Foundations of Deep Convolutional Models through Kernel Methods

Alberto Bietti

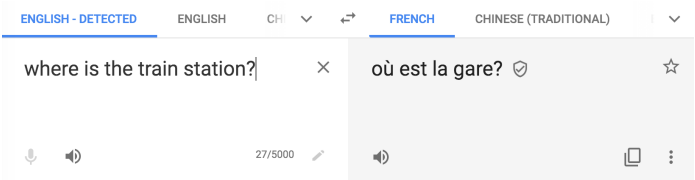
Inria

Microsoft Research, New York. January 29, 2020.

Microsoft Research - Inria
JOINT CENTRE

Success of deep learning

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



Success of deep learning

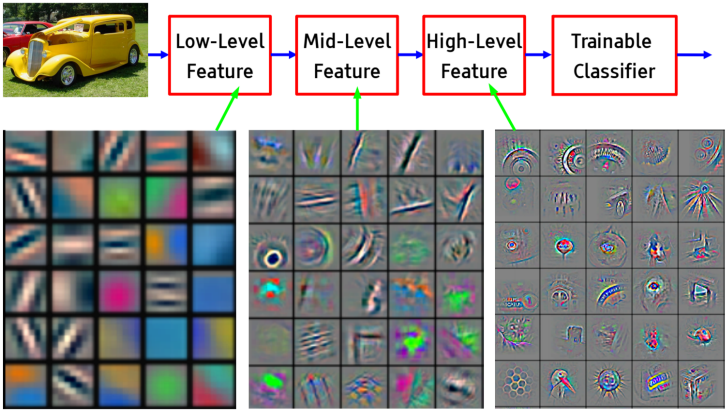
State-of-the-art models in various domains (images, speech, text, ...)

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

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

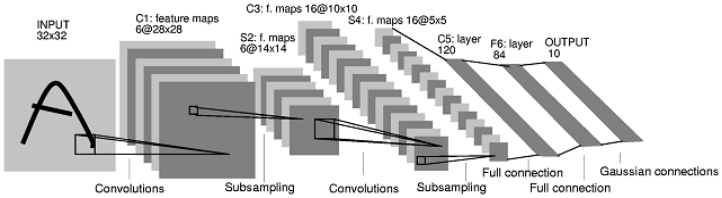
Convolutional networks

Exploiting structure of natural images (LeCun et al., 1989)



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

Convolutional networks

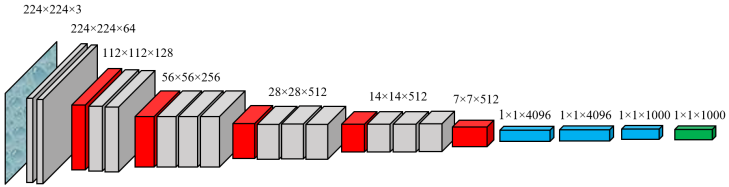


(LeCun et al., 1998)

Convolutional networks

- Model local neighborhoods at different scales
- Provide some invariance through pooling
- Useful **inductive bias** for learning efficiently on natural images

Convolutional networks



(Simonyan and Zisserman, 2014)

Convolutional networks

- Model local neighborhoods at different scales
- Provide some invariance through pooling
- Useful **inductive bias** for learning efficiently on natural images

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

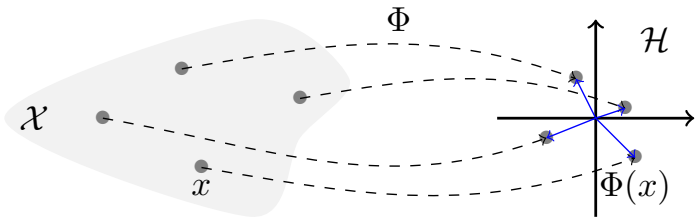
- **Over-parameterized** (millions of parameters)
- **Expressive** (can approximate any function)
- Complex **architectures** for exploiting problem structure
- Yet, **easy to optimize** with (stochastic) gradient descent!

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?

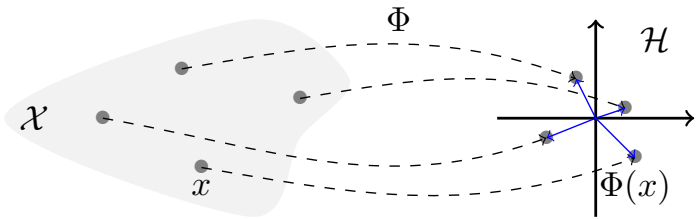
Kernels to the rescue



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

Kernels for deep models: deep kernel machines

Hierarchical kernels (Cho and Saul, 2009)

- Kernels can be constructed **hierarchically**

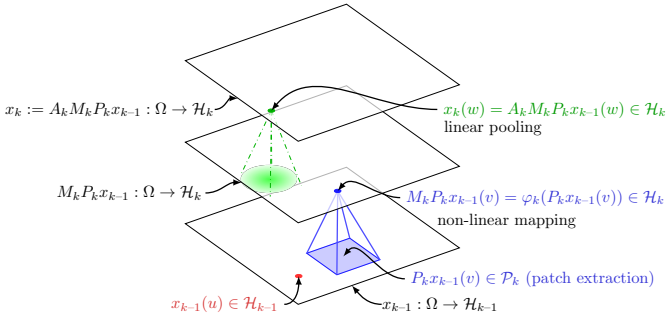
$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle \text{ with } \Phi(x) = \varphi_2(\varphi_1(x))$$

