The Mathematics of Deep Learning: Can We Open the Black Box of Deep Neural Networks?

Gitta Kutyniok
(Technische Universität Berlin)

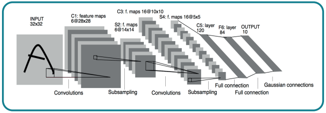
Virtual Conference on Mathematics of Data Science (MathODS)
June 11 – 12, 2020

Applied Functional Analysis Group
Technische Universität Berlin

The Dawn of Deep Learning

Self-Driving Cars

Surveillance



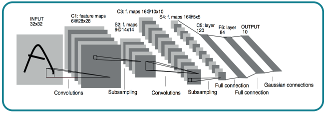
Legal Issues

Health Care

The Dawn of Deep Learning

Self-Driving Cars

Surveillance



Legal Issues

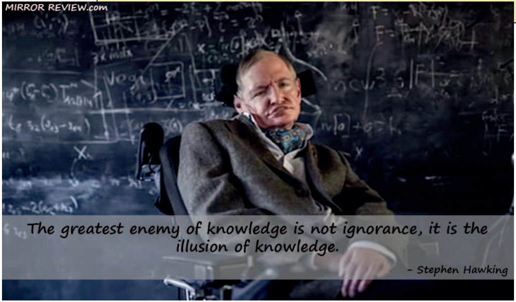
Health Care

Very few theoretical results explaining their success!

Deep Learning = Alchemy?

„Ali Rahimi, a researcher in artificial intelligence (AI) at Google in San Francisco, California, took a swipe at his field last December—and received a 40-second ovation for it. Speaking at an AI conference, Rahimi charged that **machine learning algorithms, in which computers learn through trial and error, have become a form of „alchemy."** Researchers, he said, **do not know why some algorithms work and others don't, nor do they have rigorous criteria for choosing one AI architecture over another...**"

Science, May 2018

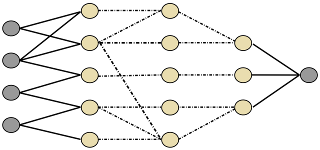


Theoretical Foundations of Deep Learning

The Mathematics of Deep Neural Networks

Definition:

Assume the following notions:



- $d \in \mathbb{N}$: Dimension of input layer.
- L: Number of layers.
- N: Number of neurons.
- $\rho : \mathbb{R} \to \mathbb{R}$: (Non-linear) function called *activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \ldots, L$: Affine linear maps.

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

$$\Phi(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*.

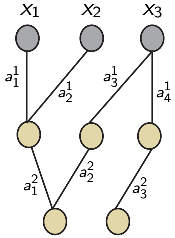
Affine Linear Maps and Weights

Remark: The affine linear map T_ℓ is defined by a matrix $A_\ell \in \mathbb{R}^{N_{\ell-1} \times N_\ell}$ and an affine part $b_\ell \in \mathbb{R}^{N_\ell}$ via

$$T_\ell(x) = A_\ell x + b_\ell.$$

$$A_1 = \begin{pmatrix} a_1^1 & a_2^1 & 0 \\ 0 & 0 & a_3^1 \\ 0 & 0 & a_4^1 \end{pmatrix}$$

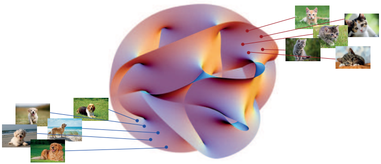
$$A_2 = \begin{pmatrix} a_1^2 & a_2^2 & 0 \\ 0 & 0 & a_3^2 \end{pmatrix}$$



Training of Deep Neural Networks

High-Level Set Up:

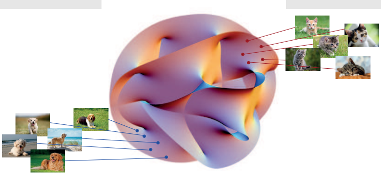
- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \ldots, K\}$.



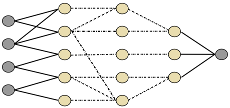
Training of Deep Neural Networks

High-Level Set Up:

- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \ldots, K\}$.



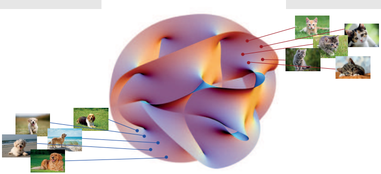
- Select an architecture of a deep neural network, i.e., a choice of d, L, $(N_\ell)_{\ell=1}^L$, and ρ.
  Sometimes selected entries of the matrices $(A_\ell)_{\ell=1}^L$, i.e., weights, are set to zero at this point.



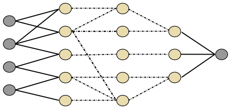
Training of Deep Neural Networks

High-Level Set Up:

- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \dots, K\}$.



- Select an architecture of a deep neural network, i.e., a choice of d, L, $(N_\ell)_{\ell=1}^L$, and ρ.
  Sometimes selected entries of the matrices $(A_\ell)_{\ell=1}^L$, i.e., weights, are set to zero at this point.



- Learn the affine-linear functions $(T_\ell)_{\ell=1}^L = (A_\ell \cdot + b_\ell)_{\ell=1}^L$ by

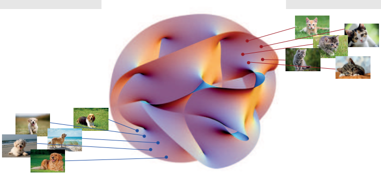
$$\min_{(A_\ell, b_\ell)_\ell} \sum_{i=1}^m \mathcal{L}(\Phi_{(A_\ell, b_\ell)_\ell}(x_i), f(x_i)) + \lambda \mathcal{R}((A_\ell, b_\ell)_\ell)$$

yielding the network $\Phi_{(A_\ell, b_\ell)_\ell} : \mathbb{R}^d \to \mathbb{R}^{N_L}$,

$$\Phi_{(A_\ell, b_\ell)_\ell}(x) = T_L \rho(T_{L-1} \rho(\dots \rho(T_1(x)))).$$

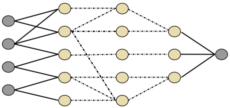
This is often done by stochastic gradient descent.

Training of Deep Neural Networks



High-Level Set Up:

- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \ldots, K\}$.



- Select an architecture of a deep neural network, i.e., a choice of d, L, $(N_\ell)_{\ell=1}^L$, and ρ.
  Sometimes selected entries of the matrices $(A_\ell)_{\ell=1}^L$, i.e., weights, are set to zero at this point.
- Learn the affine-linear functions $(T_\ell)_{\ell=1}^L = (A_\ell \cdot + b_\ell)_{\ell=1}^L$ by

$$\min_{(A_\ell, b_\ell)_\ell} \sum_{i=1}^m \mathcal{L}(\Phi_{(A_\ell, b_\ell)_\ell}(x_i), f(x_i)) + \lambda \mathcal{R}((A_\ell, b_\ell)_\ell)$$

yielding the network $\Phi_{(A_\ell, b_\ell)_\ell} : \mathbb{R}^d \to \mathbb{R}^{N_L}$,

$$\Phi_{(A_\ell, b_\ell)_\ell}(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x)))).$$

This is often done by stochastic gradient descent.

Goal: $\Phi_{(A_\ell, b_\ell)_\ell} \approx f$

Fundamental Questions concerning Deep Neural Networks

- *Expressivity:*
  - How powerful is the network architecture?
  - Can it indeed represent the correct functions?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

Fundamental Questions concerning Deep Neural Networks

- *Expressivity:*
  - How powerful is the network architecture?
  - Can it indeed represent the correct functions?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

- *Learning:*
  - Why does the current learning algorithm produce anything reasonable?
  - What are good starting values?

$\rightsquigarrow$ *Differential Geometry, Optimal Control, Optimization, ...*

Fundamental Questions concerning Deep Neural Networks

- *Expressivity:*
  - How powerful is the network architecture?
  - Can it indeed represent the correct functions?

