When do neural networks outperform kernel methods?

Song Mei

Stanford University

June 29, 2020

Joint work with Behrooz Ghorbani, Theodor Misiakiewicz, and Andrea Montanari
Neural tangent model

- Multi-layers NN: $f_N(x; \theta), \ x \in \mathbb{R}^d, \ \theta \in \mathbb{R}^N$

- Expanding around $\theta_0$:
  
  \[ f_N(x; \theta) = f_N(x; \theta_0) + \langle \theta - \theta_0, \nabla_{\theta} f_N(x; \theta_0) \rangle + o(||\theta - \theta_0||_2). \]

- Neural tangent model:
  
  \[ f_{NT,N}(x; \beta, \theta_0) = \langle \beta, \nabla_{\theta} f_N(x; \theta_0) \rangle. \]

- Coupled gradient flow:
  
  \[
  \begin{align*}
  \frac{d}{dt} \theta^t &= - \nabla_{\theta} \hat{E}[(y - f_N(x; \theta^t))^2], \quad \theta^0 = \theta_0, \\
  \frac{d}{dt} \beta^t &= - \nabla_{\beta} \hat{E}[(y - f_{NT,N}(x; \beta^t, \theta_0))^2], \quad \beta^0 = 0. 
  \end{align*}
  \]

- Under proper initialization and over-parameterization:
  
  \[
  \lim_{N \to \infty} |f_N(x; \theta^t) - f_{NT,N}(x; \beta^t)| = 0.
  \]
Neural tangent model

- Multi-layers NN: $f_N(x; \theta), x \in \mathbb{R}^d, \theta \in \mathbb{R}^N$

- Expanding around $\theta_0$:

$$f_N(x; \theta) = f_N(x; \theta_0) + \langle \theta - \theta_0, \nabla_{\theta} f_N(x; \theta_0) \rangle + o(||\theta - \theta_0||_2).$$

- Neural tangent model:

$$f_{NT,N}(x; \beta, \theta_0) = \langle \beta, \nabla_{\theta} f_N(x; \theta_0) \rangle.$$

- Coupled gradient flow:

$$\frac{d}{dt} \theta^t = -\nabla_{\theta} \mathbb{E}[(y - f_N(x; \theta^t))^2], \quad \theta^0 = \theta_0,$$

$$\frac{d}{dt} \beta^t = -\nabla_{\beta} \mathbb{E}[(y - f_{NT,N}(x; \beta^t, \theta_0))^2], \quad \beta^0 = 0.$$

- Under proper initialization and over-parameterization:

$$\lim_{N \to \infty} |f_N(x; \theta^t) - f_{NT,N}(x; \beta^t)| = 0.$$
Neural tangent model

- Multi-layers NN: \( f_N(x; \theta), x \in \mathbb{R}^d, \theta \in \mathbb{R}^N \)

- Expanding around \( \theta_0 \):
  \[
  f_N(x; \theta) = f_N(x; \theta_0) + \langle \theta - \theta_0, \nabla_{\theta} f_N(x; \theta_0) \rangle + o(||\theta - \theta_0||_2).
  \]

- Neural tangent model:
  \[
  f_{NT,N}(x; \beta, \theta_0) = \langle \beta, \nabla_{\theta} f_N(x; \theta_0) \rangle.
  \]

- Coupled gradient flow:
  \[
  \frac{d}{dt} \theta^t = -\nabla_{\theta} \mathbb{E}[(y - f_N(x; \theta^t))^2], \quad \theta^0 = \theta_0,
  \]
  \[
  \frac{d}{dt} \beta^t = -\nabla_{\beta} \mathbb{E}[(y - f_{NT,N}(x; \beta^t, \theta_0))^2], \quad \beta^0 = 0.
  \]

- Under proper initialization and over-parameterization:
  \[
  \lim_{N \to \infty} |f_N(x; \theta^t) - f_{NT,N}(x; \beta^t)| = 0.
  \]

---

[Jacot, Gabriel, Hongler, 2018], [Du, Zhai, Poczos, Singh, 2018], [Chizat, Bach, 2018b], ....
Neural tangent model

