Notes on the Neural Tangent Kernel

Narutatsu Ri

April 14th, 2023

TL;DR

Given any model with a defined loss, the Tangent Kernel arises when we take the derivative of the model's output w.r.t. the time step. The Neural Tangent Kernel (NTK) is a specific case when the model is a Neural Net (NN).

Change in Model Output

The NTK can be represented with Gaussian Processes in the infinite-width case. [LSdP+18]
The change in the model's outputs during training of infinite-width NN can be approximated with a linearlized model where the expression for NTK appears and is deterministic based on the NN's architecture [JGH20].
The same can be shown to hold for sufficiently wide finite NNs [ADH+19].

Update to Model

The updates made to the weights in the NN largely remain constant during training (Hessian of model is zero), i.e. roughly the same term is added to the model weights at every step. [JGH20]

Trained Model

The trained NN's output behavior will be ϵ-close to the output behavior of the result of kernel regression with the NTK as the kernel. [ADH+19]
Note: Not true in practice!

1 Notation

Define $\mathbb{R}^P$ parameter space, $\mathbb{R}^{n_0}$ input space, $\mathbb{R}^{n_L}$ output space (Assume model is neural net with L layers. Note that parameter space is equivalent to product of input and output space), $f(x, \theta)$ as model with parameters $\theta \in \mathbb{R}^P$, per-sample loss as $l : \mathbb{R}^P \to \mathbb{R}_+$, loss as $\mathcal{L} : \mathbb{R}^P \to \mathbb{R}_+, \mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} l(f(x, \theta); y^{(i)})$.

2 Overview

2.1 Weakly-trained Nets

Early literature that looked into the analysis of neural networks include Neal [Nea96], who connect a one-layer neural network with Gaussian processes; specifically, a single hidden-layer NN with i.i.d random parameters from $N(0,1)$ (also known as LeCun initialization) is a function drawn from a Gaussian Process. Lee et al. [LSdP+18] extended this line of work to multi-layer NNs, but all studies considered a *weakly-trained* net—a NN where the intermediate weights are fixed and only the last layer is updated.

However, NNs are not weakly-trained NNs in practice; this limitation was added for theoreical convenience. The Neural Tangent Kernel is one approach in attempting to analyze the fully-trained NN model.

2.2 Neural Tangent Kernel

Given a model $f(x;\theta)$ with some parameter θ, the total loss is written as:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} l(f(x^{(i)},\theta); y^{(i)})$$

Taking the partial derivative w.r.t. θ,

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \nabla_f l(f(x^{(i)};\theta), y^{(i)}) \cdot \nabla_\theta f(x^{(i)};\theta)$$

During training, each update made to the parameters θ can be written as

$$\frac{d\theta}{dt} \approx -\eta \nabla_\theta \mathcal{L}(\theta)$$

where η is the learning rate. With small enough η, the above equality is exact (c.f. Expectation of SGD equals gradient flow). Applying chain rule, we can obtain the expression for the change of the model's output w.r.t. the time step:

$$\begin{aligned}\frac{df(x;\theta)}{dt} = \frac{df(x;\theta)}{d\theta}\frac{d\theta}{dt} &= -\frac{1}{N}\sum_{i=1}^{N} \nabla_\theta f(x;\theta)\cdot\nabla_\theta \mathcal{L}(\theta) \\ &= -\frac{1}{N}\sum_{i=1}^{N} \nabla_\theta f(x;\theta)\cdot\nabla_\theta f(x^{(i)};\theta)\cdot\nabla_f l(f(x^{(i)};\theta), y^{(i)}) \qquad (1)\end{aligned}$$

Note the expression $\nabla_\theta f(x;\theta)\cdot\nabla_\theta f(x^{(i)};\theta)$; this is a kernel with the feature map $\Phi(x) = \nabla_\theta f(x;\theta)$.

Notice there is no specificity in the model $f(x;\theta)$ and loss $\mathcal{L}(\theta)$. When $f(x;\theta)$ is a neural net, $\Phi(x)\cdot\Phi(x')$ is defined as the **Neural Tangent Kernel**. For simplicitly of analysis, many papers use the mean squared error (MSE), i.e. $\mathcal{L}(\theta) = \frac{1}{2N}\sum_{i=1}^{N} ||y^{(i)} - f(x^{(i)};\theta)||^2$. For this case, the remaining expression in the above equation becomes $\nabla_f l(f(x^{(i)};\theta), y^{(i)}) = y^{(i)} - f(x^{(i)};\theta)$.

