A Kernel Perspective for Regularizing Deep Neural Networks

Julien Mairal
Inria Grenoble

Imaging and Machine Learning, IHP, 2019
Publications

Theoretical Foundations


Practical aspects

Convolutional Neural Networks
Short Introduction and Current Challenges
Learning a predictive model

The goal is to learn a **prediction function** $f : \mathbb{R}^p \rightarrow \mathbb{R}$ given labeled training data $(x_i, y_i)_{i=1,...,n}$ with $x_i$ in $\mathbb{R}^p$, and $y_i$ in $\mathbb{R}$:

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \lambda \Omega(f).$$

**empirical risk, data fit**

**regularization**

Julien Mairal
A Kernel Perspective for Regularizing NN
Convolutional Neural Networks

The goal is to learn a prediction function $f : \mathbb{R}^p \to \mathbb{R}$ given labeled training data $(x_i, y_i)_{i=1,...,n}$ with $x_i$ in $\mathbb{R}^p$, and $y_i$ in $\mathbb{R}$:

$$
\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \lambda \Omega(f) .
$$

What is specific to multilayer neural networks?

- The “neural network” space $\mathcal{F}$ is explicitly parametrized by:

$$
f(x) = \sigma_k(W_k \sigma_{k-1}(W_{k-1} \ldots \sigma_2(W_2 \sigma_1(W_1 x)) \ldots)) .
$$

- Linear operations are either unconstrained (fully connected) or share parameters (e.g., convolutions).

- Finding the optimal $W_1, W_2, \ldots, W_k$ yields a non-convex optimization problem in huge dimension.
What are the main features of CNNs?

- they capture **compositional** and **multiscale** structures in images;
- they provide some **invariance**;
- they model **local stationarity** of images at several scales;
- they are **state-of-the-art** in many fields.
Convolutional Neural Networks

The keywords: multi-scale, compositional, invariant, local features.

Picture from Y. LeCun’s tutorial:

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

Picture from Olah et al. [2017]:

Edges (layer conv2d0)  Textures (layer mixed3a)  Patterns (layer mixed4a)
Convolutional Neural Networks

Picture from Olah et al. [2017]:

Patterns (layer mixed4a)

Parts (layers mixed4b & mixed4c)

Objects (layers mixed4d & mixed4e)
Convolutional Neural Networks: Challenges

What are current high-potential problems to solve?

1. lack of **stability** (see next slide).
2. learning with **few labeled data**.
3. learning with **no supervision** (see Tab. from Bojanowski and Joulin, 2017).

<table>
<thead>
<tr>
<th>Method</th>
<th>Acc@1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random (Noroozi &amp; Favaro, 2016)</td>
<td>12.0</td>
</tr>
<tr>
<td>SIFT+FV (Sánchez et al., 2013)</td>
<td>55.6</td>
</tr>
<tr>
<td>Wang &amp; Gupta (2015)</td>
<td>29.8</td>
</tr>
<tr>
<td>Doersch et al. (2015)</td>
<td>30.4</td>
</tr>
<tr>
<td>Zhang et al. (2016)</td>
<td>35.2</td>
</tr>
<tr>
<td>¹Noroozi &amp; Favaro (2016)</td>
<td>38.1</td>
</tr>
<tr>
<td>BiGAN (Donahue et al., 2016)</td>
<td>32.2</td>
</tr>
<tr>
<td>NAT</td>
<td>36.0</td>
</tr>
</tbody>
</table>

*Table 3. Comparison of the proposed approach to state-of-the-art unsupervised feature learning on ImageNet. A full multi-layer perceptron is retrained on top of the features. We compare to several self-supervised approaches and an unsupervised approach.*
Convolutional Neural Networks: Challenges

Illustration of instability. Picture from Kurakin et al. [2016].

Figure: Adversarial examples are generated by computer; then printed on paper; a new picture taken on a smartphone fools the classifier.
Convolutional Neural Networks: Challenges

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \lambda \Omega(f).$$

The issue of regularization

- today, heuristics are used (DropOut, weight decay, early stopping)...
- ...but they are not sufficient.
- how to control variations of prediction functions?

$$|f(x) - f(x')|$$ should be close if $x$ and $x'$ are “similar”.

- what does it mean for $x$ and $x'$ to be “similar”?
- what should be a good regularization function $\Omega$?
Deep Neural Networks from a Kernel Perspective
A kernel perspective

Recipe

- Map data $x$ to high-dimensional space, $\Phi(x)$ in $\mathcal{H}$ (RKHS), with Hilbertian geometry (projections, barycenters, angles, . . . , exist!).
- Predictive models $f$ in $\mathcal{H}$ are linear forms in $\mathcal{H}$: $f(x) = \langle f, \Phi(x) \rangle_\mathcal{H}$.
- Learning with a positive definite kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle_\mathcal{H}$.

[Schölkopf and Smola, 2002, Shawe-Taylor and Cristianini, 2004]...
A kernel perspective

Recipe

- Map data \( x \) to **high-dimensional space**, \( \Phi(x) \) in \( \mathcal{H} \) (RKHS), with Hilbertian geometry (projections, barycenters, angles, . . . , exist!).
- Predictive models \( f \) in \( \mathcal{H} \) are **linear forms** in \( \mathcal{H} \): \( f(x) = \langle f, \Phi(x) \rangle_\mathcal{H} \).
- Learning with a positive definite kernel \( K(x, x') = \langle \Phi(x), \Phi(x') \rangle_\mathcal{H} \).

What is the relation with deep neural networks?

- It is possible to design a RKHS \( \mathcal{H} \) where a large class of deep neural networks live [Mairal, 2016].

\[
    f(x) = \sigma_k(W_k \sigma_{k-1}(W_{k-1} \ldots \sigma_2(W_2 \sigma_1(W_1 x)) \ldots)) = \langle f, \Phi(x) \rangle_\mathcal{H}.
\]

- This is the construction of **“convolutional kernel networks”**.

[Schölkopf and Smola, 2002, Shawe-Taylor and Cristianini, 2004],...
A kernel perspective

Recipe

- Map data $x$ to high-dimensional space, $\Phi(x)$ in $\mathcal{H}$ (RKHS), with Hilbertian geometry (projections, barycenters, angles, . . . , exist!).
- Predictive models $f$ in $\mathcal{H}$ are linear forms in $\mathcal{H}$: $f(x) = \langle f, \Phi(x) \rangle_\mathcal{H}$.
- Learning with a positive definite kernel $K(x, x') = \langle \Phi(x), \Phi(x') \rangle_\mathcal{H}$.

