$ can be computed recursively as following:

$$K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + \alpha^2\sigma_v^2\sigma_w^2 T\left(\begin{bmatrix} K^{(L)}(\boldsymbol{x},\boldsymbol{x}) & K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) & K^{(L)}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}) \end{bmatrix}\right), \quad (5)$$

$$K^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{\sigma_w^2}{d}\boldsymbol{x}^\top\tilde{\boldsymbol{x}}.$$

NTK for ResNet

- We obtain the limiting GP kernel and NTK for the ResNet model in (2). The model's structure allows us to use general results for the existence of the limits [Yang, 2019]. The results resemble those in [Huang et al., 2020], which considered a similar model but with fixed weights at the first and last layers.
- We also provide a theorem on the stability of the NTK during training (extending the technique that was used in [Lee et al., 2019] for MLP). No stability result appears in [Huang et al., 2020].

Theorem (GP kernel at initialization)

Consider the ResNet model in (2).
$\hat{K}_0^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) := f(\boldsymbol{x};\boldsymbol{\theta}_0)f(\tilde{\boldsymbol{x}};\boldsymbol{\theta}_0) \xrightarrow{n\to\infty} K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}}\left[f(\boldsymbol{x};\boldsymbol{\theta})f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})\right]$,
where $K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}})$ can be computed recursively as following:

$$K^{(L+1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) + \alpha^2\sigma_v^2\sigma_w^2 T\left(\begin{bmatrix} K^{(L)}(\boldsymbol{x},\boldsymbol{x}) & K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) \\ K^{(L)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) & K^{(L)}(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}) \end{bmatrix}\right), \quad (5)$$

$$K^{(1)}(\boldsymbol{x},\tilde{\boldsymbol{x}}) = \frac{\sigma_w^2}{d}\boldsymbol{x}^\top\tilde{\boldsymbol{x}}.$$

NTK for ResNet

Theorem (NTK at initialization)

Consider the ResNet model in (2) and let the element-wise non-linearities be bounded uniformly by $\mathrm{e}^{(cx^2-\epsilon)}$ for some $c, \epsilon > 0$. We have that
$\hat{\Theta}_0^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta}_0)}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta}_0)}{\partial \boldsymbol{\theta}} \right\rangle \xrightarrow{n \longrightarrow \infty} \Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \mathbb{E}_{\boldsymbol{\theta}} \left\langle \frac{\partial f(\boldsymbol{x};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \frac{\partial f(\tilde{\boldsymbol{x}};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle$,
where $\Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ is given by

$$\begin{aligned}\Theta^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) &= K^{(L+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + \Pi^{(0)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \cdot K^{(1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \\ &\quad + \alpha^2 \sum_{\ell=1}^{L} \Pi^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \cdot \left(\Sigma^{(\ell+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) + K^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \cdot \dot{\Sigma}^{(\ell+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \right)\end{aligned} \tag{6}$$

such that

$$\Sigma^{(\ell+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \sigma_v^2 \sigma_w^2 T \left(\begin{bmatrix} K^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}) & K^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \\ K^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) & K^{(\ell)}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}) \end{bmatrix} \right), \quad \dot{\Sigma}^{(\ell+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) := \sigma_v^2 \sigma_w^2 \dot{T} \left(\begin{bmatrix} K^{(\ell)}(\boldsymbol{x}, \boldsymbol{x}) & K^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \\ K^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) & K^{(\ell)}(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}) \end{bmatrix} \right),$$

$\{K^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})\}$ are given in (5), and $\{\Pi^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}})\}$ can be computed using the following recursive expression

$$\Pi^{(\ell)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \Pi^{(\ell+1)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \left(1 + \alpha^2 \dot{\Sigma}^{(\ell+2)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \right), \quad \Pi^{(L)}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = 1.$$

NTK for ResNet

Theorem (Stability of the NTK during training)

