Hierarchical Learning Machines: Derived Kernels and the Neural Response

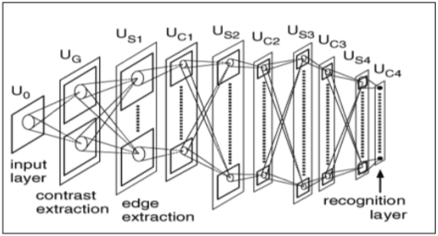
Andre Wibisono
Lorenzo Rosasco

MIT 9.520 Class 16

April 5, 2010

Hierarchical/Deep Learning

Neocognitron, from Fukushima et al., 1980



Convolutional Neural Networks (LeCun)

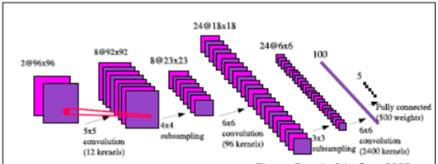


Figure: Bengio & LeCun, 2007

CBCL Model

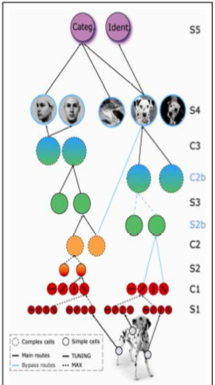
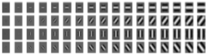
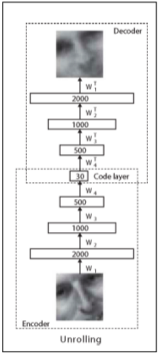


Figure: T. Serre



Hinton's Deep Autoencoder



from: G. Hinton, Science 2007.

About this class

Goal: to introduce a mathematical counterpart to the visual cortex model described in the previous two lectures.

- Describe a recursive definition of a similarity kernel.
- Describe theoretical analyses.

S. Smale, L. Rosasco, J. Bouvrie, A. Caponnetto, and T. Poggio. "Mathematics of the Neural Response", Foundations of Computational Mathematics (2010) 10: 67–91

and

J. Bouvrie, T. Poggio, L. Rosasco, S. Smale. A. Wibisono. "Properties of Hierarchical Learning Machines", in preparation.

Plan

1. **Background**
2. Derived Kernels and the Neural Response
3. Connection to Neuroscience
4. **Extensions**
5. Theoretical Analysis

Biologically Inspired Hierarhical Learning Machines

- **Human-Machine Comparison**: Chomsky's poverty of the stimulus argument: biological organisms can learn complex concepts and tasks from extraordinarily small empirical samples.
- *Hierarchical organization is the key?* circuits found in the human brain facilitate robust learning from few examples via the discovery of *invariances*, while promoting circuit modularity and reuse of redundant sub-circuits, leading also to greater energy and space efficiency.

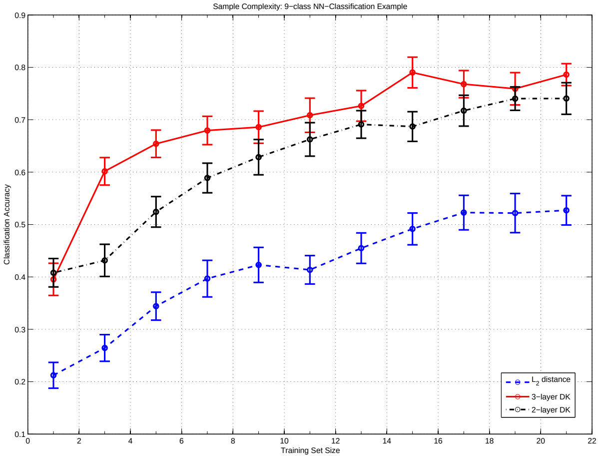
Why Hierarchical Learning Machines?

When and why is a hierarchical architecture preferred?