Why do we care?

- $\Phi(x)$ is related to the network architecture and is independent of training data. Is it stable? Does it lose signal information?
- $f$ is a predictive model. Can we control its stability?

\[
|f(x) - f(x')| \leq \|f\|_\mathcal{H} \|\Phi(x) - \Phi(x')\|_\mathcal{H}.
\]

- $\|f\|_\mathcal{H}$ controls both stability and generalization!
Summary of the results from Bietti and Mairal [2019]

Multi-layer construction of the RKHS $\mathcal{H}$

- Contains CNNs with smooth homogeneous activations functions.
Summary of the results from Bietti and Mairal [2019]

Multi-layer construction of the RKHS $\mathcal{H}$
- Contains CNNs with smooth homogeneous activations functions.

Signal representation: Conditions for
- Signal preservation of the multi-layer kernel mapping $\Phi$.
- Stability to deformations and non-expansiveness for $\Phi$.
- Constructions to achieve group invariance.
Summary of the results from Bietti and Mairal [2019]

Multi-layer construction of the RKHS $\mathcal{H}$
- Contains CNNs with smooth homogeneous activations functions.

Signal representation: Conditions for
- Signal preservation of the multi-layer kernel mapping $\Phi$.
- Stability to deformations and non-expansiveness for $\Phi$.
- Constructions to achieve group invariance.

On learning
- Bounds on the RKHS norm $\| \cdot \|_\mathcal{H}$ to control stability and generalization of a predictive model $f$.

\[ |f(x) - f(x')| \leq \| f \|_\mathcal{H} \| \Phi(x) - \Phi(x') \|_\mathcal{H}. \]

[Mallat, 2012]
Smooth homogeneous activations functions

\[ z \mapsto \text{ReLU}(w^T z) \quad \Rightarrow \quad z \mapsto \|z\| \sigma(w^T z / \|z\|). \]
A kernel perspective: regularization

Assume we have an RKHS $\mathcal{H}$ for deep networks:

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2.$$  

$\|\cdot\|_{\mathcal{H}}$ encourages smoothness and stability w.r.t. the geometry induced by the kernel (which depends itself on the choice of architecture).
A kernel perspective: regularization

Assume we have an RKHS $\mathcal{H}$ for deep networks:

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \frac{\lambda}{2} \| f \|^2_{\mathcal{H}}.$$ 

$\| . \|_{\mathcal{H}}$ encourages smoothness and stability w.r.t. the geometry induced by the kernel (which depends itself on the choice of architecture).

Problem

Multilayer kernels developed for deep networks are **typically intractable**.

One solution [Mairal, 2016]

do kernel approximations at each layer, which leads to non-standard CNNs called convolutional kernel networks (CKNs).
A kernel perspective: regularization

Assume we have an RKHS $\mathcal{H}$ for deep networks:

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2.$$ 

$\|\cdot\|_{\mathcal{H}}$ encourages smoothness and stability w.r.t. the geometry induced by the kernel (which depends itself on the choice of architecture).

Problem

Multilayer kernels developed for deep networks are typically intractable.

One solution [Mairal, 2016]
do kernel approximations at each layer, which leads to non-standard CNNs called convolutional kernel networks (CKNs).

not the subject of this talk.
A kernel perspective: regularization

Consider a classical CNN parametrized by $\theta$, which live in the RKHS:

$$\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \frac{\lambda}{2} \|f_\theta\|_H^2.$$ 

This is different than CKNs since $f_\theta$ admits a classical parametrization.
Consider a classical CNN parametrized by $\theta$, which live in the RKHS:

$$\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \frac{\lambda}{2} \| f_\theta \|_H^2.$$ 

This is different than CKNs since $f_\theta$ admits a classical parametrization.

Problem

$\| f_\theta \|_H$ is intractable...

One solution [Bietti et al., 2019]

use approximations (lower- and upper-bounds), based on mathematical properties of $\| \cdot \|_H$. 
A kernel perspective: regularization

Consider a classical CNN parametrized by $\theta$, which live in the RKHS:

$$\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \frac{\lambda}{2} \|f_\theta\|^2_\mathcal{H}.$$ 

This is different than CKNs since $f_\theta$ admits a classical parametrization.

Problem

$\|f_\theta\|_\mathcal{H}$ is intractable...

One solution [Bietti et al., 2019]

use approximations (lower- and upper-bounds), based on mathematical properties of $\|\cdot\|_\mathcal{H}$.

This is the subject of this talk.
Construction of the RKHS for continuous signals

Initial map $x_0$ in $L^2(\Omega, \mathcal{H}_0)$

$x_0 : \Omega \rightarrow \mathcal{H}_0$: continuous input signal

- $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images).
- $x_0(u) \in \mathcal{H}_0$: input value at location $u$ ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images).
Construction of the RKHS for continuous signals

Initial map $x_0$ in $L^2(\Omega, \mathcal{H}_0)$

$x_0 : \Omega \to \mathcal{H}_0$: continuous input signal

- $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images).
- $x_0(u) \in \mathcal{H}_0$: input value at location $u$ ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images).

Building map $x_k$ in $L^2(\Omega, \mathcal{H}_k)$ from $x_{k-1}$ in $L^2(\Omega, \mathcal{H}_{k-1})$

$x_k : \Omega \to \mathcal{H}_k$: 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 $u$ ($P_k x_{k-1}(u)$ is a patch centered at $u$).
Construction of the RKHS for continuous signals

Initial map $x_0$ in $L^2(\Omega, \mathcal{H}_0)$

$x_0 : \Omega \to \mathcal{H}_0$: continuous input signal

- $u \in \Omega = \mathbb{R}^d$: location $(d = 2$ for images$)$. 
- $x_0(u) \in \mathcal{H}_0$: input value at location $u$ $(\mathcal{H}_0 = \mathbb{R}^3$ for RGB images$)$.

Building map $x_k$ in $L^2(\Omega, \mathcal{H}_k)$ from $x_{k-1}$ in $L^2(\Omega, \mathcal{H}_{k-1})$

$x_k : \Omega \to \mathcal{H}_k$: 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 $u$ ($P_k x_{k-1}(u)$ is a patch centered at $u$).
- $M_k$: non-linear mapping operator, maps each patch to a new Hilbert space $\mathcal{H}_k$ with a pointwise non-linear function $\varphi_k(\cdot)$. 