Consider the ResNet model in (2) with activation function that satisfies some conditions (given in the paper). Assume that $\lambda_{min}(\mathbf{\Theta}) > 0$, the training set $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ is contained in some compact set and $\boldsymbol{x} \neq \tilde{\boldsymbol{x}}$ for all $\boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathcal{X}$. Then, for $\delta_0 > 0$ there exist $R_0 > 0$, N and $K > 1$, such that for every $n > N$ when applying GD with LR $\eta_0 < 2(\lambda_{min}(\mathbf{\Theta}) + \lambda_{max}(\mathbf{\Theta}))^{-1}$ on ℓ_2 loss, the following holds with probability at least $1 - \delta_0$ over the random initialization

$$\sqrt{\sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{D}} (f(\boldsymbol{x}_i; \boldsymbol{\theta}_t) - y_i)^2} \leq \left(1 - \frac{\eta_0}{3} \lambda_{min}(\mathbf{\Theta})\right)^t R_0,$$

$$\sum_{j=1}^{t} \|\boldsymbol{\theta}_j - \boldsymbol{\theta}_{j-1}\|_2 \leq \frac{3KR_0}{\lambda_{min}(\mathbf{\Theta})},$$

$$\sup_t \|\hat{\mathbf{\Theta}}_t - \hat{\mathbf{\Theta}}_0\|_F \leq \frac{6K^3 R_0}{\lambda_{min}(\mathbf{\Theta})} n^{-0.5}.$$

NTK for ResNet

Theorem (Stability of the NTK during training)

... with high probability over the random initialization

$$\sqrt{\sum_{(\boldsymbol{x}_i, y_i)\in\mathcal{D}} (f(\boldsymbol{x}_i;\boldsymbol{\theta}_t) - y_i)^2} \leq \left(1 - \frac{\eta_0}{3}\lambda_{min}(\boldsymbol{\Theta})\right)^t R_0,$$

$$\sum_{j=1}^{t} \|\boldsymbol{\theta}_j - \boldsymbol{\theta}_{j-1}\|_2 \leq \frac{3KR_0}{\lambda_{min}(\boldsymbol{\Theta})},$$

$$\sup_t \|\hat{\boldsymbol{\Theta}}_t - \hat{\boldsymbol{\Theta}}_0\|_F \leq \frac{6K^3R_0}{\lambda_{min}(\boldsymbol{\Theta})} n^{-0.5}.$$

- The first line implies convergence to zero training loss
- The second line implies stability of the weights during training (the bound on their amount of change does not depend on the network width n)
- The third line shows the stability of the NTK Gram matrix, which implies $\hat{\boldsymbol{\Theta}}_t \xrightarrow{n\to\infty} \boldsymbol{\Theta}$

NTK for ResNet

Theorem (Stability of the NTK during training)

... with high probability over the random initialization

$$\sqrt{\sum_{(\boldsymbol{x}_i, y_i)\in\mathcal{D}} (f(\boldsymbol{x}_i; \boldsymbol{\theta}_t) - y_i)^2} \leq \left(1 - \frac{\eta_0}{3}\lambda_{min}(\boldsymbol{\Theta})\right)^t R_0,$$

$$\sum_{j=1}^{t} \|\boldsymbol{\theta}_j - \boldsymbol{\theta}_{j-1}\|_2 \leq \frac{3KR_0}{\lambda_{min}(\boldsymbol{\Theta})},$$

$$\sup_t \|\hat{\boldsymbol{\Theta}}_t - \hat{\boldsymbol{\Theta}}_0\|_F \leq \frac{6K^3R_0}{\lambda_{min}(\boldsymbol{\Theta})} n^{-0.5}.$$

- The first line implies convergence to zero training loss
- The second line implies stability of the weights during training (the bound on their amount of change does not depend on the network width n)
- The third line shows the stability of the NTK Gram matrix, which implies $\hat{\boldsymbol{\Theta}}_t \xrightarrow{n\to\infty} \boldsymbol{\Theta}$

NTK for ResNet

Theorem (Stability of the NTK during training)

... with high probability over the random initialization

$$\sqrt{\sum_{(\boldsymbol{x}_i, y_i)\in\mathcal{D}} (f(\boldsymbol{x}_i;\boldsymbol{\theta}_t) - y_i)^2} \leq \left(1 - \frac{\eta_0}{3}\lambda_{min}(\boldsymbol{\Theta})\right)^t R_0,$$