1. Invariance versus selectivity.
2. Computational properties.
3. Adaptive tuning.
4. Sample complexity.

For tasks that can be decomposed into a hierarchy of parts, how can we show that a supervised classifier trained using a hierarchical feature map will generalize better than an off-the-shelf non-hierarchical alternative?

Hierarchica Learning: Empirical Motivation

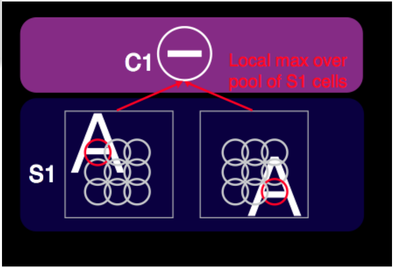


9-class digits problem, nearest neighbor classifier, Euclidean distance vs. 3-layer derived distance ($u = 12$, $v = 20$, 500 templates/layer, 3-pixel image translations).

Plan

Towards a Theory

Unit types	Pooling	Computation	Operation
Simple		Selectivity / template matching	Gaussian-tuning / and-like
Complex		Invariance	Soft-max / or-like



We will borrow concepts and operations underlying the visual cortex model.

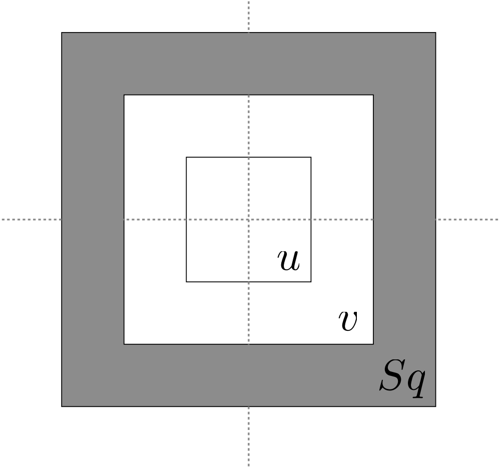
Defining a model

The ingredients needed to define the derived kernel consist of:

- A finite *architecture* of nested domains. We'll call them patches.
- A suitable family of *function spaces* defined on each patch.
- A set of *transformations* defined on patches.
- A set of *templates* which connect the mathematical model to a real world setting.

An Architecture of Patches

We first consider an architecture composed of *three* layers of patches: u, v and Sq in $\mathbb{R}^2$, with $u \subset v \subset Sq$,



Images as Functions

We consider a function space on Sq, denoted by

$$\mathrm{Im}(Sq) = \{f : Sq \to [0,1]\},$$

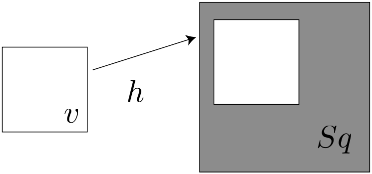
as well as the function spaces $\mathrm{Im}(u)$, $\mathrm{Im}(v)$ defined on subpatches u, v, respectively.

Functions can be interpreted as grey scale images when working with a vision problem for example.

Transformations

Next, we assume a set H_u of *transformations* that are maps from the smallest patch to the next larger patch

$$h : u \to v.$$



Similarly H_v with $h : v \to Sq$.

The sets of transformations are assumed to be finite.

These transformations act on the *domain* of a function (image).

Examples

Examples of transformations are primarily translations, but also scalings and rotations...

Translations and Scalings

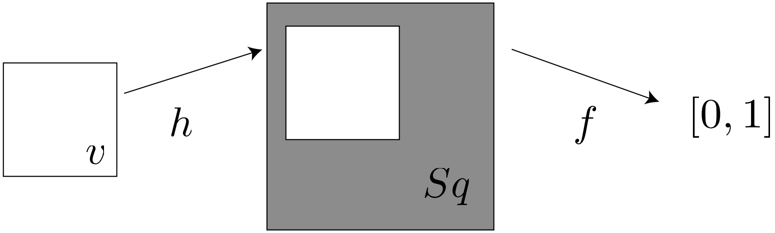
we have transformations of the form $h = h_\beta h_\alpha$ with

$$h_\alpha(x) = \alpha x, \text{ and } h_\beta(x') = x' + \beta,$$

where $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}^2$ is such that $h_\beta h_\alpha(u) \subset v$.