Construction of the RKHS for continuous signals

Initial map $x_0$ in $L^2(\Omega, \mathcal{H}_0)$

$x_0 : \Omega \rightarrow \mathcal{H}_0$: continuous input signal

- $u \in \Omega = \mathbb{R}^d$: location ($d = 2$ for images).
- $x_0(u) \in \mathcal{H}_0$: input value at location $u$ ($\mathcal{H}_0 = \mathbb{R}^3$ for RGB images).

Building map $x_k$ in $L^2(\Omega, \mathcal{H}_k)$ from $x_{k-1}$ in $L^2(\Omega, \mathcal{H}_{k-1})$

$x_k : \Omega \rightarrow \mathcal{H}_k$: 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 $u$ ($P_k x_{k-1}(u)$ is a patch centered at $u$).
- $M_k$: non-linear mapping operator, maps each patch to a new Hilbert space $\mathcal{H}_k$ with a pointwise non-linear function $\varphi_k(\cdot)$.
- $A_k$: (linear) pooling operator at scale $\sigma_k$. 
Construction of the RKHS for continuous signals

\[ x_k := A_k M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k \]

\[ x_k(w) = A_k M_k P_k x_{k-1}(w) \in \mathcal{H}_k \] (linear pooling)

\[ M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k \]

\[ M_k P_k x_{k-1}(v) = \varphi_k(P_k x_{k-1}(v)) \in \mathcal{H}_k \] (kernel mapping)

\[ x_{k-1}(u) \in \mathcal{H}_{k-1} \]

\[ x_{k-1} : \Omega \rightarrow \mathcal{H}_{k-1} \] (patch extraction)
Construction of the RKHS for continuous signals

Assumption on $x_0$

- $x_0$ is typically a **discrete** signal acquired with physical device.
- Natural assumption: $x_0 = A_0x$, with $x$ the original continuous signal, $A_0$ local integrator with scale $\sigma_0$ (**anti-aliasing**).
Construction of the RKHS for continuous signals

Assumption on $x_0$

- $x_0$ is typically a **discrete** signal acquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with $x$ the original continuous signal, $A_0$ local integrator with scale $\sigma_0$ (**anti-aliasing**).

Multilayer representation

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

- $\sigma_k$ grows exponentially in practice (i.e., fixed with subsampling).
Construction of the RKHS for continuous signals

Assumption on $x_0$

- $x_0$ is typically a **discrete** signal acquired with physical device.
- Natural assumption: $x_0 = A_0 x$, with $x$ the original continuous signal, $A_0$ local integrator with scale $\sigma_0$ (**anti-aliasing**).

Multilayer representation

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

- $\sigma_k$ grows exponentially in practice (i.e., fixed with subsampling).

Prediction layer

- e.g., linear $f(x) = \langle w, \Phi_n(x) \rangle$.
- “linear kernel” $K(x, x') = \langle \Phi_n(x), \Phi_n(x') \rangle = \int_\Omega \langle x_n(u), x'_n(u) \rangle du$. 
Practical Regularization Strategies
A kernel perspective: regularization

Another point of view: consider a classical CNN parametrized by $\theta$, which live in the RKHS:

$$\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_{\theta}(x_i)) + \frac{\lambda}{2} \| f_{\theta} \|_H^2.$$ 

Upper-bounds

$$\| f_{\theta} \|_H \leq \omega(\| W_k \|, \| W_{k-1} \|, \ldots, \| W_1 \|) \quad \text{(spectral norms)},$$

where the $W_j$’s are the convolution filters. The bound suggests controlling the spectral norm of the filters.

[Cisse et al., 2017, Miyato et al., 2018, Bartlett et al., 2017]...
A kernel perspective: regularization

Another point of view: consider a classical CNN parametrized by $\theta$, which live in the RKHS:

$$
\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_{\theta}(x_i)) + \frac{\lambda}{2} \|f_{\theta}\|_H^2.
$$

Lower-bounds

$$
\|f\|_\mathcal{H} = \sup_{\|u\|_\mathcal{H} \leq 1} \langle f, u \rangle_\mathcal{H} \geq \sup_{u \in U} \langle f, u \rangle_\mathcal{H} \quad \text{for} \quad U \subseteq B_{\mathcal{H}}(1).
$$

We design a set $U$ that leads to a tractable approximation, but it requires some knowledge about the properties of $\mathcal{H}, \Phi$. 
A kernel perspective: regularization

Adversarial penalty

We know that $\Phi$ is non-expansive and $f(x) = \langle f, \Phi(x) \rangle$. Then,

$$U = \{ \Phi(x + \delta) - \Phi(x) : x \in \mathcal{X}, \|\delta\|_2 \leq 1 \}$$

leads to

$$\lambda\|f\|_2^2 = \sup_{x \in \mathcal{X}, \|\delta\|_2 \leq \lambda} f(x + \delta) - f(x).$$

The resulting strategy is related to adversarial regularization (but it is decoupled from the loss term and does not use labels).

$$\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \sup_{x \in \mathcal{X}, \|\delta\|_2 \leq \lambda} f_\theta(x + \delta) - f_\theta(x).$$

[Madry et al., 2018]
A kernel perspective: regularization

Adversarial penalty

We know that $\Phi$ is non-expansive and $f(x) = \langle f, \Phi(x) \rangle$. Then,

$$U = \{ \Phi(x + \delta) - \Phi(x) : x \in X, \|\delta\|_2 \leq 1 \}$$

leads to

$$\lambda \|f\|_\delta^2 = \sup_{x \in X, \|\delta\|_2 \leq \lambda} f(x + \delta) - f(x).$$

The resulting strategy is related to adversarial regularization (but it is decoupled from the loss term and does not use labels).

vs, for adversarial regularization,

$$\min_{\theta \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} \sup_{\|\delta\|_2 \leq \lambda} L(y_i, f_\theta(x_i + \delta)).$$

[Madry et al., 2018]
Gradient penalties

We know that $\Phi$ is non-expansive and $f(x) = \langle f, \Phi(x) \rangle$. Then,

$$U = \{ \Phi(x + \delta) - \Phi(x) : x \in \mathcal{X}, \|\delta\|_2 \leq 1 \}$$

leads to

$$\|\nabla f\| = \sup_{x \in \mathcal{X}} \|\nabla f(x)\|_2.$$

Related penalties have been used to stabilize the training of GANs and gradients of the loss function have been used to improve robustness.

