On the Sample Complexity of Learning under Invariance and Geometric Stability

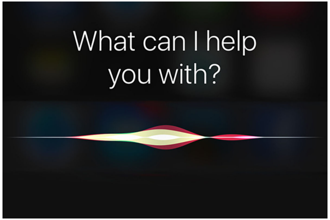
Alberto Bietti

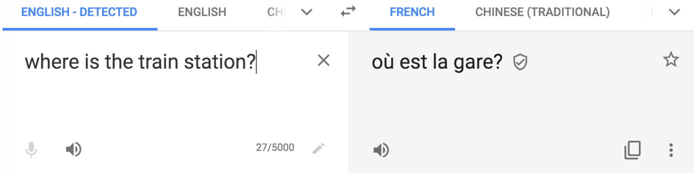
NYU

UC Berkeley. Sept. 15, 2021.

Success of deep learning

State-of-the-art models in various domains (images, speech, text, ...)





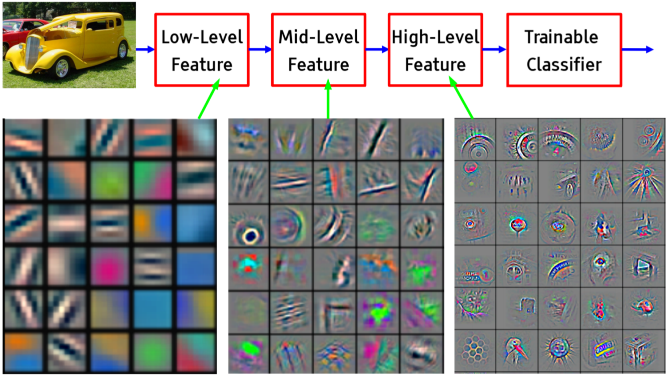
Success of deep learning

State-of-the-art models in various domains (images, speech, text, ...)

$$f(x) = W_n \sigma(W_{n-1} \cdots \sigma(W_1 x) \cdots)$$

Recipe: **huge models** $+$ **lots of data** $+$ **compute** $+$ **simple algorithms**

Exploiting data structure through architectures

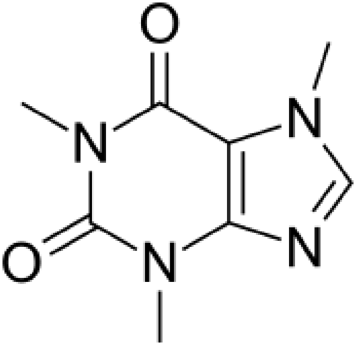


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Modern architectures (CNNs, GNNs, Transformers, ...)

- Provide some invariance through pooling
- Model (local) interactions at different scales, hierarchically
- Useful **inductive biases** for learning efficiently on structured data

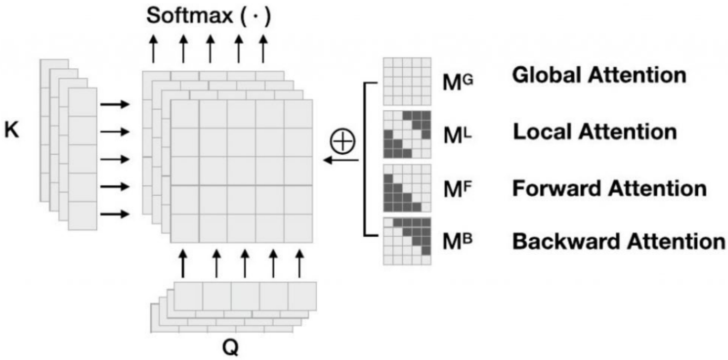
Exploiting data structure through architectures



Modern architectures (CNNs, GNNs, Transformers, ...)

- Provide some invariance through pooling
- Model (local) interactions at different scales, hierarchically
- Useful **inductive biases** for learning efficiently on structured data

Exploiting data structure through architectures



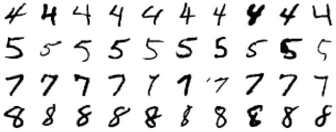
Modern architectures (CNNs, GNNs, Transformers, ...)

- Provide some invariance through pooling
- Model (local) interactions at different scales, hierarchically
- Useful **inductive biases** for learning efficiently on structured data

Geometric stability to deformations

Deformations

- $\phi : \Omega \to \Omega$: C^1-diffeomorphism (*e.g.*, $\Omega = \mathbb{R}^2$)
- $\phi \cdot x(u) = x(\phi^{-1}(u))$: action operator
- Much richer group of transformations than translations



- Studied for wavelet-based scattering transform (Mallat, 2012; Bruna and Mallat, 2013)

Geometric stability to deformations

Deformations

- $\phi : \Omega \to \Omega$: C^1-diffeomorphism (*e.g.*, $\Omega = \mathbb{R}^2$)
- $\phi \cdot x(u) = x(\phi^{-1}(u))$: action operator
- Much richer group of transformations than translations

Geometric stability

- A function $f(\cdot)$ is **stable** (Mallat, 2012) if:

$$f(\phi \cdot x) \approx f(x) \quad \text{when} \quad \|\nabla \phi - I\|_\infty \leq \epsilon$$

- In particular, near-invariance to translations ($\nabla \phi = I$)

Understanding deep learning

The challenge of deep learning theory

- **Over-parameterized** (millions of parameters)
- **Expressive** (can approximate any function)
- Complex **architectures** for exploiting problem structure
- Yet, **easy to optimize** with (stochastic) gradient descent!

Understanding deep learning

The challenge of deep learning theory

- **Over-parameterized** (millions of parameters)
- **Expressive** (can approximate any function)
- Complex **architectures** for exploiting problem structure
- Yet, **easy to optimize** with (stochastic) gradient descent!

A functional space viewpoint

- View deep networks as functions in some functional space
- Non-parametric models, natural measures of complexity (*e.g.*, norms)

Understanding deep learning

The challenge of deep learning theory

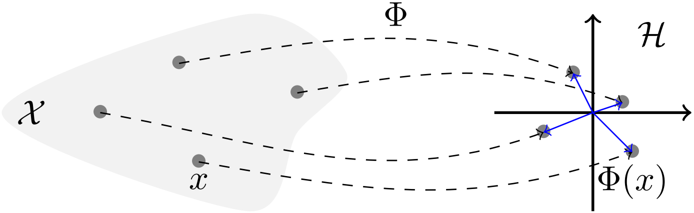
- **Over-parameterized** (millions of parameters)
- **Expressive** (can approximate any function)
- Complex **architectures** for exploiting problem structure
- Yet, **easy to optimize** with (stochastic) gradient descent!

A functional space viewpoint

- View deep networks as functions in some functional space
- Non-parametric models, natural measures of complexity (*e.g.*, norms)

What is an appropriate functional space?

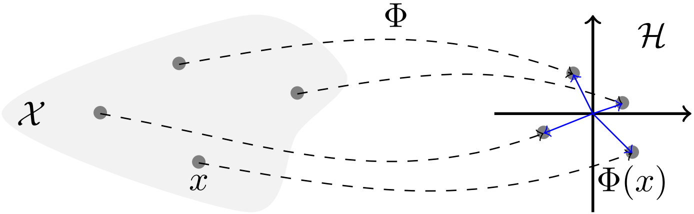
Kernels to the rescue



Kernels?

- Map data x to high-dimensional space, $\Phi(x) \in \mathcal{H}$ ($\mathcal{H}$: "RKHS")
- Functions $f \in \mathcal{H}$ are linear in features: $f(x) = \langle f, \Phi(x) \rangle$ (f can be non-linear in x!)
- Learning with a positive definite kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$
  - $\mathcal{H}$ can be infinite-dimensional! (*kernel trick*)
  - Need to compute kernel matrix $K = [K(x_i, x_j)]_{ij} \in \mathbb{R}^{N \times N}$

Kernels to the rescue



Clean and well-developed theory

- Tractable methods (convex optimization)
- Statistical and approximation properties well understood for many kernels
- Costly (kernel matrix of size N^2) but approximations are possible

Studying architecture benefits through kernels

Hierarchical kernels (Cho and Saul, 2009)

- Kernels can be constructed **hierarchically**

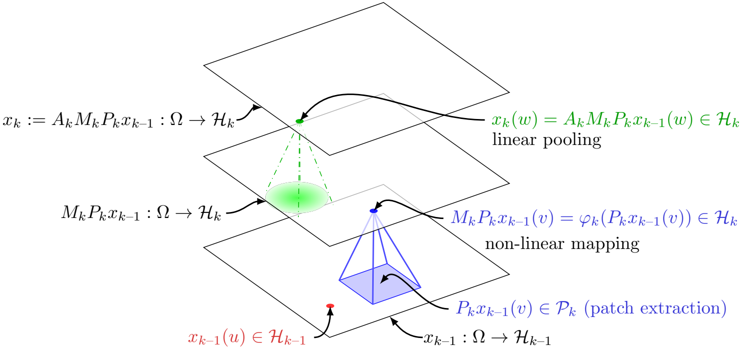
$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle \text{ with } \Phi(x) = \varphi_2(\varphi_1(x))$$

- *e.g.*, dot-product kernels on the sphere

$$K(x, x') = \kappa_2(\langle \varphi_1(x), \varphi_1(x') \rangle) = \kappa_2(\kappa_1(x^\top x'))$$

Studying architecture benefits through kernels

Convolutional kernels for images (Mairal et al., 2014; Mairal, 2016; Shankar et al., 2020)



- Good empirical performance with tractable approximations (Nyström)

Studying architecture benefits through kernels

Links with infinite-width networks

- Over-parameterized networks can lead to similar structured kernels
- "Kernel regimes":
  - Random feature kernels (RF, Neal, 1996; Rahimi and Recht, 2007)
  - Neural tangent kernels (NTK, Jacot et al., 2018; Chizat et al., 2019)
- Well-defined for many architectures

Studying architecture benefits through kernels

Links with infinite-width networks

- Over-parameterized networks can lead to similar structured kernels
- "Kernel regimes":
  - Random feature kernels (RF, Neal, 1996; Rahimi and Recht, 2007)
  - Neural tangent kernels (NTK, Jacot et al., 2018; Chizat et al., 2019)
- Well-defined for many architectures

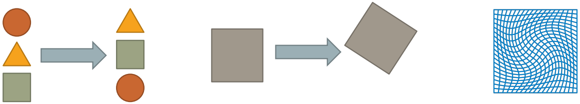
Goal: study sample complexity benefits of architectures through kernels

Outline

1. Sample complexity under invariance and stability (B., Venturi, and Bruna, 2021)

2. Locality and depth (B., 2021)

Geometric priors

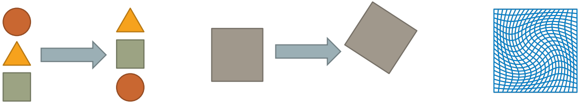


Functions $f : \mathcal{X} \to \mathbb{R}$ that are "smooth" along known transformations of input x

- *e.g.*, translations, rotations, permutations, deformations
- We consider: **permutations** $\sigma \in G$

$$(\sigma \cdot x)[u] = x[\sigma^{-1}(u)]$$

Geometric priors



Functions $f : \mathcal{X} \to \mathbb{R}$ that are "smooth" along known transformations of input x

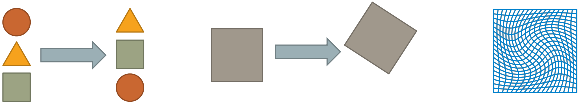
- *e.g.*, translations, rotations, permutations, deformations
- We consider: **permutations** $\sigma \in G$

$$(\sigma \cdot x)[u] = x[\sigma^{-1}(u)]$$

Group invariance: If G is a group (*e.g.*, cyclic shifts, all permutations), we want

$$f(\sigma \cdot x) = f(x), \quad \sigma \in G$$

Geometric priors



Functions $f : \mathcal{X} \to \mathbb{R}$ that are "smooth" along known transformations of input x

- *e.g.*, translations, rotations, permutations, deformations
- We consider: **permutations** $\sigma \in G$

$$(\sigma \cdot x)[u] = x[\sigma^{-1}(u)]$$

Group invariance: If G is a group (*e.g.*, cyclic shifts, all permutations), we want

$$f(\sigma \cdot x) = f(x), \quad \sigma \in G$$

Geometric stability: For other sets G (*e.g.*, local shifts, deformations), we want

$$f(\sigma \cdot x) \approx f(x), \quad \sigma \in G$$

Geometric priors: symmetrization operator

$$S_G f(x) := \frac{1}{|G|} \sum_{\sigma \in G} f(\sigma \cdot x)$$

Assumptions on a target function f^*

- G-invariant: $S_G f^* = f^*$
- G-stable: $f^* = S_G g^*$, for some g^*
  - More generally, $f^* = S_G^r g^*$ for some r
  - Similarity to *source conditions* in kernel methods or inverse problems

Geometric priors: symmetrization operator

$$S_G f(x) := \frac{1}{|G|} \sum_{\sigma \in G} f(\sigma \cdot x)$$

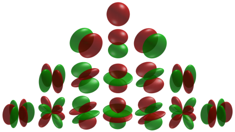
Assumptions on a target function f^*

- G-invariant: $S_G f^* = f^*$
- G-stable: $f^* = S_G g^*$, for some g^*
  - More generally, $f^* = S_G^r g^*$ for some r
  - Similarity to *source conditions* in kernel methods or inverse problems

How do these interact with generic smoothness properties of f^*?

