Neural Tangent Kernel

Convergence and Generalization in DNNs

Arthur Jacot, Franck Gabriel, Clément Hongler

Ecole Polytechnique Fédérale de Lausanne

March 18, 2019

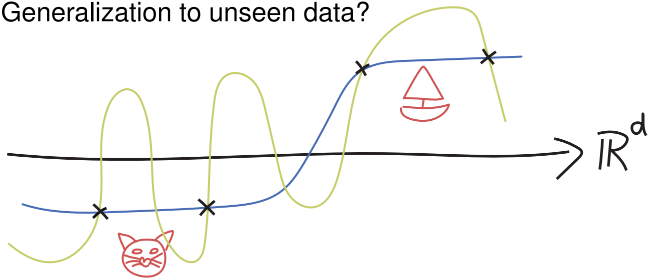
Searching in a function space

images labels

- Training set (x_i, y_i) of size N
- Optimize in a function space $\mathcal{F}$:

$$\min_{f \in \mathcal{F}} C(f) = \min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - y_i)^2$$

  - Efficiency for large datasets/input dimensions?
  - Generalization to unseen data?



Linear Model: Random Features

- Choose P features $f^{(p)} \in \mathcal{F}$
- Define a parametrization of functions $F : \mathbb{R}^P \to \mathcal{F}$:

parameters ↓

$$F(\theta) := f_\theta = \frac{1}{\sqrt{P}} \sum_{p=1}^{P} \theta_p f^{(p)}$$

Example:
choose the features $f^{(p)}$ iid with $\mathbb{E}[f^{(p)}(x) f^{(p)}(y)] = K(x, y)$.

What does it converge to?

Gradient descent on the composition $C \circ F$

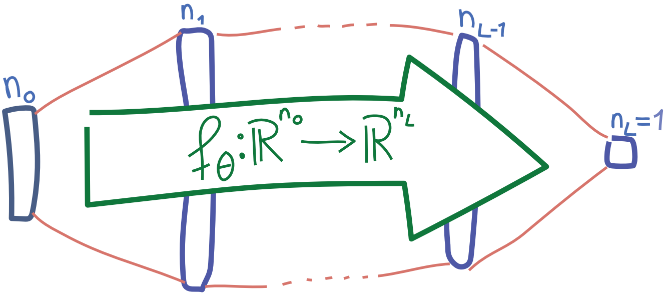
$$\mathbb{R}^P \xrightarrow{F} \mathcal{F} \xrightarrow{C} \mathbb{R}$$

1. Underparametrized $P < N$:
   Strictly convex $\Rightarrow$ unique solution
2. Overparametrized $P > N$:
   Convex $\Rightarrow$ minimal norm solution
3. Infinite parameters limit $P \to \infty$:
   Kernel regression w.r.t the kernel K
   Gaussian process prior $\mathcal{N}(0, K)$

Nonlinear Model: Neural Networks

- $L+1$ layers each containing n_ℓ neurons
- Non-linearity function $\sigma : \mathbb{R} \to \mathbb{R}$ e.g. $\sigma(x) = \max(x, 0)$
- Parameters $\theta = \left(W^{(0)}, ..., W^{(L-1)}\right)$, $W^{(\ell)} : \mathbb{R}^{n_\ell} \to \mathbb{R}^{n_{\ell+1}}$
- Non-linear parametrization $F^{(L)}(\theta) = f_\theta$:

$$\alpha^{(0)}(x) = x \xrightarrow{\frac{1}{\sqrt{n_0}} W^{(0)}} \tilde{\alpha}^{(1)} \xrightarrow{\sigma} \alpha^{(1)} \xrightarrow{\frac{1}{\sqrt{n_0}} W^{(1)}} \dots \xrightarrow{\frac{1}{\sqrt{n_{L-1}}} W^{(L-1)}} \tilde{\alpha}^{(L)} =: f_\theta$$

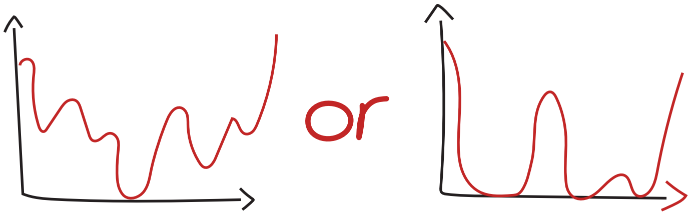


Loss surface

The loss $C \circ F^{(L)}$ is non-convex

1. Symmetries: swapping neurons
2. No bad local minima if the network is large enough
3. Similarity to physical models

$\Rightarrow$ gradient descent works well in practice for large networks
$\Rightarrow$ study the infinite-width limit ($n_1, \ldots n_{L-1} \to \infty$)