A kernel perspective: regularization

Adversarial deformation penalties

We know that $\Phi$ is stable to deformations and $f(x) = \langle f, \Phi(x) \rangle$. Then,

$$U = \{ \Phi(L_\tau x) - \Phi(x) : x \in \mathcal{X}, \tau \}$$

leads to

$$\|f\|_\tau^2 = \sup_{x \in \mathcal{X}, \tau \text{ small deformation}} f(L_\tau x) - f(x).$$

This is related to data augmentation and tangent propagation.

[Engstrom et al., 2017, Simard et al., 1998]
Experiments with Few labeled Samples

Table: Accuracies on CIFAR10 with 1000 examples for standard architectures VGG-11 and ResNet-18. With / without data augmentation.

<table>
<thead>
<tr>
<th>Method</th>
<th>1k VGG-11</th>
<th>1k ResNet-18</th>
</tr>
</thead>
<tbody>
<tr>
<td>No weight decay</td>
<td>50.70 / 43.75</td>
<td>45.23 / 37.12</td>
</tr>
<tr>
<td>Weight decay</td>
<td>51.32 / 43.95</td>
<td>44.85 / 37.09</td>
</tr>
<tr>
<td>SN projection</td>
<td>54.14 / <strong>46.70</strong></td>
<td>47.12 / 37.28</td>
</tr>
<tr>
<td>PGD-(\ell_2)</td>
<td>51.25 / 44.40</td>
<td>45.80 / 41.87</td>
</tr>
<tr>
<td>grad-(\ell_2)</td>
<td><strong>55.19</strong> / 43.88</td>
<td><strong>49.30</strong> / 44.65</td>
</tr>
<tr>
<td>(|f|_\delta^2) penalty</td>
<td>51.41 / 45.07</td>
<td>48.73 / 43.72</td>
</tr>
<tr>
<td>(|\nabla f|_2^2) penalty</td>
<td>54.80 / 46.37</td>
<td><strong>48.99</strong> / <strong>44.97</strong></td>
</tr>
<tr>
<td>PGD-(\ell_2) + SN proj</td>
<td>54.19 / <strong>46.66</strong></td>
<td>47.47 / 41.25</td>
</tr>
<tr>
<td>grad-(\ell_2) + SN proj</td>
<td><strong>55.32</strong> / <strong>46.88</strong></td>
<td>48.73 / 42.78</td>
</tr>
<tr>
<td>(|f|_\delta^2) + SN proj</td>
<td>54.02 / <strong>46.72</strong></td>
<td>48.12 / 43.56</td>
</tr>
<tr>
<td>(|\nabla f|_2^2) + SN proj</td>
<td><strong>55.24</strong> / <strong>46.80</strong></td>
<td><strong>49.06</strong> / <strong>44.92</strong></td>
</tr>
</tbody>
</table>
Experiments with Few labeled Samples

Table: Accuracies with 300 or 1 000 examples from MNIST, using deformations. (*) indicates that random deformations were included as training examples,

<table>
<thead>
<tr>
<th>Method</th>
<th>300 VGG</th>
<th>1k VGG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight decay</td>
<td>89.32</td>
<td>94.08</td>
</tr>
<tr>
<td>SN projection</td>
<td>90.69</td>
<td>95.01</td>
</tr>
<tr>
<td>grad-$\ell_2$</td>
<td>93.63</td>
<td>96.67</td>
</tr>
<tr>
<td>$|f|_2^2$ penalty</td>
<td>94.17</td>
<td>96.99</td>
</tr>
<tr>
<td>$|\nabla f|_2^2$ penalty</td>
<td>94.08</td>
<td>96.82</td>
</tr>
<tr>
<td>Weight decay (*)</td>
<td>92.41</td>
<td>95.64</td>
</tr>
<tr>
<td>grad-$\ell_2$ (*)</td>
<td>95.05</td>
<td>97.48</td>
</tr>
<tr>
<td>$|D_\tau f|_2^2$ penalty</td>
<td>94.18</td>
<td>96.98</td>
</tr>
<tr>
<td>$|f|_T^2$ penalty</td>
<td>94.42</td>
<td>97.13</td>
</tr>
<tr>
<td>$|f|_T^2 + |\nabla f|_2^2$</td>
<td>94.75</td>
<td>97.40</td>
</tr>
<tr>
<td>$|f|<em>T^2 + |f|</em>\delta^2$</td>
<td>95.23</td>
<td>97.66</td>
</tr>
<tr>
<td>$|f|<em>T^2 + |f|</em>\delta^2$ (*)</td>
<td>95.53</td>
<td>97.56</td>
</tr>
<tr>
<td>$|f|<em>T^2 + |f|</em>\delta^2 +$ SN proj</td>
<td>95.20</td>
<td>97.60</td>
</tr>
<tr>
<td>$|f|<em>T^2 + |f|</em>\delta^2 +$ SN proj (*)</td>
<td>95.40</td>
<td>97.77</td>
</tr>
</tbody>
</table>
Experiments with Few labeled Samples

Table: AUROC50 for protein homology detection tasks using CNN, with or without data augmentation (DA).

<table>
<thead>
<tr>
<th>Method</th>
<th>No DA</th>
<th>DA</th>
</tr>
</thead>
<tbody>
<tr>
<td>No weight decay</td>
<td>0.446</td>
<td>0.500</td>
</tr>
<tr>
<td>Weight decay</td>
<td>0.501</td>
<td>0.546</td>
</tr>
<tr>
<td>SN proj</td>
<td>0.591</td>
<td>0.632</td>
</tr>
<tr>
<td>PGD-$\ell_2$</td>
<td>0.575</td>
<td>0.595</td>
</tr>
<tr>
<td>grad-$\ell_2$</td>
<td>0.540</td>
<td>0.552</td>
</tr>
<tr>
<td>$|f|_{\delta}^2$</td>
<td>0.600</td>
<td>0.608</td>
</tr>
<tr>
<td>$|\nabla f|^2$</td>
<td>0.585</td>
<td>0.611</td>
</tr>
<tr>
<td>PGD-$\ell_2$ + SN proj</td>
<td>0.596</td>
<td>0.627</td>
</tr>
<tr>
<td>grad-$\ell_2$ + SN proj</td>
<td>0.592</td>
<td>0.624</td>
</tr>
<tr>
<td>$|f|_{\delta}^2$ + SN proj</td>
<td>0.630</td>
<td>0.644</td>
</tr>
<tr>
<td>$|\nabla f|^2$ + SN proj</td>
<td>0.603</td>
<td>0.625</td>
</tr>
</tbody>
</table>

Note: statistical tests have been conducted for all of these experiments (see paper).
Experiments with Few labeled Samples

Table: AUROC50 for protein homology detection tasks using CNN, with or without data augmentation (DA).

