Mad Max:
Affine Spline Insights into Deep Learning

Richard Baraniuk
Hype Cycle for Emerging Technologies, 2018

Source: Gartner (August 2018)
© 2018 Gartner, Inc. and/or its affiliates. All rights reserved.
When Humans Transcend Biology

The Singularity is Near

Ray Kurzweil

Author of the National Bestseller "The Age of Spiritual Machines"

Singularity Timeline

Rise in human intellect could be driven by integrating with machines in the future
**greek questions for the babylonians**

- Why is deep learning so **effective**?
- Can we derive deep learning systems from **first principles**?
- When and why does deep learning **fail**?
- How can deep learning systems be improved and extended in a **principled** fashion?
- Where is the **foundational framework** for theory?

See also Mallat, Soatto, Arora, Poggio, Tishby, [growing community] ...
splines and deep learning

R. Balestriero & B
“A Spline Theory of Deep Networks,” ICML 2018
“Mad Max: Affine Spline Insights into Deep Learning,” arxiv.org/abs/1805.06576, 2018
“From Hard to Soft: Understanding Deep Network Nonlinearities...,” ICLR 2019
“A Max-Affine Spline Perspective of RNNs,” ICLR 2019 (w/ J. Wang)
prediction problem

• Unknown **function/operator** $f$ mapping data to labels

$$\mathbf{y} = f(\mathbf{x})$$

**Goal:** Learn an **approximation** to $f$ using **training data**

$$\hat{y} = f_\Theta(\mathbf{x})$$

$$\{(x_i, y_i)\}_{i=1}^{n}$$
deep nets approximate

- Deep nets solve a **function approx** problem (black box)

\[ \hat{y} = f_\Theta(x) \]
Deep nets approximate

• Deep nets solve a function approx problem hierarchically

\[ \hat{y} = f_\Theta(x) = \left( f_{\theta(L)}^{(L)} \circ \cdots \circ f_{\theta(3)}^{(3)} \circ f_{\theta(2)}^{(2)} \circ f_{\theta(1)}^{(1)} \right)(x) \]
Deep nets and splines

- Deep nets solve a function approx problem hierarchically using a very special family of splines
deep nets and splines

Piecewise convexity of artificial neural networks
Blaine Rister\textsuperscript{a,*}, Daniel L. Rubin\textsuperscript{b}
\textsuperscript{a} Stanford University, Department of Electrical Engineering, 1201 Welch Rd, Stanford, CA, 94305, USA
\textsuperscript{b} Stanford University, Department of Radiology (Biomedical Informatics Research), 1201 Welch Rd Stanford, CA, 94305, USA

On the Number of Linear Regions of Deep Neural Networks

Guido Montúfar
Max Planck Institute for Mathematics in the Sciences
montufar@mis.mpg.de

Razvan Pascanu
Université de Montréal
pascanur@iro.umontreal.ca

Kyunghyun Cho
Université de Montréal
kyunghyun.cho@umontreal.ca

Yoshua Bengio
Université de Montréal, CIFAR Fellow
yoshua.bengio@umontreal.ca

A representer theorem for deep neural networks
Michael Unser
spline approximation

• A **spline** function approximation consists of
  - a *partition* $\Omega$ of the independent variable (input space)
  - a (simple) **local mapping** on each region of the partition
    (our focus: piecewise-affine mappings)
spline approximation

- A spline function approximation consists of
  - a **partition** $\Omega$ of the independent variable (input space)
  - a (simple) **local mapping** on each region of the partition

- **Powerful splines**
  - free, unconstrained partition $\Omega$ (ex: “free-knot” splines)
  - jointly optimize both the partition and local mappings
    (highly nonlinear, computationally intractable)

- **Easy splines**
  - fixed partition (ex: uniform grid, dyadic grid)
  - need only optimize the local mappings
max-affine spline (MAS)


• Consider **piecewise-affine approximation** of a convex function over $R$ regions

  - Affine functions: $a_r^T x + b_r, \quad r = 1, \ldots, R$
  
  - Convex approximation: $z(x) = \max_{r=1,\ldots,R} a_r^T x + b_r$
max-affine spline (MAS)


• **Key:** Any set of affine parameters \((a_r, b_r), \; r = 1, \ldots, R\) implicitly determines a spline partition

  - Affine functions: \(a_r^T x + b_r, \; r = 1, \ldots, R\)

  - Convex approximation: \(z(x) = \max_{r=1,\ldots,R} a_r^T x + b_r\)
Scale \( x \) by \( a \) + bias \( b \) \mid \text{ReLU:} \quad z(x) = \max(0, ax + b)

- Affine functions: \((a_1, b_1) = (0, 0), (a_2, b_2) = (a, b)\)

- Convex approximation: \( z(x) = \max_{r=1,2} a_r^T x + b_r \)
max-affine spline operator (MASO)

- **MAS** for $x \in \mathbb{R}^D$ has affine parameters $a_r \in \mathbb{R}^D, b_r \in \mathbb{R}$

