An Information Theoretic Approach to Validate Deep Learning-Based Algorithms

Gitta Kutyniok

(Ludwig-Maximilians-Universität München and University of Tromsø)

Workshop on Safety and Security of Deep Learning
ICERM (virtual), April 10 – 11, 2021

Collaborators:

Cosmas Heiß
(TU Berlin)

Ron Levie
(LMU Munich)

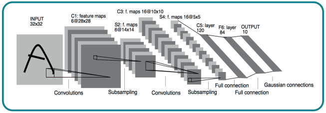
Cinjon Resnick
(NYU Courant)

Joan Bruna
(NYU Courant)

The Dawn of Deep Learning

Self-Driving Cars

Surveillance



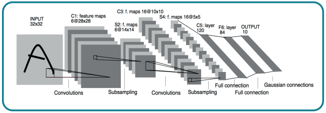
Legal Issues

Health Care

The Dawn of Deep Learning

Self-Driving Cars

Surveillance



Legal Issues

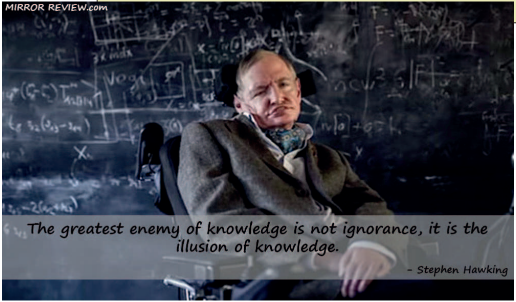
Health Care

Deep Neural Networks still act as a Black Box!

Deep Learning = Alchemy? ...Safety?

„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

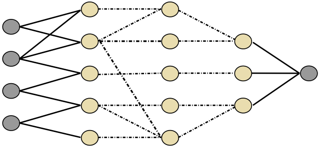


Theoretical Foundations of Deep Learning

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

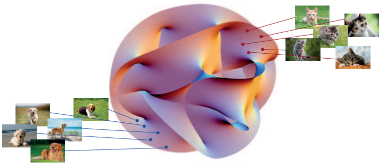
$$\Phi(x) = T_L \rho(T_{L-1} \rho(\ldots \rho(T_1(x)))), \quad x \in \mathbb{R}^d,$$

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

Training of Deep Neural Networks

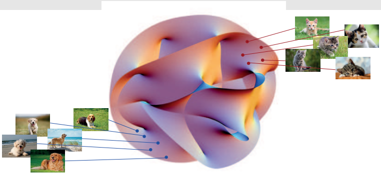
High-Level Set Up:

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

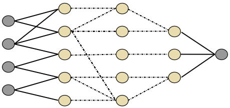


Training of Deep Neural Networks

High-Level Set Up:



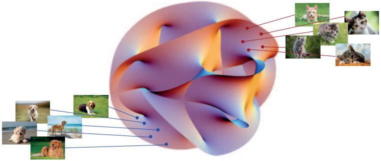
- Samples $(x_i, f(x_i))_{i=1}^m$ of a function such as $f : \mathcal{M} \to \{1, 2, \dots, K\}$.



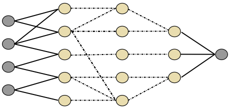
- Select an architecture of a deep neural network, i.e., a choice of d, L, $(N_\ell)_{\ell=1}^L$, and ρ.
  Sometimes selected entries of the matrices $(A_\ell)_{\ell=1}^L$, i.e., weights, are set to zero at this point.

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

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

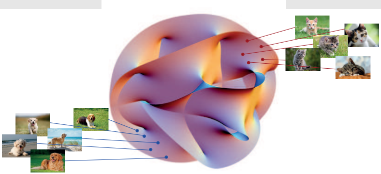
yielding the network $\Phi_{(A_\ell, b_\ell)_\ell} : \mathbb{R}^d \to \mathbb{R}^{N_L}$,

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

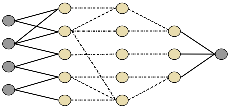
This is often done by stochastic gradient descent.