<table>
<thead>
<tr>
<th>Method</th>
<th>No DA</th>
<th>DA</th>
</tr>
</thead>
<tbody>
<tr>
<td>No weight decay</td>
<td>0.446</td>
<td>0.500</td>
</tr>
<tr>
<td>Weight decay</td>
<td>0.501</td>
<td>0.546</td>
</tr>
<tr>
<td>SN proj</td>
<td>0.591</td>
<td>0.632</td>
</tr>
<tr>
<td>PGD-$\ell_2$</td>
<td>0.575</td>
<td>0.595</td>
</tr>
<tr>
<td>grad-$\ell_2$</td>
<td>0.540</td>
<td>0.552</td>
</tr>
<tr>
<td>$|f|_2^2$</td>
<td>0.600</td>
<td>0.608</td>
</tr>
<tr>
<td>$|\nabla f|_2^2$</td>
<td>0.585</td>
<td>0.611</td>
</tr>
<tr>
<td>PGD-$\ell_2$ + SN proj</td>
<td>0.596</td>
<td>0.627</td>
</tr>
<tr>
<td>grad-$\ell_2$ + SN proj</td>
<td>0.592</td>
<td>0.624</td>
</tr>
<tr>
<td>$|f|_2^2$ + SN proj</td>
<td>0.630</td>
<td>0.644</td>
</tr>
<tr>
<td>$|\nabla f|_2^2$ + SN proj</td>
<td>0.603</td>
<td>0.625</td>
</tr>
</tbody>
</table>

**Note:** statistical tests have been conducted for all of these experiments (see paper).
Adversarial Robustness: Trade-offs

![Graph showing robustness trade-off curves of different regularization methods for VGG11 on CIFAR10. Each plot shows test accuracy vs adversarial test accuracy. Different points on a curve correspond to training with different regularization strengths.]

**Figure:** Robustness trade-off curves of different regularization methods for VGG11 on CIFAR10. Each plot shows test accuracy vs adversarial test accuracy. Different points on a curve correspond to training with different regularization strengths.
Conclusions from this work on regularization

What the kernel perspective brings us

- gives a unified perspective on many regularization principles.
- useful both for generalization and robustness.
- related to robust optimization.

Future work

- regularization based on kernel approximations.
- semi-supervised learning to exploit unlabeled data.
- relation with implicit regularization.
Invariance and Stability to Deformations

(probably for another time)
A signal processing perspective
plus a bit of harmonic analysis

- consider images defined on a **continuous** domain $\Omega = \mathbb{R}^d$.
- $\tau : \Omega \to \Omega$: $c^1$-diffeomorphism.
- $L_\tau x(u) = x(u - \tau(u))$: action operator.
- much richer group of transformations than translations.

A signal processing perspective
plus a bit of harmonic analysis

- consider images defined on a **continuous** domain $\Omega = \mathbb{R}^d$.
- $\tau : \Omega \rightarrow \Omega$: $C^1$-diffeomorphism.
- $L_\tau \ell(u) = \ell(u - \tau(u))$: action operator.
- much richer group of transformations than translations.

relation with deep convolutional representations
stability to deformations studied for wavelet-based scattering transform.

A signal processing perspective
plus a bit of harmonic analysis

- consider images defined on a **continuous** domain $\Omega = \mathbb{R}^d$.
- $\tau : \Omega \to \Omega$: $c^1$-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.
Construction of the RKHS for continuous signals

\[ x_k := A_k M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k \]

Kernel mapping

\[ M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k \]

Linear pooling

\[ x_{k-1}(u) \in \mathcal{H}_{k-1} \]

Patch extraction

\[ P_k x_{k-1}(v) \in \mathcal{P}_k \]

Kernel function

\[ \varphi_k(P_k x_{k-1}(v)) \in \mathcal{H}_k \]

\[ x_k(w) = A_k M_k P_k x_{k-1}(w) \in \mathcal{H}_k \]
Patch extraction operator $P_k$

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

- $S_k$: patch shape, e.g. box.
- $P_k$ is linear, and preserves the norm: $\|P_k x_{k-1}\| = \|x_{k-1}\|$.
- Norm of a map: $\|x\|^2 = \int_{\Omega} \|x(u)\|^2 du < \infty$ for $x$ in $L^2(\Omega, \mathcal{H})$. 

$P_k x_{k-1}(v) \in P_k$ (patch extraction)
Non-linear pointwise 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 pointwise mapping operator $M_k$

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

- $\varphi_k : \mathcal{P}_k \rightarrow \mathcal{H}_k$ pointwise non-linearity on patches.
- We assume non-expansivity

\[ \|\varphi_k(z)\| \leq \|z\| \quad \text{and} \quad \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\|. \]

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

\[ \|M_k x\| \leq \|x\| \quad \text{and} \quad \|M_k x - M_k x'\| \leq \|x - x'\|. \]
\( \varphi_k \) from kernels

- Kernel mapping of \textbf{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 \( b_j \geq 0, \kappa_k(1) = 1 \).

- \( \|\varphi_k(z)\| = K_k(z, z)^{1/2} = \|z\| \) \textbf{(norm preservation)}.

- \( \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\| \) if \( \kappa'_k(1) \leq 1 \) \textbf{(non-expansiveness)}.
\( \varphi_k \) 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 \( b_j \geq 0 \), \( \kappa_k(1) = 1 \).

- \( \|\varphi_k(z)\| = K_k(z, z)^{1/2} = \|z\| \) (norm preservation).

- \( \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\| \) if \( \kappa'_k(1) \leq 1 \) (non-expansiveness).

**Examples**

- \( \kappa_{\exp}(\langle z, z' \rangle) = e^{\langle z, z' \rangle} - 1 = e^{-\frac{1}{2}\|z-z'\|^2} \) (if \( \|z\| = \|z'\| = 1 \)).

- \( \kappa_{\text{inv}-\text{poly}}(\langle z, z' \rangle) = \frac{1}{2 - \langle z, z' \rangle} \).