- **Multi-layers NN:** \( f_N(x; \theta), \ x \in \mathbb{R}^d, \ \theta \in \mathbb{R}^N \)

- **Expanding around \( \theta_0 \):**
  \[
  f_N(x; \theta) = f_N(x; \theta_0) + \langle \theta - \theta_0, \nabla_\theta f_N(x; \theta_0) \rangle + o(||\theta - \theta_0||_2).
  \]

- **Neural tangent model:**
  \[
  f_{NT,N}(x; \beta, \theta_0) = \langle \beta, \nabla_\theta f_N(x; \theta_0) \rangle.
  \]

- **Coupled gradient flow:**
  \[
  \frac{d}{dt} \theta^t = - \nabla_\theta \mathbb{E}[(y - f_N(x; \theta^t))^2], \quad \theta^0 = \theta_0,
  \]
  \[
  \frac{d}{dt} \beta^t = - \nabla_\beta \mathbb{E}[(y - f_{NT,N}(x; \beta^t, \theta_0))^2], \quad \beta^0 = 0.
  \]

- **Under proper initialization and over-parameterization:**
  \[
  \lim_{N \to \infty} |f_N(x; \theta^t) - f_{NT,N}(x; \beta^t)| = 0.
  \]

---

[Jacot, Gabriel, Hongler, 2018], [Du, Zhai, Poczos, Singh, 2018], [Chizat, Bach, 2018b], ...
Neural tangent model

- Multi-layers NN: \( f_N(x; \theta), x \in \mathbb{R}^d, \theta \in \mathbb{R}^N \)

- Expanding around \( \theta_0 \):
  \[
  f_N(x; \theta) = f_N(x; \theta_0) + \langle \theta - \theta_0, \nabla_{\theta} f_N(x; \theta_0) \rangle + o(||\theta - \theta_0||_2).
  \]

- Neural tangent model:
  \[
  f_{NT,N}(x; \beta, \theta_0) = \langle \beta, \nabla_{\theta} f_N(x; \theta_0) \rangle.
  \]

- Coupled gradient flow:
  \[
  \frac{d}{dt} \theta^t = - \nabla_{\theta} \mathbb{E}[(y - f_N(x; \theta^t))^2], \quad \theta^0 = \theta_0,
  \]
  \[
  \frac{d}{dt} \beta^t = - \nabla_{\beta} \mathbb{E}[(y - f_{NT,N}(x; \beta^t, \theta_0))^2], \quad \beta^0 = 0.
  \]

- Under proper initialization and over-parameterization:
  \[
  \lim_{N \to \infty} |f_N(x; \theta^t) - f_{NT,N}(x; \beta^t)| = 0.
  \]

---

Jacot, Gabriel, Hongler, 2018], [Du, Zhai, Poczos, Singh, 2018], [Chizat, Bach, 2018b], ...
How about generalization?

- [Arora, Du, Hu, Li, Salakhutdinov, Wang, 2019]: Cifar10 experiments. NT: 23% test error. NN: less than 5% test error.

- [Arora, Du, Li, Salakhutdinov, Wang, Yu, 2019]: Small dataset, NT sometimes generalize better than NN.


Sometimes there is a large gap, while sometimes the gap is small.
How about generalization?

- [Arora, Du, Hu, Li, Salakhutdinov, Wang, 2019]:
  Cifar10 experiments. NT: 23% test error. NN: less than 5% test error.

- [Arora, Du, Li, Salakhutdinov, Wang, Yu, 2019]:
  Small dataset, NT sometimes generalize better than NN.

  [Li, Wang, Yu, Du, Hu, Salakhutdinov, Arora, 2019]:
  Smaller gap between NT and NN on Cifar10 (10% for NT).