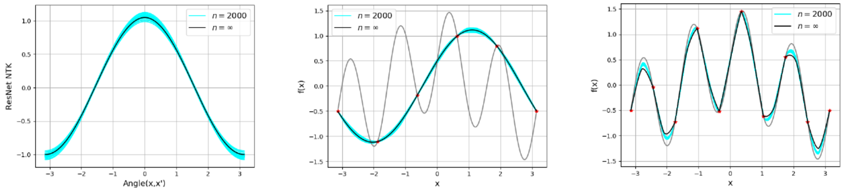
$$\sum_{j=1}^{t} \|\boldsymbol{\theta}_j - \boldsymbol{\theta}_{j-1}\|_2 \leq \frac{3KR_0}{\lambda_{min}(\boldsymbol{\Theta})},$$

$$\sup_t \|\hat{\boldsymbol{\Theta}}_t - \hat{\boldsymbol{\Theta}}_0\|_F \leq \frac{6K^3R_0}{\lambda_{min}(\boldsymbol{\Theta})} n^{-0.5}.$$

- The first line implies convergence to zero training loss
- The second line implies stability of the weights during training (the bound on their amount of change does not depend on the network width n)
- The third line shows the stability of the NTK Gram matrix, which implies $\hat{\boldsymbol{\Theta}}_t \xrightarrow{n\rightarrow\infty} \boldsymbol{\Theta}$

NTK for ResNet

Numerical ResNet NTK results:



(a) Empirical and asymptotic NTK (b) Interpolation with 6 samples (c) Interpolation with 10 samples

Figure 1: Empirical (finite width of $n = 2000$ and 30 different Gaussian initializations) and asymptotic NTK for ResNet with $L = 5$ nonlinear layers, ReLU nonlinearities, $\alpha = 0.1$, $\sigma_v = \sigma_w = 1$, for inputs on the sphere (circle) in $\mathbb{R}^2$. (1a): The kernel shape. (1b)-(1c): Interpolation using the closed-form NTK solution and using gradient descent training of the finite-width ResNet (5K iterations with lr 0.05 in (1b), and 10K iterations with lr 0.5 in (1c)).

Comparing the Smoothness of ResNet and MLP NTKs

- We compare the smoothness of the results of ResNet and MLP in the NTK regime using different evaluation methodologies.
- We start with comparing upper bounds on the norm of the models' "input-output Jacobians" $\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f(\boldsymbol{x}) \right\|_2$ *after training*, which is possible due to the NTK regime. These quantities can indicate smoothness (similarly to the way Lipschitz continuity of the gradient is used in the optimization literature).

Comparing the Smoothness of ResNet and MLP NTKs

- We compare the smoothness of the results of ResNet and MLP in the NTK regime using different evaluation methodologies.
- We start with comparing upper bounds on the norm of the models' "input-output Jacobians" $\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f(\boldsymbol{x}) \right\|_2$ *after training*, which is possible due to the NTK regime. These quantities can indicate smoothness (similarly to the way Lipschitz continuity of the gradient is used in the optimization literature).

Comparing the Smoothness of ResNet and MLP NTKs

- Starting with the Jacobians of MLP and ResNet, we have

$$\sup_{\mathbf{x}} \left\| \frac{\partial}{\partial \mathbf{x}} f_{MLP}(\mathbf{x}) \right\|_2 \leq \frac{\sigma_w}{\sqrt{n}} \|\mathbf{w}^{(L+1)}\|_2 C_\phi \frac{\sigma_w}{\sqrt{d}} \|\mathbf{W}^{(1)}\| \prod_{\ell=2}^{L} C_\phi \frac{\sigma_w}{\sqrt{n}} \|\mathbf{W}^{(\ell)}\|$$

$$\sup_{\mathbf{x}} \left\| \frac{\partial}{\partial \mathbf{x}} f_{ResNet}(\mathbf{x}) \right\|_2 \leq \frac{\sigma_w}{\sqrt{n}} \|\mathbf{w}^{(L+1)}\|_2 \frac{1}{\sqrt{d}} \|\mathbf{U}\| \cdot \prod_{\ell=1}^{L} \left(1 + \alpha C_\phi \frac{\sigma_v}{\sqrt{n}} \|\mathbf{V}^{(\ell)}\| \frac{\sigma_w}{\sqrt{n}} \|\mathbf{W}^{(\ell)}\| \right)$$

- In the NTK regime (of both ResNet and MLP) the spectral norm of the weights can be easily bounded. This is in contrast with the general case where there is no precise way to control the weights of DNNs *after* training.
- The spectral norm of, e.g., $\mathbf{W}_t^{(\ell)} \in \mathbb{R}^{n \times n}$ obeys

$$\|\mathbf{W}_t^{(\ell)}\| \leq \|\mathbf{W}_0^{(\ell)}\| + \|\mathbf{W}_t^{(\ell)} - \mathbf{W}_0^{(\ell)}\| \overset{(a)}{\leq} 2\sqrt{n} + \delta + C \overset{(b)}{\leq} 3\sqrt{n},$$