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 $A_k$

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

- $h_{\sigma_k}$: pooling filter at scale $\sigma_k$.
- $h_{\sigma_k}(u) := \sigma_k^{-d} h(u/\sigma_k)$ with $h(u)$ \textbf{Gaussian}.
- \textbf{Linear, non-expansive operator}: $\|A_k\| \leq 1$ (operator norm).
Recap: $P_k, M_k, A_k$

\[ x_k := A_k M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k \]

Kernel mapping

\[ M_k P_k x_{k-1} : \Omega \rightarrow \mathcal{H}_k \]

Linear pooling

\[ x_{k-1}(u) \in \mathcal{H}_{k-1} \]

\[ x_{k-1} : \Omega \rightarrow \mathcal{H}_{k-1} \]

\[ P_k x_{k-1}(v) \in \mathcal{P}_k \text{ (patch extraction)} \]

\[ M_k P_k x_{k-1}(v) = \varphi_k(P_k x_{k-1}(v)) \in \mathcal{H}_k \]

\[ x_k(w) = A_k M_k P_k x_{k-1}(w) \in \mathcal{H}_k \]
Invariance, definitions

- $\tau : \Omega \rightarrow \Omega$: $C^1$-diffeomorphism with $\Omega = \mathbb{R}^d$.
- $L_{\tau}x(u) = x(u - \tau(u))$: action operator.
- Much richer group of transformations than translations.

[Invariance to Translations]

Two dimensional group: $\mathbb{R}^2$

Translations and Deformations

- Patterns are translated and deformed

Invariance, definitions

- \( \tau : \Omega \rightarrow \Omega \): \( C^1 \)-diffeomorphism with \( \Omega = \mathbb{R}^d \).
- \( L_\tau x(u) = x(u - \tau(u)) \): 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.
- \( C_2 \rightarrow 0 \): translation invariance.

Warmup: translation invariance

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve translation invariance?

- Translation: \( L_c x(u) = x(u - c) \).
Warmup: translation invariance

Representation

$$\Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x.$$ 

How to achieve translation invariance?

- Translation: $L_c x(u) = x(u - c)$.
- Equivariance - all operators commute with $L_c$: $\square L_c = L_c \square$.

\[
\| \Phi_n(L_c x) - \Phi_n(x) \| = \| L_c \Phi_n(x) - \Phi_n(x) \|
\leq \| L_c A_n - A_n \| \cdot \| M_n P_n \Phi_{n-1}(x) \|
\leq \| L_c A_n - A_n \| \| x \|.
\]
Warmup: translation invariance

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve translation invariance?

- Translation: \( L_c x(u) = x(u - c) \).

- Equivariance - all operators commute with \( L_c \): \( \square L_c = L_c \square \).

\[
\| \Phi_n(L_c x) - \Phi_n(x) \| = \| L_c \Phi_n(x) - \Phi_n(x) \|
\leq \| L_c A_n - A_n \| \cdot \| M_n P_n \Phi_{n-1}(x) \|
\leq \| L_c A_n - A_n \| \| x \|.
\]

- Mallat [2012]: \( \| L_\tau A_n - A_n \| \leq \frac{C_2}{\sigma_n} \| \tau \|_\infty \) (operator norm).
Warmup: translation invariance

Representation

$$\Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x.$$ 

How to achieve translation invariance?

- Translation: $$L_c x(u) = x(u - c).$$
- Equivariance - all operators commute with $$L_c$$: $$\square L_c = L_c \square$$.

$$\| \Phi_n(L_c x) - \Phi_n(x) \| = \| L_c \Phi_n(x) - \Phi_n(x) \|$$
$$\leq \| L_c A_n - A_n \| \cdot \| M_n P_n \Phi_{n-1}(x) \|$$
$$\leq \| L_c A_n - A_n \| \| x \|.$$

- Mallat [2012]: $$\| L_c A_n - A_n \| \leq \frac{C_2}{\sigma_n} c$$ (operator norm).
- Scale $$\sigma_n$$ of the last layer controls translation invariance.
Stability to deformations

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve stability to deformations?

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!
Stability to deformations

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve stability to deformations?

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!
- \[ \| A_k L_\tau - L_\tau A_k \| \leq C_1 \| \nabla \tau \|_{\infty} \] [from Mallat, 2012].
Stability to deformations

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve stability to deformations?

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!
- \( \|[A_k, L_\tau]\| \leq C_1 \|\nabla \tau\|_\infty \) [from Mallat, 2012].
Stability to deformations

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve stability to deformations?

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!
- \( \|[A_k, L_\tau]\| \leq C_1 \|\nabla \tau\|_\infty \) [from Mallat, 2012].
- But: \( [P_k, L_\tau] \) is unstable at high frequencies!
Stability to deformations

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve stability to deformations?

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!
- \( \|[A_k, L_\tau]\| \leq C_1 \|\nabla \tau\|_\infty \) [from Mallat, 2012].
- But: \([P_k, L_\tau]\) is unstable at high frequencies!
- Adapt to current layer resolution, patch size controlled by \( \sigma_{k-1} \):

\[
\|[P_k A_{k-1}, L_\tau]\| \leq C_1,\kappa \|\nabla \tau\|_\infty \quad \sup_{u \in S_k} |u| \leq \kappa \sigma_{k-1}
\]
Stability to deformations

Representation

\[ \Phi_n(x) \triangleq A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x. \]

How to achieve stability to deformations?

- Patch extraction \( P_k \) and pooling \( A_k \) do not commute with \( L_\tau \)!
- \( \|[A_k, L_\tau]\| \leq C_1 \|
\n\n\ngrad\tau\|_\infty \) [from Mallat, 2012].
- But: \([P_k, L_\tau]\) is unstable at high frequencies!
- Adapt to current layer resolution, patch size controlled by \( \sigma_{k-1} \):  

\[ \|[P_k A_{k-1}, L_\tau]\| \leq C_{1,\kappa} \|
\n\n\ngrad\tau\|_\infty \sup_{u \in S_k} |u| \leq \kappa \sigma_{k-1} \]

- \( C_{1,\kappa} \) grows as \( \kappa^{d+1} \implies \) more stable with small patches (e.g., 3x3, VGG et al.).
Stability to deformations: final result

Theorem

If \( \| \nabla \tau \|_{\infty} \leq 1/2 \),

\[
\| \Phi_n(L_\tau x) - \Phi_n(x) \| \leq \left( C_{1,\kappa} (n + 1) \| \nabla \tau \|_{\infty} + \frac{C_2}{\sigma_n} \| \tau \|_{\infty} \right) \| x \|.
\]

