Learning structured data with random features and kernel methods

Theodor Misiakiewicz

Stanford University

April 28th, 2021

ML lunch

Joint work with Behrooz Ghorbani (Google Research), Song Mei (UC Berkeley) and Andrea Montanari (Stanford)

- 'Mysteries' of Deep Learning:
  - **Optimization:** training is a *highly non-convex* problem, but we are still able to find near global optimizers.
  - **Generalization:** *overparametrized* models. The solution often interpolates the training data while generalizing well on test data.
- In this talk, I will focus on the connection between neural networks (NNs) and kernel machines.

- 'Mysteries' of Deep Learning:
  - **Optimization:** training is a *highly non-convex* problem, but we are still able to find near global optimizers.
  - **Generalization:** *overparametrized* models. The solution often interpolates the training data while generalizing well on test data.

- In this talk, I will focus on the connection between neural networks (NNs) and kernel machines.

Neural Tangent (NT) model

- Neural network: $\mathrm{NN}_p(\boldsymbol{x}; \boldsymbol{\theta})$, $\boldsymbol{x} \in \mathbb{R}^d$, $\boldsymbol{\theta} \in \mathbb{R}^p$.
  e.g., fully-connected neural network:

$$\mathrm{NN}_p(\boldsymbol{x}; \boldsymbol{\theta}) = \langle \boldsymbol{a}, \sigma(\dots \boldsymbol{W}_2 \sigma(\boldsymbol{W}_1 \boldsymbol{x})) \rangle.$$

- Linearization around random initialization $\boldsymbol{\theta}_0$:

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

- Neural Tangent (NT) model: [Chizat et al.,'18], [Du et al.,'18]

$$\mathrm{NT}_p(x; \beta, \theta_0) = \langle \beta, \nabla_\theta \mathrm{NN}_p(x; \theta_0) \rangle.$$

- Finite-width approximation of limiting kernel ($p \to \infty$ + iid initialization):

$$K_{\mathrm{NT}}(x, y) = \mathbb{E}_{\theta_0}[\langle \nabla_\theta \mathrm{NN}_p(x; \theta_0), \nabla_\theta \mathrm{NN}_p(y; \theta_0) \rangle].$$

Neural Tangent (NT) model

- Neural network: $\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta})$, $\boldsymbol{x} \in \mathbb{R}^d$, $\boldsymbol{\theta} \in \mathbb{R}^p$.
  e.g., fully-connected neural network:

$$\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}) = \langle \boldsymbol{a}, \sigma(\dots \boldsymbol{W}_2 \sigma(\boldsymbol{W}_1 \boldsymbol{x})) \rangle.$$

- Linearization around random initialization $\boldsymbol{\theta}_0$:

$$\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}) = \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) + \underline{\underline{\langle \boldsymbol{\theta} - \boldsymbol{\theta}_0, \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) \rangle}} + o(\|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2).$$

- Neural Tangent (NT) model: [Chizat et al.,'18], [Du et al.,'18]

$$\mathrm{NT}_p(\boldsymbol{x};\boldsymbol{\beta},\boldsymbol{\theta}_0) = \langle \boldsymbol{\beta}, \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) \rangle.$$

- Finite-width approximation of limiting kernel ($p \to \infty$ + iid initialization):

$$K_{\mathrm{NT}}(\boldsymbol{x},\boldsymbol{y}) = \mathbb{E}_{\boldsymbol{\theta}_0}[\langle \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0), \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{y};\boldsymbol{\theta}_0) \rangle].$$

Neural Tangent (NT) model

- Neural network: $\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta})$, $\boldsymbol{x} \in \mathbb{R}^d$, $\boldsymbol{\theta} \in \mathbb{R}^p$.
  e.g., fully-connected neural network:

$$\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}) = \langle \boldsymbol{a}, \sigma(\dots \boldsymbol{W}_2 \sigma(\boldsymbol{W}_1 \boldsymbol{x})) \rangle.$$

- Linearization around random initialization $\boldsymbol{\theta}_0$:

$$\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}) = \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) + \underline{\underline{\langle \boldsymbol{\theta} - \boldsymbol{\theta}_0, \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) \rangle}} + o(\|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2).$$

- Neural Tangent (NT) model: [Chizat et al.,'18], [Du et al.,'18]

$$\mathrm{NT}_p(\boldsymbol{x};\boldsymbol{\beta},\boldsymbol{\theta}_0) = \langle \boldsymbol{\beta}, \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) \rangle.$$

- Finite-width approximation of limiting kernel ($p \to \infty$ + iid initialization):

$$K_{\mathrm{NT}}(x,y) = \mathbb{E}_{\theta_0}[\langle \nabla_\theta \mathrm{NN}_p(x;\theta_0), \nabla_\theta \mathrm{NN}_p(y;\theta_0) \rangle].$$

Neural Tangent (NT) model

- Neural network: $\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta})$, $\boldsymbol{x} \in \mathbb{R}^d$, $\boldsymbol{\theta} \in \mathbb{R}^p$.
  e.g., fully-connected neural network:

$$\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}) = \langle \boldsymbol{a}, \sigma(\dots \boldsymbol{W}_2 \sigma(\boldsymbol{W}_1 \boldsymbol{x})) \rangle.$$

- Linearization around random initialization $\boldsymbol{\theta}_0$:

$$\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}) = \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) + \underline{\underline{\langle \boldsymbol{\theta} - \boldsymbol{\theta}_0, \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) \rangle}} + o(\|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2).$$

- Neural Tangent (NT) model: [Chizat et al.,'18], [Du et al.,'18]

$$\mathrm{NT}_p(\boldsymbol{x};\boldsymbol{\beta},\boldsymbol{\theta}_0) = \langle \boldsymbol{\beta}, \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0) \rangle.$$

- Finite-width approximation of limiting kernel ($p \to \infty$ + iid initialization):

$$K_{\mathrm{NT}}(\boldsymbol{x},\boldsymbol{y}) = \mathbb{E}_{\boldsymbol{\theta}_0}[\langle \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}_0), \nabla_{\boldsymbol{\theta}} \mathrm{NN}_p(\boldsymbol{y};\boldsymbol{\theta}_0) \rangle].$$

GD trained NNs $\approx$ NT model in some regime

Coupled gradient descent on empirical squared loss:

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\theta}^t = -\nabla_{\boldsymbol{\theta}}\hat{\mathbb{E}}[(y - \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}^t))^2], \qquad \boldsymbol{\theta}^0 = \boldsymbol{\theta}_0,$$
$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\beta}^t = -\nabla_{\boldsymbol{\beta}}\hat{\mathbb{E}}[(y - \mathrm{NT}_p(\boldsymbol{x};\boldsymbol{\beta}^t,\boldsymbol{\theta}_0))^2], \qquad \boldsymbol{\beta}^0 = 0.$$

Theorem (informal)

For any $\varepsilon > 0$, for large enough NNs and proper random initialization θ_0 (Xavier initialization), we have with high probability

$$\sup_{t\geq 0}\sup_{\|x\|_2=1} |\mathrm{NN}_p(x;\theta^t) - \mathrm{NT}_p(x;\beta^t,\theta_0)| \leq \varepsilon.$$

$\Longrightarrow$ Intuition: *weights do not change much* and $\|\theta^t - \theta_0\|_2$ remains small.

[Jacot, Gabriel, Hongler,'18] , [Du, Zhai, Poczos, Singh,'18], [Du, Lee, Li, Wang, Zhai,'18], [Allen-Zhu, Li, Song,'18] , [Zou, Cao, Zhou, Gu,'18], [Oymak, Soltanolkotabi,'18], [Chizat, Oyallon, Bach,'19], ...

GD trained NNs $\approx$ NT model in some regime

Coupled gradient descent on empirical squared loss:

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\theta}^t = -\nabla_{\boldsymbol{\theta}}\hat{\mathbb{E}}[(y - \mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}^t))^2], \qquad \boldsymbol{\theta}^0 = \boldsymbol{\theta}_0,$$
$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\beta}^t = -\nabla_{\boldsymbol{\beta}}\hat{\mathbb{E}}[(y - \mathrm{NT}_p(\boldsymbol{x};\boldsymbol{\beta}^t,\boldsymbol{\theta}_0))^2], \qquad \boldsymbol{\beta}^0 = 0.$$

Theorem (informal)

For any $\varepsilon > 0$, for large enough NNs and proper random initialization $\boldsymbol{\theta}_0$ (Xavier initialization), we have with high probability

$$\sup_{t\geq 0}\sup_{\|\boldsymbol{x}\|_2=1} |\mathrm{NN}_p(\boldsymbol{x};\boldsymbol{\theta}^t) - \mathrm{NT}_p(\boldsymbol{x};\boldsymbol{\beta}^t,\boldsymbol{\theta}_0)| \leq \varepsilon.$$

$\Longrightarrow$ Intuition: *weights do not change much* and $\|\boldsymbol{\theta}^t - \boldsymbol{\theta}_0\|_2$ remains small.