Sometimes there is a large gap, while sometimes the gap is small.
Focus of this talk

When is there a large performance gap between NN and NT?
Two-layers neural networks

Neural networks:

\[ \mathcal{F}_{NN, N} = \left\{ f_N(x; \Theta) = \sum_{i=1}^{N} a_i \sigma(\langle w_i, x \rangle) : a_i \in \mathbb{R}, w_i \in \mathbb{R}^d \right\}. \]

Linearization:

\[ f_N(x; \Theta) = f_N(x; \Theta^0) + \sum_{i=1}^{N} \Delta a_i \sigma(\langle w_i^0, x \rangle) + \sum_{i=1}^{N} a_i^0 \sigma'(\langle w_i^0, x \rangle) \langle \Delta w_i, x \rangle + o(\cdot). \]

\[ \text{Top layer linearization} \quad \text{Bottom layer linearization} \]

Linearized neural networks (\( W = (w_i)_{i \in [N]} \sim iid \text{ Unif}(S^{d-1}) \)):

\[ \mathcal{F}_{RF, N}(W) = \left\{ f = \sum_{i=1}^{N} a_i \sigma(\langle w_i, x \rangle) : a_i \in \mathbb{R}, i \in [N] \right\}, \]

\[ \mathcal{F}_{NT, N}(W) = \left\{ f = \sum_{i=1}^{N} \sigma'(\langle w_i, x \rangle) \langle b_i, x \rangle : b_i \in \mathbb{R}^d, i \in [N] \right\}. \]
Two-layers neural networks

Neural networks:

\[ \mathcal{F}_{NN,N} = \left\{ f_N(x; \Theta) = \sum_{i=1}^{N} a_i \sigma(\langle w_i, x \rangle) : a_i \in \mathbb{R}, w_i \in \mathbb{R}^d \right\}. \]

Linearization:

\[ f_N(x; \Theta) = f_N(x; \Theta^0) + \sum_{i=1}^{N} \Delta a_i \sigma(\langle w_i^0, x \rangle) + \sum_{i=1}^{N} a_i^0 \sigma'(\langle w_i^0, x \rangle) \langle \Delta w_i, x \rangle + o(\cdot). \]

Linearized neural networks \((W = (w_i)_{i \in [N]} \sim iid \text{ Unif}(S^{d-1})):\)

\[ \mathcal{F}_{RF,N}(W) = \left\{ f = \sum_{i=1}^{N} a_i \sigma(\langle w_i, x \rangle) : a_i \in \mathbb{R}, i \in [N] \right\}, \]

\[ \mathcal{F}_{NT,N}(W) = \left\{ f = \sum_{i=1}^{N} \sigma'(\langle w_i, x \rangle) \langle b_i, x \rangle : b_i \in \mathbb{R}^d, i \in [N] \right\}. \]
Spiked features model

- Signal features and junk features
  \[ x = (x_1, x_2) \in \mathbb{R}^d, \quad x_1 \in \mathbb{R}^{d_s}, \quad x_2 \in \mathbb{R}^{d-d_s}, \]
  \[ d_s = d^\eta, \quad 0 \leq \eta \leq 1, \]
  \[ \text{Cov}(x_1) = \text{snr}_f \cdot I_{d_s}, \quad \text{Cov}(x_2) = I_{d-d_s}, \]
  \[ \text{snr}_f = d^{\kappa}, \quad 0 \leq \kappa < \infty \quad \text{(feature SNR)}. \]

- Response depend on signal features
  \[ y = f_\star(x) + \epsilon, \quad f_\star(x) = \varphi(x_1). \]

- Feature SNR: \( \text{snr}_f = d^{\kappa} \geq 1. \)
- Effective dimension: \( d_{\text{eff}} = d_s \vee (d/\text{snr}_f). \) We have \( d_s \leq d_{\text{eff}} \leq d. \)
- Larger \( \text{snr}_f \) induces smaller \( d_{\text{eff}}. \)

More precisely: \( x \sim \text{Unif}(S_{d_s}^d(r\sqrt{d_s})) \times \text{Unif}(S_{d-d_s}^d(\sqrt{d})). \) Generalizable to multi-spheres.
Spiked features model

- Signal features and junk features
  \[ x = (x_1, x_2) \in \mathbb{R}^d, \quad x_1 \in \mathbb{R}^{d_s}, \quad x_2 \in \mathbb{R}^{d-d_s}, \]
  \[ d_s = d^\eta, \quad 0 \leq \eta \leq 1, \]
  \[ \text{Cov}(x_1) = \text{snr}_f \cdot I_{d_s}, \quad \text{Cov}(x_2) = I_{d-d_s}, \]
  \[ \text{snr}_f = d^\kappa, \quad 0 \leq \kappa < \infty \quad \text{(feature SNR)}. \]