- translation invariance: large \( \sigma_n \).
- stability: small patch sizes.
- signal preservation: subsampling factor \( \approx \) patch size.
- \( \implies \text{needs several layers} \).

related work on stability [Wiatowski and Bölcskei, 2017]
Stability to deformations: final result

Theorem

If $\|\nabla \tau\|_{\infty} \leq 1/2,$

$$\|\Phi_n(L_{\tau}x) - \Phi_n(x)\| \leq \left( C_{1,\kappa} (n + 1) \|\nabla \tau\|_{\infty} + \frac{C_2}{\sigma_n} \|\tau\|_{\infty} \right) \|x\|.$$  

- translation invariance: large $\sigma_n$.  
- stability: small patch sizes.  
- signal preservation: subsampling factor $\approx$ patch size.  
- $\implies$ needs several layers.  
- requires additional discussion to make stability non-trivial.  

related work on stability [Wiatowski and Bölcskei, 2017]
Beyond the translation group

Can we achieve invariance to other groups?

- Group action: \( L_g x(u) = x(g^{-1}u) \) (e.g., rotations, reflections).
- Feature maps \( x(u) \) defined on \( u \in G \) (\( G \): locally compact group).
Beyond the translation group

Can we achieve invariance to other groups?

- Group action: $L_g x(u) = x(g^{-1}u)$ (e.g., rotations, reflections).
- Feature maps $x(u)$ defined on $u \in G$ ($G$: locally compact group).

Recipe: Equivariant inner layers + global pooling in last layer

- **Patch extraction:**
  
  \[ Px(u) = (x(uv))_{v \in S}. \]

- **Non-linear mapping:** equivariant because pointwise!
- **Pooling** ($\mu$: left-invariant Haar measure):
  
  \[ Ax(u) = \int_G x(uv)h(v)d\mu(v) = \int_G x(v)h(u^{-1}v)d\mu(v). \]

related work [Sifre and Mallat, 2013, Cohen and Welling, 2016, Raj et al., 2016]...
Group invariance and stability

Previous construction is similar to Cohen and Welling [2016] for CNNs.

A case of interest: the roto-translation group

- $G = \mathbb{R}^2 \rtimes SO(2)$ (mix of translations and rotations).
- **Stability** with respect to the translation group.
- **Global invariance** to rotations (only global pooling at final layer).
  - Inner layers: only pool on translation group.
  - Last layer: global pooling on rotations.
  - Cohen and Welling [2016]: pooling on rotations in inner layers hurts performance on Rotated MNIST
Discretization and signal preservation: example in 1D

- Discrete signal $\bar{x}_k$ in $\ell^2(\mathbb{Z}, \mathcal{H}_k)$ vs continuous ones $x_k$ in $L^2(\mathbb{R}, \mathcal{H}_k)$.
- $\bar{x}_k$: subsampling factor $s_k$ after pooling with scale $\sigma_k \approx s_k$:

$$\bar{x}_k[n] = \bar{A}_k \bar{M}_k \bar{P}_k \bar{x}_{k-1}[ns_k].$$

Warning: no claim that recovery is practical and/or stable.
Discretization and signal preservation: example in 1D

- Discrete signal $\bar{x}_k$ in $\ell^2(\mathbb{Z}, \mathcal{H}_k)$ vs continuous ones $x_k$ in $L^2(\mathbb{R}, \mathcal{H}_k)$.

- $\bar{x}_k$: subsampling factor $s_k$ after pooling with scale $\sigma_k \approx s_k$:
  \[
  \bar{x}_k[n] = \bar{A}_k \bar{M}_k \bar{P}_k \bar{x}_{k-1}[ns_k].
  \]

- **Claim**: We can recover $\bar{x}_{k-1}$ from $\bar{x}_k$ if factor $s_k \leq \text{patch size}$. 
Discretization and signal preservation: example in 1D

- Discrete signal $\bar{x}_k$ in $\ell^2(\mathbb{Z}, \mathcal{H}_k)$ vs continuous ones $x_k$ in $L^2(\mathbb{R}, \mathcal{H}_k)$.
- $\bar{x}_k$: subsampling factor $s_k$ after pooling with scale $\sigma_k \approx s_k$:
  
  $$\bar{x}_k[n] = \bar{A}_k \bar{M}_k \bar{P}_k \bar{x}_{k-1}[ns_k].$$

- **Claim**: We can recover $\bar{x}_{k-1}$ from $\bar{x}_k$ if factor $s_k \leq$ patch size.
- **How?** Recover patches with **linear functions** (contained in $\mathcal{H}_k$)
  
  $$\langle f_w, \bar{M}_k \bar{P}_k \bar{x}_{k-1}(u) \rangle = f_w(\bar{P}_k \bar{x}_{k-1}(u)) = \langle w, \bar{P}_k \bar{x}_{k-1}(u) \rangle,$$

  and
  
  $$\bar{P}_k \bar{x}_{k-1}(u) = \sum_{w \in B} \langle f_w, \bar{M}_k \bar{P}_k \bar{x}_{k-1}(u) \rangle w.$$
Discretization and signal preservation: example in 1D

- Discrete signal $\bar{x}_k$ in $\ell^2(\mathbb{Z}, \mathcal{H}_k)$ vs continuous ones $x_k$ in $L^2(\mathbb{R}, \mathcal{H}_k)$.
- $\bar{x}_k$: subsampling factor $s_k$ after pooling with scale $\sigma_k \approx s_k$:
  \[
  \bar{x}_k[n] = \bar{A}_k \bar{M}_k \bar{P}_k \bar{x}_{k-1}[ns_k].
  \]

- **Claim**: We can recover $\bar{x}_{k-1}$ from $\bar{x}_k$ if factor $s_k \leq$ patch size.
- **How?** Recover patches with **linear functions** (contained in $\mathcal{H}_k$)
  \[
  \langle f_w, \bar{M}_k \bar{P}_k \bar{x}_{k-1}(u) \rangle = f_w(\bar{P}_k \bar{x}_{k-1}(u)) = \langle w, \bar{P}_k \bar{x}_{k-1}(u) \rangle,
  \]
  and
  \[
  \bar{P}_k \bar{x}_{k-1}(u) = \sum_{w \in B} \langle f_w, \bar{M}_k \bar{P}_k \bar{x}_{k-1}(u) \rangle w.
  \]

**Warning**: no claim that recovery is practical and/or stable.
Discretization and signal preservation: example in 1D

\[ \bar{x}_{k-1} \]

deconvolution

\[ \bar{A}_k \bar{x}_{k-1} \]

recovery with linear measurements

\[ \bar{x}_k \]

downsampling

\[ \bar{A}_k \bar{M}_k \bar{P}_k \bar{x}_{k-1} \]

linear pooling

\[ \bar{M}_k \bar{P}_k \bar{x}_{k-1} \]

dot-product kernel

\[ \bar{x}_{k-1} \]

\[ \bar{P}_k \bar{x}_{k-1}(u) \in \mathcal{P}_k \]
RKHS of patch kernels $K_k$

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

What does the RKHS contain?