$\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*

- *Learning:*
  - Why does the current learning algorithm produce anything reasonable?
  - What are good starting values?

$\rightsquigarrow$ *Differential Geometry, Optimal Control, Optimization, ...*

- *Generalization:*
  - Why do deep neural networks perform that well on data sets, which do not belong to the input-output pairs from a training set?
  - What impact has the depth of the network?

$\rightsquigarrow$ *Learning Theory, Optimization, Statistics, ...*

Fundamental Questions concerning Deep Neural Networks

- *Expressivity:*
  - How powerful is the network architecture?
  - Can it indeed represent the correct functions?

  $\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*
- *Learning:*
  - Why does the current learning algorithm produce anything reasonable?
  - What are good starting values?

  $\rightsquigarrow$ *Differential Geometry, Optimal Control, Optimization, ...*
- *Generalization:*
  - Why do deep neural networks perform that well on data sets, which do not belong to the input-output pairs from a training set?
  - What impact has the depth of the network?

  $\rightsquigarrow$ *Learning Theory, Optimization, Statistics, ...*
- *Interpretability:*
  - Why did a trained deep neural network reach a certain decision?
  - Which components of the input do contribute most?

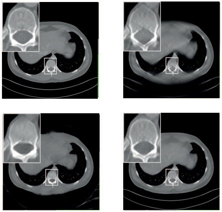
  $\rightsquigarrow$ *Information Theory, Uncertainty Quantification, ...*

Impact of Deep Learning on Mathematics

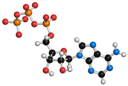
Impact of Deep Learning on Mathematics

Some Examples:

- Inverse Problems
  - $\leadsto$ *Image denoising (Burger, Schuler, Harmeling; 2012)*
  - $\leadsto$ *Superresolution (Klatzer, Soukup, Kobler, Hammernik, Pock; 2017)*
  - $\leadsto$ *Limited-angle tomography (Bubba, K, Lassas, März, Samek, Siltanen, Srinivan; 2018)*
  - $\leadsto$ *Edge detection (Andrade-Loarca, K, Öktem, Petersen; 2019)*



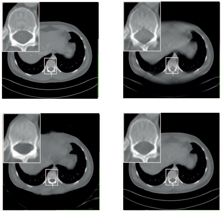
- Numerical Analysis of Partial Differential Equations
  - $\leadsto$ *Schrödinger equation (Rupp, Tkatchenko, Müller, von Lilienfeld; 2012 –)*
  - $\leadsto$ *Parametric PDEs (K, Petersen, Raslan, Schneider; 2019) (Geist, Petersen, Raslan, Schneider, K; 2020)*



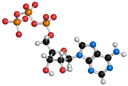
Impact of Deep Learning on Mathematics

Some Examples:

- Inverse Problems
  - $\rightsquigarrow$ *Image denoising (Burger, Schuler, Harmeling; 2012)*
  - $\rightsquigarrow$ *Superresolution (Klatzer, Soukup, Kobler, Hammernik, Pock; 2017)*
  - $\rightsquigarrow$ *Limited-angle tomography (Bubba, K, Lassas, März, Samek, Siltanen, Srinivan; 2018)*
  - $\rightsquigarrow$ *Edge detection (Andrade-Loarca, K, Öktem, Petersen; 2019)*



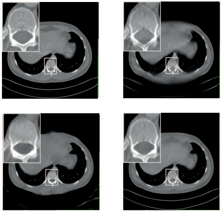
- Numerical Analysis of Partial Differential Equations
  - $\rightsquigarrow$ *Schrödinger equation (Rupp, Tkatchenko, Müller, von Lilienfeld; 2012 –)*
  - $\rightsquigarrow$ *Parametric PDEs (K, Petersen, Raslan, Schneider; 2019) (Geist, Petersen, Raslan, Schneider, K; 2020)*



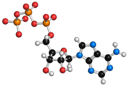
Impact of Deep Learning on Mathematics

Some Examples:

- Inverse Problems
  - ⇝ *Image denoising (Burger, Schuler, Harmeling; 2012)*
  - ⇝ *Superresolution (Klatzer, Soukup, Kobler, Hammernik, Pock; 2017)*
  - ⇝ *Limited-angle tomography (Bubba, K, Lassas, März, Samek, Siltanen, Srinivan; 2018)*
  - ⇝ *Edge detection (Andrade-Loarca, K, Öktem, Petersen; 2019)*



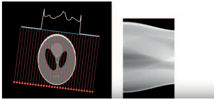
- Numerical Analysis of Partial Differential Equations
  - ⇝ *Schrödinger equation (Rupp, Tkatchenko, Müller, von Lilienfeld; 2012 –)*
  - ⇝ *Parametric PDEs (K, Petersen, Raslan, Schneider; 2019) (Geist, Petersen, Raslan, Schneider, K; 2020)*



Deep Learning for Inverse Problems

Example: Limited-Angle Computed Tomography

A CT scanner samples the *Radon transform*



$$\mathcal{R}f(\phi, s) = \int_{L(\phi,s)} f(x)dS(x),$$

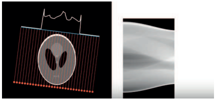
for $L(\phi, s) = \{x \in \mathbb{R}^2 : x_1 \cos(\phi) + x_2 \sin(\phi) = s\}$, $\phi \in [-\pi/2, \pi/2)$, and $s \in \mathbb{R}$.

Deep Learning for Inverse Problems

Example: Limited-Angle Computed Tomography

A CT scanner samples the *Radon transform*

$$\mathcal{R}f(\phi, s) = \int_{L(\phi,s)} f(x)dS(x),$$



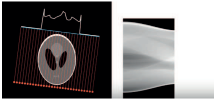
for $L(\phi, s) = \left\{x \in \mathbb{R}^2 : x_1 \cos(\phi) + x_2 \sin(\phi) = s\right\}$, $\phi \in [-\pi/2, \pi/2)$, and $s \in \mathbb{R}$.

Challenging inverse problem if $\mathcal{R}f(\cdot, s)$ is only sampled on $[-\phi, \phi]$, $\phi < \pi/2$

Deep Learning for Inverse Problems

Example: Limited-Angle Computed Tomography

A CT scanner samples the *Radon transform*



$$\mathcal{R}f(\phi, s) = \int_{L(\phi,s)} f(x) dS(x),$$

for $L(\phi, s) = \{x \in \mathbb{R}^2 : x_1 \cos(\phi) + x_2 \sin(\phi) = s\}$, $\phi \in [-\pi/2, \pi/2)$, and $s \in \mathbb{R}$.

Challenging inverse problem if $\mathcal{R}f(\cdot, s)$ is only sampled on $[-\phi, \phi]$, $\phi < \pi/2$

Learn the Invisible (Bubba, K, Lassas, März, Samek, Siltanen, Srinivan; 2018):

Step 1: Use model-based methods as far as possible

- Solve with sparse regularization using shearlets.

Step 2: Use data-driven methods where it is necessary

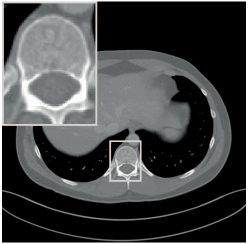
- Use a deep neural network to recover the missing components.

Step 3: Carefully combine both worlds

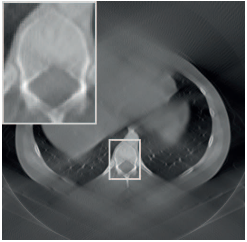
- Combine outcome of Step 1 and 2.