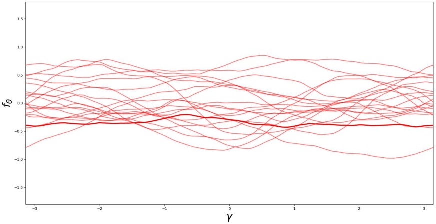


Initialization: DNNs as Gaussian processes

- Initialize the parameters $\theta \sim \mathcal{N}(0, Id_P)$.
- In the infinite width limit $n_1, ..., n_{L-1} \to \infty$ the preactivations $\tilde{\alpha}_i^{(\ell)}(\cdot;\theta) : \mathbb{R}^{n_0} \to \mathbb{R}$ are iid Gaussian processes of covariance $\Sigma^{(\ell)}$:

$$\Sigma^{(1)}(x,y) = x^T y + 1$$
$$\Sigma^{(\ell+1)}(x,y) = \mathbb{E}_{\alpha \sim \mathcal{N}\left(0,\Sigma^{(\ell)}\right)}\left[\sigma(\alpha(x))\sigma(\alpha(y))\right]$$

- In particular f_θ is a Gaussian processes of covariance $\Sigma^{(L)}$.

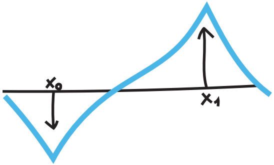


Training: Neural Tangent Kernel

- Gradient descent:

$$\partial_t \theta_p = -\partial_\theta (C \circ F^{(L)}) = \frac{2}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i)) \partial_{\theta_p} f_\theta(x_i)$$

- Evolution of f_θ:



$$\partial_t f_\theta(x) = \sum_{p=1}^{P} \partial_t \theta_p \partial_{\theta_p} f_\theta(x)$$

$$= \frac{2}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i)) \left(\sum_{p=1}^{P} \partial_{\theta_p} f_\theta(x_i) \partial_{\theta_p} f_\theta(x) \right)$$

- Neural Tangent Kernel (NTK):

$$\Theta^{(L)}(x, y) := \sum_{p=1}^{P} \partial_{\theta_p} f_\theta(x) \partial_{\theta_p} f_\theta(y)$$

Asymptotics of the NTK

Problem:
The NTK is random at initialization and varies during training!

Theorem *(NeurIPS 2018)*: Let $n_1, \ldots, n_{L-1} \to \infty$, for any $t < T$:

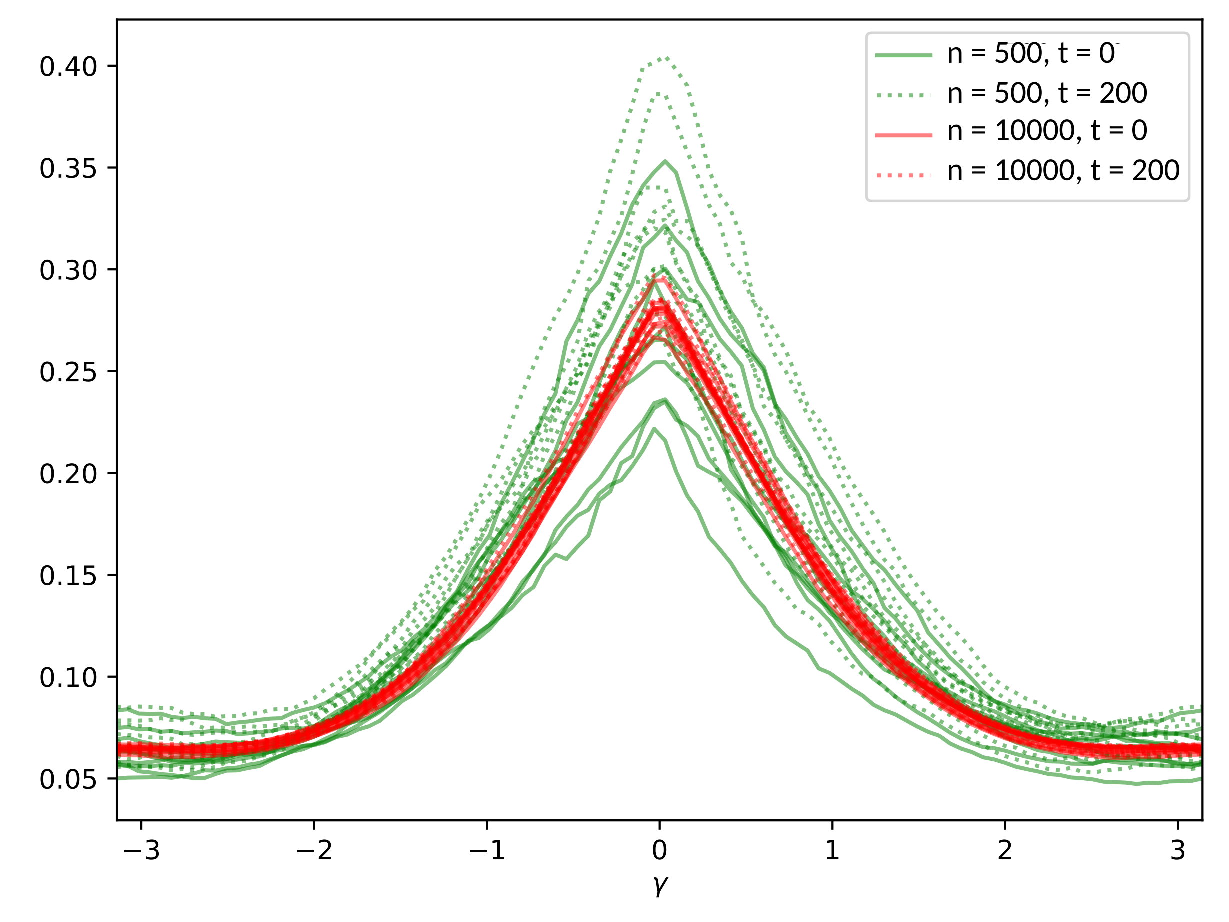
$$\Theta^{(L)}(t) \to \Theta^{(L)}_{\infty}$$