Digestion of the Expression

The change in the model output can be represented as a product between the NTK and the gradient of the loss w.r.t. the model outputs. The latter term is easy to analyze (i.e. the value is deterministic given data $(x^{(i)}, y^{(i)})_{i\in[N]}$ and model f), but the former term could be difficult to analyze. It turns out in the infinite-width case (and sufficient width case), the NTK is easy to analyze as the expression is deterministic

2.3 NTK and Gaussian Processes

[LSdP+18] show an L-layer infinite-width NNs can be represented as a recursive sequence of Gaussian Processes:

$$\begin{aligned}
\Sigma^{(1)}(\mathbf{x}, \mathbf{x}') &= \frac{1}{n_0}\mathbf{x}^\top \mathbf{x}' \\
\lambda^{(l+1)}(\mathbf{x}, \mathbf{x}') &= \begin{bmatrix} \Sigma^{(l)}(\mathbf{x}, \mathbf{x}) & \Sigma^{(l)}(\mathbf{x}, \mathbf{x}') \\ \Sigma^{(l)}(\mathbf{x}', \mathbf{x}) & \Sigma^{(l)}(\mathbf{x}', \mathbf{x}') \end{bmatrix} \\
\Sigma^{(l+1)}(\mathbf{x}, \mathbf{x}') &= \mathbb{E}_{f\sim\mathcal{N}(0,\lambda^{(l)})}[\sigma(f(\mathbf{x}))\sigma(f(\mathbf{x}'))]
\end{aligned}$$

This construction makes it possible to represent the NTK for the infinite-width NN as a combination of Gaussian Processes:

$$\Theta^{(L)}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{h=1}^{L+1} \left(\Sigma^{(h-1)}(\boldsymbol{x}, \boldsymbol{x}') \cdot \prod_{h'=h}^{L+1} \Sigma^{(h')}(\boldsymbol{x}, \boldsymbol{x}') \right)$$

2.4 NTK is Deterministic in Infinite-with Setting

In the infinite-width case, the NTK $\Theta^{(L)}$ converges in probability to a deterministic expression $\Theta_\infty^{(L)}$ and stays constant during training. In other words, the initialization has no effect on the value of the NTK which does not change over the course of training.

2.5 Linearlizing Infinite-width NNs

Because the NTK is simply a fixed four-dimesional tensor for the infinite-width case, the change of the model's output w.r.t. the time step (equation (1)) for the mean squared error loss can be written as:

$$\begin{aligned}
\frac{df(\theta)}{dt} &= -\nabla_\theta f(\theta)^\top \nabla_\theta f(\theta) \nabla_f \mathcal{L}(\theta) \\
&= -\Theta_\infty^{(L)} \nabla_f \mathcal{L}(\theta)
\end{aligned} \tag{2}$$

Now, consider a simplified approach to analyzing the NN and examine the Maclaurin expansion (Taylor expansion at 0) of the model. Under the assumptions:

- Assume the model is linear in θ (clearly a non-trivial assumption at first glance; the next section will justify this assumption)
- The time step t is infinitesimal

Then, the model can be approximated with a linear expression:

$$
\begin{aligned}
& f(\theta(t)) \approx f(\theta(0)) + \nabla_\theta f(\theta(0))(\theta(t) - \theta(0)) \\
& \Leftrightarrow f(\theta(t)) = f(\theta(0)) - \nabla_\theta f(\theta(0)) \nabla_\theta f(\theta) \cdot \nabla_f \mathcal{L} \quad (\because \theta(t) - \theta(0) = -\nabla_\theta \mathcal{L}(\theta) = -\nabla_\theta f(\theta) \cdot \nabla_f \mathcal{L}) \\
& \Leftrightarrow \frac{df}{dt} = -\Theta_\infty^{(L)} \nabla_f \mathcal{L}(\theta). \quad (\because t \to 0, \text{NTK} = \Theta_\infty^{(L)} \text{ for infinite width})
\end{aligned}
$$

The final line is precisely what we obtain when we consider the infinite-width setting in Eq. (2). In other words, the expression for the change in the model outputs is *linear* in the NTK for infinite-width networks.

Note that this only states the change in the output is the same and does not state equivalence between the final models after the changes.

2.6 Lazy Training

Notice the above Taylor expansion has small error only in the case where the change in the gradient of θ is negligible, i.e. θ changes at a constant pace (and hence linear). Here, we show this is the case for the infinite-width NNs.

Consider the setting where the infinite-width network is over-parameterized (a reasonable assumption to make). In this case, there exists a finite time step T where the training loss $f(\theta(T)) = 0$. To show linearlization is reasonable, we consider the linearized model and show specific terms become negligible as width goes to infinity.