Spherical harmonics, dot-product kernels

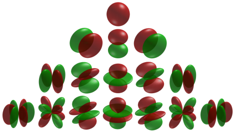
Harmonic analysis on the sphere



- $x \sim \tau$ uniform distribution on the sphere $\mathbb{S}^{d-1}$
- $L^2(\tau)$ basis of **spherical harmonics** $Y_{k,j}$
- $N(d, k)$ harmonics of degree k, form a basis of $V_{d,k}$

Spherical harmonics, dot-product kernels

Harmonic analysis on the sphere



- $x \sim \tau$ uniform distribution on the sphere $\mathbb{S}^{d-1}$
- $L^2(\tau)$ basis of **spherical harmonics** $Y_{k,j}$
- $N(d,k)$ harmonics of degree k, form a basis of $V_{d,k}$

Dot-product kernels and their RKHS $\quad K(x,x') = \kappa(\langle x, x' \rangle)$

$$\mathcal{H} = \left\{ f = \sum_{k=0}^{\infty} \sum_{j=1}^{N(d,k)} a_{k,j} Y_{k,j}(\cdot) \ \text{ s.t. } \ \|f\|_{\mathcal{H}}^2 := \sum_{k,j} \frac{a_{k,j}^2}{\mu_k} < \infty \right\}$$

- $\mu_k = \frac{\omega_{d-2}}{\omega_{d-1}} \int_{-1}^{1} \kappa(t) P_{d,k}(t)(1-t^2)^{\frac{d-3}{2}} dt$: eigenvalues of **integral operator** T_K, each with multiplicity $N(d,k)$ ($P_{d,k}$: **Legendre**/**Gegenbauer** polynomial)
- **decay** $\leftrightarrow$ **regularity**: $\mu_k \asymp k^{-2\beta} \leftrightarrow \|f\|_{\mathcal{H}} = \|T_K^{-1/2} f\|_{L^2(\tau)} \approx \|\Delta_{\mathbb{S}^{d-1}}^{\beta/2} f\|_{L^2(\tau)}$

Invariant harmonics

Key properties of S_G for invariant case (Mei, Misiakiewicz, and Montanari, 2021)

- S_G acts as projection from $V_{d,k}$ (dim $N(d,k)$) to $\overline{V}_{d,k}$ (dim $\overline{N}(d,k)$)
- The number of invariant spherical harmonics $\overline{N}$ can be estimated using:

$$\gamma_d(k) := \frac{\overline{N}(d,k)}{N(d,k)} = \frac{1}{|G|} \sum_{\sigma \in G} \mathbb{E}_x[P_{d,k}(\langle \sigma \cdot x, x \rangle)].$$

Invariant harmonics

Key properties of S_G for invariant case (Mei, Misiakiewicz, and Montanari, 2021)

- S_G acts as projection from $V_{d,k}$ (dim $N(d,k)$) to $\overline{V}_{d,k}$ (dim $\overline{N}(d,k)$)
- The number of invariant spherical harmonics $\overline{N}$ can be estimated using:

$$\gamma_d(k) := \frac{\overline{N}(d,k)}{N(d,k)} = \frac{1}{|G|} \sum_{\sigma \in G} \mathbb{E}_x[P_{d,k}(\langle \sigma \cdot x, x \rangle)].$$

Invariant kernels (Haasdonk and Burkhardt, 2007; Mroueh et al., 2015)

$$K_G(x, x') = \frac{1}{|G|} \sum_{\sigma \in G} \kappa(\langle \sigma \cdot x, x' \rangle)$$

- Corresponds to (full-width) convolution + global pooling
- Note that $T_{K_G} = S_G T_K$

Counting invariant harmonics

$$\gamma_d(k) := \frac{\overline{N}(d,k)}{N(d,k)} = \frac{1}{|G|} \sum_{\sigma \in G} \mathbb{E}_x[P_{d,k}(\langle \sigma \cdot x, x \rangle)].$$

Proposition ((B., Venturi, and Bruna, 2021))

As $k \to \infty$, we have

$$\gamma_d(k) = \frac{1}{|G|} + O(k^{-d+c}),$$

where c is the maximal number of cycles of any permutation $\sigma \in G \setminus \{Id\}$.

Counting invariant harmonics

$$\gamma_d(k) := \frac{\overline{N}(d,k)}{N(d,k)} = \frac{1}{|G|} \sum_{\sigma \in G} \mathbb{E}_x[P_{d,k}(\langle \sigma \cdot x, x \rangle)].$$

Proposition ((B., Venturi, and Bruna, 2021))

As $k \to \infty$, we have

$$\gamma_d(k) = \frac{1}{|G|} + O(k^{-d+c}),$$

where c is the maximal number of cycles of any permutation $\sigma \in G \setminus \{Id\}$.

- Relies on singularity analysis of density of $\langle \sigma \cdot x, x \rangle$ (Saldanha and Tomei, 1996)
- c can be large ($= d - 1$) for some groups (e.g. cyclic on blocks of size 2, $|G| = 2^{d/2}$)
- Can use upper bounds with faster decays but larger constants

Counting invariant harmonics

$$\gamma_d(k) := \frac{\overline{N}(d,k)}{N(d,k)} = \frac{1}{|G|} \sum_{\sigma \in G} \mathbb{E}_x[P_{d,k}(\langle \sigma \cdot x, x \rangle)].$$

Proposition ((B., Venturi, and Bruna, 2021))

As $k \to \infty$, we have

$$\gamma_d(k) = \frac{1}{|G|} + O(k^{-d+c}),$$

where c is the maximal number of cycles of any permutation $\sigma \in G \setminus \{Id\}$.

- Relies on singularity analysis of density of $\langle \sigma \cdot x, x \rangle$ (Saldanha and Tomei, 1996)
- c can be large $(= d - 1)$ for some groups (e.g. cyclic on blocks of size 2, $|G| = 2^{d/2}$)
- Can use upper bounds with faster decays but larger constants
- Comparison to Mei et al. (2021): they study $d \to \infty$ with fixed k ($\gamma_d(k) = \Theta_d(d^{-\alpha})$), we study $k \to \infty$ with fixed d

Sample complexity of invariant kernel

Assumptions for Kernel Ridge Regression

- (G-invariance) $f^*(x) = \mathbb{E}[y|x]$ is G-invariant
- (*capacity*) $\lambda_m(T_K) \leq C_K m^{-\alpha}$
- (*source*) $f^* = T_K^r g^*$ with $\|g^*\|_{L^2} \leq C_{f^*}$

Sample complexity of invariant kernel

Assumptions for Kernel Ridge Regression

- (G-invariance) $f^*(x) = \mathbb{E}[y|x]$ is G-invariant
- (*capacity*) $\lambda_m(T_K) \leq C_K m^{-\alpha}$
- (*source*) $f^* = T_K^r g^*$ with $\|g^*\|_{L^2} \leq C_{f^*}$

Theorem ((B., Venturi, and Bruna, 2021))

Let $\ell_n := \sup\{\ell : D(\ell) \lesssim \nu_d(\ell)^{\frac{2\alpha r}{2\alpha r+1}} n^{\frac{1}{2\alpha r+1}}\}$. *(replace* $\nu_d(\ell_n)$ *by* 1 *for non-invariant kernel)*

$$\mathbb{E}\,\|\hat{f} - f^*\|^2_{L^2(\tau)} \leq C \left(\frac{\nu_d(\ell_n)}{n}\right)^{\frac{2\alpha r}{2\alpha r+1}}$$

Sample complexity of invariant kernel

Assumptions for Kernel Ridge Regression

- (G-invariance) $f^*(x) = \mathbb{E}[y|x]$ is G-invariant
- (*capacity*) $\lambda_m(T_K) \leq C_K m^{-\alpha}$
- (*source*) $f^* = T_K^r g^*$ with $\|g^*\|_{L^2} \leq C_{f^*}$

Theorem ((B., Venturi, and Bruna, 2021))

Let $\ell_n := \sup\{\ell : D(\ell) \lesssim \nu_d(\ell)^{\frac{2\alpha r}{2\alpha r+1}} n^{\frac{1}{2\alpha r+1}}\}$. *(replace* $\nu_d(\ell_n)$ *by* 1 *for non-invariant kernel)*

$$\mathbb{E}\|\hat{f} - f^*\|^2_{L^2(\tau)} \leq C \left(\frac{\nu_d(\ell_n)}{n}\right)^{\frac{2\alpha r}{2\alpha r+1}}$$

- We have $\nu_d(\ell_n) = \frac{1}{|G|} + O(n^{\frac{-\beta}{(d-1)(2\alpha r+1)+2\beta\alpha r}})$ when $\gamma_d(k) = 1/|G| + O(k^{-\beta})$
- $\implies$ **Improvement in sample complexity** by a factor $|G|$!
- C may depend on d, but is optimal in a minimax sense over non-invariant f^*

Synthetic experiments

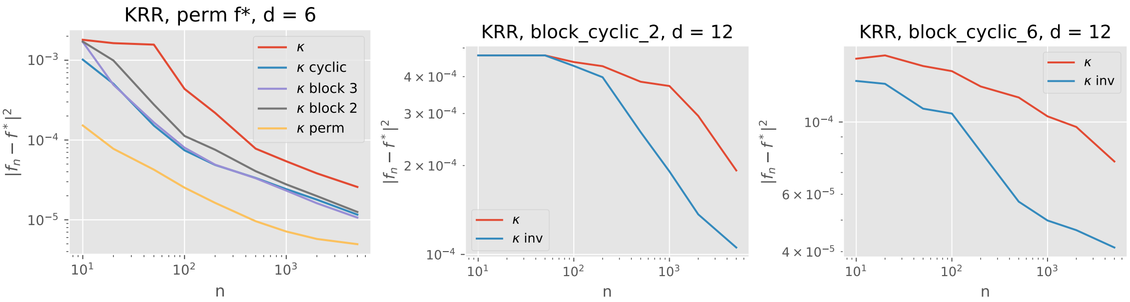


Figure: Comparison of KRR with invariant and non-invariant kernels.

