Lecture 08: The Neural Tangent Kernel Approximation

Geoff Pleiss

In the last lecture, we discussed a linear approximation to neural networks where the hidden weights are fixed at a random initialized from an i.i.d. standard normal distribution. To summarize:

- In the limit of infinite width ($d \to \infty$), the space of linearly-approximated neural networks converges almost surely to the RKHS with a closed-form kernel.
- With one hidden layer, this limiting kernel is the **arc-cosine kernel**:

$$k_{\text{arccos}}(\boldsymbol{x}, \boldsymbol{x}') := \frac{1}{2\pi}\|\boldsymbol{x}\|\,\|\boldsymbol{x}'\|\left[\sin(\varphi) + (\pi - \varphi)\cos(\varphi)\right], \qquad \varphi = \cos^{-1}\left(\frac{\boldsymbol{x}^\top \boldsymbol{x}'}{\|\boldsymbol{x}\|\,\|\boldsymbol{x}'\|}\right). \tag{1}$$

  Networks with multiple hidden layers have a limiting kernel given by a recursive application of the arc-cosine kernel (see [Lee et al., 2018]).
- This limiting RKHS $\mathcal{H}_{\text{arccos}}$ is a univesally approximating space, meaning that we can any continuous function with arbitrary precision with some function in $\mathcal{H}_{\text{arccos}}$.
- The number of learnable parameters in the linear approximation is d (the number of hidden units), regardless of the depth of the neural network.

We now will turn to a different linear approximation of neural networks; one that is perhaps less intuitive but—as we will see—is far more predictive of actual neural network behaviour.

---

1) Linear Approximation 2: The Neural Tangent Kernel

Recall the equation for our neural network:

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{1}{\sqrt{d}}\sum_{i=1}^{d} \beta_i \sigma(\boldsymbol{w}_i^\top \boldsymbol{x}), \qquad \boldsymbol{\theta} := \begin{bmatrix} \underbrace{\boldsymbol{w}_1^\top}_{\in\mathbb{R}^p} & \cdots & \underbrace{\boldsymbol{w}_d^\top}_{\in\mathbb{R}^p} & \underbrace{\beta_1}_{\in\mathbb{R}} & \cdots & \underbrace{\beta_d}_{\in\mathbb{R}} \end{bmatrix}, \qquad \sigma(z) := \max\{0, z\}. \tag{2}$$

We will now instead assume that all of the neural network parameters ($\boldsymbol{\theta}$) are learnable but that the neural network can be well-approximated by its first order Taylor expansion about its parameters. Defining, $\boldsymbol{\theta}^{(0)}$ (or alternatively $\begin{bmatrix} \boldsymbol{w}_1^{(0)\top} & \cdots & \boldsymbol{w}_d^{(0)\top} & \beta_1^{(0)} & \cdots & \beta_d^{(0)} \end{bmatrix}$) are the randomly initialized parameters of the entire neural network.

$$f(\boldsymbol{x};\boldsymbol{\theta}) \approx f(\boldsymbol{x};\boldsymbol{\theta}^{(0)}) + \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}^{(0)})^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)}) \tag{3}$$

$$= \frac{1}{\sqrt{d}} \begin{bmatrix} \sigma(\boldsymbol{w}_1^{(0)\top}\boldsymbol{x}) \\ \vdots \\ \sigma(\boldsymbol{w}_d^{(0)\top}\boldsymbol{x}) \end{bmatrix}^\top \begin{bmatrix} \beta_1^{(0)} \\ \vdots \\ \beta_d^{(0)} \end{bmatrix} + \frac{1}{\sqrt{d}} \begin{bmatrix} \beta_1^{(0)}\dot{\sigma}(\boldsymbol{w}_1^{(0)\top}\boldsymbol{x})\boldsymbol{x} \\ \vdots \\ \beta_d^{(0)}\dot{\sigma}(\boldsymbol{w}_d^{(0)\top}\boldsymbol{x})\boldsymbol{x} \\ \sigma(\boldsymbol{w}_1^{(0)\top}\boldsymbol{x}) \\ \vdots \\ \sigma(\boldsymbol{w}_d^{(0)\top}\boldsymbol{x}) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{w}_1 - \boldsymbol{w}_1^{(0)} \\ \vdots \\ \boldsymbol{w}_d - \boldsymbol{w}_d^{(0)} \\ \beta_1 - \beta_1^{(0)} \\ \vdots \\ \beta_d - \beta_d^{(0)} \end{bmatrix}$$