[Jacot, Gabriel, Hongler,'18] , [Du, Zhai, Poczos, Singh,'18], [Du, Lee, Li, Wang, Zhai,'18], [Allen-Zhu, Li, Song,'18] , [Zou, Cao, Zhou, Gu,'18], [Oymak, Soltanolkotabi,'18], [Chizat, Oyallon, Bach,'19], ...

The 'Kernel Regime'

- Optimization success:

With sufficient over-parametrization and proper initialization, gradient descent on training loss of NN converges linearly to global minimum.

- . . . but:
  - **Lazy Training:** weights hardly change, there is 'no feature learning'. [Chizat et al.,'18]
  - Empirically, NNs perform often better than their linearized counterparts.

However, offer theoretical insights [Bartlett, Montanari, Rakhlin,'21]:

- Tractability via overparametrization.
- Explain some of the trade-off between overparametrization and generalization (benign overfitting, double descent).

The 'Kernel Regime'

- Optimization success:

With sufficient over-parametrization and proper initialization, gradient descent on training loss of NN converges linearly to global minimum.

- ... but:
  - **Lazy Training:** weights hardly change, there is 'no feature learning'. [Chizat et al.,'18]
  - Empirically, NNs perform often better than their linearized counterparts.

However, offer theoretical insights [Bartlett, Montanari, Rakhlin,'21]:

- Tractability via overparametrization.
- Explain some of the trade-off between overparametrization and generalization (benign overfitting, double descent).

The 'Kernel Regime'

- Optimization success:

With sufficient over-parametrization and proper initialization, gradient descent on training loss of NN converges linearly to global minimum.

- ... but:
  - **Lazy Training:** weights hardly change, there is 'no feature learning'. [Chizat et al.,'18]
  - Empirically, NNs perform often better than their linearized counterparts.

However, offer theoretical insights [Bartlett, Montanari, Rakhlin,'21]:

- Tractability via overparametrization.
- Explain some of the trade-off between overparametrization and generalization (benign overfitting, double descent).

New approach for kernel engineering

- Progress on CIFAR-10:

Paper	Method	Accuracy
[Coates, Lee, Ng,'11]	feature learning + linear regression	77%
[Arora et al.,'19]	CNTK (data independent)	77%
[Li et al.,'19]	CRFK + data dependent preprocessing	89%
[Shankar et al.,'20]	CRFK (data independent)	90%
-	CNN	$> 99\%$

- NTK achieves near state-of-the-art results on UCI dataset [Arora et al.,'19].

Observations:

- NNs often outperform kernel methods.
- Kernel methods sometimes achieve comparable accuracy to NNs.
- Kernels induced by NNs perform much better than previous handcrafted kernels.

New approach for kernel engineering

- Progress on CIFAR-10:

Paper	Method	Accuracy
[Coates, Lee, Ng,'11]	feature learning + linear regression	77%
[Arora et al.,'19]	CNTK (data independent)	77%
[Li et al.,'19]	CRFK + data dependent preprocessing	89%
[Shankar et al.,'20]	CRFK (data independent)	90%
-	CNN	$> 99\%$

- NTK achieves near state-of-the-art results on UCI dataset [Arora et al.,'19].

Observations:

- NNs often outperform kernel methods.
- Kernel methods sometimes achieve comparable accuracy to NNs.
- Kernels induced by NNs perform much better than previous handcrafted kernels.

New approach for kernel engineering

- Progress on CIFAR-10:

Paper	Method	Accuracy
[Coates, Lee, Ng,'11]	feature learning + linear regression	77%
[Arora et al.,'19]	CNTK (data independent)	77%
[Li et al.,'19]	CRFK + data dependent preprocessing	89%
[Shankar et al.,'20]	CRFK (data independent)	90%
-	CNN	$> 99\%$

- NTK achieves near state-of-the-art results on UCI dataset [Arora et al.,'19].

Observations:

- NNs often outperform kernel methods.
- Kernel methods sometimes achieve comparable accuracy to NNs.
- Kernels induced by NNs perform much better than previous handcrafted kernels.

Outline of the talk

Questions

- Why are kernel methods not as good as NNs in general?
- When can we expect kernel methods to perform well, comparable to NNs?
- Can we quantify the benefits of using convolutional kernels against inner-product kernels (e.g., Gaussian kernel)?

Three 'stylized' scenarios:

A. **Isotropic covariates model:**
   - large gap between NN and kernel methods.

B. **Spiked covariates model:**
   - smaller gap between NN and kernel methods.

C. **Invariant function estimation:**
   - quantify the benefits of convolutional kernels over inner-product kernels.

Outline of the talk

Questions

- Why are kernel methods not as good as NNs in general?
- When can we expect kernel methods to perform well, comparable to NNs?
- Can we quantify the benefits of using convolutional kernels against inner-product kernels (e.g., Gaussian kernel)?

Three 'stylized' scenarios:

A. **Isotropic covariates model:**
   - large gap between NN and kernel methods.

B. **Spiked covariates model:**
   - smaller gap between NN and kernel methods.

C. **Invariant function estimation:**
   - quantify the benefits of convolutional kernels over inner-product kernels.

A. Isotropic covariates model

Setting

- Given n iid samples $\{(y_i, \boldsymbol{x}_i)\}_{i\in[n]}$:

$$y_i = f_\star(\boldsymbol{x}_i) + \varepsilon_i, \qquad \boldsymbol{x}_i \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1}(\sqrt{d})), \qquad \varepsilon_i \sim_{iid} N(0, \tau^2).$$

- Two-layers neural networks: $\mathrm{NN}_N(\boldsymbol{x}; \theta) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)$.

- Random Features model [Rahimi, Recht,'08]:

$$\mathrm{RF}_N(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{W}) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) = \langle \boldsymbol{a}, \nabla_{\boldsymbol{a}} \mathrm{NN}_p(\boldsymbol{x}; \theta_0)\rangle.$$

- Ridge regression with RF model (with fixed $\boldsymbol{W} = (\boldsymbol{w}_i)_{i\in[N]} \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1})$)

$$\hat{\boldsymbol{a}}(\lambda) = \arg\min_{\boldsymbol{a}} \left\{ \frac{1}{n} \sum_{i=1}^n \left(y_i - \sum_{j=1}^N a_j \sigma(\langle \boldsymbol{w}_j, \boldsymbol{x}_i\rangle) \right)^2 + \lambda \|\boldsymbol{a}\|_2^2 \right\}.$$

Denote the solution $\widehat{\mathrm{RF}}_{n,N,\lambda} = \mathrm{RF}_N(\cdot; \hat{\boldsymbol{a}}(\lambda), \boldsymbol{W})$.

Setting

- Given n iid samples $\{(y_i, \boldsymbol{x}_i)\}_{i\in[n]}$:

$$y_i = f_\star(\boldsymbol{x}_i) + \varepsilon_i, \qquad \boldsymbol{x}_i \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1}(\sqrt{d})), \qquad \varepsilon_i \sim_{iid} N(0, \tau^2).$$

- Two-layers neural networks: $\mathrm{NN}_N(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)$.

- Random Features model [Rahimi, Recht,'08]:

$$\mathrm{RF}_N(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{W}) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) = \langle \boldsymbol{a}, \nabla_{\boldsymbol{a}} \mathrm{NN}_p(\boldsymbol{x}; \boldsymbol{\theta}_0)\rangle.$$

- Ridge regression with RF model (with fixed $\boldsymbol{W} = (\boldsymbol{w}_i)_{i\in[N]} \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1})$)

$$\hat{\boldsymbol{a}}(\lambda) = \arg\min_{\boldsymbol{a}} \left\{ \frac{1}{n} \sum_{i=1}^n \left(y_i - \sum_{j=1}^N a_j \sigma(\langle \boldsymbol{w}_j, \boldsymbol{x}_i\rangle) \right)^2 + \lambda \|\boldsymbol{a}\|_2^2 \right\}.$$

Denote the solution $\widehat{\mathrm{RF}}_{n,N,\lambda} = \mathrm{RF}_N(\cdot; \hat{\boldsymbol{a}}(\lambda), \boldsymbol{W})$.

Setting

- Given n iid samples $\{(y_i, \boldsymbol{x}_i)\}_{i\in[n]}$:

$$y_i = f_\star(\boldsymbol{x}_i) + \varepsilon_i, \qquad \boldsymbol{x}_i \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1}(\sqrt{d})), \qquad \varepsilon_i \sim_{iid} N(0, \tau^2).$$

- Two-layers neural networks: $\mathrm{NN}_N(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)$.