Stability

- S_G is no longer a projection, but its eigenvalues satisfy $\gamma_d(k) = (\sum_{j=1}^{N(d,k)} \lambda_{k,j})/N(d,k)$
- Source condition adapted to S_G: $f^* = S_G^r T_K^r g^*$ with $\|g^*\|_{L^2} \leq C_{f^*}$

Theorem ((B., Venturi, and Bruna, 2021))

Let $\ell_n := \sup\{\ell : D(\ell) \lesssim \nu_d(\ell)^{\frac{2r}{2\alpha r+1}} n^{\frac{1}{2\alpha r+1}}\}$. *(replace* $\nu_d(\ell_n)$ *by* 1 *for non-invariant kernel)*

$$\mathbb{E}\,\|\hat{f} - f^*\|^2_{L^2(\tau)} \leq C \left(\frac{\nu_d(\ell_n)^{1/\alpha}}{n} \right)^{\frac{2\alpha r}{2\alpha r+1}}$$

Stability

- S_G is no longer a projection, but its eigenvalues satisfy $\gamma_d(k) = (\sum_{j=1}^{N(d,k)} \lambda_{k,j})/N(d,k)$
- Source condition adapted to S_G: $f^* = S_G^r T_K^r g^*$ with $\|g^*\|_{L^2} \leq C_{f^*}$

Theorem ((B., Venturi, and Bruna, 2021))

Let $\ell_n := \sup\{\ell : D(\ell) \lesssim \nu_d(\ell)^{\frac{2r}{2\alpha r+1}} n^{\frac{1}{2\alpha r+1}}\}$. *(replace* $\nu_d(\ell_n)$ *by* 1 *for non-invariant kernel)*

$$\mathbb{E}\,\|\hat{f} - f^*\|^2_{L^2(\tau)} \leq C\left(\frac{\nu_d(\ell_n)^{1/\alpha}}{n}\right)^{\frac{2\alpha r}{2\alpha r+1}}$$

Toy model for deformations ("small $\|\nabla\sigma - I\|$")

$$G := \{\sigma \in \mathcal{S}_d \ : \ |\sigma(u) - \sigma(u') - (u - u')| \leq \varepsilon|u - u'|\}$$

- Can achieve $\gamma_d(k) \leq \tau^d + O(k^{-\Theta(d)})$, with $\tau < 1 \implies$ this leads to gains by a factor **exponential** in d with a rate independent of d in $\nu_d(\ell_n)$!

Discussion

Curse of dimensionality

- For Lipschitz targets, cursed rate $n^{-\frac{2\alpha r}{2\alpha r+1}} = n^{-\frac{2}{2+d-1}}$ (unimprovable)
- Improving this rate requires more structural assumptions, and better architectures (up next!) or adaptivity (Bach, 2017)

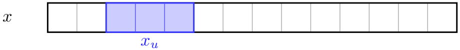
Comparison with $(Mei, Misiakiewicz, and Montanari, \ 2021)$

- Different asymptotics (us: $n \to \infty$ with d fixed, them: $d \to \infty$ with $n \sim d^\ell$)
- Their regimes only allow gains by polynomial factors in d
- We may achieve gains by exponential factors (when $|G|$ is exponential in d), but only asymptotically

Outline

1. Sample complexity under invariance and stability (B., Venturi, and Bruna, 2021)
2. Locality and depth (B., 2021)

Breaking the curse of dimensionality with locality

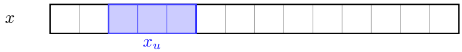


One-layer local convolutional kernel: localized patches $x_u = (x[u], \ldots, x[u+s])$ (1D)

$$K(x, x') = \sum_{u \in \Omega} k(x_u, x'_u)$$

- RKHS $\mathcal{H}_K$ contains functions $f(x) = \sum_{u \in \Omega} g_u(x_u)$ with $g_u \in \mathcal{H}_k$
- **No curse**: Smoothness requirement on g_u scales with s instead of d
- **Pooling** further encourages similarities between the g_u,

Breaking the curse of dimensionality with locality



One-layer local convolutional kernel: localized patches $x_u = (x[u], \ldots, x[u+s])$ (1D)

$$K(x, x') = \sum_{u \in \Omega} k(x_u, x'_u)$$

- RKHS $\mathcal{H}_K$ contains functions $f(x) = \sum_{u \in \Omega} g_u(x_u)$ with $g_u \in \mathcal{H}_k$
- **No curse**: Smoothness requirement on g_u scales with s instead of d
- **Pooling** further encourages similarities between the g_u,

Generalization

$$\mathbb{E}\, L(\hat{f}_n) - L(f^*) \lesssim \|f^*\|_{\mathcal{H}_K} \sqrt{\frac{\mathbb{E}_x\, K(x, x)}{n}}$$

- For invariant targets, $\|f^*\|$ independent of pooling, $\mathbb{E}_x\, K(x, x)$ improves with pooling
- Fast rates possible (Favero et al., 2021)

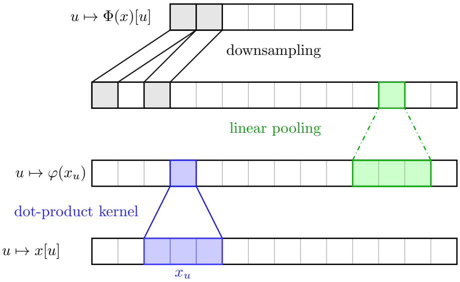
Adding convolutional layers

- Fully-connected kernels: deep = shallow (B. and Bach, 2021; Chen and Xu, 2021)
- Can depth help for structured kernels?

Adding convolutional layers

- Fully-connected kernels: deep = shallow (B. and Bach, 2021; Chen and Xu, 2021)
- Can depth help for structured kernels?

Convolutional Kernel networks (Mairal, 2016)



Some experiments on Cifar10

2-layers, 3x3 patches, pooling/downsampling sizes (2,5). Patch kernels κ_1, κ_2.

κ_1	κ_2	Test acc. (10k examples)	Test acc. (50k examples)
Exp	Exp	80.5%	87.9% (84.1%)
Exp	Poly3	80.5%	87.7% (84.1%)
Exp	Poly2	79.4%	86.9% (83.4%)
Poly2	Exp	77.4%	- (81.5%)
Poly2	Poly2	75.1%	- (81.2%)
Exp	- (Lin)	74.2%	- (76.3%)

In parentheses: Nyström approximation of the kernel (Mairal, 2016) with [256,4096] filters, instead of the full kernel.

Structured interaction models via depth and pooling

RKHS of 2-layer convolutional kernel with quadratic κ_2: Contains functions

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} g_{u,v}^{pq}(x_u, x_v),$$

with $g_{u,v}^{pq} = 0$ if $|u - v - (p - q)| > \mathrm{diam}(\mathrm{supp}(h_1))$.

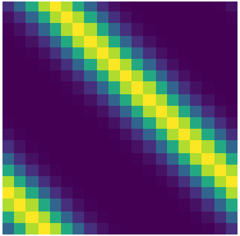
Structured interaction models via depth and pooling

RKHS of 2-layer convolutional kernel with quadratic κ_2: Contains functions

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} g_{u,v}^{pq}(x_u, x_v),$$

with $g_{u,v}^{pq} = 0$ if $|u - v - (p - q)| > \mathrm{diam}(\mathrm{supp}(h_1))$.

- Tensor-product ANOVA model: $g_{u,v}^{pq} \in \mathcal{H}_k \otimes \mathcal{H}_k$
- Still no curse if $2s \ll d$
- Pooling layers encourage similarities between different $g_{u,v}^{pq}$



Improvements in generalization

$$\mathbb{E}\, L(\hat{f}_n) - L(f^*) \lesssim \|f^*\|_{\mathcal{H}_K} \sqrt{\frac{\mathbb{E}_x\, K(x,x)}{n}}$$

- Consider $f^*(x) = \sum_{u,v \in \Omega} g^*(x_u, x_v)$ with $g^* \in \mathcal{H}_k \otimes \mathcal{H}_k$
- Assume $\mathbb{E}_x[k(x_u, x_{u'})k(x_v, x_{v'})] \leq \epsilon$ if $u \neq u'$ or $v \neq v'$
- Obtained bound for different pooling layers (h_1, h_2) and patch sizes $(|S_2|)$:

h_1	h_2	$\lvert S_2 \rvert$	$\|f^*\|_K$	$\mathbb{E}_x\, K(x,x)$	Bound $(\epsilon = 0)$
δ	δ	$\lvert\Omega\rvert$	$\lvert\Omega\rvert \|g\|$	$\lvert\Omega\rvert^3 + \epsilon\lvert\Omega\rvert^3$	$\|g\| \lvert\Omega\rvert^{2.5}/\sqrt{n}$
δ	$\mathbf{1}$	$\lvert\Omega\rvert$	$\lvert\Omega\rvert \|g\|$	$\lvert\Omega\rvert^2 + \epsilon\lvert\Omega\rvert^3$	$\|g\| \lvert\Omega\rvert^{2}/\sqrt{n}$
$\mathbf{1}$	$\mathbf{1}$	$\lvert\Omega\rvert$	$\sqrt{\lvert\Omega\rvert} \|g\|$	$\lvert\Omega\rvert + \epsilon\lvert\Omega\rvert^3$	$\|g\| \lvert\Omega\rvert/\sqrt{n}$
$\mathbf{1}$	δ or $\mathbf{1}$	1	$\sqrt{\lvert\Omega\rvert} \|g\|$	$\lvert\Omega\rvert^{-1} + \epsilon\lvert\Omega\rvert$	$\|g\|/\sqrt{n}$

Note: larger polynomial improvements in $|\Omega|$ possible with higher-order interactions.

Conclusion and perspectives

Summary

- Improved sample complexity for invariance and stability through pooling
- Locality breaks the curse
- Depth and pooling in convolutional models captures rich interaction models with invariances

Future directions

- Empirical benefits for kernels beyond two-layers?
- Invariance groups need to be built-in, can we adapt to them?
- Adaptivity to structure beyond one-layer:
  - low-dimensional structures (Gabor) at first layer?
  - more structured interactions at second layer?
  - optimization beyond kernel regimes?

Conclusion and perspectives

Summary

- Improved sample complexity for invariance and stability through pooling
- Locality breaks the curse
- Depth and pooling in convolutional models captures rich interaction models with invariances

Future directions

- Empirical benefits for kernels beyond two-layers?
- Invariance groups need to be built-in, can we adapt to them?
- Adaptivity to structure beyond one-layer:
  - low-dimensional structures (Gabor) at first layer?
  - more structured interactions at second layer?
  - optimization beyond kernel regimes?

Thank you!

References I

A. B. Approximation and learning with deep convolutional models: a kernel perspective. *arXiv preprint arXiv:2102.10032*, 2021.

A. B. and F. Bach. Deep equals shallow for relu networks in kernel regimes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

A. B. and J. Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *Journal of Machine Learning Research (JMLR)*, 20(25):1–49, 2019.

A. B., L. Venturi, and J. Bruna. On the sample complexity of learning with geometric stability. *arXiv preprint arXiv:2106.07148*, 2021.

F. Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research (JMLR)*, 18(19):1–53, 2017.

A. Bietti and J. Mairal. On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(8):1872–1886, 2013.

L. Chen and S. Xu. Deep neural tangent kernel and laplace kernel have the same rkhs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

References II

L. Chizat, E. Oyallon, and F. Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Y. Cho and L. K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

C. Ciliberto, F. Bach, and A. Rudi. Localized structured prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

T. Cohen and M. Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning (ICML)*, 2016.

A. Favero, F. Cagnetta, and M. Wyart. Locality defeats the curse of dimensionality in convolutional teacher-student scenarios. *arXiv preprint arXiv:2106.08619*, 2021.

B. Haasdonk and H. Burkhardt. Invariant kernel functions for pattern analysis and machine learning. *Machine learning*, 68(1):35–61, 2007.

