

# Master's Thesis Notes

Shreyas Kalvankar

## Contents

<b>1</b>	<b>Formalization of the Neural Tangent Kernel (NTK)</b>	<b>1</b>
1.1	Setup and Assumptions	1
1.2	Linearization around Initialization	1
1.3	Neural Tangent Kernel	2
1.4	NTK Characterization (one hidden layer, no biases)	2
1.5	One Hidden Layer: Adding Biases	3
1.6	Extension to Deep Fully-Connected Networks	4
1.7	Training Dynamics under NTK	4
1.8	Lazy Training Regime	5

## 1 Formalization of the Neural Tangent Kernel (NTK)

### 1.1 Setup and Assumptions

We consider a supervised learning setting with data  $\{(x_i, y_i)\}_{i=1}^n$ , where  $x_i \in \mathbb{R}^d$  and  $y_i \in \mathbb{R}$ . A neural network  $f : \mathbb{R}^d \times \Theta \rightarrow \mathbb{R}$  is parameterized by  $\theta \in \Theta \subseteq \mathbb{R}^p$ , where  $p$  is the number of parameters.

**Assumption 1.1** (Model and training).

1. (*Differentiability*) The network  $f(x; \theta)$  is differentiable in  $\theta$ .
2. (*Random initialization*) We initialize parameters  $\theta_0$  i.i.d. with zero mean and variance scaled according to layer input dimensions (e.g. NTK or Xavier/He schemes), so that activations and gradients remain well-behaved as depth/width grow [3, 1, 2].
3. (*Training*) We train  $f$  by gradient descent on the squared loss:

$$L(\theta) = \frac{1}{2} \sum_{i=1}^n (f(x_i; \theta) - y_i)^2.$$

### 1.2 Linearization around Initialization

A first-order Taylor expansion of  $f(x; \theta)$  around  $\theta_0$  gives

$$f(x; \theta) \approx f(x; \theta_0) + \nabla_{\theta} f(x; \theta_0)^{\top} (\theta - \theta_0).$$

- $f(x; \theta_0)$  is the network output at initialization (a bias term).
- $\phi(x) := \nabla_{\theta} f(x; \theta_0)$  is the feature vector induced at initialization.

Thus, locally, the network behaves as a linear model in  $\theta$ :

$$f(x; \theta) \approx f(x; \theta_0) + \phi(x)^{\top} (\theta - \theta_0).$$

### 1.3 Neural Tangent Kernel

**Definition 1.2** (Neural Tangent Kernel [1]). Given initialization  $\theta_0$ , the Neural Tangent Kernel (NTK) is

$$K(x, x') = \nabla_{\theta} f(x; \theta_0)^{\top} \nabla_{\theta} f(x'; \theta_0).$$

The NTK captures how parameter updates couple the outputs of  $x$  and  $x'$ .

*Remark 1.3.*

- $K(x, x')$  is positive semidefinite [1].
- In the infinite-width limit, under common initializations,  $K(x, x')$  converges almost surely to a deterministic kernel depending only on architecture and activation [1, 2].
- For finite but large width,  $K$  is still a random kernel due to random initialization, but it concentrates around its infinite-width expectation. Fluctuations vanish at rate  $O(1/\sqrt{m})$  as width  $m \rightarrow \infty$  [1, 2].

### 1.4 NTK Characterization (one hidden layer, no biases)

Consider a width- $m$  one-hidden-layer network

$$f(x) = \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r \sigma(w_r^{\top} x),$$

with parameters  $\theta = \{(a_r, w_r)\}_{r=1}^m$ , activation  $\sigma$ , and random initialization

$$a_r \sim \mathcal{N}(0, \sigma_a^2), \quad w_r \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{d} I_d\right),$$

independently across  $r$ .

#### Finite-width NTK at initialization

By definition,

$$K_m(x, x') = \nabla_{\theta} f(x)^{\top} \nabla_{\theta} f(x') = \sum_{r=1}^m \left[ \underbrace{\nabla_{a_r} f(x) \nabla_{a_r} f(x')}_{\text{output-weight part}} + \underbrace{\nabla_{w_r} f(x)^{\top} \nabla_{w_r} f(x')}_{\text{hidden-weight part}} \right].$$

Compute the gradients:

$$\nabla_{a_r} f(x) = \frac{1}{\sqrt{m}} \sigma(w_r^{\top} x), \quad \nabla_{w_r} f(x) = \frac{1}{\sqrt{m}} a_r \sigma'(w_r^{\top} x) x.$$