- Random Features model [Rahimi, Recht,'08]:

$$\mathrm{RF}_N(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{W}) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) = \langle \boldsymbol{a}, \nabla_{\boldsymbol{a}} \mathrm{NN}_p(\boldsymbol{x}; \boldsymbol{\theta}_0)\rangle.$$

- Ridge regression with RF model (with fixed $\boldsymbol{W} = (\boldsymbol{w}_i)_{i\in[N]} \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1})$)

$$\hat{\boldsymbol{a}}(\lambda) = \arg\min_{\boldsymbol{a}} \left\{ \frac{1}{n} \sum_{i=1}^n \left(y_i - \sum_{j=1}^N a_j \sigma(\langle \boldsymbol{w}_j, \boldsymbol{x}_i\rangle) \right)^2 + \lambda \|\boldsymbol{a}\|_2^2 \right\}.$$

Denote the solution $\widehat{\mathrm{RF}}_{n,N,\lambda} = \mathrm{RF}_N(\cdot; \hat{\boldsymbol{a}}(\lambda), \boldsymbol{W})$.

Setting

- Given n iid samples $\{(y_i, \boldsymbol{x}_i)\}_{i\in[n]}$:

$$y_i = f_\star(\boldsymbol{x}_i) + \varepsilon_i, \qquad \boldsymbol{x}_i \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1}(\sqrt{d})), \qquad \varepsilon_i \sim_{iid} N(0, \tau^2).$$

- Two-layers neural networks: $\mathrm{NN}_N(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)$.

- Random Features model [Rahimi, Recht,'08]:

$$\mathrm{RF}_N(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{W}) = \sum_{i=1}^N a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle) = \langle \boldsymbol{a}, \nabla_{\boldsymbol{a}} \mathrm{NN}_p(\boldsymbol{x}; \boldsymbol{\theta}_0)\rangle.$$

- Ridge regression with RF model (with fixed $\boldsymbol{W} = (\boldsymbol{w}_i)_{i\in[N]} \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1})$)

$$\hat{\boldsymbol{a}}(\lambda) = \arg\min_{\boldsymbol{a}} \left\{ \frac{1}{n} \sum_{i=1}^n \left(y_i - \sum_{j=1}^N a_j \sigma(\langle \boldsymbol{w}_j, \boldsymbol{x}_i\rangle) \right)^2 + \lambda \|\boldsymbol{a}\|_2^2 \right\}.$$

Denote the solution $\widehat{\mathrm{RF}}_{n,N,\lambda} = \mathrm{RF}_N(\cdot; \hat{\boldsymbol{a}}(\lambda), \boldsymbol{W})$.

Test error of RF model

Theorem (Mei, Misiakiewicz, Montanari, 2021)

Assume $d^{\ell+\delta} \le \min(n, N) \le d^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

$\mathrm{P}_{>\ell}$: projection orthogonal to the space of degree-ℓ polynomials.

- For d^ℓ parameters and samples, RF fits the best degree-ℓ polynomial approximation.
- n and N play a symmetric role:

$$\text{Test error}(n, N) = \max\Big\{\text{approx. error } (n = \infty, N), \text{statistical error } (n, N = \infty)\Big\}.$$

- Statistical error $(n, N = \infty)$: kernel method with an inner-product kernel $K(x_1, x_2) = h(\langle x_1, x_2\rangle/d)$ (e.g., NTK of fully-connected NNs of any depth).

Test error of RF model

Theorem (Mei, Misiakiewicz, Montanari, 2021)

Assume $d^{\ell+\delta} \le \min(n, N) \le d^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

$\mathrm{P}_{>\ell}$: projection orthogonal to the space of degree-ℓ polynomials.

- For d^ℓ parameters and samples, RF fits the best degree-ℓ polynomial approximation.
- n and N play a symmetric role:

$$\text{Test error}(n, N) = \max\Big\{\text{approx. error } (n = \infty, N), \text{statistical error } (n, N = \infty)\Big\}.$$

- Statistical error $(n, N = \infty)$: kernel method with an inner-product kernel $K(x_1, x_2) = h(\langle x_1, x_2\rangle/d)$ (e.g., NTK of fully-connected NNs of any depth).

Test error of RF model

Theorem (Mei, Misiakiewicz, Montanari, 2021)

Assume $d^{\ell+\delta} \le \min(n, N) \le d^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

$\mathrm{P}_{>\ell}$: projection orthogonal to the space of degree-ℓ polynomials.

- For d^ℓ parameters and samples, RF fits the best degree-ℓ polynomial approximation.
- n and N play a symmetric role:

$$\text{Test error}(n, N) = \max\Big\{\text{approx. error } (n = \infty, N), \text{ statistical error } (n, N = \infty)\Big\}.$$

- Statistical error $(n, N = \infty)$: kernel method with an inner-product kernel $K(x_1, x_2) = h(\langle x_1, x_2\rangle/d)$ (e.g., NTK of fully-connected NNs of any depth).

Test error of RF model

Theorem (Mei, Misiakiewicz, Montanari, 2021)

Assume $d^{\ell+\delta} \le \min(n, N) \le d^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

$\mathrm{P}_{>\ell}$: projection orthogonal to the space of degree-ℓ polynomials.

- For d^ℓ parameters and samples, RF fits the best degree-ℓ polynomial approximation.
- n and N play a symmetric role:

$$\text{Test error}(n, N) = \max\Big\{\text{approx. error } (n = \infty, N), \text{ statistical error } (n, N = \infty)\Big\}.$$

- Statistical error $(n, N = \infty)$: kernel method with an inner-product kernel $K(\boldsymbol{x}_1, \boldsymbol{x}_2) = h(\langle \boldsymbol{x}_1, \boldsymbol{x}_2\rangle/d)$ (e.g., NTK of fully-connected NNs of any depth).

The truncated staircase decay

$$f = \mathsf{P}_0 f + \mathsf{P}_1 f + \mathsf{P}_2 f + \mathsf{P}_3 f + \mathsf{P}_4 f.$$

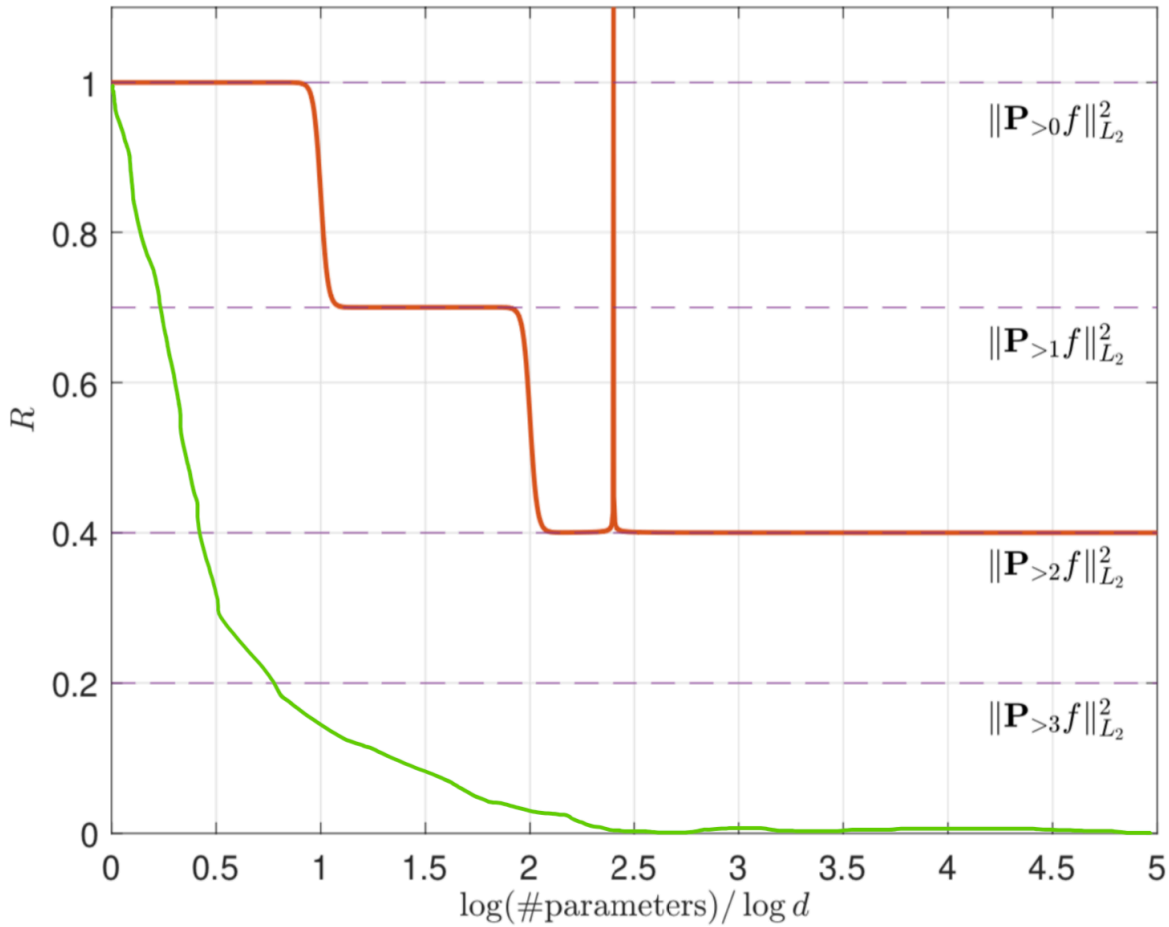


Figure: Test error (in red) v.s. $\log(N)/\log(d)$ for $n = d^{2.4}$ and $\lambda = 0^+$. Train error in green.