A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

R. Kondor and S. Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

References III

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

Y. Lin. Tensor product space anova models. *Annals of Statistics*, 28(3):734–755, 2000.

J. Mairal. End-to-End Kernel Learning with Supervised Convolutional Kernel Networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid. Convolutional kernel networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

S. Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10): 1331–1398, 2012.

S. Mei, T. Misiakiewicz, and A. Montanari. Learning with invariances in random features and kernel models. In *Conference on Learning Theory (COLT)*, 2021.

Y. Mroueh, S. Voinea, and T. A. Poggio. Learning with group invariant features: A kernel perspective. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

R. M. Neal. *Bayesian learning for neural networks*. Springer, 1996.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

References IV

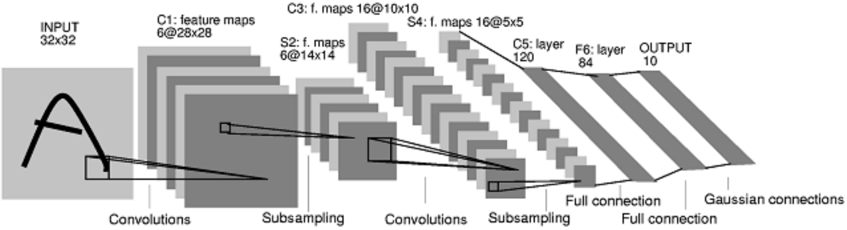
N. C. Saldanha and C. Tomei. The accumulated distribution of quadratic forms on the sphere. *Linear algebra and its applications*, 245:335–351, 1996.

M. Scetbon and Z. Harchaoui. Harmonic decompositions of convolutional networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

V. Shankar, A. Fang, W. Guo, S. Fridovich-Keil, L. Schmidt, J. Ragan-Kelley, and B. Recht. Neural kernels without tangents. *arXiv preprint arXiv:2003.02237*, 2020.

G. Wahba. *Spline models for observational data*, volume 59. Siam, 1990.

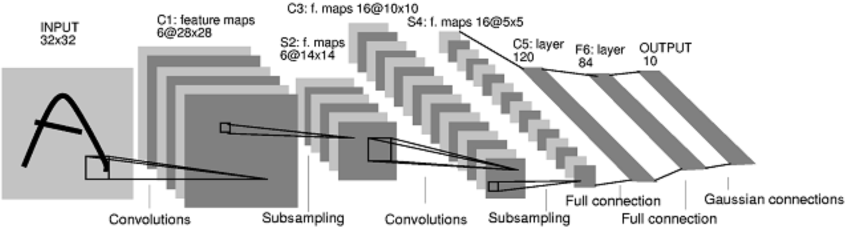
Folklore properties of convolutional models



Convolutional architectures:

- Capture **multi-scale** and **compositional** structure in natural signals
- Model **local stationarity**
- Provide some **translation invariance**

Folklore properties of convolutional models



Convolutional architectures:

- Capture **multi-scale** and **compositional** structure in natural signals
- Model **local stationarity**
- Provide some **translation invariance**

Beyond translation invariance?

One-layer convolutional kernel with pooling

- $h[u]$: pooling filter (*e.g.*, Gaussian)
- A_h: (circular) convolution operator $A_h x[u] = \sum_{v \in \Omega} h[u-v]x[v]$
- $\Phi(x)[u] = \varphi(x_u) \in \mathcal{H}$ (φ: kernel mapping of k)

1-layer convolutional kernel

$$K(x, x') = \sum_{u \in \Omega} \sum_{v, v'} h[u-v]h[u-v']k(x_v, x'_{v'}) = \langle A_h \Phi(x), A_h \Phi(x') \rangle_{L^2(\Omega, \mathcal{H})}$$

Functions in RKHS: Same functions $f(x) = \sum_u g[u](x_u)$, different **penalty**: $\|A_h^\dagger g\|^2_{L^2(\Omega,\mathcal{H})}$.

One-layer convolutional kernel with pooling

- $h[u]$: pooling filter (*e.g.*, Gaussian)
- A_h: (circular) convolution operator $A_h x[u] = \sum_{v \in \Omega} h[u-v]x[v]$
- $\Phi(x)[u] = \varphi(x_u) \in \mathcal{H}$ (φ: kernel mapping of k)

1-layer convolutional kernel

$$K(x, x') = \sum_{u \in \Omega} \sum_{v, v'} h[u-v]h[u-v']k(x_v, x'_{v'}) = \langle A_h \Phi(x), A_h \Phi(x') \rangle_{L^2(\Omega, \mathcal{H})}$$

Functions in RKHS: Same functions $f(x) = \sum_u g[u](x_u)$, different **penalty**: $\|A_h^\dagger g\|^2_{L^2(\Omega,\mathcal{H})}$.

$\implies$ Encourages spatial smoothness: for $g_z[u] := g[u](z)$, we have

$$\widehat{A_h^\dagger g_z}[w] = \frac{\hat{g}_z[w]}{\hat{h}[w]}$$

Large pooling $\leftrightarrow$ fast decay of $\hat{h}[w]$ $\leftrightarrow$ stronger penalty on high frequencies of g_z.

Generalization benefits of pooling

Translation-invariant target $f^*(x) = \sum_u g(x_u)$, with $g \in \mathcal{H}$.

Learn using kernel K_h with pooling with filter $h \geq 0$, $\|h\|_1 = 1$, *e.g.*:

- **no pooling**: $h[u] = \delta_{u,0}$
- **global pooling**: $h[u] = 1/|\Omega|$
- $A_h(g, \dots, g) = (g, \dots, g) \implies$ **same RKHS norm** for any h!

Generalization benefits of pooling

Translation-invariant target $f^*(x) = \sum_u g(x_u)$, with $g \in \mathcal{H}$.

Learn using kernel K_h with pooling with filter $h \geq 0$, $\|h\|_1 = 1$, *e.g.*:

- **no pooling**: $h[u] = \delta_{u,0}$
- **global pooling**: $h[u] = 1/|\Omega|$
- $A_h(g, \dots, g) = (g, \dots, g) \implies$ **same RKHS norm** for any h!

Basic generalization bound with 1-Lipschitz loss on $\mathcal{F} = \{\|f\|_{K_h} \leq B\}$

$$\mathbb{E}\, L(f_n) - \min_{f \in \mathcal{F}} L(f) \lesssim \frac{B\sqrt{\mathbb{E}_x[K_h(x,x)]}}{\sqrt{n}}$$

Under simple data models, $\mathbb{E}_x[k(x_u, x_u)] = 1$, $\mathbb{E}_x[k(x_u, x_v)] \leq \epsilon \ll 1$ for $u \neq v$

- **no pooling**: $\mathbb{E}[K_h(x,x)] = |\Omega|$
- **global pooling**: $\mathbb{E}[K_h(x,x)] \leq 1 + \epsilon|\Omega|$

Generalization benefits of pooling

Translation-invariant target $f^*(x) = \sum_u g(x_u)$, with $g \in \mathcal{H}$.

Learn using kernel K_h with pooling with filter $h \geq 0$, $\|h\|_1 = 1$, *e.g.*:

- **no pooling**: $h[u] = \delta_{u,0}$
- **global pooling**: $h[u] = 1/|\Omega|$
- $A_h(g, \dots, g) = (g, \dots, g) \implies$ **same RKHS norm** for any h!

Basic generalization bound with 1-Lipschitz loss on $\mathcal{F} = \{\|f\|_{K_h} \leq B\}$

$$\mathbb{E}\, L(f_n) - \min_{f \in \mathcal{F}} L(f) \lesssim \frac{B\sqrt{\mathbb{E}_x[K_h(x,x)]}}{\sqrt{n}}$$

Under simple data models, $\mathbb{E}_x[k(x_u, x_u)] = 1$, $\mathbb{E}_x[k(x_u, x_v)] \leq \epsilon \ll 1$ for $u \neq v$

- **no pooling**: $\mathbb{E}[K_h(x,x)] = |\Omega|$
- **global pooling**: $\mathbb{E}[K_h(x,x)] \leq 1 + \epsilon|\Omega| \implies$ **need $\sim |\Omega|$ fewer samples!**

Generalization benefits of pooling

Translation-invariant target $f^*(x) = \sum_u g(x_u)$, with $g \in \mathcal{H}$.

Learn using kernel K_h with pooling with filter $h \geq 0$, $\|h\|_1 = 1$, *e.g.*:

- **no pooling**: $h[u] = \delta_{u,0}$
- **global pooling**: $h[u] = 1/|\Omega|$
- $A_h(g, \ldots, g) = (g, \ldots, g) \implies$ **same RKHS norm** for any h!

Basic generalization bound with 1-Lipschitz loss on $\mathcal{F} = \{\|f\|_{K_h} \leq B\}$

$$\mathbb{E}\, L(f_n) - \min_{f \in \mathcal{F}} L(f) \lesssim \frac{B\sqrt{\mathbb{E}_x[K_h(x,x)]}}{\sqrt{n}}$$

Under simple data models, $\mathbb{E}_x[k(x_u, x_u)] = 1$, $\mathbb{E}_x[k(x_u, x_v)] \leq \epsilon \ll 1$ for $u \neq v$

- **no pooling**: $\mathbb{E}[K_h(x,x)] = |\Omega|$
- **global pooling**: $\mathbb{E}[K_h(x,x)] \leq 1 + \epsilon|\Omega| \implies$ **need $\sim |\Omega|$ fewer samples!**
- General h: $\mathbb{E}[K_h(x,x)] \leq |\Omega|\|h\|_2^2 + \epsilon|\Omega|(1 - \|h\|_2^2)$

Two-layer convolutional kernel

- Quadratic patch kernel $k_2(z, z') = (z^\top z')^2 = \langle z \otimes z, z' \otimes z' \rangle_{(\mathcal{H} \otimes \mathcal{H})^{|S_2| \times |S_2|}}$
- $\mathcal{H} \otimes \mathcal{H}$: contains functions $g(x_u, x_v)$ of 2 patches (Wahba, 1990)

Two-layer convolutional kernel

- Quadratic patch kernel $k_2(z, z') = (z^\top z')^2 = \langle z \otimes z, z' \otimes z' \rangle_{(\mathcal{H} \otimes \mathcal{H})^{|S_2| \times |S_2|}}$
- $\mathcal{H} \otimes \mathcal{H}$: contains functions $g(x_u, x_v)$ of 2 patches (Wahba, 1990)

RKHS of 2-layer convolutional kernel (patch size $|S_2| = 1$): Contains functions

$$f(x) = \sum_{u,v \in \Omega} G[u, v](x_u, x_v),$$

with $G[u, v] = 0$ if $|u - v| > \mathrm{diam}(\mathrm{supp}(h_1))$. **Penalty**:

$$\|A_2^\dagger \operatorname{diag}((A_1 \otimes A_1)^\dagger G)\|^2_{L^2(\Omega_2, \mathcal{H} \otimes \mathcal{H})}$$

Two-layer convolutional kernel

- Quadratic patch kernel $k_2(z, z') = (z^\top z')^2 = \langle z \otimes z, z' \otimes z' \rangle_{(\mathcal{H} \otimes \mathcal{H})^{|S_2| \times |S_2|}}$
- $\mathcal{H} \otimes \mathcal{H}$: contains functions $g(x_u, x_v)$ of 2 patches (Wahba, 1990)

RKHS of 2-layer convolutional kernel (patch size $|S_2| = 1$): Contains functions

$$f(x) = \sum_{u,v \in \Omega} G[u, v](x_u, x_v),$$

with $G[u, v] = 0$ if $|u - v| > \mathrm{diam}(\mathrm{supp}(h_1))$. **Penalty**:

$$\|A_2^\dagger \, \mathrm{diag}((A_1 \otimes A_1)^\dagger G)\|^2_{L^2(\Omega_2, \mathcal{H} \otimes \mathcal{H})}$$