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

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

Kernels for deep models: deep kernel machines

Convolutional kernels networks (CKNs) for images (Mairal et al., 2014; Mairal, 2016)



- Good empirical performance with tractable approximations (Nyström)

Kernels for deep models: infinite-width networks

$$f_\theta(x) = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} v_i \sigma(w_i^\top x), \qquad m \to \infty$$

Random feature kernels (RF, Neal, 1996; Rahimi and Recht, 2007)

- $\theta = (v_i)_i$, fixed random weights $w_i \sim N(0, I)$

$$K_{RF}(x, x') = \mathbb{E}_{w \sim N(0,I)}[\sigma(w^\top x)\sigma(w^\top x')]$$

Kernels for deep models: infinite-width networks

$$f_\theta(x) = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} v_i \sigma(w_i^\top x), \qquad m \to \infty$$

Random feature kernels (RF, Neal, 1996; Rahimi and Recht, 2007)

- $\theta = (v_i)_i$, fixed random weights $w_i \sim N(0, I)$

$$K_{RF}(x, x') = \mathbb{E}_{w \sim N(0,I)}[\sigma(w^\top x)\sigma(w^\top x')]$$

Neural tangent kernels (NTK, Jacot et al., 2018)

- $\theta = (v_i, w_i)_i$, initialization $\theta_0 \sim N(0, I)$
- **Lazy training** (Chizat et al., 2019): θ stays close to θ_0 when training with large m

$$f_\theta(x) \approx f_{\theta_0}(x) + \langle \theta - \theta_0, \nabla_\theta f_\theta(x)|_{\theta=\theta_0} \rangle.$$

Kernels for deep models: infinite-width networks

$$f_\theta(x) = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} v_i \sigma(w_i^\top x), \qquad m \to \infty$$

Random feature kernels (RF, Neal, 1996; Rahimi and Recht, 2007)

- $\theta = (v_i)_i$, fixed random weights $w_i \sim N(0, I)$

$$K_{RF}(x, x') = \mathbb{E}_{w \sim N(0,I)}[\sigma(w^\top x)\sigma(w^\top x')]$$

Neural tangent kernels (NTK, Jacot et al., 2018)

- $\theta = (v_i, w_i)_i$, initialization $\theta_0 \sim N(0, I)$
- **Lazy training** (Chizat et al., 2019): θ stays close to θ_0 when training with large m

$$f_\theta(x) \approx f_{\theta_0}(x) + \langle \theta - \theta_0, \nabla_\theta f_\theta(x)|_{\theta=\theta_0} \rangle.$$

- Gradient descent for $m \to \infty \approx$ kernel ridge regression with **neural tangent kernel**

$$K_{NTK}(x, x') = \lim_{m \to \infty} \langle \nabla_\theta f_{\theta_0}(x), \nabla_\theta f_{\theta_0}(x') \rangle$$

Kernels for deep models: infinite-width networks

$$f_\theta(x) = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} v_i \sigma(w_i^\top x), \qquad m \to \infty$$

Random feature kernels (RF, Neal, 1996; Rahimi and Recht, 2007)

- $\theta = (v_i)_i$, fixed random weights $w_i \sim N(0, I)$

$$K_{RF}(x, x') = \mathbb{E}_{w \sim N(0,I)}[\sigma(w^\top x)\sigma(w^\top x')]$$

Neural tangent kernels (NTK, Jacot et al., 2018)

- $\theta = (v_i, w_i)_i$, initialization $\theta_0 \sim N(0, I)$
- **Lazy training** (Chizat et al., 2019): θ stays close to θ_0 when training with large m

$$f_\theta(x) \approx f_{\theta_0}(x) + \langle \theta - \theta_0, \nabla_\theta f_\theta(x)|_{\theta=\theta_0} \rangle.$$

- Gradient descent for $m \to \infty \approx$ kernel ridge regression with **neural tangent kernel**

$$K_{NTK}(x, x') = \lim_{m \to \infty} \langle \nabla_\theta f_{\theta_0}(x), \nabla_\theta f_{\theta_0}(x') \rangle$$

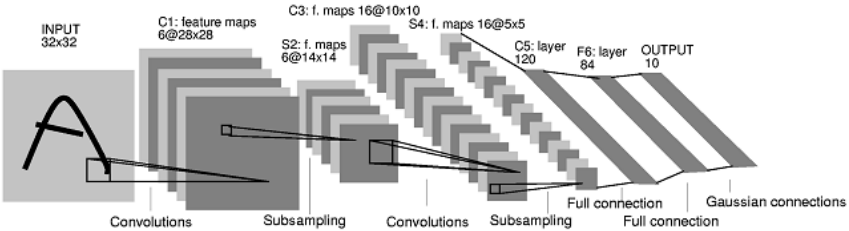
RF and NTK extend to deep architectures

Outline

1. Convolutional kernels and their stability
2. Approximation and complexity
3. Applications to regularization and robustness
4. Conclusions and perspectives

Outline

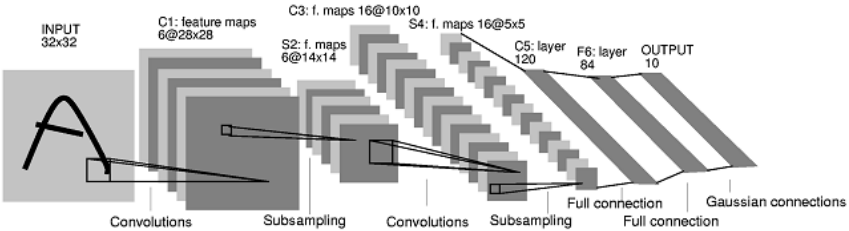
Properties of convolutional models



Convolutional architectures:

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

Properties of convolutional models



Convolutional architectures:

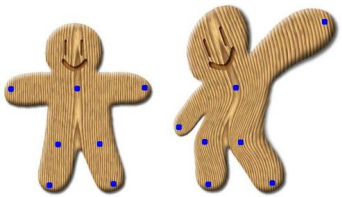
- Capture **multi-scale** and **compositional** structure in natural signals
- Model **local stationarity**
- Provide some **translation invariance**

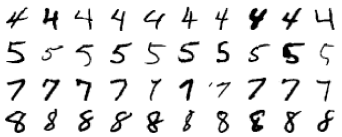
Beyond translation invariance?

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, 2019b)

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)

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

$$P_k x_{k-1}$$

  - 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 $x_1, \dots, x_n$

- $x_0 : \Omega \to \mathcal{H}_0$: 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

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

  - 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 $x_1, \dots, x_n$

- $x_0 : \Omega \to \mathcal{H}_0$: 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

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

  - 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 $x_1, \dots, x_n$

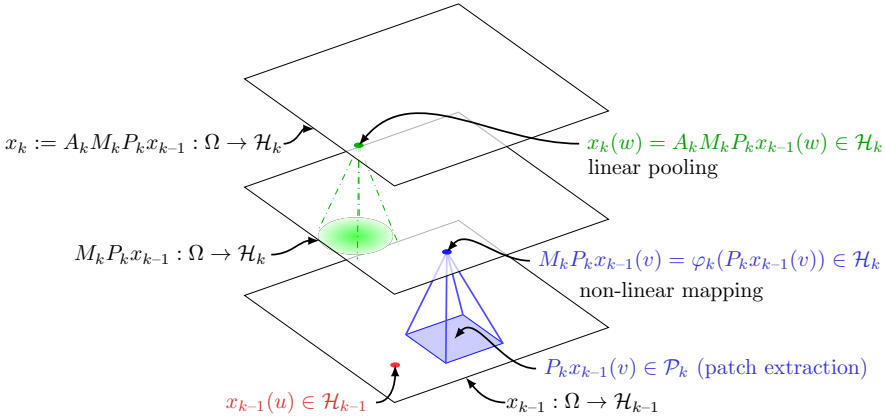
- $x_0 : \Omega \to \mathcal{H}_0$: initial (**continuous**) signal
  - $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images)
  - $x_0(u) \in \mathcal{H}_0$: value ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images)
- $x_k : \Omega \to \mathcal{H}_k$: **feature map** at layer k

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

  - 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

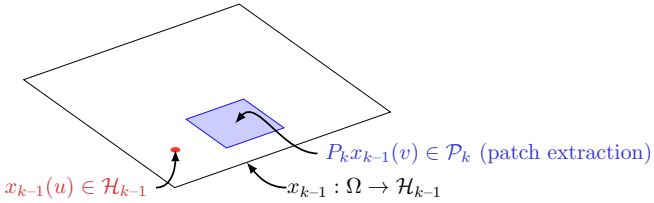
Goal: control stability of these operators through their norms

CKN construction



Patch extraction operator P_k

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



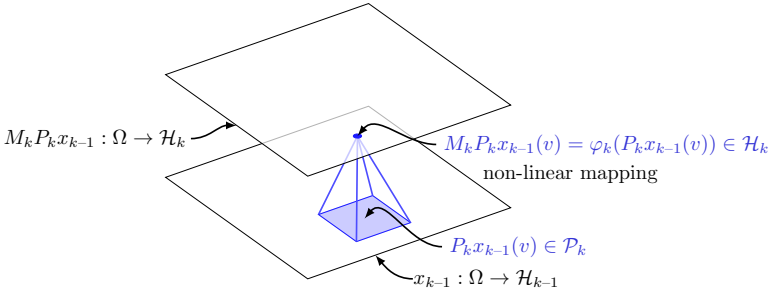
Patch extraction operator P_k

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

- 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**:

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

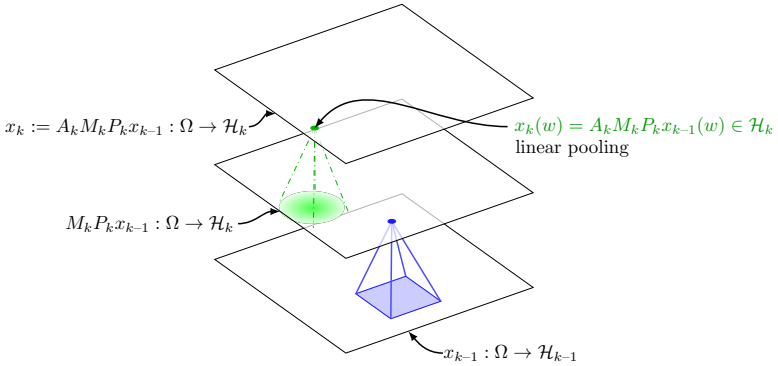
$\kappa_k(u) = \sum_{j=0}^{\infty} b_j u^j$ with $b_j \geq 0$, $\kappa_k(1) = 1$