- Response depend on signal features
  \[ y = f_*(x) + \varepsilon, \quad f_*(x) = \varphi(x_1). \]

Feature SNR: \( \text{snr}_f = d^\kappa \geq 1. \)

Effective dimension: \( d_{\text{eff}} = d_s \vee (d/\text{snr}_f). \) We have \( d_s \leq d_{\text{eff}} \leq d. \)

Larger \( \text{snr}_f \) induces smaller \( d_{\text{eff}}. \)

More precisely: \( x \sim \text{Unif}(S^{d_s}(r\sqrt{d_s})) \times \text{Unif}(S^{d-d_s}(\sqrt{d})). \) Generalizable to multi-spheres.
Spiked features model

- Signal features and junk features
  \[
  \mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2) \in \mathbb{R}^d, \quad \mathbf{x}_1 \in \mathbb{R}^{d_s}, \quad \mathbf{x}_2 \in \mathbb{R}^{d-d_s}, \\
  d_s = d^\eta, \quad 0 \leq \eta \leq 1, \\
  \text{Cov}(\mathbf{x}_1) = \text{snr}_f \cdot \mathbf{I}_{d_s}, \quad \text{Cov}(\mathbf{x}_2) = \mathbf{I}_{d-d_s}, \\
  \text{snr}_f = d^\kappa, \quad 0 \leq \kappa < \infty \quad \text{(feature SNR)}.
  \]

- Response depend on signal features
  \[
  y = f_*(\mathbf{x}) + \varepsilon, \quad f_*(\mathbf{x}) = \varphi(\mathbf{x}_1).
  \]

- Feature SNR: \( \text{snr}_f = d^\kappa \geq 1 \).
- Effective dimension: \( d_{\text{eff}} = d_s \lor (d/\text{snr}_f) \). We have \( d_s \leq d_{\text{eff}} \leq d \).
- Larger \( \text{snr}_f \) induces smaller \( d_{\text{eff}} \).

\[ f_*(\mathbf{x}_1, \mathbf{x}_2) = \varphi(\mathbf{x}_1) \]

**Figure:** Isotropic features: \( \kappa = 0, \text{snr}_f = 1 \)

More precisely: \( \mathbf{x} \sim \text{Unif}(\mathbb{S}^{d_s}(r\sqrt{d_s})) \times \text{Unif}(\mathbb{S}^{d-d_s}(\sqrt{d})) \). Generalizable to multi-spheres.
Approximation error with $N$ neurons

Approximation error: $R(f_*, \mathcal{F}) = \inf_{f \in \mathcal{F}} \|f_* - f\|_{L^2}^2$.

**Theorem (Ghorbani, Mei, Misiakiewicz, Montanari, 2020)**

Assume $d_{\text{eff}}^{\ell+\delta} \leq N \leq d_{\text{eff}}^{\ell+1-\delta}$ and “generic condition” on $\sigma$, we have

$$R(f_*, \mathcal{F}_{RF,N}(W)) = \|P_{\ell} f_*\|_{L^2}^2 + o_{d,P}(\cdot),$$

$$R(f_*, \mathcal{F}_{NT,N}(W)) = \|P_{\ell+1} f_*\|_{L^2}^2 + o_{d,P}(\cdot).$$

On the contrary, assume $d_s^{\ell+\delta} \leq N \leq d_s^{\ell+1-\delta}$, we have

$$R(f_*, \mathcal{F}_{NN,N}) \leq \|P_{\ell+1} f_*\|_{L^2}^2 + o_d(\cdot).$$

Moreover, $R(f_*, \mathcal{F}_{NN,N})$ is independent of $\text{snr}_f$.

---

$P_{\ell}$: projection orthogonal to the space of degree-$\ell$ polynomials.
Approximation error with $N$ neurons

Dim $d_{\text{eff}} \equiv d_s \lor (d/\text{snr}_f)$ and $d_s \leq d_{\text{eff}} \leq d$.