(a) holds with prob. above $1 - 2\mathrm{e}^{-\delta^2/2}$ since $\mathbf{W}_0^{(\ell)}$ has i.i.d. standard normal entries, and due to $\boldsymbol{\theta}_t \in B(\boldsymbol{\theta}_0, C := \frac{3KR_0}{\lambda_{min}(\boldsymbol{\Theta})})$ (from NTK stability theorem); (b) with high prob. for $\sqrt{n} \gg C$.

Comparing the Smoothness of ResNet and MLP NTKs

- Starting with the Jacobians of MLP and ResNet, we have

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{MLP}(\boldsymbol{x}) \right\|_2 \leq \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{w}^{(L+1)}\|_2 C_\phi \frac{\sigma_w}{\sqrt{d}} \|\boldsymbol{W}^{(1)}\| \prod_{\ell=2}^{L} C_\phi \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{W}^{(\ell)}\|$$

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{ResNet}(\boldsymbol{x}) \right\|_2 \leq \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{w}^{(L+1)}\|_2 \frac{1}{\sqrt{d}} \|\boldsymbol{U}\| \cdot \prod_{\ell=1}^{L} \left(1 + \alpha C_\phi \frac{\sigma_v}{\sqrt{n}} \|\boldsymbol{V}^{(\ell)}\| \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{W}^{(\ell)}\| \right)$$

- In the NTK regime (of both ResNet and MLP) the spectral norm of the weights can be easily bounded. This is in contrast with the general case where there is no precise way to control the weights of DNNs *after* training.
- The spectral norm of, e.g., $\boldsymbol{W}_t^{(\ell)} \in \mathbb{R}^{n \times n}$ obeys

$$\|\boldsymbol{W}_t^{(\ell)}\| \leq \|\boldsymbol{W}_0^{(\ell)}\| + \|\boldsymbol{W}_t^{(\ell)} - \boldsymbol{W}_0^{(\ell)}\| \overset{(a)}{\leq} 2\sqrt{n} + \delta + C \overset{(b)}{\leq} 3\sqrt{n},$$

(a) holds with prob. above $1 - 2e^{-\delta^2/2}$ since $\boldsymbol{W}_0^{(\ell)}$ has i.i.d. standard normal entries, and due to $\boldsymbol{\theta}_t \in B(\boldsymbol{\theta}_0, C := \frac{3KR_0}{\lambda_{min}(\Theta)})$ (from NTK stability theorem); (b) with high prob. for $\sqrt{n} \gg C$.

Comparing the Smoothness of ResNet and MLP NTKs

- Starting with the Jacobians of MLP and ResNet, we have

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{MLP}(\boldsymbol{x}) \right\|_2 \leq \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{w}^{(L+1)}\|_2 C_\phi \frac{\sigma_w}{\sqrt{d}} \|\boldsymbol{W}^{(1)}\| \prod_{\ell=2}^{L} C_\phi \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{W}^{(\ell)}\|$$

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{ResNet}(\boldsymbol{x}) \right\|_2 \leq \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{w}^{(L+1)}\|_2 \frac{1}{\sqrt{d}} \|\boldsymbol{U}\| \cdot \prod_{\ell=1}^{L} \left(1 + \alpha C_\phi \frac{\sigma_v}{\sqrt{n}} \|\boldsymbol{V}^{(\ell)}\| \frac{\sigma_w}{\sqrt{n}} \|\boldsymbol{W}^{(\ell)}\| \right)$$

- In the NTK regime (of both ResNet and MLP) the spectral norm of the weights can be easily bounded. This is in contrast with the general case where there is no precise way to control the weights of DNNs *after* training.
- The spectral norm of, e.g., $\boldsymbol{W}_t^{(\ell)} \in \mathbb{R}^{n \times n}$ obeys

$$\|\boldsymbol{W}_t^{(\ell)}\| \leq \|\boldsymbol{W}_0^{(\ell)}\| + \|\boldsymbol{W}_t^{(\ell)} - \boldsymbol{W}_0^{(\ell)}\| \overset{(a)}{\leq} 2\sqrt{n} + \delta + C \overset{(b)}{\leq} 3\sqrt{n},$$

(a) holds with prob. above $1 - 2\mathrm{e}^{-\delta^2/2}$ since $\boldsymbol{W}_0^{(\ell)}$ has i.i.d. standard normal entries, and due to $\boldsymbol{\theta}_t \in B(\boldsymbol{\theta}_0, C := \frac{3KR_0}{\lambda_{min}(\boldsymbol{\Theta})})$ (from NTK stability theorem); (b) with high prob. for $\sqrt{n} \gg C$.