Interpretation

In the vision interpretation, a translation h can be thought of as moving the image over the "receptive field" v



$$f \circ h : v \to [0, 1]$$

Figure: *A transformation "restricts" an image to a specific patch.*

Templates

Template sets are finite,
$T_u \subset \mathrm{Im}(u)$ and $T_v \subset \mathrm{Im}(v)$

- they are image patches sampled from some set of unlabeled images.
- link the mathematical development to real world problems.

The space of images can be endowed with a "mother" probability measure ρ. Templates can be seen as images frequently encountered in the early stages of life.

Reproducing Kernel

Given a set X, a function $K : X \times X \to \mathbb{R}$ is a reproducing kernel if it is a symmetric and positive definite kernel, i.e.

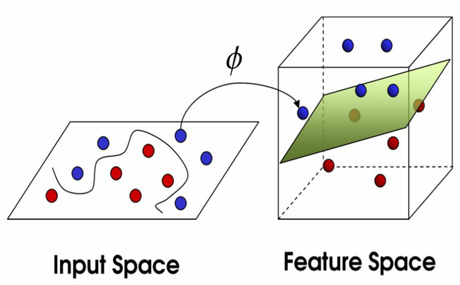
$$\sum_{i,j=1}^{n} \alpha_i \alpha_j K(x_i, x_j) \geq 0,$$

for any $n \in \mathbb{N}$, $x_1, \ldots, x_n \in X$ and $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$.

Dot Products and Feature map

Consider a feature map:

$$\Phi : X \to \mathcal{F}$$



Inner product kernels are an instance of reproducing kernels:

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

is a reproducing kernel.

Normalization

We assume $K(x,x) \neq 0$ for all $x \in X$ and let

$$\widehat{K}(x,x') = \frac{K(x,x')}{\sqrt{K(x,x)K(x',x')}}.$$

Clearly $\widehat{K}$ is a reproducing kernel and $\widehat{K}(x,x) \equiv 1$ for all $x \in X$.

- Allows interpretation of and comparison between different instances.
- Is nice for correspondence with a distance.

On the normalization

To make sense of the normalization we rule out the functions such that $K(f, f)$ is zero.

This assumption is quite natural in the context of images:

If $K(f, f)$ is zero, the responses of f is identically zero at *all* possible templates by definition:

"one can't see the contents of the image".

Derived Kernel and Neural Response

Construction

We'll give a bottom-up description of a three layer architecture before giving the general recursive definition.

Initial Kernel

Consider a *normalized* non-negative valued reproducing kernel on $\mathrm{Im}(u) \times \mathrm{Im}(u)$ denoted by $\widehat{K}_u(f, g)$.

example

Consider the inner product of square integrable functions on u

$$K_u(f, g) = \int_u f(x)g(x)dx.$$

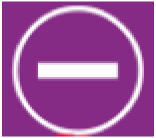
DEFINITION: Neural Response

We define the *neural response* of f at t:

$$N_v(f)(t) = \max_{h \in H} \widehat{K}_u(f \circ h, t),$$

where $f \in \mathrm{Im}(v)$, $t \in T_u$ and $H = H_u$.

NOTE: f is not the whole image here.



Neural Response (cont.)

By denoting with $D = |T_u|$ the cardinality of the template set T_u, we can interpret the neural response as a vector in $\mathbb{R}^D$,

$$f \in \mathrm{Im}(v) \longmapsto (N_v(f)(t_1), N_v(f)(t_2), \ldots, N_v(f)(t_D)).$$

This is just the collection of best responses of each template within the *sub-patch* $f \in \mathrm{Im}(v)$.

If K_u is the Euclidean dot-product, and H_u is all translations: compare to normalized cross-correlation.