To approx. a degree-$\ell$ poly. in $x_1$:

- NN need at most $d_s \ell$ parameters*
- RF need $d_{\text{eff}} \ell$ parameters
- NT need $d_{\text{eff}}^{\ell-1} \cdot d$ parameters

Approximation power: NN $\geq$ RF $\geq$ NT.

* If we don’t count parameters with value 0.
Extreme case: low feature SNR

Fix $0 < \eta < 1$, low $\text{snr}_f$: $\kappa = 0$.

To approx. a degree-$\ell$ poly. in $x_1$:

- NN need at most $d^{\eta \ell}$ parameters*.
- RF need $d^\ell$ parameters.
- NT need $d^\ell$ parameters.

Approximation power: $\text{NN} > \text{RF} = \text{NT}$.

* If we don’t count parameters with value 0.

Figure: Isotropic features: $\kappa = 0$, $\text{snr}_f = 1$
Extreme case: high feature SNR

Fix $0 < \eta < 1$, high $\text{snr}_f$: $\kappa \gg 1$.

To approx. a degree-$\ell$ poly. in $x_1$:

- NN need at most $d^{\eta \ell}$ parameters*.
- RF need $d^{\eta \ell}$ parameters.
- NT need $d^{\eta (\ell - 1) + 1}$ parameters.

Approximation power: NN $\sim$ RF $>$ NT.

* If we don’t count parameters with value 0.

**Figure:** Anisotropic features: $\kappa > 0$, $\text{snr}_f > 1$
Numerical simulations

![Graph showing numerical simulations results]

Colorbar: $\kappa \in [0, 1]$.
Dot-dashed: NN.
Dashed lines: RF;
Continuous lines: NT;
Dimension: $d = 1024$.
Eff. dim: $d_s = 16$.

Conclusion

(a) Power: $\text{NN} \geq \text{RF} \geq \text{NT}$. (b) Risk of NN independent of $\text{snr}_f$.
(c) Larger $\text{snr}_f$ induces larger power of $\{\text{RF, NT}\}$. 
Similar results for generalization error with finite samples $n$
Extreme case: low feature SNR

Fix $0 < \eta < 1$, low $\text{snr}_f$: $\kappa = 0$.

To fit a degree-$\ell$ poly. in $x_1$:

- $\exists \nu$, NN need at most $d^{\ell}$ samples.
- $\{\text{RF, NT}\}$ kernel need $d^{\ell}$ samples.

Potential generalization power:
NN > Kernel methods.
Extreme case: high feature SNR

Fix \( 0 < \eta < 1 \), high \( \text{snr}_f \): \( \kappa \gg 1 \).

To fit a degree-\( \ell \) poly. in \( x_1 \):

- \( \exists \nu \), NN need at most \( d^{m \ell} \) samples.
- \( \{ \text{RF, NT} \} \) kernel need \( d^{\eta \ell} \) samples.

Potential generalization power:
NN \( \sim \) Kernel methods.

\[
\begin{align*}
\text{Figure: Anisotropic features:} & \\
\kappa > 0, \; \text{snr}_f > 1
\end{align*}
\]
Implications

Adding isotropic noise in features (i.e., decreasing $\text{snr}_f$), performance gap between NN and $\{\text{RF, NT}\}$ becomes larger.
Numerical simulations

Figure: Underlying assumption: labels depend on low frequency components of images.
In spiked features model, a controlling parameter of the performance gap between NN and \{RF, NT\} is

\[
\text{snr}_f = \text{Feature SNR} = \frac{\text{Signal features variance}}{\text{Junk features variance}}.
\]

- Small \text{snr}_f, there is a large separation.
- Large \text{snr}_f, \{RF, NT\} performs closer to NN.

Somewhat implicitly, NN first finds the signal features (PCA), and then perform kernel methods on these features.

\[
\text{snr}_f \neq \text{SNR} = \frac{\|f^*_\|_2^2}{\mathbb{E}[\epsilon^2]}
\]
Thank you!