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

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

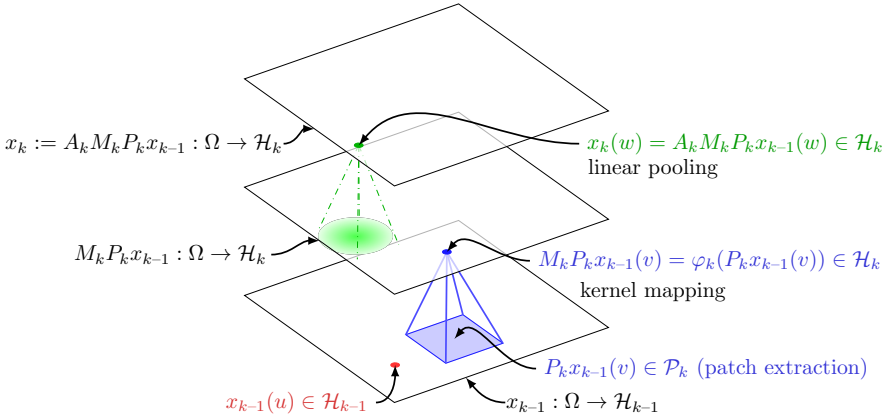
- h_{σ_k}: pooling filter at scale σ_k
- $h_{\sigma_k}(u) := \sigma_k^{-d} h(u/\sigma_k)$ with $h(u)$ **Gaussian**
- **linear, non-expansive operator**: $\|A_k\| \leq 1$

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 (Bietti and Mairal, 2019a))

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

Theorem (Stability of CKN (Bietti and Mairal, 2019a))

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** $\quad n = O(\log(\sigma_n/\sigma_0)/\log \beta)$

- Achieved by controlling norm of **commutator** $[L_\tau, P_k A_{k-1}]$
  - Extend result by Mallat (2012) for controlling $\|[L_\tau, A]\|$
  - Need patches S_k adapted to resolution σ_{k-1}: diam $S_k \leq \beta \sigma_{k-1}$

Stability to deformations

Theorem (Stability of CKN (Bietti and Mairal, 2019a))

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** $\quad n = O(\log(\sigma_n/\sigma_0)/\log \beta)$

- Achieved by controlling norm of **commutator** $[L_\tau, P_k A_{k-1}]$
  - Extend result by Mallat (2012) for controlling $\|[L_\tau, A]\|$
  - Need patches S_k adapted to resolution σ_{k-1}: diam $S_k \leq \beta \sigma_{k-1}$
- Extensions to other transformation groups, *e.g.* **roto-translations**

Stability to deformations for convolutional NTK

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

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:

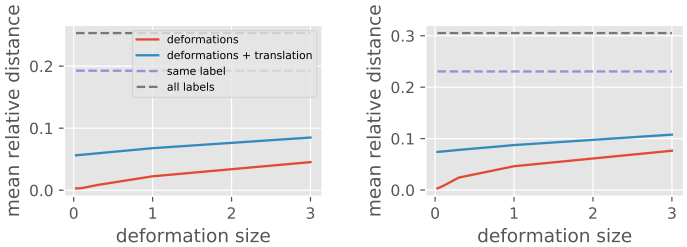
Stability to deformations for convolutional NTK

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

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:



Outline

RKHS of patch kernels K_k

$$K_k(z, z') = \|z\|\|z'\|\kappa\left(\frac{\langle z, z'\rangle}{\|z\|\|z'\|}\right), \qquad \kappa(u) = \sum_{j=0}^{\infty} b_j^2 u^j$$

- RKHS contains **homogeneous functions**:

$$f : z \mapsto \|z\|\sigma(\langle g, z\rangle / \|z\|)$$

(Bietti and Mairal, 2019a). Homogeneous version of (Zhang et al., 2016, 2017)

RKHS of patch kernels K_k

$$K_k(z, z') = \|z\|\|z'\|\kappa\left(\frac{\langle z, z'\rangle}{\|z\|\|z'\|}\right), \qquad \kappa(u) = \sum_{j=0}^{\infty} b_j^2 u^j$$

- RKHS contains **homogeneous functions**:

$$f : z \mapsto \|z\|\sigma(\langle g, z\rangle / \|z\|)$$

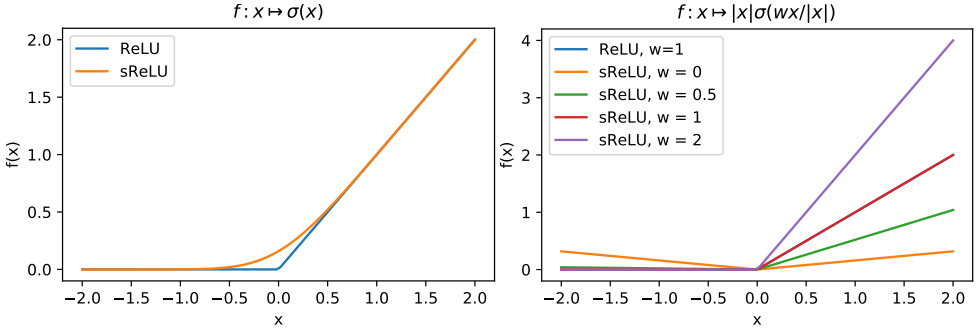
- **Smooth activations**: $\sigma(u) = \sum_{j=0}^{\infty} a_j u^j$
- Norm: $\|f\|_{\mathcal{H}_k}^2 \leq C_\sigma^2(\|g\|^2) = \sum_{j=0}^{\infty} \frac{a_j^2}{b_j^2}\|g\|^{2j} < \infty$