Performance gap between NN and kernel methods

- We expect NNs to perform way better on 'low complexity' functions

  E.g., $f_\star(\boldsymbol{x}) = \sigma(\langle \boldsymbol{w}_\star, \boldsymbol{x}\rangle)$: if $N \propto 1$ and $n \propto d$, then GD solution:

$$\|f_\star - \widehat{\mathrm{NN}}_{N,n}\|_{L^2}^2 = o_{d,\mathbb{P}}(\cdot) \qquad \text{[Bai et al.,'16]}$$

  while for KRR:

$$\|f_\star - \widehat{\mathrm{KR}}_{N,n}\|_{L^2}^2 = \|\mathrm{P}_{>1} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- More generally we expect NNs to vastly outperform kernel methods for target functions that only depend on a low dimensional subspace of the data [Bach,'17].

Intuition: fit $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)$

- For RF, $\boldsymbol{w}_i$'s are random, $\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)^2 = \langle \boldsymbol{w}_i, \boldsymbol{e}_1\rangle^2 / \|\boldsymbol{w}_i\|_2^2$ is small in HD.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

NN can 'adaptively learn' $\boldsymbol{w}_i$'s while RF cannot.

Performance gap between NN and kernel methods

- We expect NNs to perform way better on 'low complexity' functions

  E.g., $f_\star(\boldsymbol{x}) = \sigma(\langle \boldsymbol{w}_\star, \boldsymbol{x}\rangle)$: if $N \propto 1$ and $n \propto d$, then GD solution:

$$\|f_\star - \widehat{\mathrm{NN}}_{N,n}\|_{L^2}^2 = o_{d,\mathbb{P}}(\cdot) \qquad \text{[Bai et al.,'16]}$$

  while for KRR:

$$\|f_\star - \widehat{\mathrm{KR}}_{N,n}\|_{L^2}^2 = \|\mathrm{P}_{>1} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- More generally we expect NNs to vastly outperform kernel methods for target functions that only depend on a low dimensional subspace of the data [Bach,'17].

Intuition: fit $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)$

- For RF, $\boldsymbol{w}_i$'s are random, $\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)^2 = \langle \boldsymbol{w}_i, \boldsymbol{e}_1\rangle^2 / \|\boldsymbol{w}_i\|_2^2$ is small in HD.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

NN can 'adaptively learn' $\boldsymbol{w}_i$'s while RF cannot.

Performance gap between NN and kernel methods

- We expect NNs to perform way better on 'low complexity' functions

  E.g., $f_\star(\boldsymbol{x}) = \sigma(\langle \boldsymbol{w}_\star, \boldsymbol{x}\rangle)$: if $N \propto 1$ and $n \propto d$, then GD solution:

$$\|f_\star - \widehat{\mathrm{NN}}_{N,n}\|_{L^2}^2 = o_{d,\mathbb{P}}(\cdot) \qquad \text{[Bai et al.,'16]}$$

  while for KRR:

$$\|f_\star - \widehat{\mathrm{KR}}_{N,n}\|_{L^2}^2 = \|\mathrm{P}_{>1} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- More generally we expect NNs to vastly outperform kernel methods for target functions that only depend on a low dimensional subspace of the data [Bach,'17].

Intuition: fit $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)$

- For RF, $\boldsymbol{w}_i$'s are random, $\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)^2 = \langle \boldsymbol{w}_i, \boldsymbol{e}_1\rangle^2 / \|\boldsymbol{w}_i\|_2^2$ is small in HD.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

NN can 'adaptively learn' $\boldsymbol{w}_i$'s while RF cannot.

In the isotropic covariates model:

- Kernel methods suffer from the **curse of dimensionality**.

- For target functions that depend on a **low dimensional projection** of the data, NNs can **adaptively learn a good representation** of the data and vastly outperform kernel methods.

B. Spiked covariates model

Spiked covariates model

Covariate vector: orthogonal matrix $[\boldsymbol{U}, \boldsymbol{U}^\perp]$

$$\boldsymbol{x} = \boldsymbol{U}\boldsymbol{z}_1 + \boldsymbol{U}^\perp \boldsymbol{z}_2, \qquad \boldsymbol{z}_1 \in \mathbb{R}^{d_s}, \boldsymbol{z}_2 \in \mathbb{R}^{d-d_s}.$$

Signal part: $\boldsymbol{z}_1 \sim \text{Unif}\left(\mathbb{S}^{d_s-1}\left(\sqrt{\text{snr}_c \cdot d_s}\right)\right)$.

Noise part: $\boldsymbol{z}_2 \sim \text{Unif}\left(\mathbb{S}^{d-d_s-1}\left(\sqrt{d-d_s}\right)\right)$

d_s = signal dimension.

snr_c = covariate SNR.

Target function: $f_\star(\boldsymbol{x}) = \varphi(\boldsymbol{z}_1)$.

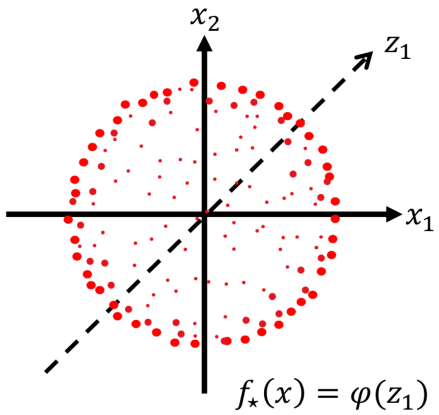


Figure: Isotropic covariates: $\text{snr}_c = 1$.

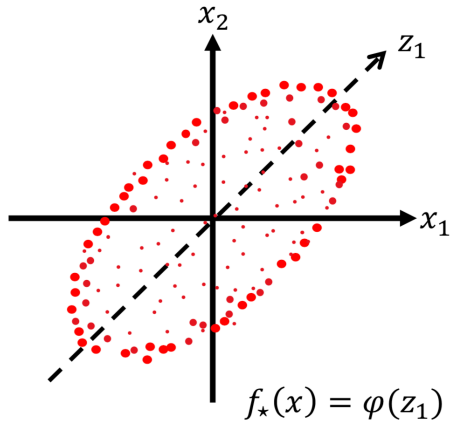


Figure: Anisotropic covariates: $\text{snr}_c > 1$.

Test error of KRR in the spiked covariates model

- Given iid samples $(\{\boldsymbol{x}_i, y_i\})_{i\in[n]}$ from the spiked covariates model.
- Effective dimension: $d_{\text{eff}} = d_s \vee (d/\text{snr}_c)$.

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

Assume ${d_{\text{eff}}}^{\ell+\delta} \le \min(n, N) \le {d_{\text{eff}}}^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d_{\text{eff}}$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\text{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\text{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- d_{eff} capture the 'effective low-dimensionality' of the data.
- d_{eff} decreases with snr_c (data more anisotropic) and kernel methods will perform better.
- On the contrary, we expect NNs to learn features $\boldsymbol{w}_i$ aligned with $\boldsymbol{z}_1$ and to depend mildly on snr_c.

Test error of KRR in the spiked covariates model

- Given iid samples $(\{\boldsymbol{x}_i, y_i\})_{i\in[n]}$ from the spiked covariates model.
- Effective dimension: $d_{\mathsf{eff}} = d_s \vee (d/\mathsf{snr}_c)$.

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

Assume ${d_{\mathsf{eff}}}^{\ell+\delta} \le \min(n, N) \le {d_{\mathsf{eff}}}^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d_{\mathsf{eff}}$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- d_{eff} capture the 'effective low-dimensionality' of the data.
- d_{eff} decreases with snr_c (data more anisotropic) and kernel methods will perform better.
- On the contrary, we expect NNs to learn features $\boldsymbol{w}_i$ aligned with $\boldsymbol{z}_1$ and to depend mildly on snr_c.

Test error of KRR in the spiked covariates model

- Given iid samples $(\{\boldsymbol{x}_i, y_i\})_{i\in[n]}$ from the spiked covariates model.
- Effective dimension: $d_{\mathsf{eff}} = d_s \vee (d/\mathsf{snr}_c)$.

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

Assume ${d_{\mathsf{eff}}}^{\ell+\delta} \le \min(n, N) \le {d_{\mathsf{eff}}}^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d_{\mathsf{eff}}$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathsf{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathsf{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- d_{eff} capture the 'effective low-dimensionality' of the data.
- d_{eff} decreases with snr_c (data more anisotropic) and kernel methods will perform better.
- On the contrary, we expect NNs to learn features $\boldsymbol{w}_i$ aligned with $\boldsymbol{z}_1$ and to depend mildly on snr_c.