Derived Kernel

The *derived kernel* is just the corresponding inner product in $L^2(T_u) = \mathbb{R}^{|T_u|}$ between neural responses, normalized by $\frac{1}{|T_u|}$

The derived kernel on $\mathrm{Im}(v) \times \mathrm{Im}(v)$ is defined as

$$K_v(f, g) = \langle N_v(f), N_v(g) \rangle_{L^2(T_u)},$$

and can be normalized to obtain the kernel $\widehat{K}_v$.

This is the correlation in the pattern of similarities to templates.

Second Layer

We now repeat the process:

second layer neural response

$$N_{Sq}(f)(t) = \max_{h \in H} \widehat{K}_v(f \circ h, t),$$

where $f \in \mathrm{Im}(Sq), t \in T_v$ and $H = H_v$.

derived kernel on $\mathrm{Im}(Sq) \times \mathrm{Im}(Sq)$

$$K_{Sq}(f, g) = \langle N_{Sq}(f), N_{Sq}(g) \rangle_{L^2(T_v)},$$

where $\langle \cdot, \cdot \rangle_{L^2(T_v)}$ is the L^2 inner product.

As before, we normalize K_{S_q} to obtain the final derived kernel $\widehat{K}_{S_q}$.

Recursive Definition

For a general n layer architecture $v_1 \subset v_2 \subset \cdots \subset v_n = Sq$, let $K_n = K_{v_n}$ and $H_n = H_{v_n}$, $T_n = T_{v_n}$.

Definition

Given a non-negative valued, normalized, reproducing kernel $\widehat{K}_1$, the m-layer derived kernel $\widehat{K}_m$, $m = 2, \ldots, n$, is obtained by normalizing

$$K_m(f, g) = \langle N_m(f), N_m(g) \rangle_{L^2(T_{m-1})}$$

where

$$N_m(f)(t) = \max_{h \in H} \widehat{K}_{m-1}(f \circ h, t), \qquad t \in T_{m-1}$$

with $H = H_{m-1}$.

The normalized neural response provides a *representation* for any function $f \in \mathrm{Im}(Sq)$.

$$\underbrace{f \in \mathrm{Im}(Sq)}_{\text{input}} \longmapsto \underbrace{\widehat{N}_{Sq}(f) \in L^2(T) = \mathbb{R}^{|T|}}_{\text{output}},$$

with $T = T_{n-1}$.

The normalization for N is that implied by the normalization of K:

$$\widehat{N}(f) = \frac{N(f)}{\|N(f)\|_{L^2(T)}}$$

where $\|x\|_{L^2(T)} = \sqrt{\langle x, x \rangle_{L^2(T)}} = \sqrt{\frac{1}{|T|}\langle x, x \rangle_{\mathbb{R}^{|T|}}}$.

Derived Distance

The derived kernel naturally defines a derived distance d on the space of images.

$$d^2(f,g) = \|\widehat{N}(f) - \widehat{N}(g)\|^2 = 2\big(1 - \widehat{K}(f,g)\big)$$

(since $\widehat{K}(f,f) = 1$ for all f)
Clearly, as the kernel "similarity" approaches its maximum value of 1, the distance goes to 0.

Plan

Neural Response vs. Simple and Complex Cells

The two key steps in the definition of neural response correspond to simple and complex cells in the visual cortex (and the CBCL model):

- S: inner products with the templates.
- C: $\max$ over the set of translations.

Simple Cells at the First Layer

Given an initial kernel K_u, let

$$N_{S1}(f \circ h)(t) = K_u(f \circ h, t)$$

with $f \in \mathrm{Im}(v)$, $h \in H_u$ and $t \in T_u$.

$N_{S1}(f \circ h)(t)$ corresponds to the response of an $S1$ cell with template t and receptive field $h \circ u$.

The operations underlying the definition of $S1$ can be thought of as "normalized convolutions".