(Bietti and Mairal, 2019a). Homogeneous version of (Zhang et al., 2016, 2017)

RKHS of patch kernels K_k

Examples:

- $\sigma(u) = u$ (linear): $C_\sigma^2(\lambda^2) = O(\lambda^2)$
- $\sigma(u) = u^p$ (polynomial): $C_\sigma^2(\lambda^2) = O(\lambda^{2p})$
- $\sigma \approx$ sin, sigmoid, smooth ReLU: $C_\sigma^2(\lambda^2) = O(e^{c\lambda^2})$



Constructing a CNN in the RKHS $\mathcal{H}_{CKN}$

- Consider a CNN with filters $W_k^{ij}(u), u \in S_k$
- "Smooth homogeneous" activations σ
- The CNN can be **constructed hierarchically** in $\mathcal{H}_{CKN}$
- Norm upper bound:

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

(Bietti and Mairal, 2019a)

Constructing a CNN in the RKHS $\mathcal{H}_{CKN}$

- Consider a CNN with filters $W_k^{ij}(u), u \in S_k$
- "Smooth homogeneous" activations σ
- The CNN can be **constructed hierarchically** in $\mathcal{H}_{CKN}$
- Norm upper bound (linear layers):

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

- Linear layers: product of spectral norms

(Bietti and Mairal, 2019a)

Link with generalization

- Simple bound on Rademacher complexity for linear/kernel methods:

$$\mathcal{F}_B = \{f \in \mathcal{H}, \|f\|_{\mathcal{H}} \leq B\} \implies \text{Rad}_N(\mathcal{F}_B) \leq O\left(\frac{BR}{\sqrt{N}}\right)$$

Link with generalization

- Simple bound on Rademacher complexity for linear/kernel methods:

$$\mathcal{F}_B = \{f \in \mathcal{H}, \|f\|_{\mathcal{H}} \leq B\} \implies \text{Rad}_N(\mathcal{F}_B) \leq O\left(\frac{BR}{\sqrt{N}}\right)$$

- **Margin bound** for a learned model $\hat{f}_N$ with margin (confidence) $\gamma > 0$

$$P(y\hat{f}_N(x) < 0) \leq O\left(\frac{\|\hat{f}_N\|_{\mathcal{H}} R}{\gamma\sqrt{N}}\right)$$

- If $\hat{f}_N$ is a CNN: related to recent generalization bounds for neural networks based on **product of spectral norms** (*e.g.*, Bartlett et al., 2017; Neyshabur et al., 2018)

Going further for the non-convolutional case

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

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

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

Going further for the non-convolutional case

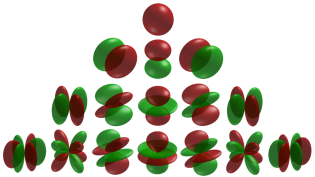
Fully-connected models $\implies$ dot-product kernels

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

- 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 $L^2(\mathbb{S}^{p-1})$ basis of **spherical harmonics** $Y_{k,j}$



Going further for the non-convolutional case

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

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

- 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 $L^2(\mathbb{S}^{p-1})$ basis of **spherical harmonics** $Y_{k,j}$

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

Going further for the non-convolutional case

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

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

- 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 $L^2(\mathbb{S}^{p-1})$ basis of **spherical harmonics** $Y_{k,j}$

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

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

NTK vs random features (Bietti and Mairal, 2019b)

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

Some experiments on Cifar10

Convolutional kernels with 3x3 patches + kernel ridge regression (very costly!)

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

Some experiments on Cifar10

Convolutional kernels with 3x3 patches + kernel ridge regression (very costly!)

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**%

Some experiments on Cifar10

Convolutional kernels with 3x3 patches + kernel ridge regression (very costly!)

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%

Li et al. (2019): no pooling before last layer, more complicated pre-processing

Outline

Regularizing deep learning models in practice

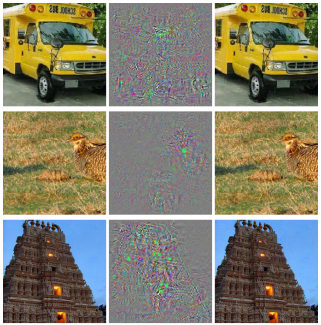
Two issues with today's deep learning models:

- Poor performance on **small datasets**
- **Lack of robustness** to adversarial perturbations

Regularizing deep learning models in practice

Two issues with today's deep learning models:

- Poor performance on **small datasets**
- **Lack of robustness** to adversarial perturbations



clean + noise $\rightarrow$ "**ostrich**"

Regularizing deep learning models in practice

Two issues with today's deep learning models:

- Poor performance on **small datasets**
- **Lack of robustness** to adversarial perturbations

(a real ostrich)

Regularizing deep learning models in practice

Two issues with today's deep learning models:

- Poor performance on **small datasets**
- **Lack of robustness** to adversarial perturbations

New approach to regularization (Bietti et al., 2019):

- View generic CNN f_θ as an element of a RKHS $\mathcal{H}$
  - CNNs f_θ with ReLUs are (approximately) **in the RKHS** for CKNs
- Regularize using $\|f_\theta\|_{\mathcal{H}}$

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints

(Bietti, Mialon, Chen, and Mairal, 2019)

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$

(Bietti, Mialon, Chen, and Mairal, 2019)