$$= \underbrace{\frac{1}{\sqrt{d}} \begin{bmatrix} \beta_1^{(0)}\dot{\sigma}(\boldsymbol{w}_1^{(0)\top}\boldsymbol{x}) \\ \vdots \\ \beta_d^{(0)}\dot{\sigma}(\boldsymbol{w}_d^{(0)\top}\boldsymbol{x}) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{x}^\top\boldsymbol{w}_1^{(0)} \\ \vdots \\ \boldsymbol{x}^\top\boldsymbol{w}_d^{(0)} \end{bmatrix}}_{(0)} + \underbrace{\frac{1}{\sqrt{d}} \begin{bmatrix} \beta_1^{(0)}\dot{\sigma}(\boldsymbol{w}_1^{(0)\top}\boldsymbol{x})\boldsymbol{x} \\ \vdots \\ \beta_d^{(0)}\dot{\sigma}(\boldsymbol{w}_d^{(0)\top}\boldsymbol{x})\boldsymbol{x} \\ \sigma(\boldsymbol{w}_1^{(0)\top}\boldsymbol{x}) \\ \vdots \\ \sigma(\boldsymbol{w}_d^{(0)\top}\boldsymbol{x}) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{w}_1 \\ \vdots \\ \boldsymbol{w}_d \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix}}_{(1)} \tag{4}$$

Note that term (0) in Eq. (4) contains no learnable parameters while term (1) is linear with respect to all $\boldsymbol{w}_i, \beta_i$ parameters. As before, the first order approximation given by Eq. (3) is a linear function of its learnable parameters. However, the number of learnable parameters is now $d + d \cdot p$, rather than d for our first linear approximation. If we were to consider deeper neural networks with L hidden layers, the resulting approximation would be a $d + d \cdot p + (L-1)d^2$-dimensional linear model; one dimension for each learnable parameter. Thus, increasing depth will increase the capacity of the neural network under this approximation.

Rearranging the terms in Eq. (4) and writing the inner products as summations, we have

$$f(\boldsymbol{x};\boldsymbol{\theta}) \approx \underbrace{\frac{1}{\sqrt{d}} \sum_{i=1}^{d} \beta_i \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x})}_{\text{fixed hidden layer approx.}} + \frac{1}{\sqrt{d}} \sum_{i=1}^{d} \beta_i^{(0)} \dot{\sigma}(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) \boldsymbol{x}^\top \left(\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\right). \tag{5}$$

The first term in the summation is simply thet fixed hidden layer approximation we discussed in the last lecture! So we can interpret this Taylor-based approximation as our previous approximation with an additional correction term.

1.1 Another Kernel Machine as $d \to \infty$

Again, we can also consider how to characterize the limiting behaviour of this linear approximation as $d \to \infty$. Assume that all learnable parameters are drawn i.i.d. from $\mathcal{N}(0,1)$. Note that $\boldsymbol{w}_i^{(0)}, \beta_i^{(0)}$ and $\boldsymbol{w}_i^{(0)}, -\beta_i^{(0)}$ are equally likely, and so as $d \to \infty$ with probability 1 we will always have pairs of parameters that cancel each other out and thus term (0) in Eq. (4) will vanish.

As we did in the last lecture, we note that the space of neural networks under this linear approximation is

an RKHS with kernel

$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{d} \begin{bmatrix} \beta_1^{(0)} \dot{\sigma}(\boldsymbol{w}_1^{(0)\top} \boldsymbol{x}) \boldsymbol{x} \\ \vdots \\ \beta_d^{(0)} \dot{\sigma}(\boldsymbol{w}_d^{(0)\top} \boldsymbol{x}) \boldsymbol{x} \\ \sigma(\boldsymbol{w}_1^{(0)\top} \boldsymbol{x}) \\ \vdots \\ \sigma(\boldsymbol{w}_d^{(0)\top} \boldsymbol{x}) \end{bmatrix}^\top \begin{bmatrix} \beta_1^{(0)} \dot{\sigma}(\boldsymbol{w}_1^{(0)\top} \boldsymbol{x}) \boldsymbol{x} \\ \vdots \\ \beta_d^{(0)} \dot{\sigma}(\boldsymbol{w}_d^{(0)\top} \boldsymbol{x}) \boldsymbol{x} \\ \sigma(\boldsymbol{w}_1^{(0)\top} \boldsymbol{x}) \\ \vdots \\ \sigma(\boldsymbol{w}_d^{(0)\top} \boldsymbol{x}) \end{bmatrix}$$