Comparing the Smoothness of ResNet and MLP NTKs

- Using these properties of the NTK regime for *finite*, yet large n, we get

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{MLP}(\boldsymbol{x}) \right\|_2 \leq 2C_\phi \sigma_w^2 (1 + 2\sqrt{\frac{n}{d}}) \left(3C_\phi \sigma_w\right)^{L-1} := B_{\mathrm{MLP}}$$

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{ResNet}(\boldsymbol{x}) \right\|_2 \leq 2\sigma_w (1 + 2\sqrt{\frac{n}{d}}) \left(1 + 9\alpha C_\phi \sigma_v \sigma_w\right)^{L} := B_{\mathrm{ResNet}}$$

- This factor $\sqrt{\frac{n}{d}}$ is not surprising, since under rather mild conditions on $\mathcal{X}$, the networks can fit any training data in the NTK regime, and thus the slope of their output is not bounded by a constant number.
- For typical value of 1 for the hyperparams., we get

$$\frac{B_{\mathrm{ResNet}}}{B_{\mathrm{MLP}}} = \frac{(1 + 9\alpha C_\phi \sigma_v \sigma_w)^L}{3^{L-1}(C_\phi \sigma_w)^L} = \frac{(1 + 9\alpha)^L}{3^{L-1}}$$

- A moderate value of α, such as 0.1, implies $B_{\mathrm{ResNet}} \leq B_{\mathrm{MLP}}$ for any $L \geq 3$, which hints that ResNet NTK will be smoother.
- For small enough α, increasing L (# nonlinear layers) is expected to increase the smoothness distinction between the NTKs.

Comparing the Smoothness of ResNet and MLP NTKs

- Using these properties of the NTK regime for *finite*, yet large n, we get

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{MLP}(\boldsymbol{x}) \right\|_2 \leq 2C_\phi \sigma_w^2 (1 + 2\sqrt{\frac{n}{d}}) \left(3C_\phi \sigma_w\right)^{L-1} := B_{\mathrm{MLP}}$$

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{ResNet}(\boldsymbol{x}) \right\|_2 \leq 2\sigma_w (1 + 2\sqrt{\frac{n}{d}}) \left(1 + 9\alpha C_\phi \sigma_v \sigma_w\right)^{L} := B_{\mathrm{ResNet}}$$

- This factor $\sqrt{\frac{n}{d}}$ is not surprising, since under rather mild conditions on $\mathcal{X}$, the networks can fit any training data in the NTK regime, and thus the slope of their output is not bounded by a constant number.
- For typical value of 1 for the hyperparams., we get

$$\frac{B_{\mathrm{ResNet}}}{B_{\mathrm{MLP}}} = \frac{(1 + 9\alpha C_\phi \sigma_v \sigma_w)^L}{3^{L-1}(C_\phi \sigma_w)^L} = \frac{(1 + 9\alpha)^L}{3^{L-1}}$$

- A moderate value of α, such as 0.1, implies $B_{\mathrm{ResNet}} \leq B_{\mathrm{MLP}}$ for any $L \geq 3$, which hints that ResNet NTK will be smoother.
- For small enough α, increasing L (# nonlinear layers) is expected to increase the smoothness distinction between the NTKs.