- $(A_1 \otimes A_1)^\dagger$: encourages 2D smoothness of "image" $G[u, v]$, bandwidth σ_1
- $A_2^\dagger$: encourage 1D smoothness along diagonal of G, bandwidth σ_2
- $\sigma_1 > \sigma_2 \implies G[u, v]$ can depend more strongly on $u - v$ than u or v

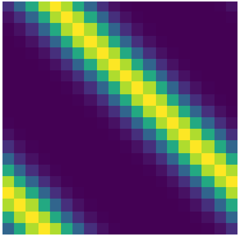
Two-layer convolutional kernel

RKHS of 2-layer convolutional kernel (any patch size $|S_2|$): Contains functions

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} G_{pq}[u,v](x_u, x_v),$$

with $G_{pq}[u,v] = 0$ if $|u - v - (p - q)| > \operatorname{diam}(\operatorname{supp}(h_1))$. **Penalty**:

$$\sum_{p,q \in S_2} \| A_2^\dagger \operatorname{diag}((L_p A_1 \otimes L_q A_1)^\dagger G_{pq}) \|^2_{L^2(\Omega_2, \mathcal{H} \otimes \mathcal{H})}$$



Two-layer convolutional kernel

RKHS of 2-layer convolutional kernel (any patch size $|S_2|$): Contains functions

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} G_{pq}[u,v](x_u, x_v),$$

with $G_{pq}[u,v] = 0$ if $|u - v - (p-q)| > \mathrm{diam}(\mathrm{supp}(h_1))$. **Penalty**:

$$\sum_{p,q \in S_2} \| A_2^\dagger \, \mathrm{diag}((L_p A_1 \otimes L_q A_1)^\dagger G_{pq}) \|^2_{L^2(\Omega_2, \mathcal{H} \otimes \mathcal{H})}$$

"Variance" term: $\sqrt{\mathbb{E}_x[K(x,x)]} \leq |\Omega||S_2|^2 \sum_v \langle h_2, L_v h_w \rangle \langle h_1, L_v h_1 \rangle^2 + O(\epsilon)$

Two-layer convolutional kernel

RKHS of 2-layer convolutional kernel (any patch size $|S_2|$): Contains functions

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} G_{pq}[u,v](x_u, x_v),$$

with $G_{pq}[u,v] = 0$ if $|u - v - (p - q)| > \mathrm{diam}(\mathrm{supp}(h_1))$. **Penalty**:

$$\sum_{p,q \in S_2} \|A_2^\dagger \operatorname{diag}((L_p A_1 \otimes L_q A_1)^\dagger G_{pq})\|^2_{L^2(\Omega_2, \mathcal{H} \otimes \mathcal{H})}$$

"Variance" term: $\sqrt{\mathbb{E}_x[K(x,x)]} \leq |\Omega||S_2|^2 \sum_v \langle h_2, L_v h_w \rangle \langle h_1, L_v h_1 \rangle^2 + O(\epsilon)$

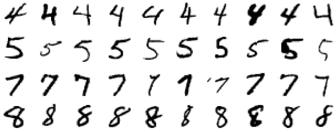
Extensions:

- κ_2 higher-degree polynomial $\implies$ higher-order interactions
- more layers: also higher-order interactions, but more structured penalty

Stability to deformations

Deformations

- $\tau : \Omega \to \Omega$: C^1-diffeomorphism
- $L_\tau x(u) = x(u - \tau(u))$: action operator
- Much richer group of transformations than translations



- Studied for wavelet-based scattering transform (Mallat, 2012; Bruna and Mallat, 2013)

Stability to deformations

Deformations

- $\tau : \Omega \to \Omega$: C^1-diffeomorphism
- $L_\tau x(u) = x(u - \tau(u))$: action operator
- Much richer group of transformations than translations

Definition of stability

- Representation $\Phi(\cdot)$ is **stable** (Mallat, 2012) if:

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1 \|\nabla \tau\|_\infty + C_2 \|\tau\|_\infty) \|x\|$$

- $\|\nabla \tau\|_\infty = \sup_u \|\nabla \tau(u)\|$ controls deformation
- $\|\tau\|_\infty = \sup_u |\tau(u)|$ controls translation
- $C_2 \to 0$: translation invariance

Smoothness and stability with kernels

Geometry of the kernel mapping: $f(x) = \langle f, \Phi(x) \rangle$

$$|f(x) - f(x')| \leq \|f\|_{\mathcal{H}} \cdot \|\Phi(x) - \Phi(x')\|_{\mathcal{H}}$$

- $\|f\|_{\mathcal{H}}$ controls **complexity** of the model
- $\Phi(x)$ encodes CNN **architecture** independently of the model (smoothness, invariance, stability to deformations)

Smoothness and stability with kernels

Geometry of the kernel mapping: $f(x) = \langle f, \Phi(x) \rangle$

$$|f(x) - f(x')| \leq \|f\|_{\mathcal{H}} \cdot \|\Phi(x) - \Phi(x')\|_{\mathcal{H}}$$

- $\|f\|_{\mathcal{H}}$ controls **complexity** of the model
- $\Phi(x)$ encodes CNN **architecture** independently of the model (smoothness, invariance, stability to deformations)

Useful kernels in practice:

- Convolutional kernel networks (**CKNs**, Mairal, 2016) with efficient approximations
- Extends to neural tangent kernels (**NTKs**, Jacot et al., 2018) of infinitely wide CNNs (Bietti and Mairal, 2019)

Construction of convolutional kernels

Construct a sequence of feature maps $x_1, \ldots, x_n$

- $x_0 : \Omega \to \mathcal{H}_0$: initial (**continuous**) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)

Construction of convolutional kernels

Construct a sequence of feature maps $x_1, \dots, x_n$

- $x_0 : \Omega \to \mathcal{H}_0$: initial (**continuous**) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \to \mathcal{H}_k$: **feature map** at layer k

$$P_k x_{k-1}$$

  - P_k: **patch extraction** operator, extract small patch of feature map x_{k-1} around each point u

Construction of convolutional kernels

Construct a sequence of feature maps $x_1, \ldots, x_n$

- $x_0 : \Omega \to \mathcal{H}_0$: initial (**continuous**) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \to \mathcal{H}_k$: **feature map** at layer k

$$M_k P_k x_{k-1}$$

  - P_k: **patch extraction** operator, extract small patch of feature map x_{k-1} around each point u
  - M_k: **non-linear mapping** operator, maps each patch to a new point with a **pointwise** non-linear function $\varphi_k(\cdot)$ (kernel mapping)

Construction of convolutional kernels

Construct a sequence of feature maps $x_1, \ldots, x_n$

- $x_0 : \Omega \to \mathcal{H}_0$: initial (**continuous**) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \to \mathcal{H}_k$: **feature map** at layer k

$$x_k = A_k M_k P_k x_{k-1}$$

  - P_k: **patch extraction** operator, extract small patch of feature map x_{k-1} around each point u
  - M_k: **non-linear mapping** operator, maps each patch to a new point with a **pointwise** non-linear function $\varphi_k(\cdot)$ (kernel mapping)
  - A_k: (linear, Gaussian) **pooling** operator at scale σ_k

Construction of convolutional kernels

Construct a sequence of feature maps $x_1, \ldots, x_n$

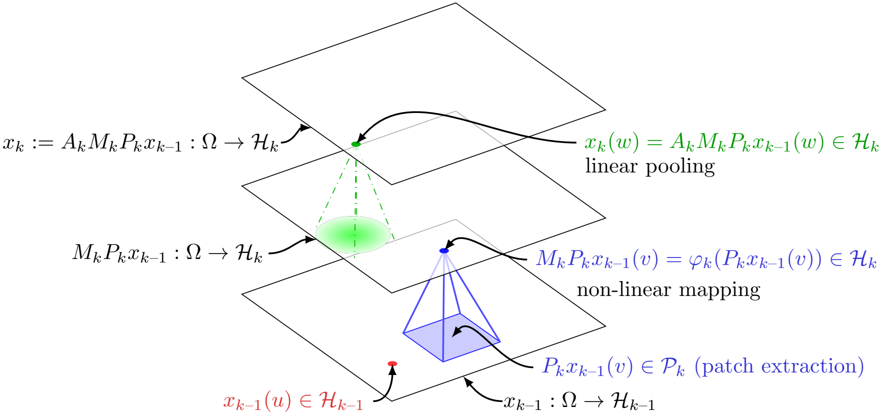
- $x_0 : \Omega \to \mathcal{H}_0$: initial (**continuous**) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \to \mathcal{H}_k$: **feature map** at layer k

$$x_k = A_k M_k P_k x_{k-1}$$

- P_k: **patch extraction** operator, extract small patch of feature map x_{k-1} around each point u
- M_k: **non-linear mapping** operator, maps each patch to a new point with a **pointwise** non-linear function $\varphi_k(\cdot)$ (kernel mapping)
- A_k: (linear, Gaussian) **pooling** operator at scale σ_k

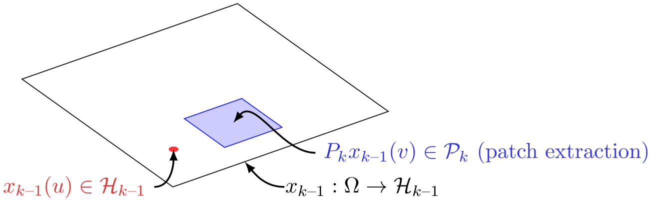
Goal: control stability of these operators through their norms

CKN construction



Patch extraction operator P_k

$$P_k x_{k-1}(u) := (x_{k-1}(u+v))_{v \in S_k} \in \mathcal{P}_k = \mathcal{H}_{k-1}^{S_k}$$



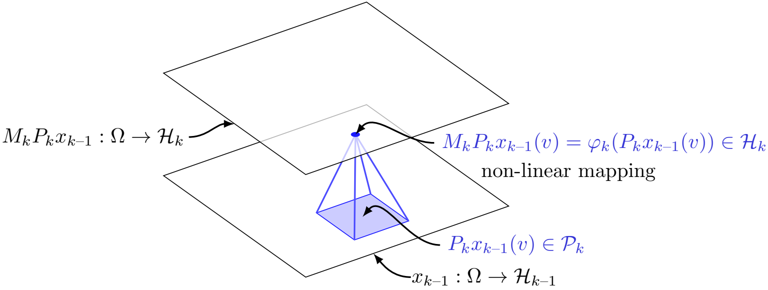
Patch extraction operator P_k

$$P_k x_{k-1}(u) := (x_{k-1}(u+v))_{v \in S_k} \in \mathcal{P}_k = \mathcal{H}_{k-1}^{S_k}$$

- S_k: patch shape, e.g. box
- P_k is **linear**, and **preserves the L^2 norm**: $\|P_k x_{k-1}\| = \|x_{k-1}\|$

Non-linear mapping operator M_k

$$M_k P_k x_{k-1}(u) := \varphi_k(P_k x_{k-1}(u)) \in \mathcal{H}_k$$



Non-linear mapping operator M_k

$$M_k P_k x_{k-1}(u) := \varphi_k(P_k x_{k-1}(u)) \in \mathcal{H}_k$$

- $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$ pointwise non-linearity on patches (kernel map)
- We assume **non-expansivity**: for $z, z' \in \mathcal{P}_k$

$$\|\varphi_k(z)\| \leq \|z\| \quad \text{and} \quad \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\|$$

- M_k then satisfies, for $x, x' \in L^2(\Omega, \mathcal{P}_k)$

$$\|M_k x\| \leq \|x\| \quad \text{and} \quad \|M_k x - M_k x'\| \leq \|x - x'\|$$

φ_k from kernels

Kernel mapping of **homogeneous dot-product kernels**:

$$K_k(z, z') = \|z\|\|z'\|\kappa_k\left(\frac{\langle z, z'\rangle}{\|z\|\|z'\|}\right) = \langle \varphi_k(z), \varphi_k(z')\rangle.$$