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball using properties of Φ

(Bietti, Mialon, Chen, and Mairal, 2019)

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball using properties of Φ

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\delta\| \leq 1} \langle f, \Phi(x+\delta) - \Phi(x) \rangle_{\mathcal{H}} \quad \text{(adversarial perturbations)}$$

(Bietti, Mialon, Chen, and Mairal, 2019)

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball using properties of Φ

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\delta\| \leq 1} f(x+\delta) - f(x) \quad \text{(adversarial perturbations)}$$

(Bietti, Mialon, Chen, and Mairal, 2019)

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball using properties of Φ

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\delta\| \leq 1} f(x+\delta) - f(x) \quad \text{(adversarial perturbations)}$$

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\tau\| \leq C} f(L_\tau x) - f(x) \quad \text{(adversarial deformations)}$$

(Bietti, Mialon, Chen, and Mairal, 2019)

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball using properties of Φ

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\delta\| \leq 1} f(x+\delta) - f(x) \quad \text{(adversarial perturbations)}$$

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\tau\| \leq C} f(L_\tau x) - f(x) \quad \text{(adversarial deformations)}$$

$$\|f\|_{\mathcal{H}} \geq \sup_{x} \|\nabla f(x)\|_2 \quad \text{(gradient penalty)}$$

(Bietti, Mialon, Chen, and Mairal, 2019)

Better models through regularization

- Controlling **upper bounds**: spectral norm penalties/constraints
- Controlling **lower bounds** using $\|f\|_{\mathcal{H}} = \sup_{\|u\|_{\mathcal{H}} \leq 1} \langle f, u \rangle$
- $\implies$ consider tractable subsets of the unit ball using properties of Φ

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\delta\| \leq 1} f(x+\delta) - f(x) \quad \text{(adversarial perturbations)}$$

$$\|f\|_{\mathcal{H}} \geq \sup_{x, \|\tau\| \leq C} f(L_\tau x) - f(x) \quad \text{(adversarial deformations)}$$

$$\|f\|_{\mathcal{H}} \geq \sup_{x} \|\nabla f(x)\|_2 \quad \text{(gradient penalty)}$$

- Best performance by combining upper + lower bound approaches

(Bietti, Mialon, Chen, and Mairal, 2019)

Regularization on small datasets: image classification

Table 2. Regularization on 300 or 1 000 examples from MNIST, using deformations from Infinite MNIST. $(*)$ indicates that random deformations were included as training examples, while $\|f\|_\tau^2$ and $\|D_\tau f\|^2$ use them as part of the regularization penalty.

Method	300 VGG	1k VGG
Weight decay	89.32	94.08
SN projection	90.69	95.01
grad-ℓ_2	93.63	96.67
$\|f\|_\delta^2$ penalty	94.17	96.99
$\|\nabla f\|^2$ penalty	94.08	96.82
Weight decay $(*)$	92.41	95.64
grad-ℓ_2 $(*)$	95.05	97.48
$\|D_\tau f\|^2$ penalty	94.18	96.98
$\|f\|_\tau^2$ penalty	94.42	97.13
$\|f\|_\tau^2 + \|\nabla f\|^2$	94.75	97.40
$\|f\|_\tau^2 + \|f\|_\delta^2$	95.23	**97.66**
$\|f\|_\tau^2 + \|f\|_\delta^2$ $(*)$	**95.53**	**97.56**
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj	95.20	**97.60**
$\|f\|_\tau^2 + \|f\|_\delta^2$ + SN proj $(*)$	**95.40**	**97.77**

(Bietti, Mialon, Chen, and Mairal, 2019)

Regularization on small datasets: protein homology detection

Table 3. Regularization on protein homology detection tasks, with or without data augmentation (DA). Fixed hyperparameters are selected using the first half of the datasets, and we report the average auROC50 score on the second half.

Method	No DA	DA
No weight decay	0.446	0.500
Weight decay	0.501	0.546
SN proj	0.591	**0.632**
PGD-ℓ_2	0.575	0.595
grad-ℓ_2	0.540	0.552
$\|f\|_\delta^2$	**0.600**	0.608
$\|\nabla f\|^2$	0.585	0.611
PGD-ℓ_2 + SN proj	**0.596**	**0.627**
grad-ℓ_2 + SN proj	0.592	**0.624**
$\|f\|_\delta^2$ + SN proj	**0.630**	**0.644**
$\|\nabla f\|^2$ + SN proj	**0.603**	**0.625**

(Bietti, Mialon, Chen, and Mairal, 2019)

Regularization for robustness

Links with robust optimization/adversarial training

- Robust optimization yields another lower bound (hinge/logistic loss)

$$\frac{1}{N}\sum_{i=1}^{N} \sup_{\|\delta\|_2 \leq \epsilon} \ell(y_i, f(x_i + \delta)) \leq \frac{1}{N}\sum_{i=1}^{N} \ell(y_i, f(x_i)) + \epsilon \|f\|_{\mathcal{H}}$$

- **But**: may only encourage **local** robustness around training data

(Bietti, Mialon, Chen, and Mairal, 2019)

Regularization for robustness

Links with robust optimization/adversarial training

- Robust optimization yields another lower bound (hinge/logistic loss)

$$\frac{1}{N}\sum_{i=1}^{N}\sup_{\|\delta\|_2\leq\epsilon}\ell(y_i, f(x_i+\delta)) \leq \frac{1}{N}\sum_{i=1}^{N}\ell(y_i, f(x_i)) + \epsilon\|f\|_{\mathcal{H}}$$