Comparing the Smoothness of ResNet and MLP NTKs

- Using these properties of the NTK regime for *finite*, yet large n, we get

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{MLP}(\boldsymbol{x}) \right\|_2 \leq 2C_\phi \sigma_w^2 (1 + 2\sqrt{\frac{n}{d}}) \left(3C_\phi \sigma_w\right)^{L-1} := B_{\mathrm{MLP}}$$

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{ResNet}(\boldsymbol{x}) \right\|_2 \leq 2\sigma_w (1 + 2\sqrt{\frac{n}{d}}) \left(1 + 9\alpha C_\phi \sigma_v \sigma_w\right)^{L} := B_{\mathrm{ResNet}}$$

- This factor $\sqrt{\frac{n}{d}}$ is not surprising, since under rather mild conditions on $\mathcal{X}$, the networks can fit any training data in the NTK regime, and thus the slope of their output is not bounded by a constant number.
- For typical value of 1 for the hyperparams., we get

$$\frac{B_{\mathrm{ResNet}}}{B_{\mathrm{MLP}}} = \frac{(1 + 9\alpha C_\phi \sigma_v \sigma_w)^L}{3^{L-1}(C_\phi \sigma_w)^L} = \frac{(1 + 9\alpha)^L}{3^{L-1}}$$

- A moderate value of α, such as 0.1, implies $B_{\mathrm{ResNet}} \leq B_{\mathrm{MLP}}$ for any $L \geq 3$, which hints that ResNet NTK will be smoother.
- For small enough α, increasing L (# nonlinear layers) is expected to increase the smoothness distinction between the NTKs.

Comparing the Smoothness of ResNet and MLP NTKs

- Using these properties of the NTK regime for *finite*, yet large n, we get

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{MLP}(\boldsymbol{x}) \right\|_2 \leq 2C_\phi \sigma_w^2 (1 + 2\sqrt{\frac{n}{d}}) \left(3C_\phi \sigma_w\right)^{L-1} := B_{\mathrm{MLP}}$$

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{ResNet}(\boldsymbol{x}) \right\|_2 \leq 2\sigma_w (1 + 2\sqrt{\frac{n}{d}}) \left(1 + 9\alpha C_\phi \sigma_v \sigma_w\right)^{L} := B_{\mathrm{ResNet}}$$

- This factor $\sqrt{\frac{n}{d}}$ is not surprising, since under rather mild conditions on $\mathcal{X}$, the networks can fit any training data in the NTK regime, and thus the slope of their output is not bounded by a constant number.
- For typical value of 1 for the hyperparams., we get

$$\frac{B_{\mathrm{ResNet}}}{B_{\mathrm{MLP}}} = \frac{(1 + 9\alpha C_\phi \sigma_v \sigma_w)^L}{3^{L-1}(C_\phi \sigma_w)^L} = \frac{(1 + 9\alpha)^L}{3^{L-1}}$$

- A moderate value of α, such as 0.1, implies $B_{\mathrm{ResNet}} \leq B_{\mathrm{MLP}}$ for any $L \geq 3$, which hints that ResNet NTK will be smoother.
- For small enough α, increasing L (# nonlinear layers) is expected to increase the smoothness distinction between the NTKs.

Comparing the Smoothness of ResNet and MLP NTKs

- Using these properties of the NTK regime for *finite*, yet large n, we get

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{MLP}(\boldsymbol{x}) \right\|_2 \leq 2C_\phi \sigma_w^2 (1 + 2\sqrt{\frac{n}{d}}) \left(3C_\phi \sigma_w\right)^{L-1} := B_{\mathrm{MLP}}$$

$$\sup_{\boldsymbol{x}} \left\| \frac{\partial}{\partial \boldsymbol{x}} f_{ResNet}(\boldsymbol{x}) \right\|_2 \leq 2\sigma_w (1 + 2\sqrt{\frac{n}{d}}) \left(1 + 9\alpha C_\phi \sigma_v \sigma_w\right)^{L} := B_{\mathrm{ResNet}}$$

- This factor $\sqrt{\frac{n}{d}}$ is not surprising, since under rather mild conditions on $\mathcal{X}$, the networks can fit any training data in the NTK regime, and thus the slope of their output is not bounded by a constant number.
- For typical value of 1 for the hyperparams., we get

$$\frac{B_{\mathrm{ResNet}}}{B_{\mathrm{MLP}}} = \frac{(1 + 9\alpha C_\phi \sigma_v \sigma_w)^L}{3^{L-1}(C_\phi \sigma_w)^L} = \frac{(1+9\alpha)^L}{3^{L-1}}$$

- A moderate value of α, such as 0.1, implies $B_{\mathrm{ResNet}} \leq B_{\mathrm{MLP}}$ for any $L \geq 3$, which hints that ResNet NTK will be smoother.
- For small enough α, increasing L ($\#$ nonlinear layers) is expected to increase the smoothness distinction between the NTKs.