Complex Cells at the First Layer

The neural response is given by

$$N_{C1}(f)(t) = \max_{h \in H}\{N_{S1}(f \circ h)(t)\}$$

with $f \in \mathrm{Im}(v)$, $H = H_u$ and $t \in T_u$ so that $N_{C1} : \mathrm{Im}(v) \to \mathbb{R}^{|T_u|}$.

$N_{C1}(f)(t)$ corresponds to the response of a $C1$ cell with template t and receptive field corresponding to v.

Plan

1. Background
2. Derived Kernels and the Neural Response
3. Connection to Neuroscience
4. **Extensions**
5. **Theoretical Analysis**

Tuning Functions

One can consider more general tuning functions, in fact any reproducing kernel $K\colon \ell^2 \times \ell^2 \to \mathbb{R}$,

$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{F}}$$

Gaussian Tuning

$$G(f, g) = e^{-\gamma d^2(f,g)},$$

where we used the (derived) distance.

Pooling Functions

One can consider more general pooling functions,

$$\Psi \colon \mathbb{R}^* = \bigcup_{n \in \mathbb{N}} \mathbb{R}^n \to \mathbb{R}.$$

Name	Expression
Average	$\Psi(\alpha(h)) = \frac{1}{\lvert H \rvert} \sum_{h \in H} \alpha(h)$
ℓ^1	$\Psi(\alpha(h)) = \sum_{h \in H} \lvert \alpha(h) \rvert$
Max	$\Psi(\alpha(h)) = \max_{h \in H} \alpha(h)$
ℓ^∞	$\Psi(\alpha(h)) = \max_{h \in H} \lvert \alpha(h) \rvert$

Templates

At some layer m, given $\Pi_m = (N_m(t_1), \ldots, N_m(t_D))$ we can consider more sophisticated templates learning schemes.

- Sparse Coding
- Non-negative matrix factorization
- Kernel PCA
- Diffusion Wavelets
- Laplacian Eigen-Maps

Many methods can be written as:

$$\|\Pi - PB\|^2 + \lambda \text{pen}(B, P)$$

General Definition

Definition (Neural Response & Derived Kernel)

Let $N_1\colon \mathrm{Im}(v_1) \to \ell_2$ be a feature map and $\mathcal{T}_1$ be a set of templates associated to $\Phi \circ N_1$.
Then

$$N_m(f)(\tau) = \Psi\big(\langle \Phi(N_{m-1}(f \circ h)), \tau\rangle\big),$$

with $f \in \mathrm{Im}(v_m)$, $\tau \in \mathcal{T}_{m-1}$, $H = H_{m-1}$ and $\mathcal{T}_{m-1}$ is a set of templates associated to $\Phi \circ N_{m-1}$. Moreover,

$$K_m(f, g) = K\big(N_m(f), N_m(g)\big)$$

Special Cases

- **CBCL Model**. Max pooling, Gaussian tuning (or normalized dot product). Templates are sampled patches.
- **Convolutional Neural Nets**. Pooling

$$\Psi = \ell^1 \circ \sigma,$$

  where ℓ^1 is the sum of the absolute values and σ is a sigmoid function. Tuning function, is typically a (normalized) inner product.
- **Neural Nets**. Take $v_1 = v_2 = \cdots = v_n$. Sigmoid Pooling functions. Tuning given by inner product.

Plan

1. Background
2. Derived Kernels and the Neural Response
3. Connection to Neuroscience
4. **Extensions**
5. **Theoretical Analysis**

Theoretical Analysis

Formulating the model in careful, mathematical terms was the first step towards a comprehensive theory.