- A **MASO** is simply a concatenation of $K$ MASs
modern deep nets

- Focus: The lion-share of today’s deep net architectures (convnets, resnets, skip-connection nets, inception nets, recurrent nets, …) employ **piecewise linear (affine) layers** (fully connected, conv; (leaky) ReLU, abs value; max/mean/channel-pooling)

\[
\hat{y} = f_{\Theta}(x) = \left( f^{(L)}_{\theta(L)} \circ \cdots \circ f^{(3)}_{\theta(3)} \circ f^{(2)}_{\theta(2)} \circ f^{(1)}_{\theta(1)} \right)(x)
\]
theorems

- Each deep net layer is a MASO
  - convex wrt each output dimension, piecewise-affine operator
Each deep net layer is a **MASO**
- **convex**, piecewise-affine operator

A deep net is a **composition of MASOs**
- **non-convex** piecewise-affine spline operator

\[ \text{WLOG ignore output softmax} \]
theorems

- A deep net is a **composition of MASOs**
  - **non-convex** piecewise-affine spline operator

- A deep net is a **convex MASO** iff the convolution/fully connected weights in all but the first layer are nonnegative and the intermediate nonlinearities are nondecreasing
MASO spline partition

- The parameters of each **deep net layer** (MASO) induce a **partition** of its input space with **convex regions**
  - vector quantization (info theory)
  - $k$-means (statistics)
  - Voronoi tiling (geometry)
MASO spline partition

- The $L$ layer-partitions of an $L$-layer deep net combine to form the **global input signal space partition**
  - affine spline operator
  - non-convex regions

Toy example: **3-layer “deep net”**
- Input $\mathbf{x}$: 2-D (4 classes)
- Fully connected | ReLU (45-D output)
- Fully connected | ReLU (3-D output)
- Fully connected | (softmax) (4-D output)
- Output $\mathbf{y}$: 4-D
The $L$ layer-partitions of an $L$-layer deep net combine to form the **global input signal space partition**
- affine spline operator
- non-convex regions

Toy example: 3-layer “deep net”
- **Input $x$:** 2-D (4 classes)
- Fully connected | ReLU (45-D output)
- Fully connected | ReLU (3-D output)
- Fully connected | (softmax) (4-D output)
- **Output $y$:** 4-D
MASO spline partition

- **Toy example: 3-layer “deep net”**
  - Input $x$: 2-D (4 classes)
  - Fully connected | ReLU (45-D output)
  - Fully connected | ReLU (3-D output)
  - Fully connected | (softmax) (4-D output)
  - Output $y$: 4-D

- **VQ partition of layer 1** depicted in the input space
  - **convex** regions
MASO spline partition

• Toy example: 3-layer “deep net”
  - Input $x$: 2-D (4 classes)
  - Fully connected | ReLU (45-D output)
  - Fully connected | ReLU (3-D output)
  - Fully connected | (softmax) (4-D output)
  - Output $y$: 4-D

• Given the partition region $Q(x)$ containing $x$ the layer input/output mapping is affine

$$z(x) = A_{Q(x)}x + b_{Q(x)}$$
MASO spline partition

- Toy example: 3-layer “deep net”
  - Input $x$: 2-D (4 classes)
  - Fully connected | ReLU (45-D output)
  - **Fully connected | ReLU** (3-D output)
  - Fully connected | (softmax) (4-D output)
  - Output $y$: 4-D

- VQ partition of **layer 2** depicted in the input space
  - **non-convex** regions due to visualization in the input space
MASO spline partition

- **Toy example:** 3-layer “deep net”
  - Input $\mathbf{x}$: 2-D (4 classes)
  - Fully connected | ReLU (45-D output)
  - **Fully connected | ReLU** (3-D output)
  - Fully connected | (softmax) (4-D output)
  - Output $\mathbf{y}$: 4-D

- Given the partition region $Q(\mathbf{x})$ containing $\mathbf{x}$ the layer input/output mapping is affine

$$\mathbf{z}(\mathbf{x}) = \mathbf{A}_{Q(\mathbf{x})}\mathbf{x} + \mathbf{b}_{Q(\mathbf{x})}$$
MASO spline partition

- Toy example: “Deep” net layer
  - Input $\mathbf{x}$: 2-D (4 classes)
  - Fully connected | ReLU (45-D output)
  - Fully connected | ReLU (3-D output)
  - Fully connected | (softmax) (4-D output)
  - Output $\mathbf{y}$: 4-D

- VQ partition of layers 1 & 2 depicted in the input space
  - non-convex regions
learning

layers 1 & 2

learning epochs (time)
local affine mapping – CNN

Example: Classical CNN architecture with conv/ReLU/max-pooling layers terminating in a linear classifier comprising one fully connected layer and softmax.