Learn the Invisible (LtI)

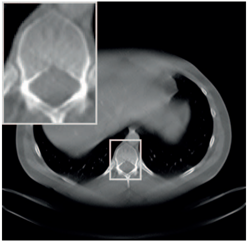
(Bubba, K, Lassas, März, Samek, Siltanen, Srinivan; 2018)



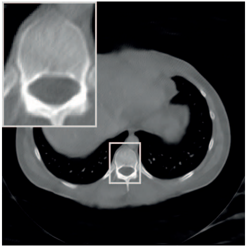
Original



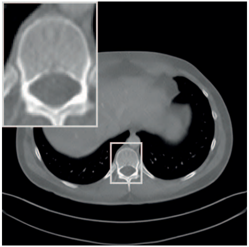
Filtered Backprojection



Sparse Regularization with Shearlets



[Gu & Ye, 2017]

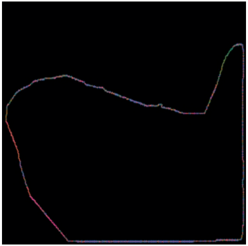


Learn the Invisible (LtI)

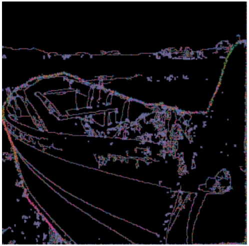
Deep Network Shearlet Edge Extractor (DeNSE)

(Andrade-Loarca, K, Öktem, Petersen; 2019)

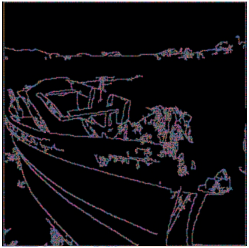
Original



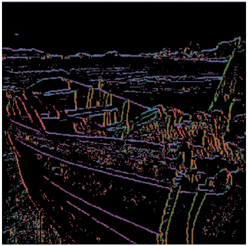
Human Annotation



SEAL [Yu et al; 2018]



CoShREM [Reisenhofer et al.; 2015]

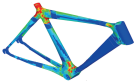


DeNSE

Deep Learning for (Parametric) PDEs

Parametric Map:

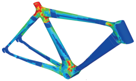
$$\mathbb{R}^p \supset \mathcal{Y} \ni y \mapsto u_y \in \mathcal{H} \quad \text{such that} \quad \mathcal{L}(u_y, y) = f_y.$$



Deep Learning for (Parametric) PDEs

Parametric Map:

$$\mathbb{R}^p \supset \mathcal{Y} \ni y \mapsto u_y \in \mathcal{H} \quad \text{such that} \quad \mathcal{L}(u_y, y) = f_y.$$

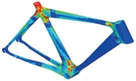


Curse of Dimensionality: Computational cost is exponential in p!

Deep Learning for (Parametric) PDEs

Parametric Map:

$$\mathbb{R}^p \supset \mathcal{Y} \ni y \;\mapsto\; u_y \in \mathcal{H} \quad \text{such that} \quad \mathcal{L}(u_y, y) = f_y.$$



Curse of Dimensionality: Computational cost is exponential in p!

Theoretical Approach (K, Petersen, Raslan, Schneider; 2019):

Expressivity

- We show the existence of a neural network Φ which approximates the parametric map, i.e., $\|\Phi - u_y\| \le \epsilon$ for all $y \in \mathcal{Y}$.

Complexity Analysis

- We prove that the complexity of the neural network Φ does not suffer from the curse of dimensionality.

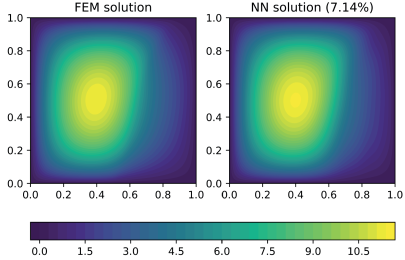
Numerical Results

(Geist, Petersen, Raslan, Schneider, K; 2020)

Set-Up:

- Parametric diffusion equation with various parametrizations
- Fixed neural network: 11 layers and 0.2-LReLU
- Training set: 20000 i.i.d. parameter samples

Example ($p = 91$):



The performance does not suffer from the curse of dimensionality!

Data-Driven Versus Model-Based Approaches?

Optimal balancing of
data-driven and model-based approaches!

Mathematics of Deep Neural Networks

- *Expressivity:*
  - How powerful is the network architecture?
  - Can it indeed represent the correct functions?

  $\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*
- *Learning:*
  - Why does the current learning algorithm produce anything reasonable?
  - What are good starting values?

  $\rightsquigarrow$ *Differential Geometry, Optimal Control, Optimization, ...*
- *Generalization:*
  - Why do deep neural networks perform that well on data sets, which do not belong to the input-output pairs from a training set?
  - What impact has the depth of the network?

  $\rightsquigarrow$ *Learning Theory, Optimization, Statistics, ...*
- *Interpretability:*
  - Why did a trained deep neural network reach a certain decision?
  - Which components of the input do contribute most?

  $\rightsquigarrow$ *Information Theory, Uncertainty Quantification, ...*

Expressivity

Main Research Goal

Some Questions:

- Which architecture to choose for a particular application?
- What is the expressive power of a given architecture?
- What effect has the depth of a neural network in this respect?
- What is the complexity of the approximating neural network?
- Can deep neural networks beat the curse of dimensionality?

General Mathematical Problem:

Given a function class $\mathcal{C}$ and $f \in \mathcal{C}$ as well as an accuracy $\varepsilon > 0$, does there exist a neural network Φ such that

$$\|\Phi - f\| \leq \varepsilon,$$

and which complexity does Φ have?

Function Approximation in a Nutshell

Goal: Given $\mathcal{C} \subseteq L^2(\mathbb{R}^d)$ and $(\varphi_i)_{i \in I} \subseteq L^2(\mathbb{R}^d)$. Measure the suitability of $(\varphi_i)_{i \in I}$ for uniformly approximating functions from $\mathcal{C}$.

Definition: The *error of best N-term approximation* of some $f \in \mathcal{C}$ is given by

$$\|f - f_N\|_2 := \inf_{I_N \subset I, \# I_N = N, (c_i)_{i \in I_N}} \|f - \sum_{i \in I_N} c_i \varphi_i\|_2.$$

The largest $\gamma > 0$ such that

$$\sup_{f \in \mathcal{C}} \|f - f_N\|_2 = O(N^{-\gamma}) \qquad \text{as } N \to \infty$$

determines the *optimal (sparse) approximation rate* of $\mathcal{C}$ by $(\varphi_i)_{i \in I}$.

Function Approximation in a Nutshell

Goal: Given $\mathcal{C} \subseteq L^2(\mathbb{R}^d)$ and $(\varphi_i)_{i \in I} \subseteq L^2(\mathbb{R}^d)$. Measure the suitability of $(\varphi_i)_{i \in I}$ for uniformly approximating functions from $\mathcal{C}$.

Definition: The *error of best N-term approximation* of some $f \in \mathcal{C}$ is given by

$$\|f - f_N\|_2 := \inf_{I_N \subset I, \# I_N = N, (c_i)_{i \in I_N}} \|f - \sum_{i \in I_N} c_i \varphi_i\|_2.$$

The largest $\gamma > 0$ such that

$$\sup_{f \in \mathcal{C}} \|f - f_N\|_2 = O(N^{-\gamma}) \qquad \text{as } N \to \infty$$

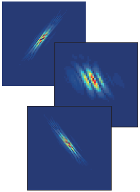
determines the *optimal (sparse) approximation rate* of $\mathcal{C}$ by $(\varphi_i)_{i \in I}$.