Now we can start looking at invariance, discrimination, and other properties that emerge from our definitions:

1. Range compression
   - Loss of dynamic range
2. Invariance of the neural response
   - Global invariance from local invariance
3. Analysis in the one dimensional case
   - Characterization of equivalence classes of the derived kernel
   - Less exhaustive architecture is less discriminative

Range Compression: Empirical Observation

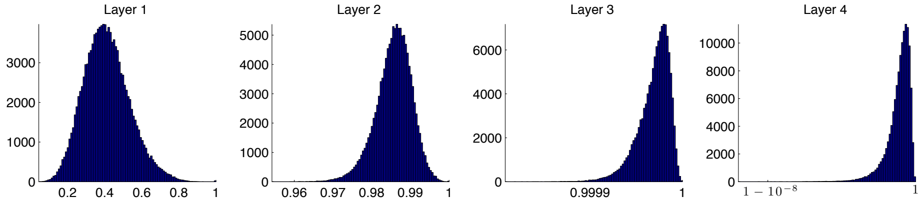
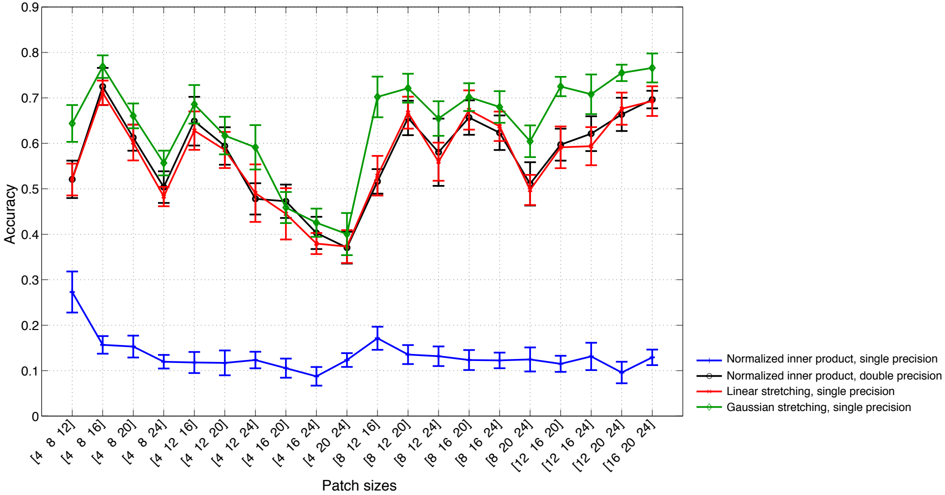


Figure: Sample distribution of $\widehat{K}_m(f,g)$ in m-layer architecture, for $1 \leq m \leq 4$. Note different scales at each plot.

- Loss of dynamic range at each layer.
- Creates problem with performance if architecture has ≥ 4 layers (with single precision) or ≥ 5 layers (with double precision).
- E.g. accuracy in 8-class MNIST dataset, single precision: 86% (3 layers) $\rightarrow$ 14% (4 layers).

Range Compression: Empirical Observation



Range Compression: Theoretical Result

Theorem

Consider the architecture with $\max$ pooling and normalized inner product kernel. If at layer $m \geq 1$ we have

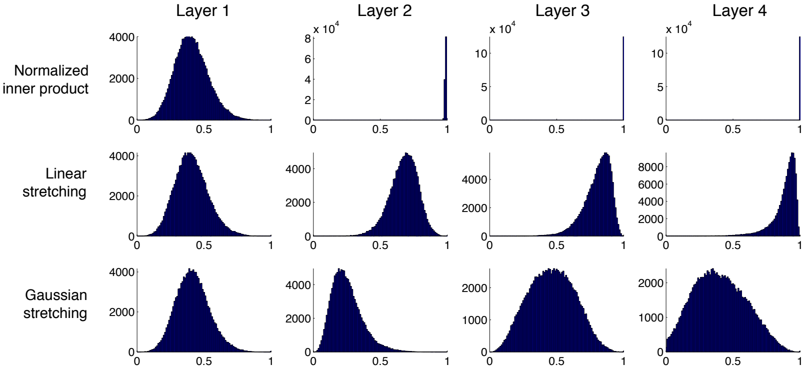
$$\widehat{K}_m(f,g) \geq a \quad \text{for all } f,g \in \mathrm{Im}(v_m),$$

then at layer $m+1$,

$$\widehat{K}_{m+1}(f,g) \geq \frac{2a}{1+a^2} \quad \text{for all } f,g \in \mathrm{Im}(v_{m+1}).$$