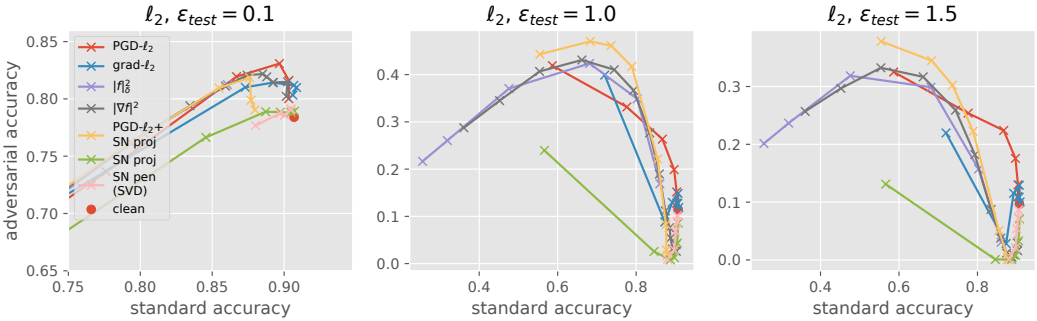
- **But**: may only encourage **local** robustness around training data

Global vs local robustness

- Controlling $\|f\|_{\mathcal{H}}$ allows a more **global** form of robustness
- Guarantees on **adversarial generalization** with ℓ_2 perturbations
  - Extension of margin-based bound, by using $\|f\|_{\mathcal{H}} \geq \|f\|_{\text{Lip}}$ near the decision boundary
- But: may cause a loss in accuracy in practice

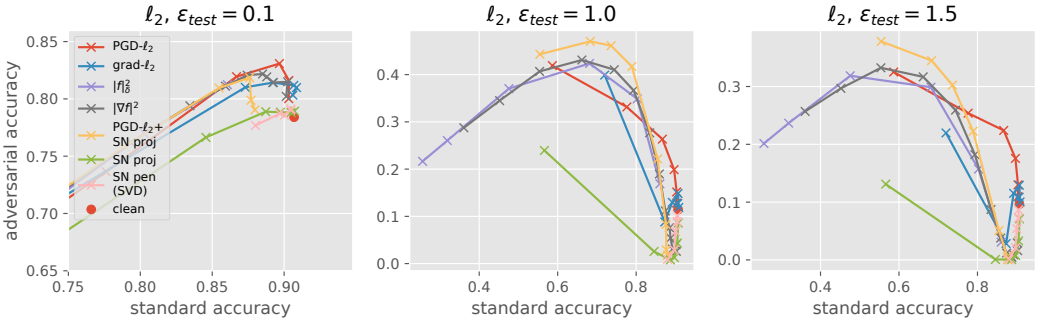
(Bietti, Mialon, Chen, and Mairal, 2019)

Robustness trade-offs on Cifar10



(Bietti, Mialon, Chen, and Mairal, 2019)

Robustness trade-offs on Cifar10



State-of-the-art robust accuracy for large ϵ_{test}

(Bietti, Mialon, Chen, and Mairal, 2019)

Outline

Conclusions

Benefits of convolutional kernels

- Translation invariance + deformation stability with small patches and pooling
- Extensions to other groups (roto-translations)
- RKHS contains CNNs with smooth activations

Conclusions

Benefits of convolutional kernels

- Translation invariance + deformation stability with small patches and pooling
- Extensions to other groups (roto-translations)
- RKHS contains CNNs with smooth activations

New practical regularization strategies

- Regularization of generic CNNs using RKHS norm
- State-of-the-art performance on adversarial robustness

Conclusions

Benefits of convolutional kernels

- Translation invariance + deformation stability with small patches and pooling
- Extensions to other groups (roto-translations)
- RKHS contains CNNs with smooth activations

New practical regularization strategies

- Regularization of generic CNNs using RKHS norm
- State-of-the-art performance on adversarial robustness

Links with over-parameterized optimization: neural tangent kernels

- NTK for CNNs takes a similar form to CKNs
- Weaker stability guarantees than RF, but better approximation properties

Perspectives

Further study of convolutional kernels

- More precise approximation guarantees?
- More empirical evaluation; usefulness in practice?

Perspectives

Further study of convolutional kernels

- More precise approximation guarantees?
- More empirical evaluation; usefulness in practice?

Beyond kernels for deep learning

- Kernels do not fully explain success of deep learning
- Simple, tractable, interpretable models that improve on kernels?
- Inductive bias of optimization beyond "lazy training"? lazy only for some layers?

Bonus: contextual bandit bake-off

Contextual bandits

- Simple setting for real-world reinforcement learning ("one-step RL")
- Observe context (user info), choose action (ad/news story), observe reward (click?)
- $\implies$ exploration/exploitation trade-off
- Several algorithms with theoretical guarantees, little empirical evaluation

Bonus: contextual bandit bake-off

Contextual bandits

- Simple setting for real-world reinforcement learning ("one-step RL")
- Observe context (user info), choose action (ad/news story), observe reward (click?)
- $\implies$ exploration/exploitation trade-off
- Several algorithms with theoretical guarantees, little empirical evaluation

The bake-off (Bietti, Agarwal, and Langford, 2020?)

- Large-scale evaluation on 500+ datasets from supervised learning
- Focus on practical methods that leverage supervised ML algorithms
- All methods implemented in Vowpal Wabbit!