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, 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, 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

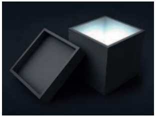
General Problem Setting

Question:

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

⇝ *Can we determine how it operates?*

Opening the Black Box!



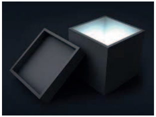
General Problem Setting

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!*

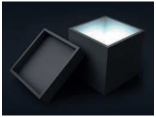
General Problem Setting

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!*

Vision for the Future:

- 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)

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)

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)

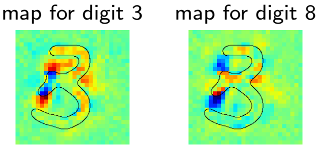
History of the Field

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!



Classification as a Classical Task for Neural Networks:

- Which features are most relevant for the decision?
  - Treat every pixel separately
  - Consider combinations of pixels
  - Incorporate additional knowledge
- How certain is the decision?

Tasks for Today

Challenges:

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

Tasks for Today

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?
- How to extend to challenging modalities?

Tasks for Today

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.*
- How to compare different relevance maps?
- How to extend to challenging modalities?

Tasks for Today

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.*
- How to compare different relevance maps?
  $\rightsquigarrow$ *Canonical framework for comparison.*
- How to extend to challenging modalities?

Tasks for Today

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.*
- How to compare different relevance maps?
  $\rightsquigarrow$ *Canonical framework for comparison.*
- How to extend to challenging modalities?
  $\rightsquigarrow$ *Conceptually general and flexible interpretability approach.*

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.

$\xrightarrow{\Phi}$ $\Phi(x) = 0.97$ "Monkey"

$\xrightarrow{\Phi}$ $\Phi(x) = 0.07$ "Not a monkey"

The Relevance Mapping Problem

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.

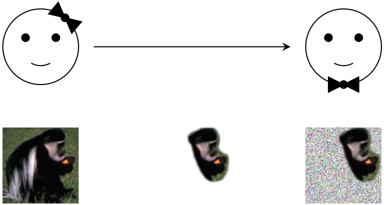
Original image x

Relevant components S

Non-relevant components S^c

Rate-Distortion Viewpoint

Alice Bob



$\Phi(x) = 0.97$ "Monkey"

Original image x Partial image S Random completion y

$\Phi(y) = 0.91$ "Monkey"

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:

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

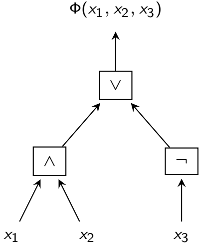
$\rightsquigarrow$ *Use this viewpoint for the definition of a relevance map!*

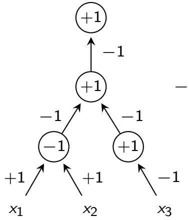
Finding a minimizer of $R(\epsilon)$

or even approximating it is very hard!

Hardness Results

Boolean Functions as ReLU Neural Networks:



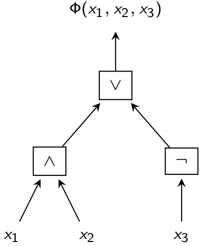


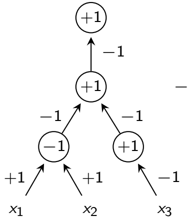
ReLU activation function $\varrho(x) = \max\{0, x\}$

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

Hardness Results

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.

Hardness Results

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!

Hardness Results

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!

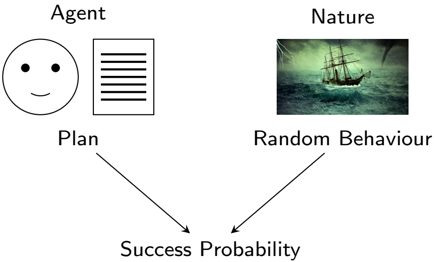
What is NP^{PP}?