(i.e. the inner product of the basis functions that we apply the learnable parameters to). Taking the limit as $d \to \infty$, we find that the limiting kernel is

$$\begin{aligned} \lim_{d\to\infty} k(\boldsymbol{x}, \boldsymbol{x}') &= \frac{1}{d} \lim_{d\to\infty} \sum_{i=1}^{d} \sigma(\boldsymbol{w}_i^{(0)\top} \boldsymbol{x}) \sigma(\boldsymbol{w}_i^{(0)\top} \boldsymbol{x}') + \frac{1}{d} \lim_{d\to\infty} \sum_{i=1}^{d} \beta_i^{(0)2} \dot{\sigma}(\boldsymbol{w}_i^{(0)\top} \boldsymbol{x}) \dot{\sigma}(\boldsymbol{w}_i^{(0)\top} \boldsymbol{x}') \boldsymbol{x}^\top \boldsymbol{x}' \\ &\overset{\text{a.s.}}{=} \underbrace{\mathbb{E}_{\boldsymbol{w}^{(0)}} \left[\sigma(\boldsymbol{w}^{(0)\top} \boldsymbol{x}) \sigma(\boldsymbol{w}^{(0)\top} \boldsymbol{x}') \right]}_{k_{\text{arccos}}(\boldsymbol{x}, \boldsymbol{x}')} + \boldsymbol{x}^\top \boldsymbol{x}' \mathbb{E}_{\boldsymbol{w}^{(0)}, \beta^{(0)}} \left[\dot{\sigma}(\boldsymbol{w}^{(0)\top} \boldsymbol{x}) \dot{\sigma}(\boldsymbol{w}^{(0)\top} \boldsymbol{x}') \right]. \end{aligned}$$

The first term is simply the arc-cosine kernel in Eq. (1) that we discussed last lecture. The second term can be derived using similar techniques to the last lecture, where we recognize that $\dot{\sigma}(z) = \mathbb{I}(z > 0)$ and that $\boldsymbol{w}^{(0)}$ and $\beta^{(0)}$ are independent. All together we have that

$$\begin{aligned} k_{\text{NTK}}(\boldsymbol{x}, \boldsymbol{x}') := \lim_{d\to\infty} k(\boldsymbol{x}, \boldsymbol{x}') &= \frac{1}{2\pi} \|\boldsymbol{x}\| \, \|\boldsymbol{x}'\| \left[\sin(\varphi) + 2(\pi - \varphi) \cos(\varphi) \right] \\ &= k_{\text{arccos}}(\boldsymbol{x}, \boldsymbol{x}') + \frac{1}{2\pi} \|\boldsymbol{x}\| \, \|\boldsymbol{x}'\| \left[(\pi - \varphi) \cos(\varphi) \right] \end{aligned} \tag{6}$$

Jacot et al. [2018] coined this kernel as the **Neural Tangent Kernel** (NTK), as it is the limiting kernel of the inner product of vectors tangent to the neural network.

1.2 Properties of the Neural Tangent Kernel

The NTK admits nearly identical properties as the arc-cosine kernel:

1. For any $\boldsymbol{x}, \boldsymbol{x}'$ with norm 1, the NTK is a **dot-product kernel**; i.e. it is a function of $\boldsymbol{x}^\top \boldsymbol{x}' = \cos(\varphi)$.
2. For most data distributions, the spectrum of the NTK will decay at a rate of $\lambda_i = i^{-\alpha}$ for some $\alpha > 1$ [Geifman et al., 2020].
3. For neural networks with multiple hidden layers, each of which is infinitely wide[1] the limiting kernel is a recursive application of the NTK. If we replace our fully-connected hidden layers with convolutional, residual, or transformer layers, we end up with a different limiting kernel but one that still matches the "flavour" of the NTK. See [Yang, 2020] for details on deriving limiting kernels for any architecture.
4. The NTK RKHS is a universally approximating function space.