Comparing the Smoothness of ResNet and MLP NTKs

- We turn to visualize the NTKs and the NTK regression results in several settings.
- We measure the smoothness of NTK regression outputs by an approximated $\mathcal{L}^2$-norm of the outputs' second derivatives.
- Given a normalized regression result $\bar{f} = f/\widehat{\|f\|_{\mathcal{L}^2}}$ we approximate $\|\bar{f}''\|_{\mathcal{L}^2}$ by

$$\mu(f) := \Big(\frac{1}{N^2}\sum_{k=-N/2}^{N/2-1} |k|^4 |F(k)|^2\Big)^{\frac{1}{2}},$$

where $F(k)$ denotes $\mathrm{FFT}[\{\bar{f}(x_q)\}](k)$.

Comparing the Smoothness of ResNet and MLP NTKs

- We turn to visualize the NTKs and the NTK regression results in several settings.
- We measure the smoothness of NTK regression outputs by an approximated $\mathcal{L}^2$-norm of the outputs' second derivatives.
- Given a normalized regression result $\bar{f} = f/\widehat{\|f\|_{\mathcal{L}^2}}$ we approximate $\|\bar{f}''\|_{\mathcal{L}^2}$ by

$$\mu(f) := \left(\frac{1}{N^2}\sum_{k=-N/2}^{N/2-1} |k|^4 |F(k)|^2\right)^{\frac{1}{2}},$$

where $F(k)$ denotes $\mathrm{FFT}[\{\bar{f}(x_q)\}](k)$.

Comparing the Smoothness of ResNet and MLP NTKs

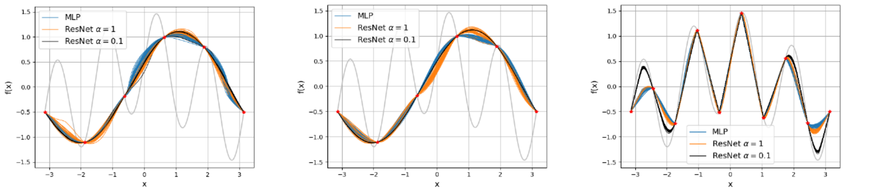
- We turn to visualize the NTKs and the NTK regression results in several settings.
- We measure the smoothness of NTK regression outputs by an approximated $\mathcal{L}^2$-norm of the outputs' second derivatives.
- Given a normalized regression result $\bar{f} = f / \widehat{\|f\|_{\mathcal{L}^2}}$ we approximate $\|\bar{f}''\|_{\mathcal{L}^2}$ by

$$\mu(f) := \left(\frac{1}{N^2} \sum_{k=-N/2}^{N/2-1} |k|^4 |F(k)|^2 \right)^{\frac{1}{2}},$$

where $F(k)$ denotes $\mathrm{FFT}[\{\bar{f}(x_q)\}](k)$.

Comparing the Smoothness of ResNet and MLP NTKs

Recall the empirical results outside the NTK regime:

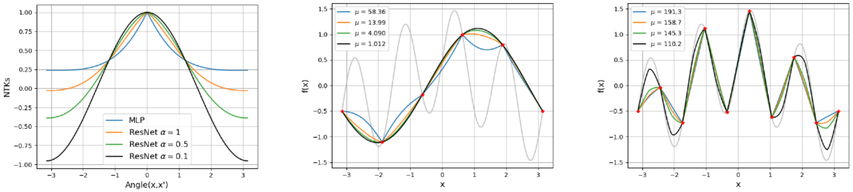


(a) Interpolation with 6 samples (Adam) (b) Interpolation with 6 samples (SGD) (c) Interpolation with 10 samples (SGD)

Figure 3: Empirical interpolations of MLP and ResNet (for different values of α) with $L = 5$ nonlinear layers, ReLU nonlinearities, $\sigma_v = \sigma_w = 1$, for inputs on the sphere (circle) in $\mathbb{R}^2$. We use practical models with width of $n = 500$, 30 different Xavier's Gaussian initializations (instead of normalizations by $1/\sqrt{n}$) and 1K iterations of SGD/Adam optimizers.