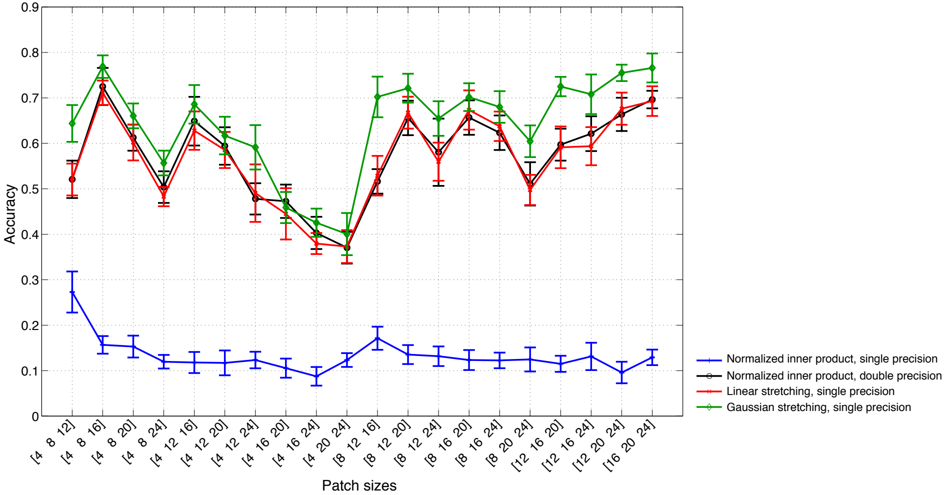
- Convergence of derived kernel and neural response as $m \to \infty$.
- Much higher rate of convergence in practice.
- Holds for more general architecture (e.g. average pooling function, ℓ_1-norm, ℓ_p-norm).
- Also holds when normalization occurs in the pooling. E.g. inner product kernel and $\ell_1 \circ \sigma$ pooling function.

Range Compression: Corrections



- Introduce tunable parameter at each layer to "stretch" the range of the derived kernel: $\widehat{K}_m \xrightarrow{\text{stretch}} \widetilde{K}_m$.
- Recover performance in higher layers (e.g. $14\% \rightarrow 72\%$).

Range Compression: Corrections



Theoretical Analysis

1. Range compression
   - Loss of dynamic range
2. Invariance of the neural response
   - Global invariance from local invariance
3. Analysis in the one dimensional case
   - Characterization of equivalence classes of the derived kernel
   - Less exhaustive architecture is less discriminative

Invariance of the Neural Response

- We can consider *invariance* of the (normalized) neural response with respect to some set of domain transformations $\mathcal{R} = \{r \mid r \colon v \to v\}$.
  (For example, in the case of vision, $\mathcal{R}$ can be the set of reflections or rotations.)
- We say that $\widehat{N}_m$ is invariant to $\mathcal{R}$ if

  $$\widehat{N}_m(f) = \widehat{N}_m(f \circ r)$$

  for every $f \in \mathrm{Im}(v_m)$ and $r \in \mathcal{R}$, or equivalently,

  $$\widehat{K}_m(f \circ r, f) = 1.$$

Invariance of the Neural Response

Assumption

For all $r \in \mathcal{R}$ and $h \in H$, there exists a unique $h' \in H$ such that

$$r \circ h = h' \circ r,$$

and there exists a unique $h'' \in H$ such that

$$h \circ r = r \circ h''.$$

Theorem

If the initial kernel satisfies $\widehat{K}_1(f, f \circ r) = 1$ for all $r \in \mathcal{R}$ and $f \in \mathrm{Im}(v_1)$, then at each layer $m \leq n$, we have

$$\widehat{K}_m(f, f \circ r) = 1$$

for all $r \in \mathcal{R}$, $f \in \mathrm{Im}(v_m)$.