Approximation accuracy $\leftrightarrow$ Complexity of approximating system in terms of sparsity

Example: Shearlet Systems

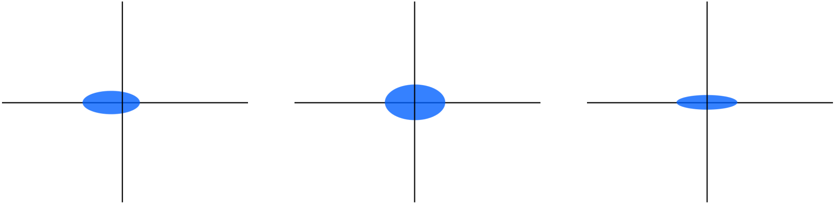
Definition (K, Labate; 2006):

$$A_j := \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2} \end{pmatrix}, \qquad S_k := \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}, \quad j,k \in \mathbb{Z}.$$



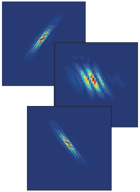
Then

$$\psi_{j,k,m} := 2^{\frac{3j}{4}} \psi(S_k A_j \cdot - m).$$



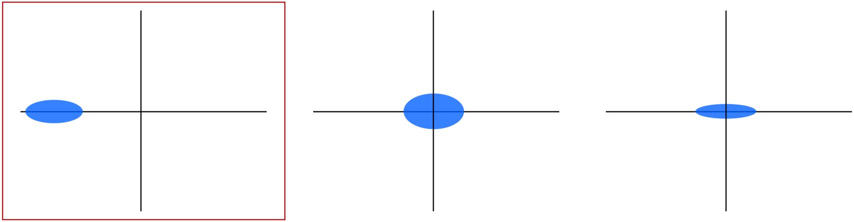
Definition (K, Labate; 2006):

$$A_j := \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2} \end{pmatrix}, \qquad S_k := \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}, \quad j,k \in \mathbb{Z}.$$



Then

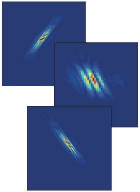
$$\psi_{j,k,m} := 2^{\frac{3j}{4}} \psi(S_k A_j \cdot - m).$$



Example: Shearlet Systems

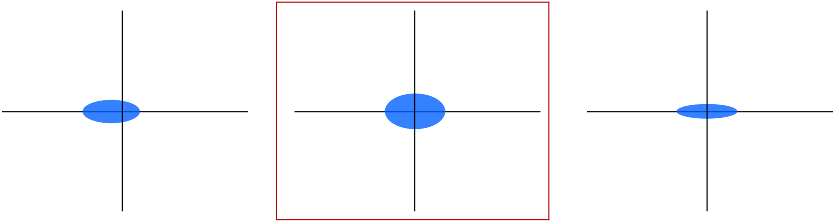
Definition (K, Labate; 2006):

$$A_j := \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2} \end{pmatrix}, \qquad S_k := \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}, \quad j,k \in \mathbb{Z}.$$



Then

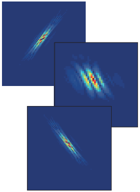
$$\psi_{j,k,m} := 2^{\frac{3j}{4}} \psi(S_k A_j \cdot - m).$$



Example: Shearlet Systems

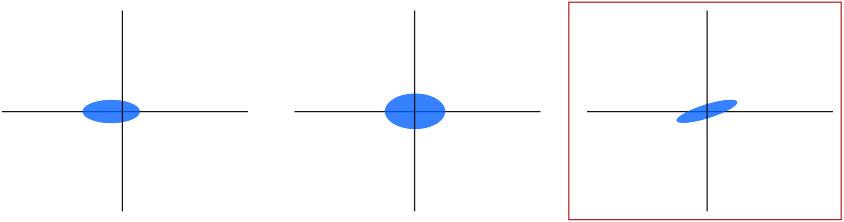
Definition (K, Labate; 2006):

$$A_j := \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2} \end{pmatrix}, \qquad S_k := \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}, \quad j, k \in \mathbb{Z}.$$



Then

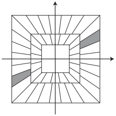
$$\psi_{j,k,m} := 2^{\frac{3j}{4}} \psi(S_k A_j \cdot - m).$$



(Cone-adapted) Discrete Shearlet Systems

Definition (K, Labate; 2006):

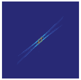
The (cone-adapted) discrete shearlet system $\mathcal{SH}(c;\phi,\psi,\tilde{\psi})$, $c>0$, generated by $\phi \in L^2(\mathbb{R}^2)$ and $\psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ is the union of



$$\{\phi(\cdot - cm) : m \in \mathbb{Z}^2\},$$

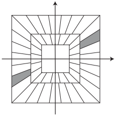
$$\{2^{3j/4}\psi(S_k A_{2^j}\ \cdot\ - cm) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\},$$

$$\{2^{3j/4}\tilde{\psi}(\tilde{S}_k \tilde{A}_{2^j}\ \cdot\ - cm) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\}.$$



Definition (K, Labate; 2006):

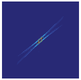
The (cone-adapted) discrete shearlet system $\mathcal{SH}(c;\phi,\psi,\tilde{\psi})$, $c>0$, generated by $\phi \in L^2(\mathbb{R}^2)$ and $\psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ is the union of



$$\{\phi(\cdot - cm) : m \in \mathbb{Z}^2\},$$

$$\{2^{3j/4}\psi(S_k A_{2^j} \cdot - cm) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\},$$

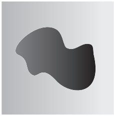
$$\{2^{3j/4}\tilde{\psi}(\tilde{S}_k \tilde{A}_{2^j} \cdot - cm) : j \geq 0, |k| \leq \lceil 2^{j/2} \rceil, m \in \mathbb{Z}^2\}.$$



Theorem (K, Lim; 2011):

Let $\phi, \psi, \tilde{\psi} \in L^2(\mathbb{R}^2)$ be compactly supported, and let $\hat{\psi}$, $\hat{\tilde{\psi}}$ satisfy certain decay condition. Then $\mathcal{SH}(\phi,\psi,\tilde{\psi})$ provides an optimally sparse approximation of cartoon-like functions f, i.e.,

$$\|f - f_N\|_2 \leq C \cdot N^{-1} \cdot (\log N)^{3/2}, \quad N \to \infty.$$



Function Approximation in a Nutshell

Goal: Given $\mathcal{C} \subseteq L^2(\mathbb{R}^d)$ and $(\varphi_i)_{i \in I} \subseteq L^2(\mathbb{R}^d)$. Measure the suitability of $(\varphi_i)_{i \in I}$ for uniformly approximating functions from $\mathcal{C}$.

Definition: The *error of best N-term approximation* of some $f \in \mathcal{C}$ is given by

$$\|f - f_N\|_2 := \inf_{I_N \subset I, \#I_N = N, (c_i)_{i \in I_N}} \|f - \sum_{i \in I_N} c_i \varphi_i\|_2.$$

The largest $\gamma > 0$ such that

$$\sup_{f \in \mathcal{C}} \|f - f_N\|_2 = O(N^{-\gamma}) \qquad \text{as } N \to \infty$$

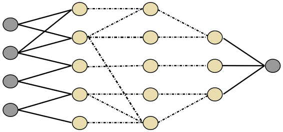
determines the *optimal (sparse) approximation rate* of $\mathcal{C}$ by $(\varphi_i)_{i \in I}$.

Approximation accuracy $\leftrightarrow$ Complexity of approximating system in terms of sparsity

Complexity of a Deep Neural Network

Complexity of a Deep Neural Network

Recall:



- $d \in \mathbb{N}$: Dimension of input layer.
- L: Number of layers.
- N: Number of neurons.
- $\rho : \mathbb{R} \to \mathbb{R}$: (Non-linear) function called *activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \ldots, L$: Affine linear maps $x \mapsto A_\ell x + b_\ell$.

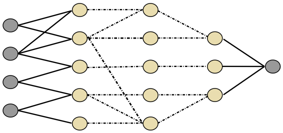
Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

$$\Phi(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*.

Complexity of a Deep Neural Network

Recall:



- $d \in \mathbb{N}$: Dimension of input layer.
- L: Number of layers.
- N: Number of neurons.
- $\rho : \mathbb{R} \to \mathbb{R}$: (Non-linear) function called *activation function*.
- $T_\ell : \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$, $\ell = 1, \ldots, L$: Affine linear maps $x \mapsto A_\ell x + b_\ell$.

Then $\Phi : \mathbb{R}^d \to \mathbb{R}^{N_L}$ given by

$$\Phi(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x))), \quad x \in \mathbb{R}^d,$$

is called *(deep) neural network (DNN)*.