where

$$\Theta^{(L)}_{\infty}(x, y) = \sum_{\ell=1}^{L} \Sigma^{(\ell)}(x, y)\dot{\Sigma}^{(\ell+1)}(x, y) \ldots \dot{\Sigma}^{(L)}(x, y)$$

with

$$\dot{\Sigma}^{(L)}(x, x') = \mathbb{E}_{\alpha \sim \mathcal{N}(0, \Sigma^{(L-1)})}[\dot{\sigma}(\alpha(x))\dot{\sigma}(\alpha(x'))]$$



0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
−3
−2
−1
0
1
2
3
γ
n = 500, t = 0
n = 500, t = 200
n = 10000, t = 0
n = 10000, t = 200

Kernel gradient descent

Kernel $\Rightarrow$ Hilbert space of functions $\Rightarrow$ Kernel Gradient

$$\partial_f C = \left\langle \nabla_{\Theta_\infty} C(f_{\theta(t)}), \cdot \right\rangle$$

Complete infinite-width dynamics:

$$f_{\theta(0)} \sim \mathcal{N}(0, \Sigma^{(L)})$$
$$\partial_t f_{\theta(t)} = -\nabla_{\Theta_\infty} C(f_{\theta(t)})$$

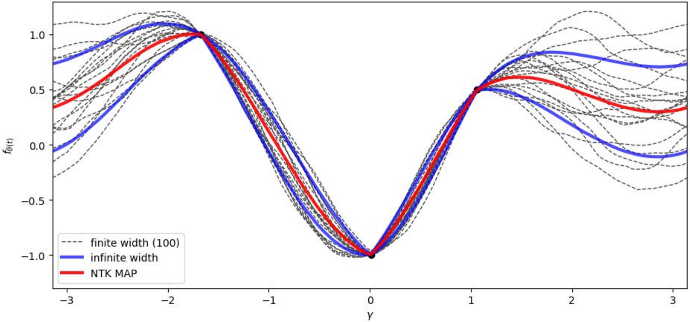
positive definite NTK $\Longrightarrow$ convergence to a global minimum

Consequences

In the infinite-width limit, DNNs converge to:

- Least-squares cost $\Rightarrow$ kernel regression (in expectation)
- Cross-entropy losses $\Rightarrow$ kernel maximum margin classifier
- Early stopping acts as a regularization

Bayesian interpretation: Gaussian process prior $\mathcal{N}(0, \Theta_\infty)$



Ideas of the proofs

- Initialization: sequential law of large numbers to show the convergence of the NTK $\Theta^{(\ell)}$ of subnetworks.
- Training: Grönwall
  - Growing number of parameters => they move less individually
  - The activations move less and less
  - The NTK $\Theta^{(\ell)}$ of subnetworks become fixed
- Appears to generalize to other architectures

Tangent kernel for linear models

- For linear models $\partial_{\theta_p} f_\theta = \frac{1}{\sqrt{P}} f^{(p)}$
- The Tangent Kernel is constant

$$\Theta^{lin}(x,y) = \frac{1}{P}\sum f^{(p)}(x) f^{(p)}(y)$$
$$\xrightarrow[P\to\infty]{} \mathbb{E}[f^{(p)}(x) f^{(p)}(y)] = K(x,y)$$

- DNNs behave like linear models when $P \to \infty$!
  - Actually $\left\| \mathcal{H} F^{(L)} \right\|_{op}$ is $\mathcal{O}(n_\ell^{-1/2+\epsilon})$
  - But there is more: $\partial_t \Theta^{(L)}$ is $\mathcal{O}(n_\ell^{-1})$

DNNs as linear models

1. Rich random features from simple and fast computations (GPUs)
2. The weights serve both as parameters and as source of randomness
3. Different architectures:
   3.1 Convolutional networks
   3.2 Recurrent networks
   3.3 Attention mechanism
   3.4 And many more
4. But there is still a gap in performance which is not explained by the NTK

Conclusion

1. The NTK gives a complete description of infinite-width DNNs
2. In this limit, DNNs behave like linear models!
3. Is there an actual advantage to the non-linearity?

Thank you!