Global invariance from local invariance!

Theoretical Analysis

1. Range compression
   - Loss of dynamic range
2. Invariance of the neural response
   - Global invariance from local invariance
3. Analysis in the one dimensional case
   - Characterization of equivalence classes of the derived kernel
   - Less exhaustive architecture is less discriminative

One Dimensional Strings

- An n-string is a function $f\colon \{1,\dots,k\} \to S$, where S is a set of finite alphabets, i.e. $f = a_1 \dots a_k \in S^k$.
- Patches are of the form $v_m = \{1,\dots,|v_m|\}, \quad |v_m| < |v_{m+1}|$.
- Function spaces are all possible strings, $\mathrm{Im}(v_m) = S^{|v_m|}$.
- H_m is the set of all possible translations $h\colon v_m \to v_{m+1}$.
- Use $\max$ pooling function and normalized inner product kernel, with all possible templates.
- Consider the initial kernel

$$\widehat{K}_1(f,g) = \frac{\#\{i \mid f(i) = g(i)\}}{|v_1|}.$$

Note that $\widehat{K}_1(f,g) = 1$ iff $f = g$.

One Dimensional Strings: Invariance

- Given $f = a_1 \dots a_k$, the reversal of f is $f \circ r = a_k \dots a_1$.
- $\widehat{K}_n$ is *reversal invariant* if $\widehat{K}_n(f, f \circ r) = 1$ for all $f \in \mathrm{Im}(v_n)$.

Theorem

Suppose $|S| \geq 2$. Then $\widehat{K}_n$ is reversal invariant if and only if $|v_1| = 1$.

Proof (sketch):

1. Local $\rightarrow$ global invariance.
2. Consider $f = abb \dots b$ and $g = bb \dots ba$, with $a \neq b$.

One Dimensional Strings: Discrimination

In a truly exhaustive architecture:

Theorem

Suppose $|v_m| = m$ for $1 \leq m \leq n$. Then $\widehat{K}_n(f, g) = 1$ if and only if:

1. $f = g$,
2. f is the reversal of g, or
3. f and g are the "checkerboard" pattern: $f = abab\ldots$, $g = baba\ldots$.

What if we start with larger initial patch size?

One Dimensional Strings: Discrimination

Theorem

Suppose $n \geq 2$, $|v_1| \geq 2$, and $|v_{m+1}| - |v_m| = 1$ for $1 \leq m \leq n-1$. Then $\widehat{K}_n(f, g) = 1$ if and only if:

1. $f = g$, or
2. f and g are the "checkerboard" pattern: $f = abab\ldots$, $g = baba\ldots$.

What if we allow "jumps" in patch sizes?

One Dimensional Strings: Discrimination

Theorem

Suppose $n \geq 2$, $|v_1| \geq 3$, and

$$\max_{1 \leq m \leq n-1} \big(|v_{m+1}| - |v_m|\big) = 2.$$

Then $\widehat{K}_n(f,g) = 1$ if and only if:

1. $f = g$,
2. $f = abab\ldots$ and $g = baba\ldots$,
3. $f = abcabc\ldots$ and $g = bcabca\ldots$, or
4. $f = abcabc\ldots$ and $g = cabcab\ldots$.

Less exhaustive architecture $\Rightarrow$ more invariance!

Summary

- We provided a compact mathematical description of a hierarchical model, based on recent feedforward models of the visual cortex.
- Preliminary theoretical and empirical analysis of the model.
- Theory is just at the beginning and many questions remain. For example:
  - Further invariance and discrimination analysis?
  - Can we show that more layers are better?
  - How can we learn the templates better?
  - Is the $\max$ operation really crucial (e.g. in terms of the performance)? Can it be replaced by some other operation (e.g. average)?
  - How to choose the parameters (number of layers, patch sizes, number of templates)?