Measure for Complexity: The *number of weights $W(\Phi)$* is defined by

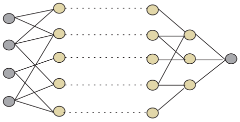
$$W(\Phi) := \sum_{\ell=1}^{L} \left(\|A_\ell\|_0 + \|b_\ell\|_0 \right).$$

We write $\Phi \in \mathcal{NN}_{L,W(\Phi),d,\rho}$.

Sparsely Connected Deep Neural Networks

Key Problem:

- Deep neural networks employed in practice often consist of hundreds of layers.
- Training and storage of such networks pose formidable (computational) challenge.



$\rightsquigarrow$ *Employ deep neural networks with sparse connectivity!*

Example of Speech Recognition:

- Typically speech recognition is performed in the cloud (e.g. SIRI).
- New speech recognition systems (e.g. Android) can operate offline and are based on a sparsely connected deep neural network.

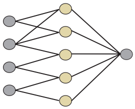
Key Challenge:

Approximation accuracy $\leftrightarrow$ *Complexity of approximating DNN in terms of sparse connectivity!*

One Size Fits All?

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):
Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

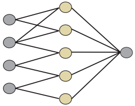
$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot \rangle - b_k)\|_\infty \leq \epsilon.$$



One Size Fits All?

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):
Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot \rangle - b_k)\|_\infty \leq \epsilon.$$

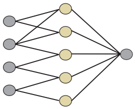


The complexity can be arbitrarily large!

One Size Fits All?

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):
Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot\rangle - b_k)\|_\infty \le \epsilon.$$



The complexity can be arbitrarily large!

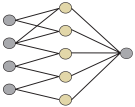
Theorem (Yarotsky; 2017): For all $f \in \mathcal{C} = C^s([0,1]^d)$ and ρ the *ReLU (Rectifiable Linear Unit $\rho(x) = \max\{0, x\}$)*, there exist neural networks $(\Phi_n)_{n\in\mathbb{N}}$ with $L(\Phi_n) \approx \log(n)$ such that

$$\|f - \Phi_n\|_\infty \lesssim W(\Phi_n)^{-\frac{s}{d}} \to 0 \quad \text{as } n \to \infty.$$

One Size Fits All?

Universal Approximation Theorem (Cybenko, 1989)(Hornik, 1991):
Let $d \in \mathbb{N}$, $K \subset \mathbb{R}^d$ compact, $f : K \to \mathbb{R}$ continuous, $\rho : \mathbb{R} \to \mathbb{R}$ continuous and not a polynomial. Then, for each $\epsilon > 0$, there exist $N \in \mathbb{N}$, $a_k, b_k \in \mathbb{R}, w_k \in \mathbb{R}^d$ such that

$$\|f - \sum_{k=1}^{N} a_k \rho(\langle w_k, \cdot \rangle - b_k)\|_\infty \leq \epsilon.$$



The complexity can be arbitrarily large!

Theorem (Yarotsky; 2017): For all $f \in \mathcal{C} = C^s([0,1]^d)$ and ρ the *ReLU (Rectifiable Linear Unit $\rho(x) = \max\{0, x\}$)*, there exist neural networks $(\Phi_n)_{n \in \mathbb{N}}$ with $L(\Phi_n) \approx \log(n)$ such that

$$\|f - \Phi_n\|_\infty \lesssim W(\Phi_n)^{-\frac{s}{d}} \to 0 \quad \text{as } n \to \infty.$$

This result is not optimal!

A Fundamental Lower Bound

Complexity of a Function Class:

The optimal exponent $\gamma^*(\mathcal{C})$ measures the complexity of $\mathcal{C} \subset L^2(\mathbb{R}^d)$.

A Fundamental Lower Bound

Complexity of a Function Class:

The optimal exponent $\gamma^*(\mathcal{C})$ measures the complexity of $\mathcal{C} \subset L^2(\mathbb{R}^d)$.

Theorem (Bölcskei, Grohs, K, and Petersen; 2019):

Let $d \in \mathbb{N}$, $\rho : \mathbb{R} \to \mathbb{R}$, and let $\mathcal{C} \subset L^2(\mathbb{R}^d)$. Further, let

$$\textbf{Learn} : (0,1) \times \mathcal{C} \to \mathcal{NN}_{\infty,\infty,d,\rho}$$

satisfy that, for each $f \in \mathcal{C}$ and $0 < \epsilon < 1$,

$$\sup_{f \in \mathcal{C}} \|f - \textbf{Learn}(\epsilon, f)\|_2 \leq \epsilon.$$

Then, for all $\gamma < \gamma^*(\mathcal{C})$,

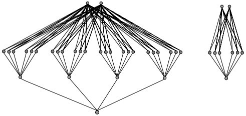
$$\epsilon^\gamma \sup_{f \in \mathcal{C}} W(\textbf{Learn}(\epsilon, f)) \to \infty, \qquad \text{as } \epsilon \to 0.$$

Conceptual bound independent on the learning algorithm!

Optimally Memory Efficient Neural Networks

Key Ideas:

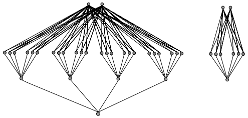
- Consider a representation system which provides an optimal approximation rate of a specific function class.
- Realize each element of a representation system by a DNN.
- Control the number of edges of those DNNs.



Optimally Memory Efficient Neural Networks

Key Ideas:

- Consider a representation system which provides an optimal approximation rate of a specific function class.
- Realize each element of a representation system by a DNN.
- Control the number of edges of those DNNs.



Choice for our Result:

Use the affine system of shearlets.

Theorem (Bölcskei, Grohs, K, and Petersen; 2019):

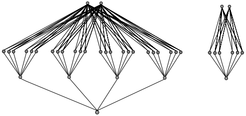
Let ρ be an admissible smooth rectifier, and let $\epsilon > 0$. Then, for all cartoon-like functions f and $N \in \mathbb{N}$, there exists $\Phi \in \mathcal{NN}_{3,O(N),2,\rho}$ with

$$\|f - \Phi\|_2 \lesssim N^{-1+\epsilon} \to 0 \quad \text{as } N \to \infty.$$

Optimally Memory Efficient Neural Networks

Key Ideas:

- Consider a representation system which provides an optimal approximation rate of a specific function class.
- Realize each element of a representation system by a DNN.
- Control the number of edges of those DNNs.



Choice for our Result:

Use the affine system of shearlets.

Theorem (Bölcskei, Grohs, K, and Petersen; 2019):

Let ρ be an admissible smooth rectifier, and let $\epsilon > 0$. Then, for all cartoon-like functions f and $N \in \mathbb{N}$, there exists $\Phi \in \mathcal{NN}_{3,O(N),2,\rho}$ with

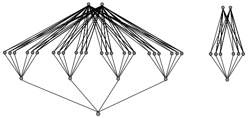
$$\|f - \Phi\|_2 \lesssim N^{-1+\epsilon} \to 0 \quad \text{as } N \to \infty.$$

This is the optimal rate; hence the first bound is sharp!

Optimally Memory Efficient Neural Networks

Key Ideas:

- Consider a representation system which provides an optimal approximation rate of a specific function class.
- Realize each element of a representation system by a DNN.
- Control the number of edges of those DNNs.



Choice for our Result:

Use the affine system of shearlets.

Theorem (Bölcskei, Grohs, K, and Petersen; 2019):

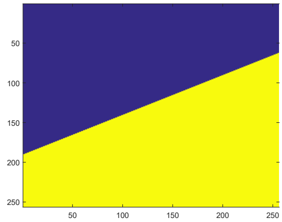
Let ρ be an admissible smooth rectifier, and let $\epsilon > 0$. Then, for all cartoon-like functions f and $N \in \mathbb{N}$, there exists $\Phi \in \mathcal{NN}_{3,O(N),2,\rho}$ with

$$\|f - \Phi\|_2 \lesssim N^{-1+\epsilon} \to 0 \quad \text{as } N \to \infty.$$

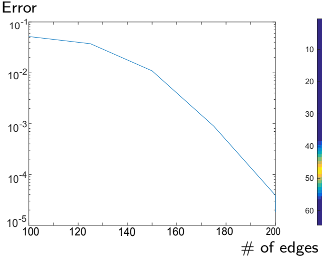
This is the optimal rate; hence the first bound is sharp!