Test error of KRR in the spiked covariates model

- Given iid samples $(\{\boldsymbol{x}_i, y_i\})_{i\in[n]}$ from the spiked covariates model.
- Effective dimension: $d_{\mathsf{eff}} = d_s \vee (d/\mathsf{snr}_c)$.

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

Assume ${d_{\mathsf{eff}}}^{\ell+\delta} \le \min(n, N) \le {d_{\mathsf{eff}}}^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d_{\mathsf{eff}}$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathsf{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathsf{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- d_{eff} capture the 'effective low-dimensionality' of the data.
- d_{eff} decreases with snr_c (data more anisotropic) and kernel methods will perform better.
- On the contrary, we expect NNs to learn features $\boldsymbol{w}_i$ aligned with $\boldsymbol{z}_1$ and to depend mildly on snr_c.

Test error of KRR in the spiked covariates model

- Given iid samples $(\{\boldsymbol{x}_i, y_i\})_{i\in[n]}$ from the spiked covariates model.
- Effective dimension: $d_{\text{eff}} = d_s \vee (d/\text{snr}_c)$.

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

Assume ${d_{\text{eff}}}^{\ell+\delta} \le \min(n, N) \le {d_{\text{eff}}}^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d_{\text{eff}}$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\text{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\text{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- d_{eff} capture the 'effective low-dimensionality' of the data.
- d_{eff} decreases with snr_c (data more anisotropic) and kernel methods will perform better.
- On the contrary, we expect NNs to learn features $\boldsymbol{w}_i$ aligned with $\boldsymbol{z}_1$ and to depend mildly on snr_c.

Test error of KRR in the spiked covariates model

- Given iid samples $(\{\boldsymbol{x}_i, y_i\})_{i\in[n]}$ from the spiked covariates model.
- Effective dimension: $d_{\mathsf{eff}} = d_s \vee (d/\mathsf{snr}_c)$.

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

Assume ${d_{\mathsf{eff}}}^{\ell+\delta} \le \min(n, N) \le {d_{\mathsf{eff}}}^{\ell+1-\delta}$ and $|\log(n/N)| \ge \delta \log d_{\mathsf{eff}}$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathsf{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathsf{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- d_{eff} capture the 'effective low-dimensionality' of the data.
- d_{eff} decreases with snr_c (data more anisotropic) and kernel methods will perform better.
- On the contrary, we expect NNs to learn features $\boldsymbol{w}_i$ aligned with $\boldsymbol{z}_1$ and to depend mildly on snr_c.

NNs vs kernel methods in the spiked covariates model

Effective dimension: $d_{\text{eff}} = d_s \vee (d/\text{snr}_c)$.

In the spiked covariates model, we expect the generalization error of:

- Kernel methods to depend on d_{eff};
- NNs to depend on d_s.

- $d_s \le d_{\text{eff}} \le d$:

  Test error: NN $\le$ Kernel methods

- d_{eff} decreases with snr_c:
  - Isotropic features: $d_{\text{eff}} = d$ (low covariate SNR)

    Test error: NN $\ll$ Kernel methods.

  - Highly anisotropic features: $d_{\text{eff}} = d_s$ (high covariate SNR)

    Test error: NN $\sim$ Kernel methods.

NNs vs kernel methods in the spiked covariates model

Effective dimension: $d_{\text{eff}} = d_s \vee (d/\text{snr}_c)$.

In the spiked covariates model, we expect the generalization error of:

- Kernel methods to depend on d_{eff};
- NNs to depend on d_s.

- $d_s \le d_{\text{eff}} \le d$:

  Test error: NN $\le$ Kernel methods

- d_{eff} decreases with snr_c:
  - Isotropic features: $d_{\text{eff}} = d$ (low covariate SNR)

    Test error: NN $\ll$ Kernel methods.

  - Highly anisotropic features: $d_{\text{eff}} = d_s$ (high covariate SNR)

    Test error: NN $\sim$ Kernel methods.

NNs vs kernel methods in the spiked covariates model

Effective dimension: $d_{\text{eff}} = d_s \vee (d/\text{snr}_c)$.

In the spiked covariates model, we expect the generalization error of:

- Kernel methods to depend on d_{eff};
- NNs to depend on d_s.

- $d_s \leq d_{\text{eff}} \leq d$:

  Test error: NN $\leq$ Kernel methods

- d_{eff} decreases with snr_c:
  - Isotropic features: $d_{\text{eff}} = d$ (low covariate SNR)

    Test error: NN $\ll$ Kernel methods.

  - Highly anisotropic features: $d_{\text{eff}} = d_s$ (high covariate SNR)

    Test error: NN $\sim$ Kernel methods.

Intuition for the gap: anisotropic case

Goal: fitting $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$.

Consider $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\mathsf{T}] = \mathrm{diag}(\mathsf{snr}_c, \mathrm{Id}_{d-1})$.

- To fit $\varphi(x_1)$, we need to use features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$ that are correlated to x_1, i.e.,

$$\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle) = \mathsf{snr}_c \cdot \frac{\langle \boldsymbol{w}_i, \boldsymbol{e}_1 \rangle^2}{\|\Sigma^{1/2} \boldsymbol{w}_i\|_2^2} = \mathsf{snr}_c \cdot O_{d,\mathbb{P}}(d^{-1}) = O_{d,\mathbb{P}}(d_{\mathsf{eff}}{}^{-1}).$$

- For RF, when snr_c is large, all random $\boldsymbol{w}_i$'s are good.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

RF automatically have access to good $\boldsymbol{w}_i$'s.

Intuition for the gap: anisotropic case

Goal: fitting $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$.

Consider $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\mathsf{T}] = \mathrm{diag}(\mathrm{snr}_c, \mathrm{Id}_{d-1})$.

- To fit $\varphi(x_1)$, we need to use features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$ that are correlated to x_1, i.e.,

$$\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle) = \mathrm{snr}_c \cdot \frac{\langle \boldsymbol{w}_i, \boldsymbol{e}_1 \rangle^2}{\|\Sigma^{1/2} \boldsymbol{w}_i\|_2^2} = \mathrm{snr}_c \cdot O_{d,\mathbb{P}}(d^{-1}) = O_{d,\mathbb{P}}(d_{\mathrm{eff}}{}^{-1}).$$

- For RF, when snr_c is large, all random $\boldsymbol{w}_i$'s are good.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

RF automatically have access to good $\boldsymbol{w}_i$'s.

Intuition for the gap: anisotropic case

Goal: fitting $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$.

Consider $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\mathsf{T}] = \mathrm{diag}(\mathrm{snr}_c, \mathrm{Id}_{d-1})$.

- To fit $\varphi(x_1)$, we need to use features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$ that are correlated to x_1, i.e.,

$$\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle) = \mathrm{snr}_c \cdot \frac{\langle \boldsymbol{w}_i, \boldsymbol{e}_1 \rangle^2}{\|\Sigma^{1/2} \boldsymbol{w}_i\|_2^2} = \mathrm{snr}_c \cdot O_{d,\mathbb{P}}(d^{-1}) = O_{d,\mathbb{P}}(d_{\mathrm{eff}}{}^{-1}).$$

- For RF, when snr_c is large, all random $\boldsymbol{w}_i$'s are good.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

RF automatically have access to good $\boldsymbol{w}_i$'s.

Intuition for the gap: anisotropic case

Goal: fitting $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$.

Consider $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\mathsf{T}] = \mathrm{diag}(\mathrm{snr}_c, \mathrm{Id}_{d-1})$.

- To fit $\varphi(x_1)$, we need to use features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$ that are correlated to x_1, i.e.,

$$\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle) = \mathrm{snr}_c \cdot \frac{\langle \boldsymbol{w}_i, \boldsymbol{e}_1 \rangle^2}{\|\Sigma^{1/2} \boldsymbol{w}_i\|_2^2} = \mathrm{snr}_c \cdot O_{d,\mathbb{P}}(d^{-1}) = O_{d,\mathbb{P}}(d_{\mathrm{eff}}{}^{-1}).$$

- For RF, when snr_c is large, all random $\boldsymbol{w}_i$'s are good.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

RF automatically have access to good $\boldsymbol{w}_i$'s.

Intuition for the gap: anisotropic case

Goal: fitting $\varphi(x_1)$ using features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$.

Consider $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\mathsf{T}] = \mathrm{diag}(\mathrm{snr}_c, \mathrm{Id}_{d-1})$.

- To fit $\varphi(x_1)$, we need to use features $\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle)$ that are correlated to x_1, i.e.,

$$\mathrm{corr}(x_1 \boldsymbol{e}_1, \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle) = \mathrm{snr}_c \cdot \frac{\langle \boldsymbol{w}_i, \boldsymbol{e}_1 \rangle^2}{\|\Sigma^{1/2} \boldsymbol{w}_i\|_2^2} = \mathrm{snr}_c \cdot O_{d,\mathbb{P}}(d^{-1}) = O_{d,\mathbb{P}}(d_{\mathrm{eff}}{}^{-1}).$$

- For RF, when snr_c is large, all random $\boldsymbol{w}_i$'s are good.
- For NN, $\boldsymbol{w}_i$'s can be chosen to have a large correlation with $\boldsymbol{e}_1$.