[1] Again, we assume that we take each hidden layer to infinite-width sequentially rather than simultaneously.

2) Goodness of the NTK Approximation

The NTK approximation is not only more expressive than the fixed hidden layer approximation (for a finite d), but it is also more predictive of the behaviour of an actual neural network.

Let's for a second replace $\sigma(z) = \max\{0, z\}$ with a "smoother" $\sigma(z)$. In particular, we will assume that:

1. $|\ddot{\sigma}(z)| \leq \gamma$ for some $\gamma > 0$ for all $z \in \mathbb{R}$ and
2. $|\sigma(z) - \sigma(z')| \leq C|z - z'|$ for some $C > 0$ for all $z \in \mathbb{R}$.

The second assumption is true for ReLU activations; the first is not. Nevertheless, we can approximate the ReLU function to arbitrary precision using a smooth function.[2]

Note that

$$
\begin{aligned}
& f(\boldsymbol{x};\boldsymbol{\theta}) - \left[f(\boldsymbol{x};\boldsymbol{\theta}^{(0)}) + \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}^{(0)})^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)})\right] \\
&= \frac{1}{\sqrt{d}}\left[\underbrace{\sum_{i=1}^{d} \beta_i \sigma(\boldsymbol{w}_i^\top \boldsymbol{x})}_{f(\boldsymbol{x};\boldsymbol{\theta})} - \underbrace{\left(\sum_{i=1}^{d} \beta_i \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) + \sum_{i=1}^{d} \beta_i^{(0)} \dot{\sigma}(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x})\boldsymbol{x}^\top\left(\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\right)\right)}_{\text{Eq. (5)}}\right] \\
&= \frac{1}{\sqrt{d}} \sum_{i=1}^{d}\Bigg[\beta_i \sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) - \beta_i \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) - \beta_i^{(0)}\sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) + \beta_i^{(0)}\sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) \\
&\qquad\qquad + \beta_i^{(0)}\sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) - \beta_i^{(0)}\sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) - \beta_i^{(0)}\dot{\sigma}(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x})\boldsymbol{x}^\top\left(\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\right)\Bigg] \\
&= \frac{1}{\sqrt{d}} \sum_{i=1}^{d} \underbrace{\left(\beta_i - \beta_i^{(0)}\right)\left(\sigma(\boldsymbol{x}^\top \boldsymbol{w}_i) - \sigma(\boldsymbol{x}^\top \boldsymbol{w}_i^{(0)})\right)}_{(a)}. \\
&\quad + \frac{1}{\sqrt{d}} \sum_{i=1}^{d} \beta_i^{(0)} \left[\underbrace{\sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) - \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) - \dot{\sigma}(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x})\boldsymbol{x}^\top\left(\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\right)}_{(b)}\right]
\end{aligned}
$$

Define $z_i^{(0)} = \boldsymbol{x}^\top \boldsymbol{w}_i^{(0)}$ and $z_i = \boldsymbol{x}^\top \boldsymbol{w}_i$S. Combining Taylor's theorem (the theorem that recursively defines the Taylor series) with our assumption, we can bound (b):

$$
\begin{aligned}
\left|\sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) - \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) - \dot{\sigma}(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x})\boldsymbol{x}^\top\left(\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\right)\right| &= \left|\int_{z_i^{(0)}}^{z_i} \ddot{\sigma}(z_i)(z_i - z_i^{(0)})dz_i\right| \\
&\leq \int_{z_i^{(0)}}^{z_i} |\ddot{\sigma}(z_i)| \left|(z_i - z_i^{(0)})\right| dz_i \\
&\leq \gamma \int_{z_i^{(0)}}^{z_i} \left|(z_i - z_i^{(0)})\right| dz_i \\
&= \frac{\gamma}{2}\left(\boldsymbol{x}^\top\left(\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\right)\right)^2 \\
&\leq \frac{\gamma}{2}\|\boldsymbol{x}\|^2 \|\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\|^2.
\end{aligned}
$$

[2]For example, the **softplus** function $\sigma_t(z) = \log(1 + \exp(tz))/t$ for $t > 0$ is one such smooth approximation. Both the approximation fidelity as well as the corresponding β constant increase as $t \to 0$.