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	$S \subseteq \{1, \dots, d\}$	
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	$\lvert S \rvert$	$\lVert s \rVert_1$

Resulting Minimization Problem:

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

Observations

Distortion:

$$D(s) = \mathbb{E}\left[\frac{1}{2}\left(\Phi(x) - \Phi(y)\right)^2\right]$$
$$= \frac{1}{2}\left(\Phi(x) - \mathbb{E}\left[\Phi(y)\right]\right)^2 + \frac{1}{2}\operatorname{cov}\left[\Phi(y)\right]$$

Obfuscation:

$$\mathbb{E}\left[y\right] = s \odot x + (1 - s) \odot \mathbb{E}\left[n\right]$$
$$\operatorname{cov}\left[y\right] = \operatorname{diag}(1 - s)\operatorname{cov}\left[n\right]\operatorname{diag}(1 - s)$$

Observations

$$\mathbb{E}[y], \mathrm{cov}[y] \xrightarrow{\quad\Phi\quad} \mathbb{E}[\Phi(y)], \mathrm{cov}[\Phi(y)]$$

Observations

$$\mathbb{E}[y], \operatorname{cov}[y] \xrightarrow{\quad\quad\quad\Phi\quad\quad\quad} \mathbb{E}[\Phi(y)], \operatorname{cov}[\Phi(y)]$$

Generic Approach:

- Estimate using sample mean and sample covariance
- Possible for any classifier function Φ
- Might require large number of samples

Observations

$$\mathbb{E}[y], \operatorname{cov}[y] \xrightarrow{\quad\quad\quad\Phi\quad\quad\quad} \mathbb{E}[\Phi(y)], \operatorname{cov}[\Phi(y)]$$

Generic Approach:

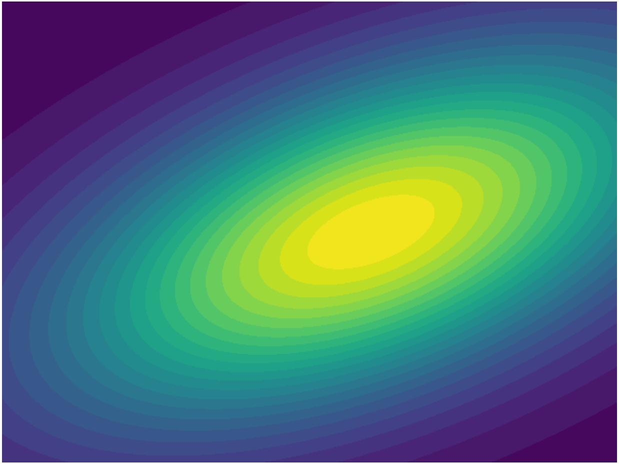
- Estimate using sample mean and sample covariance
- Possible for any classifier function Φ
- Might require large number of samples

Neural Network Approach:

- Use compositional structure of Φ
- Propagate distribution through the layers
- Project to simple family of distributions at each layer