$\kappa_k(u) = \sum_{j=0}^{\infty} b_j u^j$ with $b_j \geq 0$, $\kappa_k(1) = 1$

- Commonly used for hierarchical kernels
- $\|\varphi_k(z)\| = K_k(z, z)^{1/2} = \|z\|$
- $\|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\|$ if $\kappa_k'(1) \leq 1$
- $\implies$ **non-expansive**

φ_k from kernels

Kernel mapping of **homogeneous dot-product kernels**:

$$K_k(z, z') = \|z\|\|z'\|\kappa_k\left(\frac{\langle z, z'\rangle}{\|z\|\|z'\|}\right) = \langle \varphi_k(z), \varphi_k(z')\rangle.$$

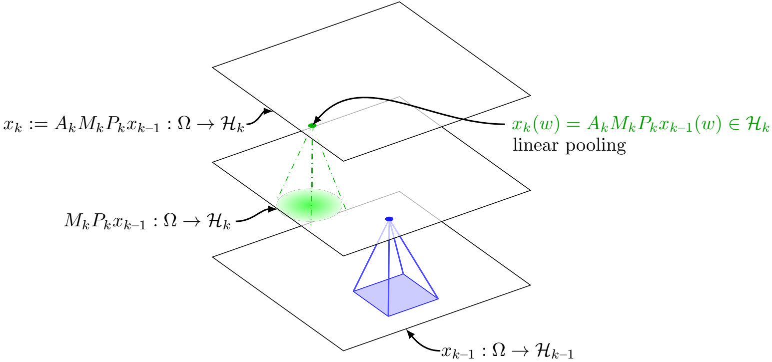
$\kappa_k(u) = \sum_{j=0}^{\infty} b_j u^j$ with $b_j \geq 0$, $\kappa_k(1) = 1$

Examples

- $\kappa_{\exp}(\langle z, z'\rangle) = e^{\langle z,z'\rangle - 1}$ (Gaussian kernel on the sphere)
- $\kappa_{\text{inv-poly}}(\langle z, z'\rangle) = \frac{1}{2 - \langle z, z'\rangle}$
- $\kappa_\sigma(\langle z, z'\rangle) = \mathbb{E}_w[\sigma(w^\top z)\sigma(w^\top z')]$ (Random features)
  - arc-cosine kernels for the ReLU $\sigma(u) = \max(0, u)$

Pooling operator A_k

$$x_k(u) = A_k M_k P_k x_{k-1}(u) = \int_{\mathbb{R}^d} h_{\sigma_k}(u - v) M_k P_k x_{k-1}(v) dv \in \mathcal{H}_k$$



Pooling operator A_k

$$x_k(u) = A_k M_k P_k x_{k-1}(u) = \int_{\mathbb{R}^d} h_{\sigma_k}(u - v) M_k P_k x_{k-1}(v) dv \in \mathcal{H}_k$$

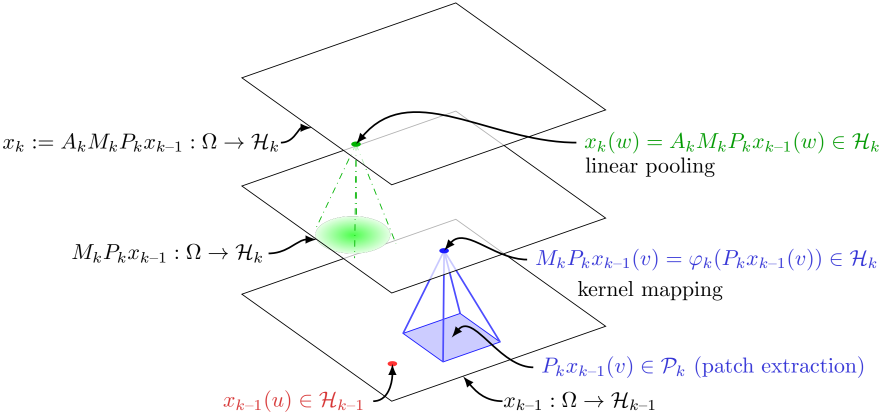
- h_{σ_k}: pooling filter at scale σ_k
- $h_{\sigma_k}(u) := \sigma_k^{-d} h(u/\sigma_k)$ with $h(u)$ **Gaussian**
- **linear, non-expansive operator**: $\|A_k\| \leq 1$

Pooling operator A_k

$$x_k(u) = A_k M_k P_k x_{k-1}(u) = \int_{\mathbb{R}^d} h_{\sigma_k}(u - v) M_k P_k x_{k-1}(v) dv \in \mathcal{H}_k$$

- h_{σ_k}: pooling filter at scale σ_k
- $h_{\sigma_k}(u) := \sigma_k^{-d} h(u/\sigma_k)$ with $h(u)$ **Gaussian**
- **linear, non-expansive operator**: $\|A_k\| \leq 1$
- In practice: **discretization**, sampling at resolution σ_k after pooling
- "Preserves information" when **subsampling** $\leq$ **patch size**

Recap: P_k, M_k, A_k



Multilayer construction

Assumption on x_0

- x_0 is typically a **discrete** signal aquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with x the original continuous signal, A_0 local integrator with scale σ_0 (**anti-aliasing**).

Multilayer construction

Assumption on x_0

- x_0 is typically a **discrete** signal aquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with x the original continuous signal, A_0 local integrator with scale σ_0 (**anti-aliasing**).

Multilayer representation

$$\Phi(x_0) = A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x_0 \in L^2(\Omega, \mathcal{H}_n).$$

- S_k, σ_k grow exponentially in practice (i.e., fixed with subsampling).

Multilayer construction

Assumption on x_0

- x_0 is typically a **discrete** signal aquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with x the original continuous signal, A_0 local integrator with scale σ_0 (**anti-aliasing**).

Multilayer representation

$$\Phi(x_0) = A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x_0 \in L^2(\Omega, \mathcal{H}_n).$$

- S_k, σ_k grow exponentially in practice (i.e., fixed with subsampling).

Final kernel

$$K_{CKN}(x, x') = \langle \Phi(x), \Phi(x') \rangle_{L^2(\Omega)} = \int_\Omega \langle x_n(u), x'_n(u) \rangle du$$

Stability to deformations

Theorem (Stability of CKN (B. and Mairal, 2019))

Let $\Phi_n(x) = \Phi(A_0 x)$ *and assume* $\|\nabla \tau\|_\infty \leq 1/2$*,*

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(C_\beta \, (n+1) \, \|\nabla \tau\|_\infty + \frac{C}{\sigma_n} \|\tau\|_\infty \right) \|x\|$$

- Translation invariance: large σ_n
- Stability: small patch sizes ($\beta \approx$ patch size, $C_\beta = O(\beta^3)$ for images)
- Signal preservation: subsampling factor $\approx$ patch size

$\Longrightarrow$ **need several layers with small patches** $\quad n = O(\log(\sigma_n/\sigma_0)/\log \beta)$

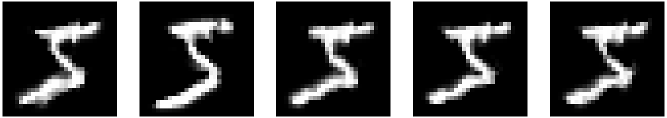
Stability to deformations for convolutional NTK

Theorem (Stability of NTK (Bietti and Mairal, 2019))

Let $\Phi_n(x) = \Phi^{NTK}(A_0 x)$, *and assume* $\|\nabla\tau\|_\infty \leq 1/2$

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(C_\beta n^{7/4} \|\nabla\tau\|_\infty^{1/2} + C'_\beta n^2 \|\nabla\tau\|_\infty + \sqrt{n+1}\frac{C}{\sigma_n}\|\tau\|_\infty \right) \|x\|,$$

Comparison with random feature CKN on deformed MNIST digits:



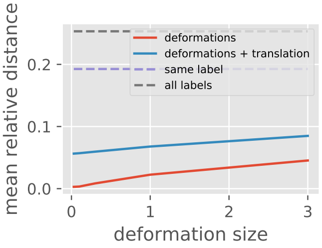
Stability to deformations for convolutional NTK

Theorem (Stability of NTK (Bietti and Mairal, 2019))

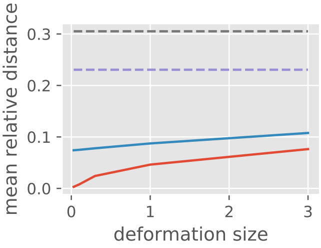
Let $\Phi_n(x) = \Phi^{NTK}(A_0 x)$, *and assume* $\|\nabla \tau\|_\infty \leq 1/2$

$$\begin{aligned} &\|\Phi_n(L_\tau x) - \Phi_n(x)\| \\ &\leq \left(C_\beta n^{7/4} \|\nabla \tau\|_\infty^{1/2} + C'_\beta n^2 \|\nabla \tau\|_\infty + \sqrt{n+1} \frac{C}{\sigma_n} \|\tau\|_\infty \right) \|x\|, \end{aligned}$$

Comparison with random feature CKN on deformed MNIST digits:



(a) CKN



(b) NTK

Experiments with convolutional kernels on Cifar10

Convolutional kernels with 3x3 patches + kernel ridge regression (danger: lots of compute!)

Conv. layers	subsampling	kernel	test acc.
2	2-5	ReLU RF	86.63%
2	2-5	ReLU NTK	87.19%

Experiments with convolutional kernels on Cifar10

Convolutional kernels with 3x3 patches + kernel ridge regression (danger: lots of compute!)

Conv. layers	subsampling	kernel	test acc.
2	2-5	ReLU RF	86.63%
2	2-5	ReLU NTK	87.19%
2	2-5	exp, $\sigma = 0.6$	87.93%
3	2-2-2	exp, $\sigma = 0.6$	**88.2**%

Experiments with convolutional kernels on Cifar10

Convolutional kernels with 3x3 patches + kernel ridge regression (danger: lots of compute!)

Conv. layers	subsampling	kernel	test acc.
2	2-5	ReLU RF	86.63%
2	2-5	ReLU NTK	87.19%
2	2-5	exp, $\sigma = 0.6$	87.93%
3	2-2-2	exp, $\sigma = 0.6$	**88.2**%
16 (Li et al., 2019)	last layer only	ReLU RF	87.28%
16 (Li et al., 2019)	last layer only	ReLU NTK	86.77%
10	every 3 layers	exp	**88.2**%

Li et al. (2019): no pooling before last layer, more complicated pre-processing
Shankar et al. (2020): similar performance to us (88.2%), reaches 90% when adding flips

Approximation with convolutional networks

- **What functions does the RKHS contain? What is their norm?**
- Role of **convolution** vs **fully-connected**?
- Role of **depth**?

Approximation with convolutional networks

- **What functions does the RKHS contain? What is their norm?**
- Role of **convolution** vs **fully-connected**?
- Role of **depth**?
- Limitations of kernels?