where the last inequality is given by plugging in the definitions for z_i and $z_i^{(0)}$ and applying Cauchy-Schwarz. Moreover, if we assume without loss of generality[3] that $|\beta_i - \beta_i^{(0)}| < D\|\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\|$ for some constant $D > 0$, then we can bound (a) as

$$\left|\left(\beta_i - \beta_i^{(0)}\right)\left(\sigma(\boldsymbol{x}^\top \boldsymbol{w}_i) - \sigma(\boldsymbol{x}^\top \boldsymbol{w}_i^{(0)})\right)\right| \leq C\left|\beta_i - \beta_i^{(0)}\right|\left|\boldsymbol{x}^\top\left(\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\right)\right| \leq C\|\boldsymbol{x}\|\|\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\|^2.$$

All together we have that

$$\begin{aligned}\left|f(\boldsymbol{x};\boldsymbol{\theta}) - \left[f(\boldsymbol{x};\boldsymbol{\theta}^{(0)}) + \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}^{(0)})^\top(\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)})\right]\right| &\leq \frac{1}{\sqrt{d}}\left(\frac{\gamma}{2}\|\boldsymbol{x}\|^2 + C\|\boldsymbol{x}\|\right)\sum_{i=1}^{d}\|\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\|^2 \\ &\leq \frac{1}{\sqrt{d}}\left(\frac{\gamma}{2}\|\boldsymbol{x}\|^2 + C\|\boldsymbol{x}\|\right)\|\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)}\|^2,\end{aligned} \tag{7}$$

Curiously Eq. (7) tells us that—if we are only $o(\sqrt{d})$ away from the initialization—this Taylor approximation becomes better and better as $d \to \infty$!

With more realistic assumptions. This technique will not work if we remove that second derivative assumption on σ (necessary for ReLU activations), nor will it work for deeper networks.

To extend this argument to ReLU activations, we will replacce Eq. (7) with a **high probability bound**. In particular, we will show that—with high probability—if $\|\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)}\|$ is not too big, then *most* of the $\boldsymbol{x}^\top \boldsymbol{w}_i$ will have the same sign as $\boldsymbol{x}^\top \boldsymbol{w}_i^{(0)}$. For these non-sign switching activations, we thus have that:

$$\sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) - \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) = (\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)})^\top \boldsymbol{x} \qquad \text{if } \boldsymbol{x}^\top \boldsymbol{w}_i > 0 \tag{8}$$

$$\sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) - \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) = 0 \qquad \text{if } \boldsymbol{x}^\top \boldsymbol{w}_i \leq 0. \tag{9}$$

We thus have that $\sigma(\boldsymbol{w}_i^\top \boldsymbol{x}) - \sigma(\boldsymbol{w}_i^{(0)\top}\boldsymbol{x}) \leq \|\boldsymbol{x}\|\|\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\|$ for most i, and from there we can replicate most of the above argument. See [Ji et al., 2021] for a detailed proof or [Allen-Zhu et al., 2019] for a proof for multi-layer networks.

When is $\|\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)}\|$ small enough? If we're close enough to the initialization, then our linear approximation will be good. However, it is possible that over the course of optimization we will move $> O(\sqrt{d})$ away from the initialization. Surprisingly, in the next lecture we will find that this is not the case for neural networks trained with gradient descent!

References

Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252, 2019.

A. Geifman, A. Yadav, Y. Kasten, M. Galun, D. Jacobs, and B. Ronen. On the similarity between the laplace and neural tangent kernels. *Advances in Neural Information Processing Systems*, 33:1451–1461, 2020.

A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, pages 8571–8580, 2018.

Z. Ji, J. Li, and M. Telgarsky. Early-stopped neural networks are consistent. *Advances in Neural Information Processing Systems*, 34:1805–1817, 2021.

[3] If $|\beta_i - \beta_i^{(0)}| > D\|\boldsymbol{w}_i - \boldsymbol{w}_i^{(0)}\|$, then we simply get a bound in terms of $(\beta_i - \beta_i^{(0)})^2$ instead. We will end up with the same upper bound

J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

G. Yang. Tensor programs II: Neural tangent kernel for any architecture. *arXiv preprint arXiv:2006.14548*, 2020.