Result: Input ($x$) to output ($z_{CNN}^{(L)}(x)$) mapping is a region-dependent affine transform.

$$z_{CNN}^{(L)}(x) = A_Q(x) x + b_Q(x)[x]$$
local affine mapping – CNN

\[
\begin{align*}
z_{\text{CNN}}^{(L)}(x) &= \left( W^{(L)} \prod_{\ell=L-1}^{1} A_{\rho}^{(\ell)}(x) A_{\sigma}^{(\ell)}(x) C^{(\ell)} \right) A_{Q(x)} x \\
&+ \sum_{\ell=1}^{L-1} \left( \prod_{j=L-1}^{\ell+1} A_{\rho}^{(j)}(x) A_{\sigma}^{(j)}(x) C^{(j)} \right) \left( A_{\rho}^{(\ell)}(x) A_{\sigma}^{(\ell)}(x) b_{C}^{(\ell)} \right) + b_{W(L)} 
\end{align*}
\]

Fixed, different \( A_{Q}(x) \), \( b_{Q}(x) \) in each partition region
matched filters
deep nets are matched filterbanks

\[
\mathbf{z}^{(L)}(\mathbf{x}) = \mathbf{A}_{Q}(\mathbf{x}) \mathbf{x} + \mathbf{b}_{Q}(\mathbf{x})
\]

- Row \(c\) of \(\mathbf{A}_{Q}(\mathbf{x})\) is a vectorized signal/image corresponding to class \(c\)
- Entry \(c\) of deep net output = inner product between row \(c\) and signal
- For classification, select largest output; \textbf{matched filter}!
deep nets are matched filterbanks

Result  Row $c$ of $A_Q(x)$ is a matched filter for class $c$ that is applied to $x$; largest inner product wins

Visualization for CIFAR10: Row of $A_{\text{net}}[x]$, inner product with $x$

Input $x$  plane, 11.7  ship, 1.1  dog, -3.4

( Converted to black & white for ease of visualization)

Matched filter can be interpreted as being applied hierarchically thru the layers

Link with saliency maps [Simonyan et al., 2013; Zeiler & Fergus, 2014]
Result: Matched filters of an infinite capacity deep net **memorize the training data** \[ \{(x_n, y_n)\}_{n=1}^N \]

\[
\text{row } c \text{ of } A_Q(x_n) = \begin{cases} 
+ \sqrt{\frac{(C-1)\alpha}{C}} x_n, & c = y_n \text{ (correct class)} \\
- \sqrt{\frac{\alpha}{C(C-1)}} x_n, & c \neq y_n \text{ (incorrect class)}
\end{cases}
\]

Experiment with MNIST, CIFAR10

Inner products between training image \( x_n \) and rows of \( A_{\text{net}}[x_n] \)

- green: correct class (large positive)
- red: incorrect classes (large negative)
Matched filter classifier is optimal only for signal + white Gaussian noise (idealized).

For more general noise/nuisance models, useful to **orthogonalize** the matched filters.

**Result** Easy to do with any deep net thanks to the affine transformation formula; simply add to the cost function a **penalty on the off-diagonal entries of** $W^{(L)}(W^{(L)})^T$.

**Bonus:** Reduced overfitting
Capture the geometry of the data space by measuring the **distance between the partition regions** inhabited by two signals $x_1$ and $x_2$.

Use **Hamming distance** between the codewords $Q(x_1)$ and $Q(x_2)$.

Easily computed in terms of **activation patterns** of ReLU/max-pooling layers.

Links with distance between **vector quantization encodings**.
partition-based signal distance

15 nearest neighbors of a test image (upper left) using spline partition (VQ) distance
partition-based signal distance

15 nearest neighbors of a test image (upper left) using spline partition (VQ) distance

(a) Training with correct labels

(b) Training with random labels

(c) No Training
additional directions

- Study the **geometry** of deep nets and signals via VQ partition
- Affine input/output formula enables explicit calculation of the **Lipschitz constant** of a deep net for the analysis of stability, adversarial examples, ...
- Theory covers many **recurrent** neural networks (RNNs)
additional directions

- Theory extends to non-piecewise-affine operators (ex: sigmoid) by replacing the “hard VQ” of a MASO with a “soft VQ”
  - soft-VQ can generate new nonlinearities (ex: swish)
summary

• A wide range of deep nets solve function approximation problems using a composition of max-affine spline operators (MASOs)
  – links to vector quantization, $k$-means, Voronoi tiling

• Input/output deep net mapping is a VQ-dependent affine transform
  – enables explicit calculation of the Lipschitz constant of a deep net for the analysis of stability, adversarial examples, . . .

• Deep nets are (learned) matched filterbanks
  – new insights into dataset memorization

• Theory is constructive
  – inspires orthogonalized deep nets
  – new geometric distance via Hamming-VQ distance
max-affine splines and deep learning

R. Balestrieri & B
“A Spline Theory of Deep Networks,” ICML 2018
“Mad Max: Affine Spline Insights into Deep Learning,” arxiv.org/abs/1805.06576, 2018
“From Hard to Soft: Understanding Deep Network Nonlinearities...,” ICLR 2019
“A Max-Affine Spline Perspective of RNNs,” ICLR 2019 (w/ J. Wang)