RF automatically have access to good $\boldsymbol{w}_i$'s.

Insight

Insight

Lower covariate SNR (data more isotropic) should lead to larger generalization gap between NNs and kernel methods.

How to test this insight?

In *image classification*, we expect

- Images to have most of their spectrum concentrated on low-frequencies (z_1 part);
- The labels to depend predominantly on the low-frequencies ($y = f_\star(x) = \varphi(z_1)$).

Experiment: add noise to the high frequency components (z_2 part) of the images (i.e., decreasing the snr_c).

Adding noise in covariates should increase the generalization gap between NN and kernel methods.

Insight

Lower covariate SNR (data more isotropic) should lead to larger generalization gap between NNs and kernel methods.

How to test this insight?

In *image classification*, we expect

- Images to have most of their spectrum concentrated on low-frequencies ($\boldsymbol{z}_1$ part);
- The labels to depend predominantly on the low-frequencies ($y = f_\star(\boldsymbol{x}) = \varphi(\boldsymbol{z}_1)$).

Experiment: add noise to the high frequency components (z_2 part) of the images (i.e., decreasing the snr_c).

Adding noise in covariates should increase the generalization gap between NN and kernel methods.

Insight

Lower covariate SNR (data more isotropic) should lead to larger generalization gap between NNs and kernel methods.

How to test this insight?

In *image classification*, we expect

- Images to have most of their spectrum concentrated on low-frequencies ($\boldsymbol{z}_1$ part);
- The labels to depend predominantly on the low-frequencies ($y = f_\star(\boldsymbol{x}) = \varphi(\boldsymbol{z}_1)$).

Experiment: add noise to the high frequency components ($\boldsymbol{z}_2$ part) of the images (i.e., decreasing the snr_c).

Adding noise in covariates should increase the generalization gap between NN and kernel methods.

Numerical simulations

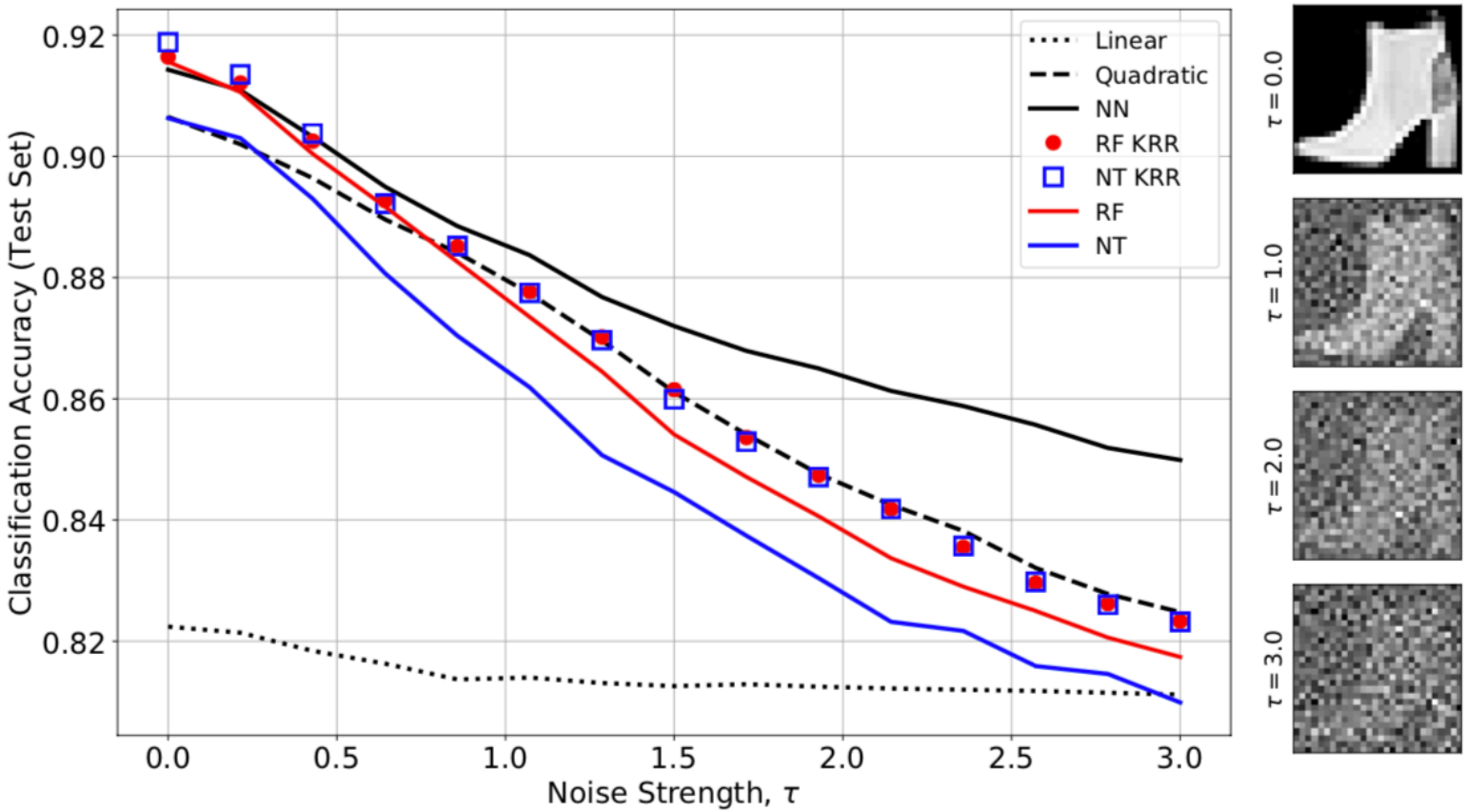


Figure: Test accuracy on FMNIST: adding noise to the high frequency components.

Spiked covariates model:
A controlling parameter of the performance gap between NN and kernel methods is

$$\mathrm{snr}_c = \text{Covariate SNR} = \frac{\text{Signal covariates variance}}{\text{Noise covariates variance}}.$$

- Small snr_c: large separation.
- Large snr_c: kernel methods perform closer to NN.

Intuition:

- When snr_c is small, RF fail to find the signal covariates.
- When snr_c is big, RF automatically find the signal covariates.
- NNs always look for the signal covariates.

C. Invariant function estimation on the sphere

Symmetries in the data

- In many learning tasks, the target function present some **natural symmetries**.

  E.g., image classification: labels **invariant under translation** of the images.

- Effectiveness of NNs architectures is often loosely attributed to their ability to encode various symmetries present in the data.

  E.g., **CNNs with translation invariance.**

- CNNs perform better than fully-connected networks.

  Convolutional kernels perform better than inner-product kernels.

- **Can we quantify this gain?**

Symmetries in the data

- In many learning tasks, the target function present some **natural symmetries**.

  E.g., image classification: labels **invariant under translation** of the images.

- Effectiveness of NNs architectures is often loosely attributed to their ability to encode various symmetries present in the data.

  E.g., **CNNs with translation invariance.**

- CNNs perform better than fully-connected networks.

  Convolutional kernels perform better than inner-product kernels.

- **Can we quantify this gain?**

Symmetries in the data

- In many learning tasks, the target function present some **natural symmetries**.

  E.g., image classification: labels **invariant under translation** of the images.

- Effectiveness of NNs architectures is often loosely attributed to their ability to encode various symmetries present in the data.

  E.g., **CNNs with translation invariance.**

- CNNs perform better than fully-connected networks.

  Convolutional kernels perform better than inner-product kernels.

- Can we quantify this gain?

Symmetries in the data

- In many learning tasks, the target function present some **natural symmetries**.

  E.g., image classification: labels **invariant under translation** of the images.

- Effectiveness of NNs architectures is often loosely attributed to their ability to encode various symmetries present in the data.

  E.g., **CNNs with translation invariance.**

- CNNs perform better than fully-connected networks.

  Convolutional kernels perform better than inner-product kernels.

- **Can we quantify this gain?**

Cyclic function estimation on the sphere

- Data $\boldsymbol{x} \sim \text{Unif}(\mathbb{S}^{d-1}(\sqrt{d}))$.
- Consider $\mathcal{G}_d = \{g_0, g_1, \ldots, g_{d-1}\}$ the cyclic group:

$$g_i \cdot \boldsymbol{x} = (x_{d-i+1}, x_{d-i+2}, \ldots, x_d, x_1, x_2, \ldots, x_{d-i}).$$

- **Goal:** learn a cyclic invaeriant function $f_\star$:

$$\text{i.e., } f_\star(g \cdot \boldsymbol{x}) = f_\star(\boldsymbol{x}) \text{ for all } g \in \mathcal{G}_d.$$

E.g., $f(\boldsymbol{x}) = \sum_{i=1}^{d} x_i x_{i+1}$.

Stylized model for an image label $y = f_\star(\boldsymbol{x})$ invariant by translation of image $\boldsymbol{x}$.