Applying Maclaurin expansion,

$$
\begin{aligned}
f(\theta(T)) &\approx f(\theta(0)) + \nabla_\theta f(\theta(0))(\theta(T) - \theta(0)) \\
\Leftrightarrow \Delta\theta = \theta(T) - \theta(0) &\approx \frac{\|f(\theta(T)) - f(\theta(0))\|}{\|\nabla_\theta f(\theta(0))\|}
\end{aligned}
$$

Now, considering the change in $\nabla_\theta f$, we take the partial derivative of the Taylor expansion:

$$
\begin{aligned}
\nabla_\theta f(\theta(T)) &\approx \nabla_\theta f(\theta(0)) + \nabla_\theta^2 f(\theta(0)) \Delta\theta \\
&= \nabla_\theta f(\theta(0)) + \nabla_\theta^2 f(\theta(0)) \frac{\|f(\theta(T)) - f(\theta(0))\|}{\|\nabla_\theta f(\theta(0))\|} \\
\Leftrightarrow \text{Change in } \left(\nabla_\theta f\right) = \nabla_\theta f(\theta(T)) - \nabla_\theta f(\theta(0)) &= \|f(\theta(T)) - f(\theta(0))\| \frac{\nabla_\theta^2 f(\theta(0))}{\|\nabla_\theta f(\theta(0))\|}
\end{aligned}
$$

Showing the final term is negligibly small equates to a constant gradient for θ means the difference between $f(\theta(T))$ and $f(\theta(0))$ can be linearlized with low approximation error.

Consider the relative change of the gradient of θ w.r.t. the initial gradient at time step $t = 0$ and define it as $\kappa(\theta)$:

$$
\kappa(\theta) = \frac{\text{Change in } \left(\nabla_\theta f\right)}{\|\nabla_\theta f(\theta(0))\|} = \|f(\theta(T)) - f(\theta(0))\| \frac{\nabla_\theta^2 f(\theta(0))}{\|\nabla_\theta f(\theta(0))\|^2}
$$

It can be shown that $\kappa(\theta) \to 0$ as the widths tend to infinity. In other words, in the infinite-width regime, the change in the model outputs can be linearlized because the change in the gradient of θ is negligible, and thus a first-order Taylor appxorimation is appropriate.

3 Follow-up Literature

3.1 Convergence to the NTK for finite-width NNs

Theorem 3.1. *Fix $\epsilon > 0$ and $\delta \in (0, 1)$. Suppose $\sigma(z) = \max(0, z)$ and $\min_{h \in [L]} \geq \Omega(\frac{L^6}{\epsilon^4} \log(L/\delta))$. Then for any inputs $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^{d_0}$ such that $\|\mathbf{x}\| \leq 1$, $\|\mathbf{x}'\| \leq 1$, with probability at least $1 - \delta$ we have:*

$$\left| \left\langle \frac{\delta f(\theta, \mathbf{x})}{\delta \theta}, \frac{\delta f(\theta, \mathbf{x}')}{\delta \theta} \right\rangle - \Theta^{(L)}(\mathbf{x}, \mathbf{x}') \right| \leq (L + 1)\epsilon.$$

3.2 Equivalence between trained NN and kernel regression

Theorem 3.2. *Suppose $\sigma(z) = \max(0, z)$, $1/\kappa = poly(1/\epsilon, \log(n/\delta))$ and $d_1 = d_2 = \ldots = d_L = m$ with $m \geq poly(1/\kappa, L, 1/\lambda_0, n, \log(1/\delta))$. Then for any $\mathbf{x}_{te} \in \mathbb{R}^d$ with $\|\mathbf{x}_{te}\| = 1$, with probability at least $1 - \delta$ over the random initialization, we have*

$$|f_{nn}(\mathbf{x}_{te}) - f_{ntk}(\mathbf{x}_{te})| \leq \epsilon,$$

where $\lambda_0 = \lambda_{\min}(\mathbf{H}^*)$, $f_{nn}(\theta, \mathbf{x}) = \kappa f(\theta, \mathbf{x})$,

$$f_{nn}(\mathbf{x}_{te}) = \lim_{t \to \infty} f_{nn}(\theta(t), \mathbf{x}_{te}),$$

and

$$f_{ntk} = (\ker_{ntk}(\mathbf{x}_{te}, \mathbf{X}))^\top (\mathbf{H}^*)^{-1} \mathbf{y}.$$

[ADH+19] observe that the training dynamics in Eq. (2) is identical to kernelized regression under gradient flow. Note that this states that the outputs of the kernel regression model and a sufficiently wide NN are ϵ-close. In other words, analyzing the outputs of the kernel regression model (which is far easier to analyze) is equivalent to analyzing the output of the NN. This theorem brings the theory around NTKs full circle.

References

[ADH+19] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net, 2019.

[JGH20] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks, 2020.

[LSdP+18] Jaehoon Lee, Jascha Sohl-dickstein, Jeffrey Pennington, Roman Novak, Sam Schoenholz, and Yasaman Bahri. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

[Nea96] Radford M. Neal. *Priors for Infinite Networks*, pages 29–53. Springer New York, New York, NY, 1996.