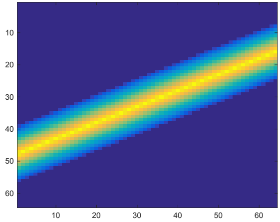
DNNs achieve optimal approx. properties of all affine systems combined!

Numerical Experiments (with ReLUs & Backpropagation)



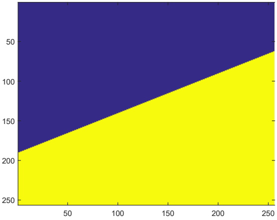
Linear Singularity





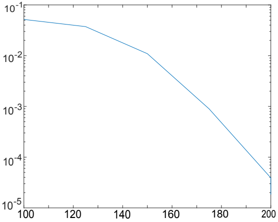
Subnetworks: Ridgelets!

Numerical Experiments (with ReLUs & Backpropagation)

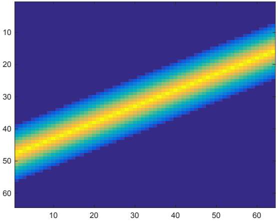


Linear Singularity

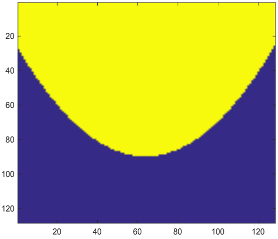
Error



# of edges

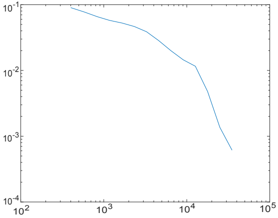


Subnetworks: Ridgelets!

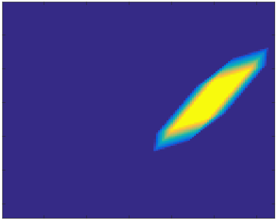


Curvilinear Singularity

Error



# of edges



Subnetworks: $\approx$ Shearlets!

Mathematics of Deep Neural Networks

- *Expressivity:*
  - How powerful is the network architecture?
  - Can it indeed represent the correct functions?

  $\rightsquigarrow$ *Applied Harmonic Analysis, Approximation Theory, ...*
- *Learning:*
  - Why does the current learning algorithm produce anything reasonable?
  - What are good starting values?

  $\rightsquigarrow$ *Differential Geometry, Optimal Control, Optimization, ...*
- *Generalization:*
  - Why do deep neural networks perform that well on data sets, which do not belong to the input-output pairs from a training set?
  - What impact has the depth of the network?

  $\rightsquigarrow$ *Learning Theory, Optimization, Statistics, ...*
- *Interpretability:*
  - Why did a trained deep neural network reach a certain decision?
  - Which components of the input do contribute most?

  $\rightsquigarrow$ *Information Theory, Uncertainty Quantification, ...*

Interpretability

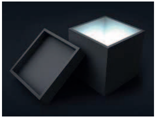
General Problem Setting

Question:

- Given a trained neural network.
- We don't know what the training data was nor how it was trained.

$\rightsquigarrow$ *Can we determine how it operates?*

Opening the Black Box!



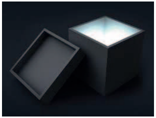
General Problem Setting

Question:

- Given a trained neural network.
- We don't know what the training data was nor how it was trained.

$\rightsquigarrow$ *Can we determine how it operates?*

Opening the Black Box!



Why is this important?

- Assume a job application is rejected.
- Imagine this rejection was done by a neural network-based algorithm.

$\rightsquigarrow$ *The applicant wants to know the reasons!*

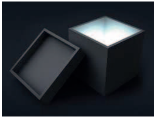
General Problem Setting

Question:

- Given a trained neural network.
- We don't know what the training data was nor how it was trained.

$\rightsquigarrow$ *Can we determine how it operates?*

Opening the Black Box!



Why is this important?

- Assume a job application is rejected.
- Imagine this rejection was done by a neural network-based algorithm.

$\rightsquigarrow$ *The applicant wants to know the reasons!*

Holy Grail: 

- Explanation of a decision indistinguishable from a human being!

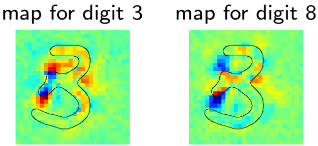
History of the Field

Previous Relevance Mapping Methods:

- Gradient based methods:
  - Sensitivity Analysis (Baehrens, Schroeter, Harmeling, Kawanabe, Hansen, Müller, 2010)
  - SmoothGrad (Smilkov, Thorat, Kim, Viégas, Wattenberg, 2017)
- Backwards propagation based methods:
  - Guided Backprop (Springenberg, Dosovitskiy, Brox, Riedmiller, 2015)
  - Layer-wise Relevance Propagation (Bach, Binder, Montavon, Klauschen, Müller, Samek, 2015)
  - Deep Taylor (Montavon, Samek, Müller, 2018)
- Surrogate model based methods:
  - LIME (Local Interpretable Model-agnostic Explanations) (Ribeiro, Singh, Guestrin, 2016)
- Game theoretic methods:
  - Shapley values (Shapley, 1953), (Kononenko, Štrumbelj, 2010)
  - SHAP (Shapley Additive Explanations) (Lundberg, Lee, 2017)

What is Relevance?

Main Goal: We aim to understand decisions of “black-box” predictors!

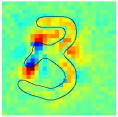


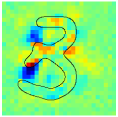
Challenges:

- What exactly is relevance in a mathematical sense?
- What is a good relevance map?
- How to compare different relevance maps?

What is Relevance?

Main Goal: We aim to understand decisions of "black-box" predictors!





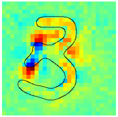
Challenges:

- What exactly is relevance in a mathematical sense?
  $\rightsquigarrow$ *Rigorous definition of relevance by information theory.*
- What is a good relevance map?
- How to compare different relevance maps?

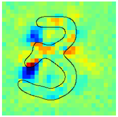
What is Relevance?

Main Goal: We aim to understand decisions of "black-box" predictors!

map for digit 3



map for digit 8



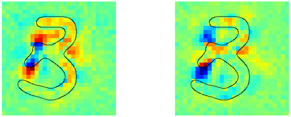
Challenges:

- What exactly is relevance in a mathematical sense?
  $\rightsquigarrow$ *Rigorous definition of relevance by information theory.*
- What is a good relevance map?
  $\rightsquigarrow$ *Formulation of interpretability as optimization problem.*
  $\rightsquigarrow$ *Theoretical analysis of complexity.*
- How to compare different relevance maps?

What is Relevance?

Main Goal: We aim to understand decisions of "black-box" predictors!

map for digit 3 map for digit 8



Challenges:

- What exactly is relevance in a mathematical sense?
  $\rightsquigarrow$ *Rigorous definition of relevance by information theory.*
- What is a good relevance map?
  $\rightsquigarrow$ *Formulation of interpretability as optimization problem.*
  $\rightsquigarrow$ *Theoretical analysis of complexity.*
- How to compare different relevance maps?
  $\rightsquigarrow$ *Canonical framework for comparison.*

The Relevance Mapping Problem

The Relevance Mapping Problem

The Setting: Let

- $\Phi \colon [0,1]^d \to [0,1]$ be a classification function,
- $x \in [0,1]^d$ be an input signal.