Comparing the Smoothness of ResNet and MLP NTKs

Numerical NTK results:



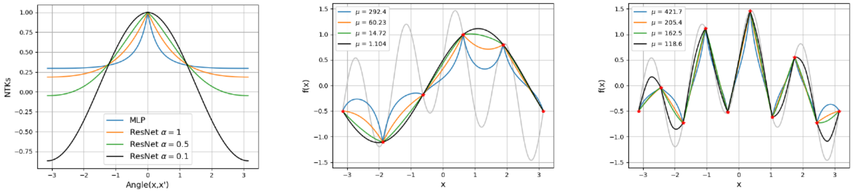
(a) NTKs (normalized to unit peak) $L = 5$ (b) Interpolation with 6 samples $L = 5$ (c) Interpolation with 10 samples $L = 5$

Figure 2: NTKs for MLP and ResNet (for different values of α) with $L = 5$ (top) and $L = 15$ (bottom) nonlinear layers, ReLU nonlinearities, $\sigma_v = \sigma_w = 1$, for inputs on the sphere (circle) in $\mathbb{R}^2$. (2a),(2d): The kernels shape. (2b)-(2c), (2e)-(2f): Interpolations by the closed-form solutions, measured by $\mu(\cdot)$ defined in (22). Note that the legend in (2a),(2d) applies to all the figures.

- Decreasing α yields a smoother ResNet NTK with smoother interpolation results. For $\alpha = 1$ the ResNet NTK is more similar to the MLP NTK, but even then it appears smoother and less "edgy" than the MLP.

Comparing the Smoothness of ResNet and MLP NTKs

Numerical NTK results:



(*d*) NTKs (normalized to unit peak) $L = 15$ (*e*) Interpolation with 6 samples $L = 15$ (*f*) Interpolation with 10 samples $L = 15$

Figure 2: NTKs for MLP and ResNet (for different values of α) with $L = 5$ (top) and $L = 15$ (bottom) nonlinear layers, ReLU nonlinearities, $\sigma_v = \sigma_w = 1$, for inputs on the sphere (circle) in $\mathbb{R}^2$. (2a),(2d): The kernels shape. (2b)-(2c), (2e)-(2f): Interpolations by the closed-form solutions, measured by $\mu(\cdot)$ defined in (22). Note that the legend in (2a),(2d) applies to all the figures.

- For the MLP NTK, increasing L clearly reduces the smoothness (as observed both visually and by the increase in μ). For the ResNet NTK this effect is moderate for $\alpha = 1$, and almost unseen for $\alpha = 0.1$.

Conclusion

- We developed the NTK for a ResNet model and proved its stability during training with gradient descent (under common NTK assumptions).
- When both networks use ReLU activations, our smoothness examination shows that ResNet, especially with moderately attenuated residual blocks, yields smoother interpolations than MLP.
- When the activation function is smooth (e.g., erf) the different NTKs have similar smoothness (see the paper).
- More technical details and numerical results for multivariate functions appears in the paper.

Conclusion

- We developed the NTK for a ResNet model and proved its stability during training with gradient descent (under common NTK assumptions).
- When both networks use ReLU activations, our smoothness examination shows that ResNet, especially with moderately attenuated residual blocks, yields smoother interpolations than MLP.
- When the activation function is smooth (e.g., erf) the different NTKs have similar smoothness (see the paper).
- More technical details and numerical results for multivariate functions appears in the paper.

Conclusion

- We developed the NTK for a ResNet model and proved its stability during training with gradient descent (under common NTK assumptions).
- When both networks use ReLU activations, our smoothness examination shows that ResNet, especially with moderately attenuated residual blocks, yields smoother interpolations than MLP.
- When the activation function is smooth (e.g., erf) the different NTKs have similar smoothness (see the paper).
- More technical details and numerical results for multivariate functions appears in the paper.

Conclusion

- We developed the NTK for a ResNet model and proved its stability during training with gradient descent (under common NTK assumptions).
- When both networks use ReLU activations, our smoothness examination shows that ResNet, especially with moderately attenuated residual blocks, yields smoother interpolations than MLP.
- When the activation function is smooth (e.g., erf) the different NTKs have similar smoothness (see the paper).
- More technical details and numerical results for multivariate functions appears in the paper.

Conclusion

- We developed the NTK for a ResNet model and proved its stability during training with gradient descent (under common NTK assumptions).
- When both networks use ReLU activations, our smoothness examination shows that ResNet, especially with moderately attenuated residual blocks, yields smoother interpolations than MLP.
- When the activation function is smooth (e.g., erf) the different NTKs have similar smoothness (see the paper).
- More technical details and numerical results for multivariate functions appears in the paper.

Thank You