Homogeneous version of [Zhang et al., 2016, 2017]
RKHS of patch kernels $K_k$

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

What does the RKHS contain?

- RKHS contains homogeneous functions:

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

Homogeneous version of [Zhang et al., 2016, 2017]
RKHS of patch kernels $K_k$

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

What does the RKHS contain?

- RKHS contains **homogeneous functions**:

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

- **Smooth activations**: $\sigma(u) = \sum_{j=0}^{\infty} a_j u^j$ with $a_j \geq 0$.

- **Norm**: $\|f\|_{H_k}^2 \leq C_\sigma^2(\|g\|^2) = \sum_{j=0}^{\infty} \frac{a_j^2}{b_j} \|g\|^2 < \infty$.

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

![Function graphs](image-url)
Constructing a CNN in the RKHS $\mathcal{H}_K$

Some CNNs live in the RKHS: “linearization” principle

$$f(x) = \sigma_k(W_k\sigma_{k-1}(W_{k-1} \ldots \sigma_2(W_2\sigma_1(W_1x)) \ldots)) = \langle f, \Phi(x) \rangle_\mathcal{H}.$$
Constructing a CNN in the RKHS $\mathcal{H}_K$

Some CNNs live in the RKHS: “linearization” principle

$$f(x) = \sigma_k(W_k \sigma_{k-1}(W_{k-1} \ldots \sigma_2(W_2 \sigma_1(W_1 x)) \ldots)) = \langle f, \Phi(x) \rangle_{\mathcal{H}}.$$ 

- Consider a CNN with filters $W_k^{ij}(u), u \in S_k$.
  - $k$: layer;
  - $i$: index of filter;
  - $j$: index of input channel.
- “Smooth homogeneous” activations $\sigma$.
- The CNN can be constructed hierarchically in $\mathcal{H}_K$.
- Norm (linear layers):
  $$\|f_\sigma\|^2 \leq \|W_{n+1}\|_2^2 \cdot \|W_n\|_2^2 \cdot \|W_{n-1}\|_2^2 \ldots \|W_1\|_2^2.$$ 
- Linear layers: product of spectral norms.
Link with generalization

Direct application of classical generalization bounds

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

\[ F_B = \{ f \in \mathcal{H}_K, \| f \| \leq B \} \implies \text{Rad}_N(F_B) \leq O\left( \frac{BR}{\sqrt{N}} \right). \]
Link with generalization

Direct application of classical generalization bounds

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

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

- Leads to margin bound \( O(\| \hat{f}_N \| R / \gamma \sqrt{N}) \) for a learned CNN \( \hat{f}_N \) with margin (confidence) \( \gamma > 0 \).

- Related to recent generalization bounds for neural networks based on **product of spectral norms** [e.g., Bartlett et al., 2017, Neyshabur et al., 2018].

[see, e.g., Boucheron et al., 2005, Shalev-Shwartz and Ben-David, 2014]...
Conclusions from the work on invariance and stability

Study of generic properties of signal representation

- **Deformation stability** with small patches, adapted to resolution.
- **Signal preservation** when subsampling $\leq$ patch size.
- **Group invariance** by changing patch extraction and pooling.

"higher capacity" is needed to discriminate small deformations.

Questions:
- How does SGD control capacity in CNNs?
- What about networks with no pooling layers? ResNet?
Conclusions from the work on invariance and stability

Study of generic properties of signal representation

- **Deformation stability** with small patches, adapted to resolution.
- **Signal preservation** when subsampling $\leq$ patch size.
- **Group invariance** by changing patch extraction and pooling.

Applies to learned models

- Same quantity $\|f\|$ controls stability and generalization.
- “higher capacity” is needed to discriminate small deformations.
Conclusions from the work on invariance and stability

Study of generic properties of signal representation

- **Deformation stability** with small patches, adapted to resolution.
- **Signal preservation** when subsampling $\leq$ patch size.
- **Group invariance** by changing patch extraction and pooling.

Applies to learned models

- Same quantity $\|f\|$ controls stability and generalization.
- “higher capacity” is needed to discriminate small deformations.

Questions:

- How does SGD control capacity in CNNs?
- What about networks with no pooling layers? ResNet?


References III


References V


References VI


References VII


Approximate $\varphi_k(z)$ by projection (Nyström approximation) on $\mathcal{F} = \text{Span}(\varphi_k(z_1), \ldots, \varphi_k(z_p))$. 

Figure: Nyström approximation.

[Williams and Seeger, 2001, Smola and Schölkopf, 2000, Zhang et al., 2008]...
Approximate $\varphi_k(z)$ by projection (Nyström approximation) on 

$$\mathcal{F} = \text{Span}(\varphi_k(z_1), \ldots, \varphi_k(z_p)).$$

Leads to **tractable**, $p$-dimensional representation $\psi_k(z)$.

Norm is preserved, and projection is non-expansive:

$$\|\psi_k(z) - \psi_k(z')\| = \|\Pi_k \varphi_k(z) - \Pi_k \varphi_k(z')\|$$

$$\leq \|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\|.$$

Anchor points $z_1, \ldots, z_p$ ($\approx$ filters) can be learned from data (K-means or backprop).

[Williams and Seeger, 2001, Smola and Schölkopf, 2000, Zhang et al., 2008]...
Convolutional kernel networks in practice.

\[ \varphi_k \] from kernel approximations: CKNs [Mairal, 2016]
Discussion

- norm of $\|\Phi(x)\|$ is of the same order (or close enough) to $\|x\|$.
- the kernel representation is non-expansive but not contractive

$$\sup_{x,x' \in L^2(\Omega,\mathcal{H}_0)} \frac{\|\Phi(x) - \Phi(x')\|}{\|x - x'\|} = 1.$$