The Task:

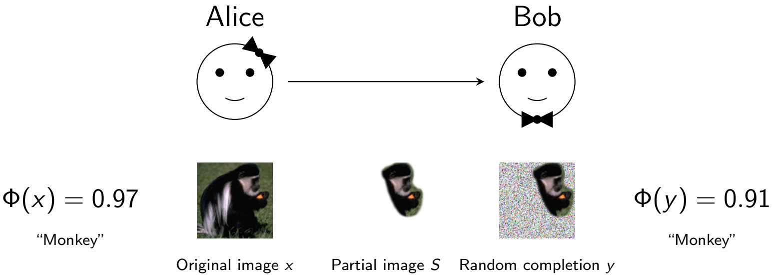
- Determine the most relevant components of x for the prediction $\Phi(x)$.
- Choose $S \subseteq \{1, \ldots, d\}$ of components that are considered relevant.
- S should be small (usually not everything is relevant).
- S^c is considered non-relevant.

Original image x

Relevant components S

Non-relevant components S^c

Rate-Distortion Viewpoint



Obfuscation: Let

- $n \sim \mathcal{V}$ be a random noise vector, and
- y be a random vector defined as $y_S = x_S$ and $y_{S^c} = n_{S^c}$.

Rate-Distortion Viewpoint

Recall: Let

- $\Phi\colon [0,1]^d \to [0,1]$ be a classification function,
- $x \in [0,1]^d$ be an input signal,
- $n \sim \mathcal{V}$ be a random noise vector, and
- y be a random vector defined as $y_S = x_S$ and $y_{S^c} = n_{S^c}$.

Expected Distortion:

$$D(S) = D(\Phi, x, S) = \mathbb{E}\left[\frac{1}{2}\left(\Phi(x) - \Phi(y)\right)^2\right]$$

Rate-Distortion Function:

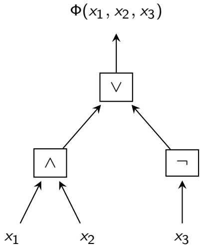
$$R(\epsilon) = \min_{S \subseteq \{1,\dots,d\}} \{|S| \,:\, D(S) \le \epsilon\}$$

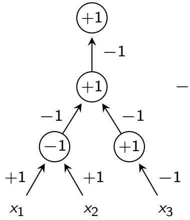
$\rightsquigarrow$ *Use this viewpoint for the definition of a relevance map!*

Finding a minimizer of $R(\epsilon)$

or even approximating it is very hard!

Boolean Functions as ReLU Neural Networks:

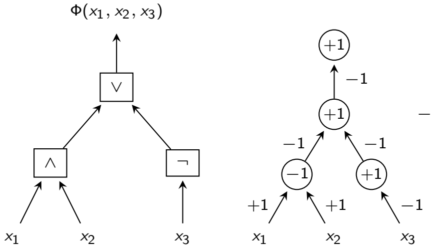




ReLU activation function $\varrho(x) = \max\{0, x\}$

$$-\varrho\left(\begin{bmatrix}-1 & -1\end{bmatrix}\varrho\left(\begin{bmatrix}1 & 1 & 0\\ 0 & 0 & -1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}+\begin{bmatrix}-1\\ 1\end{bmatrix}\right)+1\right)+1$$

Boolean Functions as ReLU Neural Networks:



ReLU activation function $\varrho(x) = \max\{0, x\}$

$$-\varrho\left(\begin{bmatrix}-1 & -1\end{bmatrix}\varrho\left(\begin{bmatrix}1 & 1 & 0\\ 0 & 0 & -1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}+\begin{bmatrix}-1\\ 1\end{bmatrix}\right)+1\right)+1$$

The Binary Setting: Let

- $\Phi\colon \{0,1\}^d \to \{0,1\}$ be classifier functions,
- $x \in \{0,1\}^d$ be signals, and
- $\mathcal{V} = \mathcal{U}(\{0,1\}^d)$ be a uniform distribution.

We consider the binary case.

Theorem (Wäldchen, Macdonald, Hauch, K, 2019):
Given Φ, x, $k \in \{1, \ldots, d\}$, and $\epsilon < \frac{1}{4}$. Deciding whether $R(\epsilon) \leq k$ is $\mathrm{NP}^{\mathrm{PP}}$-complete.

Finding a minimizer of $R(\epsilon)$ is hard!

Theorem (Wäldchen, Macdonald, Hauch, K, 2019):
Given Φ, x, and $\alpha \in (0, 1)$. Approximating $R(\epsilon)$ to within a factor of $d^{1-\alpha}$ is NP-hard.

Even the approximation problem of it is hard!

Preprint: "The Computational Complexity of Understanding Network Decisions", `https://arxiv.org/abs/1905.09163`

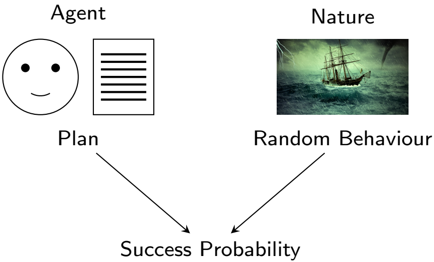
What is NP^{PP}?

The Complexity Class NP^{PP}:

Many important problems in artificial intelligence belong to this class.

Some Examples:

- Planning under uncertainties
- Finding maximum a-posteriori configurations in graphical models
- Maximizing utility functions in Bayesian networks



Our Method:

Rate-Distortion Explanation (RDE)

Problem Relaxation

	Discrete problem	Continuous problem
Relevant set	$S \subseteq \{1, \ldots, d\}$	
Obfuscation	$y_S = x_S,\ y_{S^c} = n_{S^c}$	
Distortion	$D(S)$	
Rate/Size	$\lvert S \rvert$	

Problem Relaxation

	Discrete problem	Continuous problem
Relevant set	$S \subseteq \{1, \ldots, d\}$	$s \in [0,1]^d$
Obfuscation	$y_S = x_S,\ y_{S^c} = n_{S^c}$	$y = s \odot x + (1-s) \odot n$
Distortion	$D(S)$	$D(s)$
Rate/Size	$\lvert S \rvert$	$\lVert s \rVert_1$

Problem Relaxation

	Discrete problem	Continuous problem
Relevant set	$S \subseteq \{1, \ldots, d\}$	$s \in [0,1]^d$
Obfuscation	$y_S = x_S,\ y_{S^c} = n_{S^c}$	$y = s \odot x + (1-s) \odot n$
Distortion	$D(S)$	$D(s)$
Rate/Size	$\|S\|$	$\\|s\\|_1$

Resulting Minimization Problem:

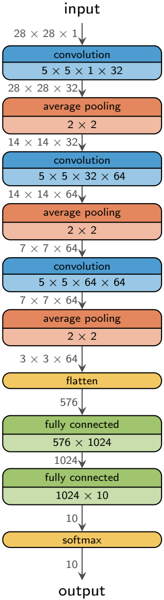
$$\boxed{\text{minimize} \quad D(s) + \lambda \|s\|_1 \quad \text{subject to} \quad s \in [0,1]^d}$$

Numerical Experiments

MNIST Experiment

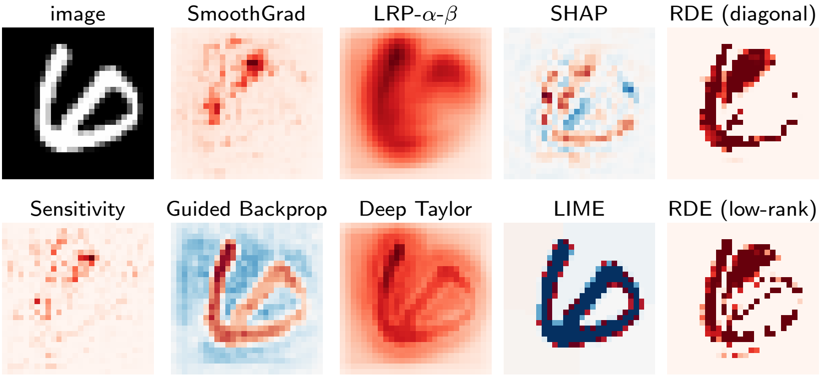
Data Set	
Image size	$28 \times 28 \times 1$
Number of classes	10
Training samples	50000