Prelude: "teacher" CNNs with smooth activations are in the RKHS

- Consider a CNN with filters $W_k^{ij}(u), u \in S_k$
- **Smooth** activations σ with smoothness controlled by some $C_{\kappa,\sigma}(\cdot)$
- The CNN can be **constructed hierarchically** in $\mathcal{H}_{CKN}$
- Complexity is controlled by the RKHS norm:

$$\|f_\sigma\|_{\mathcal{H}}^2 \leq \|W_{n+1}\|_2^2 \; C_{\kappa,\sigma}^2(\|W_n\|_2^2 \; C_{\kappa,\sigma}^2(\|W_{n-1}\|_2^2 \; C_{\kappa,\sigma}^2(\dots)))$$

(B. and Mairal, 2019)

Prelude: "teacher" CNNs with smooth activations are in the RKHS

- Consider a CNN with filters $W_k^{ij}(u), u \in S_k$
- **Smooth** activations σ with smoothness controlled by some $C_{\kappa,\sigma}(\cdot)$
- The CNN can be **constructed hierarchically** in $\mathcal{H}_{CKN}$
- Complexity is controlled by the RKHS norm (linear layers):

$$\|f_\sigma\|_{\mathcal{H}}^2 \leq \|W_{n+1}\|_2^2 \cdot \|W_n\|_2^2 \cdot \|W_{n-1}\|_2^2 \dots \|W_1\|_2^2$$

- Linear layers: product of spectral norms

(B. and Mairal, 2019)

Prelude: "teacher" CNNs with smooth activations are in the RKHS

- Consider a CNN with filters $W_k^{ij}(u), u \in S_k$
- **Smooth** activations σ with smoothness controlled by some $C_{\kappa,\sigma}(\cdot)$
- The CNN can be **constructed hierarchically** in $\mathcal{H}_{CKN}$
- Complexity is controlled by the RKHS norm (linear layers):

$$\|f_\sigma\|_{\mathcal{H}}^2 \leq \|W_{n+1}\|_2^2 \cdot \|W_n\|_2^2 \cdot \|W_{n-1}\|_2^2 \dots \|W_1\|_2^2$$

- Linear layers: product of spectral norms
- **Can we give a more precise characterization of the RKHS?**

(B. and Mairal, 2019)

The fully-connected case

Fully-connected models $\implies$ dot-product kernels

$$K(x, y) = \kappa(x^\top y) \text{ for } x, y \in \mathbb{S}^{d-1}$$

- Infinitely wide random networks (Neal, 1996; Cho and Saul, 2009; Lee et al., 2018)
- NTK for infinitely wide networks (Jacot et al., 2018)

The fully-connected case

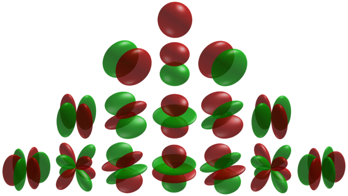
Fully-connected models $\implies$ dot-product kernels

$$K(x, y) = \kappa(x^\top y) \text{ for } x, y \in \mathbb{S}^{d-1}$$

- Infinitely wide random networks (Neal, 1996; Cho and Saul, 2009; Lee et al., 2018)
- NTK for infinitely wide networks (Jacot et al., 2018)

Precise description of the RKHS (Mercer decomposition)

- Rotation-invariant kernel on the sphere
- $\implies$ RKHS description in the $L^2(\mathbb{S}^{d-1})$ basis of **spherical harmonics** $Y_{k,j}$



The fully-connected case

Fully-connected models $\implies$ dot-product kernels

$$K(x, y) = \kappa(x^\top y) \text{ for } x, y \in \mathbb{S}^{d-1}$$

- Infinitely wide random networks (Neal, 1996; Cho and Saul, 2009; Lee et al., 2018)
- NTK for infinitely wide networks (Jacot et al., 2018)

Precise description of the RKHS (Mercer decomposition)

- Rotation-invariant kernel on the sphere
- $\implies$ RKHS description in the $L^2(\mathbb{S}^{d-1})$ basis of **spherical harmonics** $Y_{k,j}$

$$\kappa(x^\top y) = \sum_{k=0}^{\infty} \mu_k \sum_{j=1}^{N(d,k)} Y_{k,j}(x) Y_{k,j}(y), \quad \text{for } x, y \in \mathbb{S}^{d-1}$$

The fully-connected case

Fully-connected models $\implies$ dot-product kernels

$$K(x, y) = \kappa(x^\top y) \text{ for } x, y \in \mathbb{S}^{d-1}$$

- Infinitely wide random networks (Neal, 1996; Cho and Saul, 2009; Lee et al., 2018)
- NTK for infinitely wide networks (Jacot et al., 2018)

Precise description of the RKHS (Mercer decomposition)

- Rotation-invariant kernel on the sphere
- $\implies$ RKHS description in the $L^2(\mathbb{S}^{d-1})$ basis of **spherical harmonics** $Y_{k,j}$

$$\mathcal{H} = \left\{ f = \sum_{k=0}^{\infty} \sum_{j=1}^{N(d,k)} a_{k,j} Y_{k,j}(\cdot) \ \text{ s.t. } \ \|f\|_{\mathcal{H}}^2 := \sum_{k,j} \frac{a_{k,j}^2}{\mu_k} < \infty \right\}$$

Approximation for two-layer ReLU networks

Approximation of functions on the sphere (Bach, 2017)

- Decay of $\mu_k \leftrightarrow$ regularity of functions in the RKHS
- Polynomial decays $\mu_k \approx k^{-2\beta}$: similar to Sobolev space of order β, norm:

$$\|f\|_{\mathcal{H}} \approx \|\Delta_{\mathbb{S}^{d-1}}^{\beta/2} f\|_{L^2(\mathbb{S}^{d-1})}$$

- Leads to sufficient conditions for RKHS membership
- Rates of approximation for Lipschitz functions

Approximation for two-layer ReLU networks

Approximation of functions on the sphere (Bach, 2017)

- Decay of $\mu_k \leftrightarrow$ regularity of functions in the RKHS
- Polynomial decays $\mu_k \approx k^{-2\beta}$: similar to Sobolev space of order β, norm:

$$\|f\|_{\mathcal{H}} \approx \|\Delta_{\mathbb{S}^{d-1}}^{\beta/2} f\|_{L^2(\mathbb{S}^{d-1})}$$

- Leads to sufficient conditions for RKHS membership
- Rates of approximation for Lipschitz functions

NTK vs random features (Bietti and Mairal, 2019)

- f has $\beta = p/2$ η-bounded derivatives $\implies f \in \mathcal{H}_{NTK}$, $\|f\|_{\mathcal{H}_{NTK}} \leq O(\eta)$
- $\beta = p/2 + 1$ needed for RF (Bach, 2017)
- $\implies \mathcal{H}_{NTK}$ is (slightly) "**larger**" than $\mathcal{H}_{RF}$
- Similar improvement for approximation of Lipschitz functions

Deep fully-connected ReLU networks: limitations

$$\kappa_L(x^\top y) = \underbrace{\kappa \circ \cdots \circ \kappa}_{L \text{ times}}(x^\top y)$$

Deep = Shallow (B. and Bach, 2021)

- RF or NTK kernels for deep and shallow networks have the same decay! (thus same $\mathcal{H}$)
- Proof using differentiability of κ: we have $\mu_k \sim k^{d-2\nu+1}$ when

$$\kappa(1-t) = poly(t) + c_1 t^{\nu} + o(t^{\nu})$$
$$\kappa(-1+t) = poly(t) + c_{-1} t^{\nu} + o(t^{\nu}).$$

- Such expansions are preserved when taking composition with ReLU/arc-cosine kernel

Deep fully-connected ReLU networks: limitations

$$\kappa_L(x^\top y) = \underbrace{\kappa \circ \cdots \circ \kappa}_{L \text{ times}}(x^\top y)$$

Deep = Shallow (B. and Bach, 2021)

- RF or NTK kernels for deep and shallow networks have the same decay! (thus same $\mathcal{H}$)
- Proof using differentiability of κ: we have $\mu_k \sim k^{d-2\nu+1}$ when

$$\kappa(1-t) = poly(t) + c_1 t^{\nu} + o(t^{\nu})$$
$$\kappa(-1+t) = poly(t) + c_{-1} t^{\nu} + o(t^{\nu}).$$

- Such expansions are preserved when taking composition with ReLU/arc-cosine kernel

Consequences

$\implies$ kernel regime cannot explain power of depth in fully-connected nets

$\implies$ power of deep kernels comes from **architecture**

Deep = shallow: numerical experiments

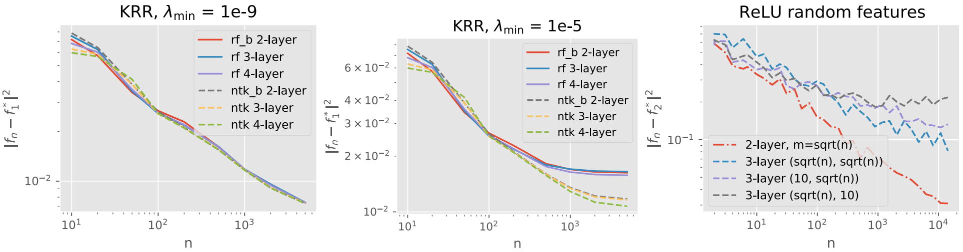


Figure 1: (left, middle) expected squared error vs sample size n for kernel ridge regression estimators with different kernels on f_1^* and with two different budgets on optimization difficulty $\lambda_{\min}$ (the minimum regularization parameter allowed). (right) ridge regression with one or two layers of random ReLU features on f_2^*, with different scalings of the number of "neurons" at each layer in terms of n.

Deep = shallow: numerical experiments

MNIST

L	RF	NTK
2	98.60 ± 0.03	98.49 ± 0.02
3	98.67 ± 0.03	98.53 ± 0.02
4	98.66 ± 0.02	98.49 ± 0.01
5	98.65 ± 0.04	98.46 ± 0.02

F-MNIST

L	RF	NTK
2	90.75 ± 0.11	90.65 ± 0.07
3	90.87 ± 0.16	90.62 ± 0.08
4	90.89 ± 0.13	90.55 ± 0.07
5	90.88 ± 0.08	90.50 ± 0.05

(on 50k samples)

Approximating functions on signals: motivation

Curse of dimensionality

- Natural signals are very high-dimensional ($d \approx |\Omega|$, where Ω is the domain)
- Approximating general f^* requires exponentially large norm or very high smoothness

Approximating functions on signals: motivation

Curse of dimensionality

- Natural signals are very high-dimensional ($d \approx |\Omega|$, where Ω is the domain)
- Approximating general f^* requires exponentially large norm or very high smoothness

Adding structure: localized functions *e.g.*, $f^*(x) = g^*(Px[u_0])$

- With fully-connected kernel, still need norm exp. large in d
- For basic convolutional kernel, norm only scales with the dimension of the patch $Px[u_0]$:

$$K(x, x') = \langle MPx, MPx' \rangle = \sum_{u \in \Omega} k(Px[u], Px'[u])$$

- See also Ciliberto et al. (2019) for similar part-based kernels for structured prediction

Warmup: one layer with pooling

$$K(x, x') = \langle AMPx, AMPx' \rangle_{L^2(\Omega, \mathcal{H})}$$

($\mathcal{H}$: RKHS of patch kernels)

- RKHS consists of functions of the form (patches denoted $x_u = Px[u] \in \mathbb{R}^p$)

$$f(x) = \sum_{u \in \Omega} G[u](x_u), \qquad G[u] \in \mathcal{H}$$

Warmup: one layer with pooling

$$K(x, x') = \langle AMPx, AMPx' \rangle_{L^2(\Omega, \mathcal{H})}$$

($\mathcal{H}$: RKHS of patch kernels)

- RKHS consists of functions of the form (patches denoted $x_u = Px[u] \in \mathbb{R}^p$)

$$f(x) = \sum_{u \in \Omega} G[u](x_u), \qquad G[u] \in \mathcal{H}$$

- Squared RKHS norm given by the minimum over such decompositions of

$$\|A^{-\top} G\|^2_{L^2(\Omega, \mathcal{H})} = \|(A^{-\top} \otimes \Gamma) G\|^2_{L^2(\Omega) \otimes L^2(\mathbb{S}^{p-1})}$$

  - G viewed in $L^2(\Omega) \otimes L^2(\mathbb{S}^{p-1})$ as $(u, z) \mapsto G[u](z)$
  - $\Gamma = T^{-1/2}$ regularization operator of $\mathcal{H}$, *e.g.*, $\Gamma = \Delta^{\beta/2}_{\mathbb{S}^{p-1}}$
- $\implies$ A (pooling) encourages smoothness of $u \mapsto G[u](z)$
- $\implies$ Γ (kernel) encourages smoothness of $z \mapsto G[u](z)$