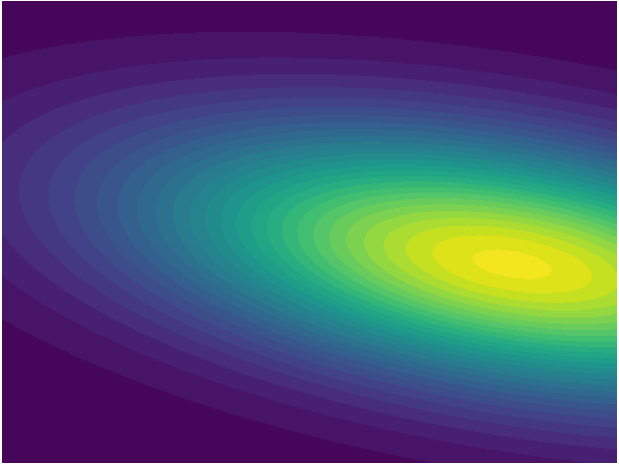
Assumed Density Filtering

Input distribution: $\mathcal{N}(\mu_{\text{in}}, \sigma_{\text{in}})$



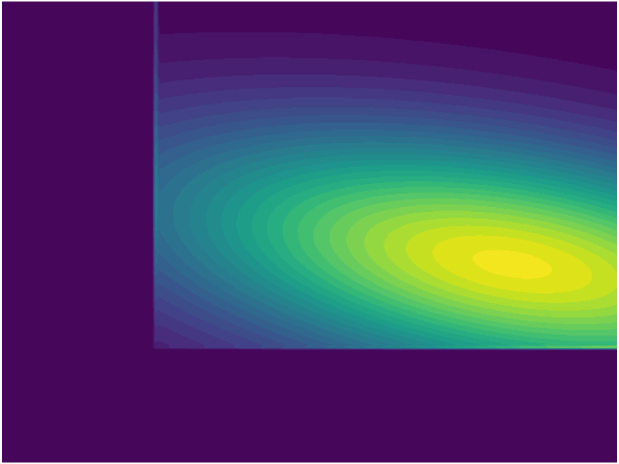
Assumed Density Filtering

Affine transform: $\mathcal{N}(W\mu_{\text{in}} + b, W\sigma_{\text{in}}W^\top)$



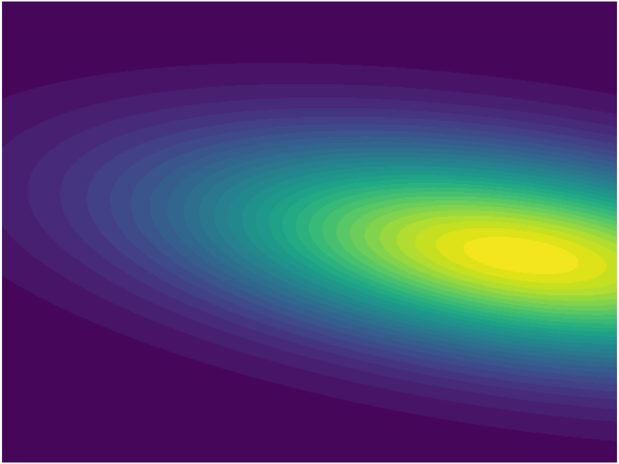
Assumed Density Filtering

ReLU activation: Not Gaussian anymore



Assumed Density Filtering

Moment matching output distribution: $\mathcal{N}(\mu_{\text{out}}, \sigma_{\text{out}})$

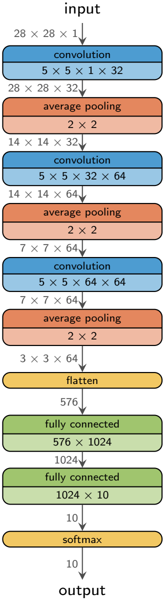


Numerical Experiments

MNIST Experiment

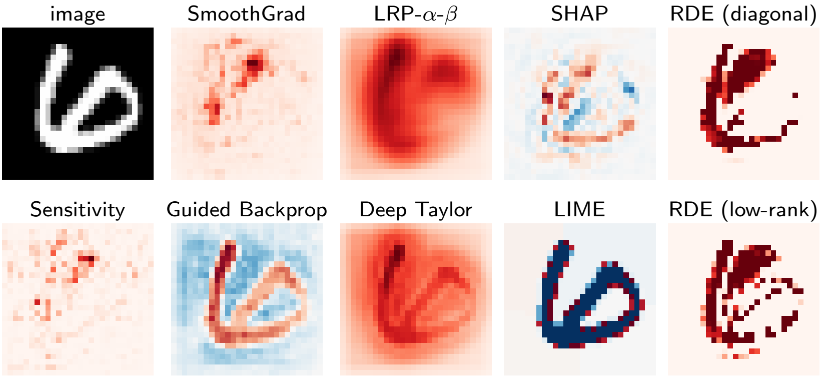
Data Set	
Image size	$28 \times 28 \times 1$
Number of classes	10
Training samples	50000