Cyclic function estimation on the sphere

- Data $\boldsymbol{x} \sim \text{Unif}(\mathbb{S}^{d-1}(\sqrt{d}))$.
- Consider $\mathcal{G}_d = \{g_0, g_1, \ldots, g_{d-1}\}$ the cyclic group:

$$g_i \cdot \boldsymbol{x} = (x_{d-i+1}, x_{d-i+2}, \ldots, x_d, x_1, x_2, \ldots, x_{d-i}).$$

- **Goal:** learn a cyclic invaeriant function $f_\star$:

$$\text{i.e., } f_\star(g \cdot \boldsymbol{x}) = f_\star(\boldsymbol{x}) \text{ for all } g \in \mathcal{G}_d.$$

E.g., $f(\boldsymbol{x}) = \sum_{i=1}^{d} x_i x_{i+1}$.

Stylized model for an image label $y = f_\star(\boldsymbol{x})$ invariant by translation of image $\boldsymbol{x}$.

Cyclic function estimation on the sphere

- Data $\boldsymbol{x} \sim \text{Unif}(\mathbb{S}^{d-1}(\sqrt{d}))$.
- Consider $\mathcal{G}_d = \{g_0, g_1, \ldots, g_{d-1}\}$ the cyclic group:

$$g_i \cdot \boldsymbol{x} = (x_{d-i+1}, x_{d-i+2}, \ldots, x_d, x_1, x_2, \ldots, x_{d-i}).$$

- **Goal:** learn a cyclic invaeriant function $f_\star$:

$$\text{i.e., } f_\star(g \cdot \boldsymbol{x}) = f_\star(\boldsymbol{x}) \text{ for all } g \in \mathcal{G}_d.$$

E.g., $f(\boldsymbol{x}) = \sum_{i=1}^{d} x_i x_{i+1}$.

Stylized model for an image label $y = f_\star(\boldsymbol{x})$ invariant by translation of image $\boldsymbol{x}$.

Cyclic function estimation on the sphere

- Data $\boldsymbol{x} \sim \text{Unif}(\mathbb{S}^{d-1}(\sqrt{d}))$.
- Consider $\mathcal{G}_d = \{g_0, g_1, \ldots, g_{d-1}\}$ the cyclic group:

$$g_i \cdot \boldsymbol{x} = (x_{d-i+1}, x_{d-i+2}, \ldots, x_d, x_1, x_2, \ldots, x_{d-i}).$$

- **Goal:** learn a cyclic invaeriant function $f_\star$:

$$\text{i.e., } f_\star(g \cdot \boldsymbol{x}) = f_\star(\boldsymbol{x}) \text{ for all } g \in \mathcal{G}_d.$$

E.g., $f(\boldsymbol{x}) = \sum_{i=1}^{d} x_i x_{i+1}$.

Stylized model for an image label $y = f_\star(\boldsymbol{x})$ invariant by translation of image $\boldsymbol{x}$.

Invariant random features and kernels

- Invariant RF model:

$$\mathrm{RF}_N^{\mathrm{inv}}(x; a, W) = \sum_{i=1}^{N} a_i \int_{g \in \mathcal{G}_d} \sigma(\langle w_i, g \cdot x \rangle) \pi(\mathrm{d}g) = \frac{1}{d} \sum_{i=1}^{N} a_i \sum_{k=1}^{d} \sigma(\langle w_i, g_k \cdot x \rangle).$$

Neural tangent model of a CNN with N filters $w_i \in \mathbb{R}^d$.
Associated invariant kernel:

$$K_{\mathrm{inv}}(x, y) = \int_{g \in \mathcal{G}_d} h(\langle g \cdot x, y \rangle / d) \pi(\mathrm{d}g).$$

- Compared to standard RF model:

$$\mathrm{RF}_N(x; a, W) = \sum_{i=1}^{N} a_i \sigma(\langle x, w_i \rangle),$$

and associated 'standard' (inner-product) kernel:

$$K(x, y) = h(\langle x, y \rangle / d).$$

Invariant random features and kernels

- Invariant RF model:

$$\mathrm{RF}_N^{\mathrm{inv}}(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{W}) = \sum_{i=1}^{N} a_i \int_{g \in \mathcal{G}_d} \sigma(\langle \boldsymbol{w}_i, g \cdot \boldsymbol{x} \rangle) \pi(\mathrm{d}g) = \frac{1}{d} \sum_{i=1}^{N} a_i \sum_{k=1}^{d} \sigma(\langle \boldsymbol{w}_i, g_k \cdot \boldsymbol{x} \rangle).$$

  Neural tangent model of a CNN with N filters $\boldsymbol{w}_i \in \mathbb{R}^d$.
  Associated invariant kernel:

$$K_{\mathrm{inv}}(\boldsymbol{x}, \boldsymbol{y}) = \int_{g \in \mathcal{G}_d} h(\langle g \cdot \boldsymbol{x}, \boldsymbol{y} \rangle / d) \pi(\mathrm{d}g).$$

- Compared to standard RF model:

$$\mathrm{RF}_N(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{W}) = \sum_{i=1}^{N} a_i \sigma(\langle \boldsymbol{x}, \boldsymbol{w}_i \rangle),$$

  and associated 'standard' (inner-product) kernel:

$$K(\boldsymbol{x}, \boldsymbol{y}) = h(\langle \boldsymbol{x}, \boldsymbol{y} \rangle / d).$$

Test error of KRR with invariant kernel

- Cyclic invariant $f_\star$: given iid samples $\{(y_i, \boldsymbol{x}_i)\}_{i\in[n]}$,

$$y_i = f_\star(\boldsymbol{x}_i) + \varepsilon_i, \qquad \boldsymbol{x}_i \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1}(\sqrt{d})), \quad \varepsilon_i \sim_{iid} N(0, \tau^2).$$

Theorem (Mei, Misiakiewicz, Montanari, 2021)

Assume $d^{\ell-1+\delta} \le \min(n, N) \le d^{\ell-\delta}$ and $|\log(n/N)| \ge \delta \log d$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}^{\mathrm{inv}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- Standard RF model:

$$\text{For } d^{\ell+\delta} \le \min(n, N) \le d^{\ell+1-\delta}, \qquad \|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_d(\cdot).$$

  Gain a factor d in features and sample complexity by using invariant RF.

- More generally, for $\mathcal{G}_d$ with 'degeneracy α', we gain a factor d^α.

Test error of KRR with invariant kernel

- Cyclic invariant $f_\star$: given iid samples $\{(y_i, \boldsymbol{x}_i)\}_{i\in[n]}$,

$$y_i = f_\star(\boldsymbol{x}_i) + \varepsilon_i, \qquad \boldsymbol{x}_i \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1}(\sqrt{d})), \quad \varepsilon_i \sim_{iid} N(0, \tau^2).$$

Theorem (Mei, Misiakiewicz, Montanari, 2021)

Assume $d^{\ell-1+\delta} \le \min(n, N) \le d^{\ell-\delta}$ and $|\log(n/N)| \ge \delta \log d$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}^{\mathrm{inv}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- Standard RF model:

  For $d^{\ell+\delta} \le \min(n, N) \le d^{\ell+1-\delta}$, $\quad \|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_d(\cdot)$.

  Gain a factor d in features and sample complexity by using invariant RF.

- More generally, for $\mathcal{G}_d$ with 'degeneracy α', we gain a factor d^α.

Test error of KRR with invariant kernel

- Cyclic invariant $f_\star$: given iid samples $\{(y_i, \boldsymbol{x}_i)\}_{i\in[n]}$,

$$y_i = f_\star(\boldsymbol{x}_i) + \varepsilon_i, \qquad \boldsymbol{x}_i \sim_{iid} \mathrm{Unif}(\mathbb{S}^{d-1}(\sqrt{d})), \quad \varepsilon_i \sim_{iid} N(0, \tau^2).$$

Theorem (Mei, Misiakiewicz, Montanari, 2021)

Assume $d^{\ell-1+\delta} \le \min(n, N) \le d^{\ell-\delta}$ and $|\log(n/N)| \ge \delta \log d$, σ satisfies some generic conditions and λ is small enough. Then,

$$\|f_\star - \widehat{\mathrm{RF}}^{\mathrm{inv}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_{d,\mathbb{P}}(\cdot).$$

- Standard RF model:

  For $d^{\ell+\delta} \le \min(n, N) \le d^{\ell+1-\delta}$, $\quad \|f_\star - \widehat{\mathrm{RF}}_{n,N,\lambda}\|_{L^2}^2 = \|\mathrm{P}_{>\ell} f_\star\|_{L^2}^2 + o_d(\cdot)$.

  Gain a factor d in features and sample complexity by using invariant RF.

- More generally, for $\mathcal{G}_d$ with 'degeneracy α', we gain a factor d^α.