Beyond one layer: empirical study

Cifar10 with full kernel (or Nyström in parentheses)

κ_1	κ_2	Test acc. (10k)	Test acc. (full)
Exp	Exp	80.5%	87.9% (84.1%)
Exp	Poly3	80.5%	87.7% (84.1%)
Exp	Poly2	79.4%	86.9% (83.4%)
Poly2	Exp	77.4%	- (81.5%)
Poly2	Poly2	75.1%	- (81.2%)
Exp	- (Lin)	74.2%	- (76.3%)

One layer is not enough

Polynomial kernel can be enough for second layer

Interlude: kernel tensor products

κ_2 polynomial $\implies$ products of patch kernels

$$K((x_1, x_2), (x_1', x_2')) = k(x_1, x_1')k(x_2, x_2') = \langle \varphi(x_1) \otimes \varphi(x_2), \varphi(x_1') \otimes \varphi(x_2') \rangle_{\mathcal{H} \otimes \mathcal{H}}$$

- RKHS $\mathcal{H} \otimes \mathcal{H}$ contains closure of functions $f(x_1, x_2) = \sum_{j=1}^m f_{1,j}(x_1) f_{2,j}(x_2)$

Interlude: kernel tensor products

κ_2 **polynomial** $\implies$ **products of patch kernels**

$$K((x_1, x_2), (x_1', x_2')) = k(x_1, x_1')k(x_2, x_2') = \langle \varphi(x_1) \otimes \varphi(x_2), \varphi(x_1') \otimes \varphi(x_2') \rangle_{\mathcal{H}\otimes\mathcal{H}}$$

- RKHS $\mathcal{H} \otimes \mathcal{H}$ contains closure of functions $f(x_1, x_2) = \sum_{j=1}^m f_{1,j}(x_1) f_{2,j}(x_2)$
- RKHS is often **much smaller** than a dot-product kernel on $x = (x_1, x_2)$
- Helpful for modeling **interactions** between variables/patches (Wahba, 1990; Lin, 2000; Scetbon and Harchaoui, 2020)

Interlude: kernel tensor products

κ_2 polynomial $\implies$ products of patch kernels

$$K((x_1, x_2), (x_1', x_2')) = k(x_1, x_1')k(x_2, x_2') = \langle \varphi(x_1) \otimes \varphi(x_2), \varphi(x_1') \otimes \varphi(x_2') \rangle_{\mathcal{H} \otimes \mathcal{H}}$$

- RKHS $\mathcal{H} \otimes \mathcal{H}$ contains closure of functions $f(x_1, x_2) = \sum_{j=1}^m f_{1,j}(x_1) f_{2,j}(x_2)$
- RKHS is often **much smaller** than a dot-product kernel on $x = (x_1, x_2)$
- Helpful for modeling **interactions** between variables/patches (Wahba, 1990; Lin, 2000; Scetbon and Harchaoui, 2020)
- Here, the **architecture** determines which interactions matter, and **pooling** will further encourage **spatial regularities** among interaction terms

RKHS of two-layer CKN with quadratic second layer

Kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$, with

$$\Phi(x) = A_2 M_2 P_2 A_1 M_1 P_1 x \in L^2 \left(\Omega, (\mathcal{H} \otimes \mathcal{H})^{|S_2| \times |S_2|} \right)$$

RKHS of two-layer CKN with quadratic second layer

Kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$, with

$$\Phi(x) = A_2 M_2 P_2 A_1 M_1 P_1 x \in L^2 \left(\Omega, (\mathcal{H} \otimes \mathcal{H})^{|S_2| \times |S_2|} \right)$$

RKHS functions of the form

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} G_{pq}[u, v](x_u, x_v)$$

RKHS of two-layer CKN with quadratic second layer

Kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$, with

$$\Phi(x) = A_2 M_2 P_2 A_1 M_1 P_1 x \in L^2 \left(\Omega, (\mathcal{H} \otimes \mathcal{H})^{|S_2| \times |S_2|} \right)$$

RKHS functions of the form

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} G_{pq}[u, v](x_u, x_v)$$

Under **localization** constraint: $G_{pq} \in \mathrm{Range}((L_p A_1 \otimes L_q A_1)^\top \operatorname{diag}(\cdot))$

Figure 2. Display of the response of the operator E_{pq} to Dirac inputs $x = \delta_u$ centered at two different locations u. These are bumps centered on points of the $p - q$ diagonal, corresponding to interactions between two patches, at distance around $p - q$.

RKHS of two-layer CKN with quadratic second layer

Kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$, with

$$\Phi(x) = A_2 M_2 P_2 A_1 M_1 P_1 x \in L^2 \left(\Omega, (\mathcal{H} \otimes \mathcal{H})^{|S_2| \times |S_2|} \right)$$

RKHS functions of the form

$$f(x) = \sum_{p,q \in S_2} \sum_{u,v \in \Omega} G_{pq}[u, v](x_u, x_v)$$

Under **localization** constraint: $G_{pq} \in \mathrm{Range}((L_p A_1 \otimes L_q A_1)^\top \operatorname{diag}(\cdot))$
RKHS norm given by the penalty

$$\sum_{p,q \in S_2} \| A_2^{-\top} \operatorname{diag}((L_p A_1 \otimes L_q A_1)^{-\top} G_{pq}) \|^2_{L^2(\Omega, \mathcal{H} \otimes \mathcal{H})}.$$

- $(L_p A_1 \otimes L_q A_1)^{-\top} G$ encourages **2D smoothness** of $(u, v) \mapsto G[u, v](z, z')$
- $A_2^{-\top}$ imposes even stronger **1D smoothness** on diagonal $u - v = p - q$

Extensions

- **Higher-order** polynomials $\implies$ higher-order interactions
- **More layers**: also capture higher-order interactions, with different structure

Extensions

- **Higher-order** polynomials $\implies$ higher-order interactions
- **More layers**: also capture higher-order interactions, with different structure
- Empirically, on Cifar10, 2 layers with degree-4 kernels at 2nd layer suffice for best performance

Conclusions

Benefits of convolutional kernels

- Translation invariance + deformation stability with small patches and pooling
- $\implies$ benefits of depth for stability
- Approximation benefits of ≥ 2 layers by efficiently capturing interactions
- Limitations of depth for fully-connected models in kernel regimes

Conclusions

Benefits of convolutional kernels

- Translation invariance + deformation stability with small patches and pooling
- $\implies$ benefits of depth for stability
- Approximation benefits of ≥ 2 layers by efficiently capturing interactions
- Limitations of depth for fully-connected models in kernel regimes

Future directions

- Empirically, any benefits of depth beyond 2 layers?
- Statistical analysis through covariance operator

Conclusions

Benefits of convolutional kernels

- Translation invariance + deformation stability with small patches and pooling
- $\implies$ benefits of depth for stability
- Approximation benefits of ≥ 2 layers by efficiently capturing interactions
- Limitations of depth for fully-connected models in kernel regimes

Future directions

- Empirically, any benefits of depth beyond 2 layers?
- Statistical analysis through covariance operator

Perspectives: beyond kernels

- Kernels provide a nice tractable model, but a limited picture of deep learning
- Feature selection through mean-field/"active" regime, at least at first layer
- Benefits of depth beyond simple interaction models, *e.g.*, through hierarchy

Convolutional NTK kernel mapping

Define

$$M(x, y)(u) = \begin{pmatrix} \varphi_0(x(u)) \otimes y(u) \\ \varphi_1(x(u)) \end{pmatrix}$$

Theorem (NTK feature map for CNN)

$$K_{NTK}(x, x') = \langle \Phi(x), \Phi(x') \rangle_{L^2(\Omega)},$$

with $\Phi(x)(u) = A_n M(x_n, y_n)(u)$, *where* $y_1(u) = x_1(u) = P_1 x(u)$ *and*

$$\begin{aligned} x_k(u) &= P_k A_{k-1} \varphi_1(x_{k-1})(u) \\ y_k(u) &= P_k A_{k-1} M(x_{k-1}, y_{k-1})(u). \end{aligned}$$

Discretization and signal preservation

- $\bar{x}_k$: subsampling factor s_k after pooling with scale $\sigma_k \approx s_k$:

$$\bar{x}_k[n] = A_k M_k P_k \bar{x}_{k-1}[n s_k]$$

Discretization and signal preservation

- $\bar{x}_k$: subsampling factor s_k after pooling with scale $\sigma_k \approx s_k$:

$$\bar{x}_k[n] = A_k M_k P_k \bar{x}_{k-1}[n s_k]$$

- **Claim**: We can recover $\bar{x}_{k-1}$ from $\bar{x}_k$ if **subsampling** $s_k \leq$ **patch size**

Discretization and signal preservation

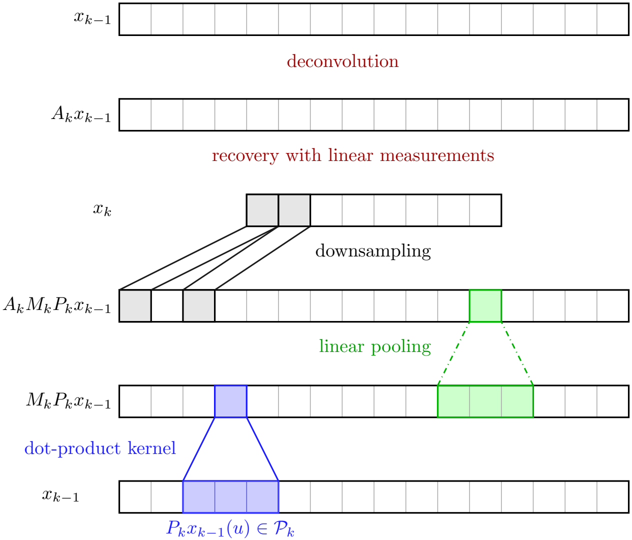
- $\bar{x}_k$: subsampling factor s_k after pooling with scale $\sigma_k \approx s_k$:

$$\bar{x}_k[n] = A_k M_k P_k \bar{x}_{k-1}[n s_k]$$

- **Claim**: We can recover $\bar{x}_{k-1}$ from $\bar{x}_k$ if **subsampling** $s_k \leq$ **patch size**
- **How**? Kernels! Recover patches with **linear functions** (contained in RKHS)

$$\langle f_w, M_k P_k x(u) \rangle = f_w(P_k x(u)) = \langle w, P_k x(u) \rangle$$

Signal recovery: example in 1D



Beyond the translation group

Global invariance to other groups?

- Rotations, reflections, roto-translations, ...
- Group action $L_g x(u) = x(g^{-1}u)$
- **Equivariance** in inner layers + **(global) pooling** in last layer
- Similar construction to Cohen and Welling (2016); Kondor and Trivedi (2018)

G-equivariant layer construction

- Feature maps $x(u)$ defined on $u \in G$ (G: locally compact group)
  - Input needs special definition when $G \neq \Omega$
- **Patch extraction**:

$$Px(u) = (x(uv))_{v \in S}$$

- **Non-linear mapping**: equivariant because pointwise!
- **Pooling** (μ: left-invariant Haar measure):

$$Ax(u) = \int_G x(uv)h(v)d\mu(v) = \int_G x(v)h(u^{-1}v)d\mu(v)$$

Group invariance and stability

Roto-translation group $G = \mathbb{R}^2 \rtimes SO(2)$ (translations + rotations)

- **Stability** w.r.t. translation group
- **Global invariance** to rotations (only global pooling at final layer)
  - Inner layers: patches and pooling only on translation group
  - Last layer: global pooling on rotations
  - Cohen and Welling (2016): pooling on rotations in inner layers hurts performance on Rotated MNIST