On the Inductive Bias of Neural Tangent Kernels

Alberto Bietti  Julien Mairal -  Inria Grenoble

Inductive Bias and Over-Parameterization

Optimization and Inductive Bias:
- Over-parameterized deep networks have great approximation power
- Optimization algorithm is plays a crucial role for generalization

Lazy Training: In certain regimes (over-parameterization, particular initialization), neural networks behave like their linearization near initialization
\[ f(x; \theta) \approx f(x; \theta_0) + \langle \theta - \theta_0, \nabla f(x; \theta_0) \rangle \]

Neural Tangent Kernels (NTK): In this regime, generalization properties are controlled by the limiting kernel [Jacot et al., 2018]
\[ \langle \nabla \phi f(x; \theta_0), \nabla \phi f(x'; \theta_0) \rangle \rightarrow K(x, x') \]
In particular, with squared loss and infinite width, we get the interpolating solution with minimum RKHS norm.

Contributions:
- Derivation of NTK for convolutional networks with generic linear patch extraction/pooling operators;
- Study of smoothness, stability, and approximation properties of functions with finite RKHS norm;
- Comparison to other ReLU kernels (e.g., training only last layer): the NTK has weaker smoothness properties but better approximation.

Neural Tangent Kernels for CNNs

Two-layer ReLU Networks: \( f(x; \theta) = \sqrt{\sum_{j=1}^{n} \phi_j(w_j^T x)} \), NTK given by
\[ K(x, x') = \|x\|\|x'\|\kappa \left( \frac{(x, x')}{\|x\|\|x'\|} \right) \]
where \( \kappa(u) = \kappa_0(u) + \kappa_1(u) \),
\[ \kappa_0(u) = \frac{1}{\pi} (\pi - \arccos(u)), \quad \kappa_1(u) = \frac{1}{\pi} \left( u (\pi - \arccos(u)) + \sqrt{1 - u^2} \right). \]

Convolutional networks:
- Signals \( x[u] \in \mathbb{R}^{2(Z^d)} \)
- Patch extraction operators \( P^d x[u] = [S_{|y|^{1/2}}(x[u + v])]_{v \in S_1} \in \mathcal{H}[|y|] \)
- Linear pooling operators \( A^d x[u] = \sum_{y \in Z^d} a^d[y] (x[u - v]) \)

Network: \( f(x; \theta) = \sqrt{\sum_{j=1}^{n} \phi_j(w_j^T x)} \), with
\[ a^d[y] = \sqrt{2} m_{y} W^d A^{d-1} \]

NTK: Consider the non-linear operator
\[ M(x, y)[u] = \langle \phi(x[u]) \otimes \phi(y[u]) \rangle_{x \in \mathbb{R}^d} \]
with \( \phi_0, \phi_1 \) are kernel mappings for kernels \( \kappa_0 \) and \( \kappa_1 \).

Proposition (NTK feature map for CNN)
The NTK is given by
\[ K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{x \in \mathbb{R}^d} \]
with \( \Phi(x)[u] = A^d M(x_0, y_0)[u], y_1[u] = x_1[u] = P^1 x[u] \) and
\[ x_0[u] = P^d A^{d-1} \phi(x_0[u]) \]
\[ y_0[u] = P^d A^{d-1} M(x_{d-1}, y_{d-1})[u], \]
\[ \text{with the notation } \phi_1(x)[u] = \phi_1(x[u]) \text{ for a signal } x. \]

Smoothness and Deformation Stability

Two-layer ReLU networks: The NTK (when training both layers) has weaker smoothness compared to training only the second layer.

Proposition (Non-Lipschitzness)
The kernel mapping \( \Phi(\cdot) \) of the two-layer NTK is not Lipschitz:
\[ \sup_{x \neq y} \frac{\|\Phi(x) - \Phi(y)\|_{\mathcal{H}[|y|]}}{\|x - y\|} \rightarrow +\infty. \]

It follows that the RKHS \( \mathcal{H}[|y|] \) contains unit-norm functions with arbitrarily large Lipschitz constant.

Proposition (Smoothness for ReLU NTK)
The kernel mapping \( \Phi \) satisfies
\[ \|\Phi(x) - \Phi(y)\|_{\mathcal{H}[|y|]} \leq \sqrt{\min(\|x\|, \|y\|)}\|x - y\| + 2\|x - y\|. \]

Deformation stability for deep ReLU CNNs: Similar assumptions to [Bietti and Mairal, 2019]
- Continuous signals \( x(u) \) in \( L^2(\mathbb{R}^d) \), deformations \( L.x(u) = x(u + \tau(u)) \)
- Anti-aliasing of the original signal: \( k \neq 0 \) instead of \( k \)
- Patch sizes controlled at current resolution: \( \sup_{x \in S_1} |v| = \beta_{k-1} \)

Proposition (Stability of NTK)
Let \( \phi_0(x) = \Phi(A_k x), \) and assume \( \|\nabla \tau\|_{\infty} \leq 1/2. \) We have:
\[ \|\phi_0(L.x) - \Phi(x)\|_{\mathcal{H}[|y|]} \leq (C_1 n^{d/2} \|\nabla \tau\|_{\infty}^{1/2} + C_2 n^{d} \|\nabla \tau\|_{\infty} + \sqrt{n} \sigma n^{-d/2} \|\nabla \tau\|_{\infty}) \|x\|. \]

Worse dependence on \( \|\nabla \tau\|_{\infty} \) for small deformations compared to training just the last layer!

Approximation Properties
Q: How rich is the RKHS for the NTK \( \kappa \) versus the simpler kernel \( \kappa_1 \) obtained by training just the second layer?

Mercer decomposition with spherical harmonics:

Proposition (Mercer decomposition)
For any \( x, y \in S^{p-1} \), we have the following decomposition of the NTK \( \kappa \)
\[
\kappa((x, y)) = \sum_{k=0}^{\infty} \sum_{j=1}^{n(k)} \mu_k Y_{j,k}(x) Y_{j,k}(y).
\]
where \( Y_{j,k} \) are spherical harmonic polynomials of degree \( k \), and the non-negative eigenvalues \( \mu_k \) satisfy \( \mu_0, \mu_1 > 0, \mu_k = 0 \text{ if } k = 2j + 1 \) with \( j \geq 1 \), and otherwise \( \mu_k \approx C(p)k^{-p} \) as \( k \rightarrow \infty \).

This gives an explicit characterization of the RKHS norm of a function.

Approximation results: (following [Bach 2017])
- The RKHS is "larger": slower decay compared to \( \kappa_1 \), for which \( \mu_k = O(k^{-p/2}) \);
- Contains functions with weaker requirements on derivatives;
- Better rates for approximating Lipschitz functions on the sphere.

Relevant References
  Breaking the curse of dimensionality with convex neural networks.
  Invariance and stability of deep convolutional representations.