Contextual bandit bake-off: takeaways and perspectives

Findings

- Methods with strong assumptions tend to dominate: **RegCB** and even **Greedy** (!)
- Many theoretically analyzed methods tend to **over-explore** but are more robust to difficult datasets when appropriately modified: **Cover**
- Importance of **design choices** (reward estimators, encoding, ability to run offline experiments)

Contextual bandit bake-off: takeaways and perspectives

Findings

- Methods with strong assumptions tend to dominate: **RegCB** and even **Greedy** (!)
- Many theoretically analyzed methods tend to **over-explore** but are more robust to difficult datasets when appropriately modified: **Cover**
- Importance of **design choices** (reward estimators, encoding, ability to run offline experiments)

Perspectives

- **Simple is good**: better understanding of greedy?
- **Robustness with adaptivity**: can we get robust methods that are competitive with greedy/RegCB in favorable scenarios?
- Closer to the **real world**: how to deal with **non-stationarity**? *e.g.*, feedback loops in recommendation systems. Need better data models?

Thanks!

References I

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

P. L. Bartlett, D. J. Foster, and M. Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

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

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

A. Bietti, G. Mialon, D. Chen, and J. Mairal. A kernel perspective for regularizing deep neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

A. Bietti, A. Agarwal, and J. Langford. A contextual bandit bake-off. *arXiv preprint arXiv:1802.04064*, 2020?

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

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.

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

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.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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.

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.

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

B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

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

References IV

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

Y. Zhang, J. D. Lee, and M. I. Jordan. ℓ_1-regularized neural networks are improperly learnable in polynomial time. In *International Conference on Machine Learning (ICML)*, 2016.

Y. Zhang, P. Liang, and M. J. Wainwright. Convexified convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

Bake-off results

Figure: Comparison between three competitive approaches: RegCB (confidence based), Cover-NU (variant of Online Cover) and Greedy. The plots show relative loss compared to supervised learning (lower is better) on all datasets with 5 actions or more. Red points indicate datasets with a statistically significant difference in loss between two methods. A greedy approach can outperform exploration methods in many cases; yet both Greedy and RegCB may fail to explore efficiently on some other datasets where Cover-NU dominates.

Bake-off results

Stat. significant "win-loss" difference. Fixed hyperparameters.

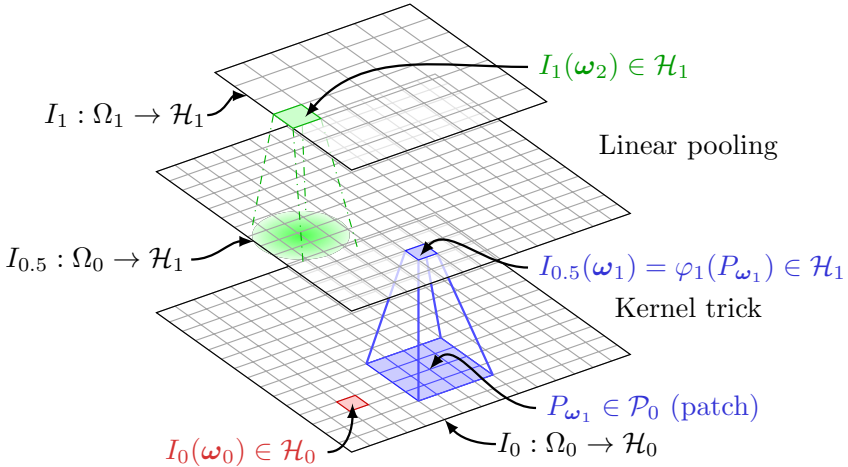
$\downarrow$ vs $\rightarrow$	G	RO	C-nu	B-g	ϵG
G	-	-7	10	50	54
RO	**7**	-	**26**	**49**	**68**
C-nu	-10	-26	-	22	57
B-g	-50	-49	-22	-	17
ϵG	-54	-68	-57	-17	-

-1/0 encoding

$\downarrow$ vs $\rightarrow$	G	RO	C-nu	B-g	ϵG
G	-	-64	-17	36	52
RO	**64**	-	**45**	**100**	**120**
C-nu	17	-45	-	45	75
B-g	-36	-100	-45	-	19
ϵG	-52	-120	-75	-19	-

0/1 encoding

Discretization and signal preservation



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

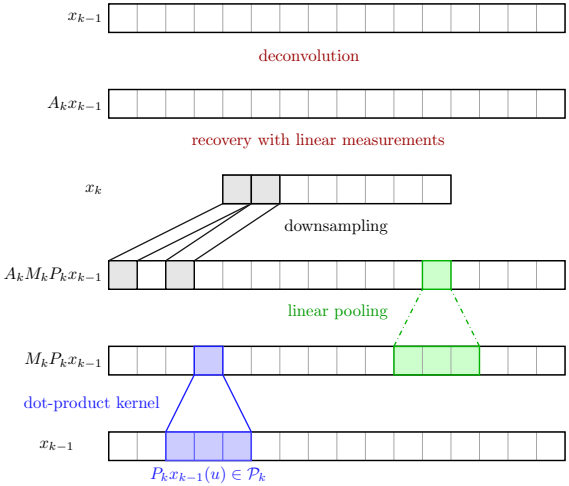
- $\bar{x}_k$: subsampling factor s_k after pooling with scale $\sigma_k \approx s_k$:

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

- **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