Numerical simulations

$$f_{\text{lin}} = \frac{1}{\sqrt{d}} \sum_{i=1}^{d} x_i, \qquad f_{\text{quad}} = \frac{1}{\sqrt{d}} \sum_{i=1}^{d} x_i x_{i+1}, \qquad f_{\text{cube}} = \frac{1}{\sqrt{d}} \sum_{i=1}^{d} x_i x_{i+1} x_{i+2}.$$

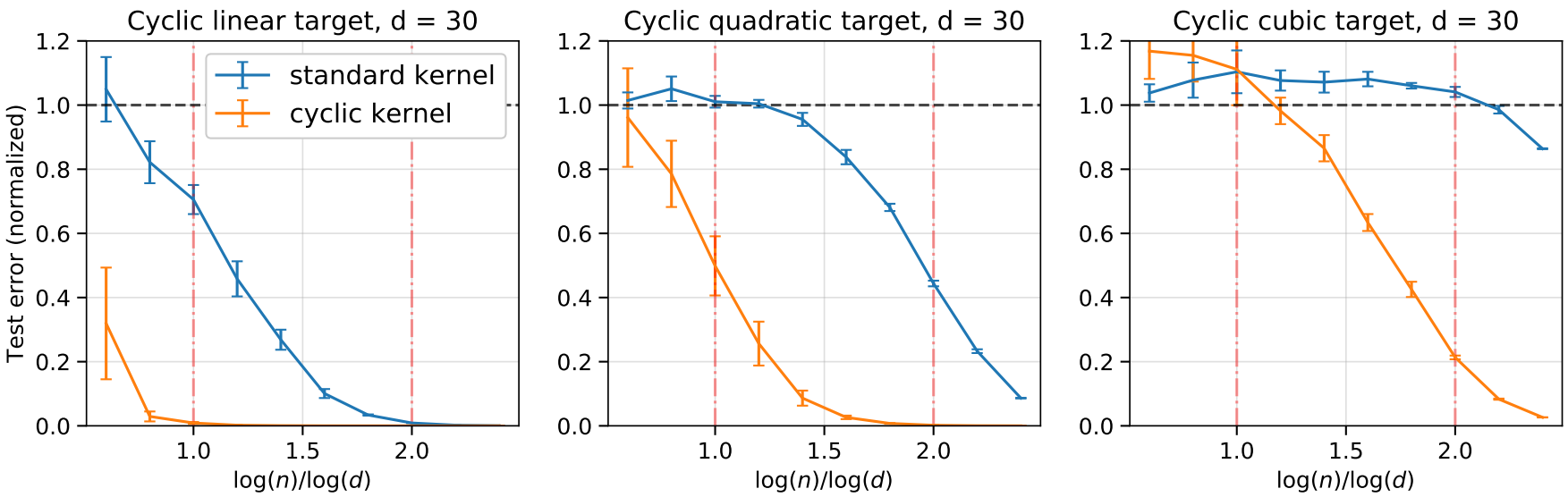
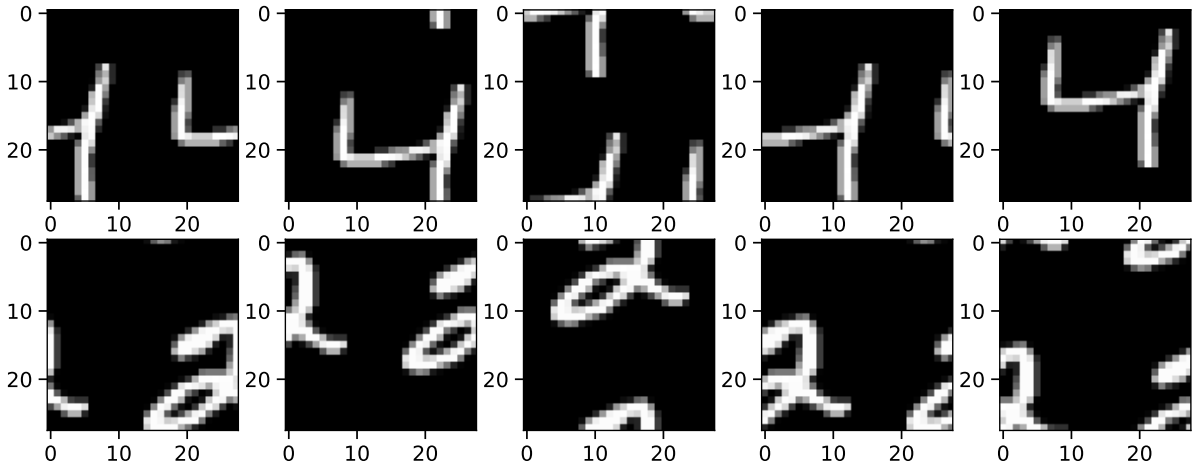


Figure: Test error of KRR with invariant kernel and inner-product kernel.

Cyclic invariant MNIST



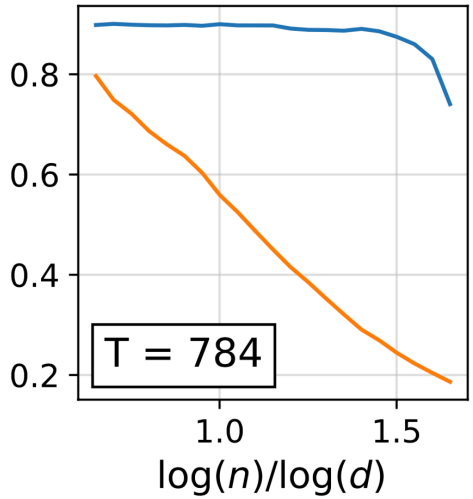


Figure: Test accuracy against number of samples (orange: cyclic kernel, blue: standard kernel).

Summary

What I illustrated in this talk:

- **Isotropic covariates model:**
  Kernel methods suffer from the *curse of dimensionality* and perform inferior to NNs due to **lack of feature learning.**

- **Spiked covariates model:**
  For large covariate SNR, **feature learning is unecessary** and kernel methods can also perform well.

- **Invariant function estimation:**
  Invariant kernels outperform standard kernels and we quantify the gain in statistical efficiency.

All the results in this talk can be derived from a general framework for getting high-dimensional asymptotics of test error of random features and kernel methods.

[Mei, **Misiakiewicz**, Montanari, '21a].

Summary

What I illustrated in this talk:

- **Isotropic covariates model:**
  Kernel methods suffer from the *curse of dimensionality* and perform inferior to NNs due to **lack of feature learning.**

- **Spiked covariates model:**
  For large covariate SNR, **feature learning is unecessary** and kernel methods can also perform well.

- **Invariant function estimation:**
  Invariant kernels outperform standard kernels and we quantify the gain in statistical efficiency.

All the results in this talk can be derived from a general framework for getting high-dimensional asymptotics of test error of random features and kernel methods.
[Mei, **Misiakiewicz**, Montanari, '21a].

Summary

What I illustrated in this talk:

- **Isotropic covariates model:**
  Kernel methods suffer from the *curse of dimensionality* and perform inferior to NNs due to **lack of feature learning.**

- **Spiked covariates model:**
  For large covariate SNR, **feature learning is unecessary** and kernel methods can also perform well.

- **Invariant function estimation:**
  Invariant kernels outperform standard kernels and we quantify the gain in statistical efficiency.

All the results in this talk can be derived from a general framework for getting high-dimensional asymptotics of test error of random features and kernel methods.
[Mei, **Misiakiewicz**, Montanari, '21a].

Summary

What I illustrated in this talk:

- **Isotropic covariates model:**
  Kernel methods suffer from the *curse of dimensionality* and perform inferior to NNs due to **lack of feature learning.**

- **Spiked covariates model:**
  For large covariate SNR, **feature learning is unecessary** and kernel methods can also perform well.

- **Invariant function estimation:**
  Invariant kernels outperform standard kernels and we quantify the gain in statistical efficiency.

All the results in this talk can be derived from a general framework for getting high-dimensional asymptotics of test error of random features and kernel methods.
[Mei, **Misiakiewicz**, Montanari, '21a].

Summary

What I illustrated in this talk:

- **Isotropic covariates model:**
  Kernel methods suffer from the *curse of dimensionality* and perform inferior to NNs due to **lack of feature learning.**

- **Spiked covariates model:**
  For large covariate SNR, **feature learning is unecessary** and kernel methods can also perform well.

- **Invariant function estimation:**
  Invariant kernels outperform standard kernels and we quantify the gain in statistical efficiency.

All the results in this talk can be derived from a general framework for getting high-dimensional asymptotics of test error of random features and kernel methods.

[Mei, **Misiakiewicz**, Montanari, '21a].

Open problems

- Show that NNs trained by gradient descent overcome the curse of dimensionality in the spiked covariates model (using mean-field dynamics, see [Chizat, Bach, '20]).
- What are other latent low-dimensional structure that we can study? (other than the spiked covariates model)
- Using our general framework to study more complicated kernels.
- Data dependent kernel methods?
- Practical advantages of using kernel methods vs NNs? Transfer learning, robustness...
- ...

Thank you!