Hence,

$$K_m(x, x') = \frac{1}{m} \sum_{r=1}^m \sigma(w_r^{\top} x) \sigma(w_r^{\top} x') + \frac{1}{m} \sum_{r=1}^m a_r^2 \sigma'(w_r^{\top} x) \sigma'(w_r^{\top} x') x^{\top} x'.$$

## Infinite-width limit

The two sums are empirical averages of i.i.d. terms. Since  $a_r$  and  $w_r$  are independent with finite moments, the (strong) law of large numbers gives, almost surely,

$$\begin{aligned} \frac{1}{m} \sum_{r=1}^m \sigma(w_r^\top x) \sigma(w_r^\top x') &\longrightarrow \mathbb{E}_w [\sigma(w^\top x) \sigma(w^\top x')], \\ \frac{1}{m} \sum_{r=1}^m a_r^2 \sigma'(w_r^\top x) \sigma'(w_r^\top x') &\longrightarrow \sigma_a^2 \mathbb{E}_w [\sigma'(w^\top x) \sigma'(w^\top x')]. \end{aligned}$$

Thus, in the infinite-width limit, the empirical NTK converges almost surely to a deterministic kernel [1, 2].

$$K_\infty(x, x') = \mathbb{E}_w [\sigma(w^\top x) \sigma(w^\top x')] + \sigma_a^2 x^\top x' \mathbb{E}_w [\sigma'(w^\top x) \sigma'(w^\top x')]$$

with  $w \sim \mathcal{N}(0, \frac{\sigma_w^2}{d} I_d)$ .

## 1.5 One Hidden Layer: Adding Biases

If we allow per-neuron biases, we get minimal changes from [subsection 1.4](#). Consider

$$f(x) = \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r \sigma(w_r^\top x + b_r), \quad a_r \sim \mathcal{N}(0, \sigma_a^2), \quad w_r \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{d} I_d\right), \quad b_r \sim \mathcal{N}(0, \sigma_b^2),$$

independently across  $r$ . Let  $u_r(x) := w_r^\top x + b_r$ .

**Finite width** Gradients are

$$\nabla_{a_r} f(x) = \frac{1}{\sqrt{m}} \sigma(u_r(x)), \quad \nabla_{w_r} f(x) = \frac{1}{\sqrt{m}} a_r \sigma'(u_r(x)) x, \quad \nabla_{b_r} f(x) = \frac{1}{\sqrt{m}} a_r \sigma'(u_r(x)).$$

Thus

$$K_m(x, x') = \frac{1}{m} \sum_{r=1}^m \sigma(u_r(x)) \sigma(u_r(x')) + \frac{1}{m} \sum_{r=1}^m a_r^2 \sigma'(u_r(x)) \sigma'(u_r(x')) (x^\top x' + 1).$$

**Infinite width (preactivation covariance picks up  $\sigma_b^2$ ).** With  $(U, V)$  jointly Gaussian:

$$\text{Var}(U) = \frac{\sigma_w^2}{d} \|x\|^2 + \sigma_b^2, \quad \text{Var}(V) = \frac{\sigma_w^2}{d} \|x'\|^2 + \sigma_b^2, \quad \text{Cov}(U, V) = \frac{\sigma_w^2}{d} x^\top x' + \sigma_b^2,$$

we have

$$K_\infty(x, x') = \mathbb{E}[\sigma(U) \sigma(V)] + \sigma_a^2 (x^\top x' + 1) \mathbb{E}[\sigma'(U) \sigma'(V)].$$

## 1.6 Extension to Deep Fully-Connected Networks

**No biases.** Let  $n_0 = d$ ,  $n_1, \dots, n_{L-1}$  be layer widths and define

$$\alpha^{(0)}(x) = x, \quad \tilde{\alpha}^{(\ell+1)}(x) = W^{(\ell)} \alpha^{(\ell)}(x), \quad \alpha^{(\ell+1)}(x) = \sigma(\tilde{\alpha}^{(\ell+1)}(x)),$$

for  $\ell = 0, \dots, L-2$ , with entries  $W^{(\ell)} \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \frac{\sigma_w^2}{n_\ell}\right)$ . The scalar output is

$$f_\theta(x) = \frac{1}{\sqrt{n_{L-1}}} \sum_{r=1}^{n_{L-1}} a_r \alpha_r^{(L-1)}(x), \quad a_r \sim \mathcal{N}(0, \sigma_a^2).$$