Test accuracy: 99.1%



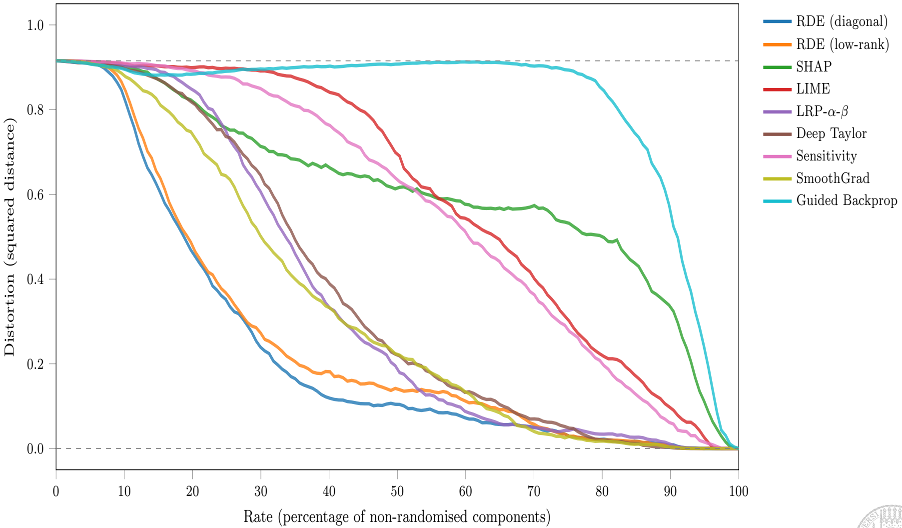
MNIST dataset of handwritten digits (LeCun, Cortes, 1998)

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



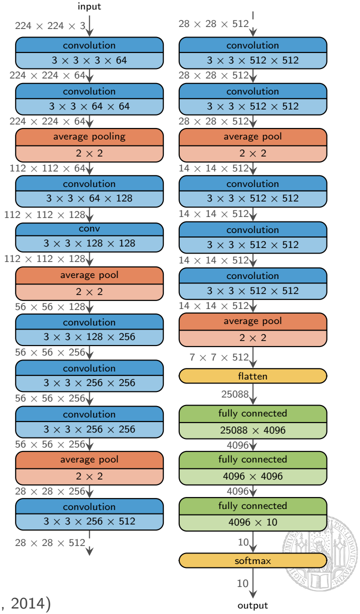
SmoothGrad (Smilkov, Thorat, Kim, Viégas, Wattenberg, 2017), Layer-wise Relevance Propagation (Bach, Binder, Montavon, Klauschen, Müller, Samek, 2015), SHAP (Lundberg, Lee, 2017), Sensitivity Analysis (Simonyan, Vedaldi, Zisserman, 2013), Guided Backprop (Springenberg, Dosovitskiy, Brox, Riedmiller,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

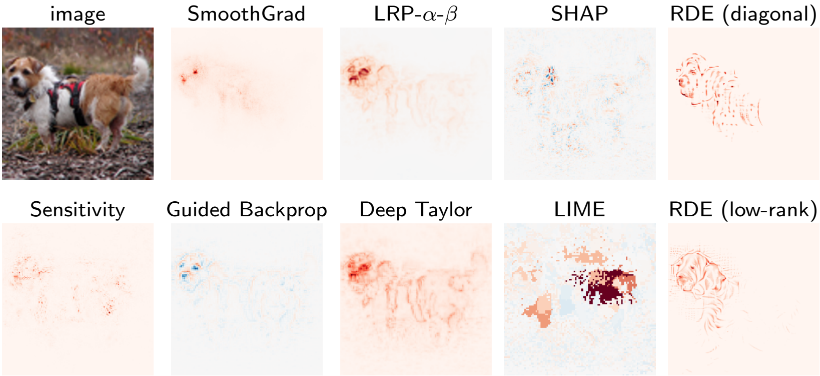
Test accuracy: 93.5%

(VGG-16 convolutions pretrained on Imagenet)