Test accuracy: 99.1%



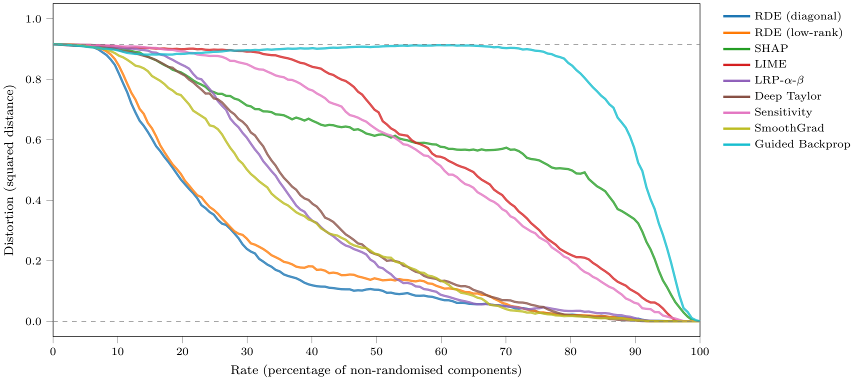
MNIST dataset of handwritten digits (LeCun, Cortes, 1998)

MNIST Experiment



SmoothGrad (Smilkov, Thorat, Kim, Viégas, Wattenberg, 2017), Layer-wise Relevance Propagation (Bach, Binder, Montavon, Klauschen, Müller, Samek, 2015), SHAP (Lundberg, Lee, 2017)
Sensitivity Analysis (Simonyan, Vedaldi, Zisserman, 2013), Guided Backprop (Springenberg, Dosovitskiy, Brox, Riedmiller, 2015), Deep Taylor Decompositions (Montavon, Samek, Müller, 2018)
LIME (Ribeiro, Singh, Guestrin, 2016)

MNIST Experiment



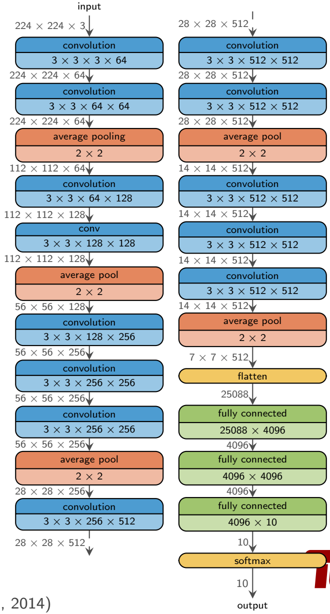
SmoothGrad (Smilkov, Thorat, Kim, Viégas, Wattenberg, 2017), Layer-wise Relevance Propagation (Bach, Binder, Montavon, Klauschen, Müller, Samek, 2015), SHAP (Lundberg, Lee, 2017)
Sensitivity Analysis (Simonyan, Vedaldi, Zisserman, 2013), Guided Backprop (Springenberg, Dosovitskiy, Brox, Riedmiller, 2015), Deep Taylor Decompositions (Montavon, Samek, Müller, 2018)
LIME (Ribeiro, Singh, Guestrin, 2016)

STL-10 Experiment

Data Set	
Image size	$96 \times 96 \times 3$ ($224 \times 224 \times 3$)
Number of classes	10
Training samples	4000

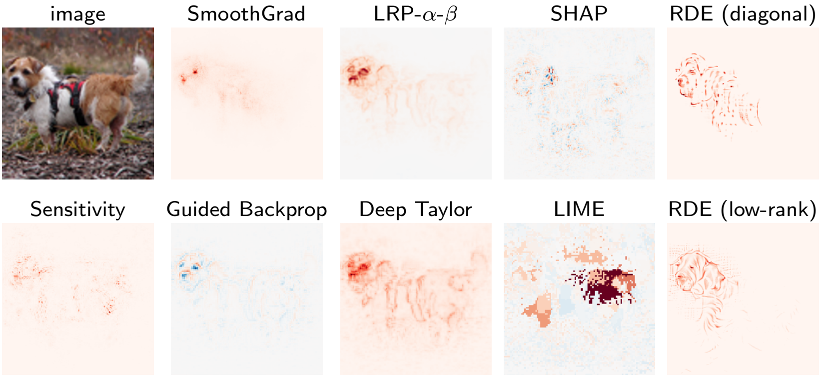
Test accuracy: 93.5%

(VGG-16 convolutions pretrained on Imagenet)



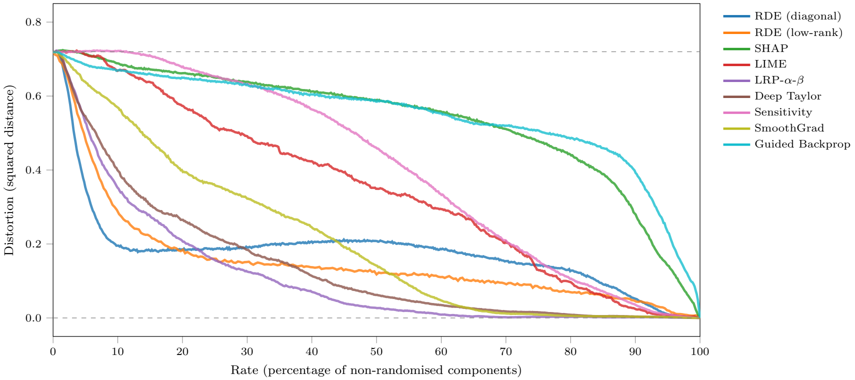
STL-10 dataset (Coates, Lee, Ng, 2011), VGG-16 network (Simonyan, Zisserman, 2014)

STL-10 Experiment



SmoothGrad (Smilkov, Thorat, Kim, Viégas, Wattenberg, 2017), Layer-wise Relevance Propagation (Bach, Binder, Montavon, Klauschen, Müller, Samek, 2015), SHAP (Lundberg, Lee, 2017)
Sensitivity Analysis (Simonyan, Vedaldi, Zisserman, 2013), Guided Backprop (Springenberg, Dosovitskiy, Brox, Riedmiller, 2015), Deep Taylor Decompositions (Montavon, Samek, Müller, 2018)
LIME (Ribeiro, Singh, Guestrin, 2016)

STL-10 Experiment



SmoothGrad (Smilkov, Thorat, Kim, Viégas, Wattenberg, 2017), Layer-wise Relevance Propagation (Bach, Binder, Montavon, Klauschen, Müller, Samek, 2015), SHAP (Lundberg, Lee, 2017)
Sensitivity Analysis (Simonyan, Vedaldi, Zisserman, 2013), Guided Backprop (Springenberg, Dosovitskiy, Brox, Riedmiller, 2015), Deep Taylor Decompositions (Montavon, Samek, Müller, 2018)
LIME (Ribeiro, Singh, Guestrin, 2016)

Conclusions

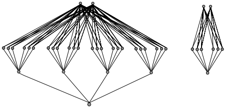
What to take Home...?

Deep Learning:

- Impressive performance *in combination with classical mathematical methods* (Inverse Problems, PDEs, ...).
- A theoretical foundation of neural networks is largely missing: *Expressivity, Learning, Generalization, and Interpretability*.

Expressivity:

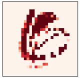
- Fundamental lower bound on the complexity, leading to the construction of optimally memory-efficient networks.



- Neural networks are as powerful approximators as classical affine systems such as wavelets, shearlets, ...

Interpretability:

- We provide a precise mathematical notion for *relevance* based on *rate-distortion theory*.



- We show that solving the optimization problem is *hard*.
- *RDE considers a relaxed version* and *outperforms current methods*.

THANK YOU!

References available at:

www.math.tu-berlin.de/~kutyniok

Related Book:

- P. Grohs and G. Kutyniok
  Theory of Deep Learning
  Cambridge University Press (in preparation)