Define the activation covariance

$$\Sigma^{(\ell)}(x, x') := \mathbb{E}[\alpha_r^{(\ell)}(x) \alpha_r^{(\ell)}(x')], \quad q^{(\ell)}(x) := \Sigma^{(\ell)}(x, x),$$

with base  $\Sigma^{(0)}(x, x') = \frac{1}{d} x^\top x'$ . Then the forward (NNGP) recursion is

$$\Sigma^{(\ell+1)}(x, x') = \mathbb{E}_{f \sim \mathcal{GP}(0, \sigma_w^2 \Sigma^{(\ell)})} [\sigma(f(x)) \sigma(f(x'))].$$

Define the derivative-correlation kernel

$$\dot{\Sigma}^{(\ell+1)}(x, x') := \mathbb{E}_{f \sim \mathcal{GP}(0, \sigma_w^2 \Sigma^{(\ell)})} [\sigma'(f(x)) \sigma'(f(x'))].$$

The limiting NTK satisfies

$$\Theta_\infty^{(1)}(x, x') = \Sigma^{(1)}(x, x'), \quad \Theta_\infty^{(\ell+1)}(x, x') = \Theta_\infty^{(\ell)}(x, x') \dot{\Sigma}^{(\ell+1)}(x, x') + \Sigma^{(\ell+1)}(x, x').$$

For a dataset  $\{x_i\}$ :

$$\Theta^{(\ell+1)} = \Theta^{(\ell)} \odot \dot{\Sigma}^{(\ell+1)} + \Sigma^{(\ell+1)}, \quad \Theta^{(1)} = \Sigma^{(1)}.$$

Setting  $L = 2$  recovers the two-term one-layer kernel.

## 1.7 Training Dynamics under NTK

We train with squared loss

$$L(\theta) = \frac{1}{2} \sum_{i=1}^n (f(x_i; \theta) - y_i)^2,$$

and consider *gradient flow* in parameter space, i.e. the continuous-time limit of gradient descent as the step size  $\eta \rightarrow 0$ :

$$\frac{d\theta_t}{dt} = -\nabla_\theta L(\theta_t).$$

Let  $f_t(x_i) := f(x_i; \theta_t)$ . By the chain rule,

$$\frac{d}{dt} f_t(x_i) = \nabla_\theta f(x_i; \theta_t)^\top \frac{d\theta_t}{dt} = -\nabla_\theta f(x_i; \theta_t)^\top \nabla_\theta L(\theta_t).$$

Compute the parameter gradient of the loss:

$$\nabla_\theta L(\theta_t) = \sum_{j=1}^n (f_t(x_j) - y_j) \nabla_\theta f(x_j; \theta_t).$$

Substituting gives

$$\frac{d}{dt}f_t(x_i) = - \sum_{j=1}^n \underbrace{\nabla_{\theta}f(x_i; \theta_t)^{\top} \nabla_{\theta}f(x_j; \theta_t)}_{=: K_t(x_i, x_j)} (f_t(x_j) - y_j).$$

Stacking  $f_t = (f_t(x_1), \dots, f_t(x_n))$  yields the vector ODE

$$\frac{d}{dt}f_t = -K_t(f_t - y),$$

where  $[K_t]_{ij} = K_t(x_i, x_j)$  is the (time-dependent) NTK matrix.

**Constant-kernel (NTK) regime.** In the infinite-width limit (or under a lazy-training approximation), the kernel remains essentially constant during training,  $K_t \approx K_0 =: K$  [1]. The ODE reduces to

$$\frac{d}{dt}f_t = -K(f_t - y).$$

Let  $r_t := f_t - y$ . Then  $\dot{r}_t = -Kr_t$  with solution  $r_t = e^{-Kt}r_0$ , i.e.

$$f_t = y + e^{-Kt}(f_0 - y).$$

The convergence rate along eigenvector  $v_j$  of  $K$  is exponential with rate  $\lambda_j$ , the corresponding eigenvalue.

## 1.8 Lazy Training Regime

Training is in the *lazy regime* if parameter updates stay small relative to initialization:

$$\|\theta_t - \theta_0\| \ll \|\theta_0\|.$$

Then  $\phi(x)$  and the NTK remain essentially constant and training is equivalent to kernel regression with fixed kernel  $K$ . When  $\|\theta_t - \theta_0\|$  is not negligible,  $\phi(x)$  evolves, yielding adaptive feature learning beyond NTK.

To be continued...

## References

- [1] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, 2018.
- [2] Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems*, 2019.
- [3] Radford M. Neal. *Priors for infinite networks*. PhD thesis, University of Toronto, 1996.