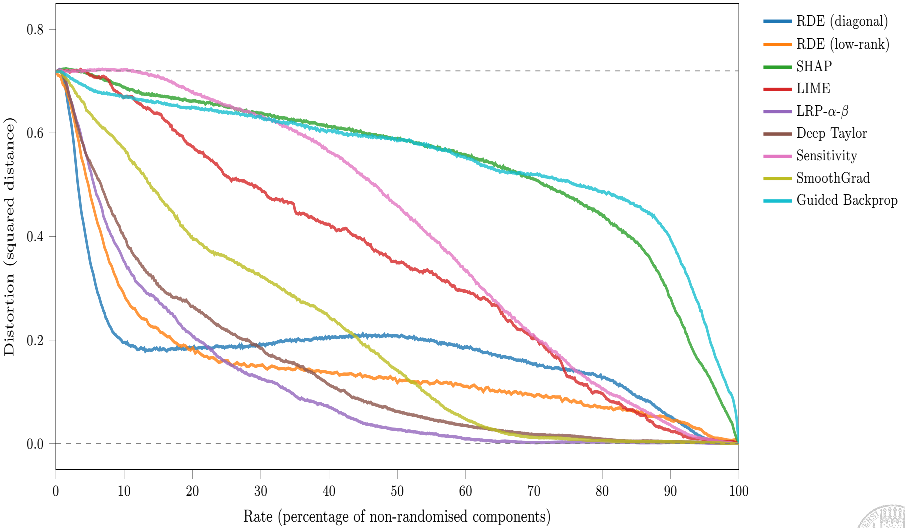
STL-10 dataset (Coates, Lee, Ng, 2011), VGG-16 network (Simonyan, Zisserman, 2014)

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)

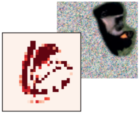
Interpretable Machine Learning

for Challenging Modalities

Desiderata

Problems:

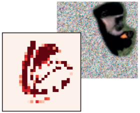
- Modifying the image with random noise or some background color might lead to the obfuscation not being in the domain of the network.
  $\rightsquigarrow$ *Does this give meaningful information about why the network made its decisions?*
- The explanations are pixel-based.
  $\rightsquigarrow$ *Does this lead to useful information for different modalities?*



Desiderata

Problems:

- Modifying the image with random noise or some background color might lead to the obfuscation not being in the domain of the network.
  $\rightsquigarrow$ *Does this give meaningful information about why the network made its decisions?*
- The explanations are pixel-based.
  $\rightsquigarrow$ *Does this lead to useful information for different modalities?*



Goal:

- Take the conditional data distribution into account.
- Ensure that specifics of various modalities can be handled.

Obfuscating Correctly

Recall for $s \in [0,1]^d$:

$$D(s) = \mathbb{E}_{y \sim \Upsilon_s}\left[\frac{1}{2}\left(\Phi(x) - \Phi(y)\right)^2\right]$$

How do we obfuscate according to the conditional data distribution?

Obfuscating Correctly

Recall for $s \in [0,1]^d$:

$$D(s) = \mathbb{E}_{y \sim \Upsilon_s} \left[\frac{1}{2} \left(\Phi(x) - \Phi(y) \right)^2 \right]$$

How do we obfuscate according to the conditional data distribution?

Generative Transformation (Chang, Creager, Goldenberg, Duvenaud; '19):

- Let $\mathcal{D}$ be the training data distribution.
- Use an inpainting network G so that a critic has trouble deciding whether the obfuscation

$$y := x \odot s + G(x, s, n) \odot (1 - s)$$

came from $\mathcal{D}$.

$\rightsquigarrow$ *Sampling from the conditional data distribution* $\mathcal{D}|_{y_s = x_s}$.

General Approach (Heiß, Levie, Resnick, K, Bruna; '20)

Optimization Problem:

We consider the following minimization problem:

$$\min_{s\in\{0,1\}^d} \mathbb{E}_{y\sim\Upsilon_s} \left[\frac{1}{2}(\Phi(x) - \Phi(y))^2\right] + \lambda\|s\|_1,$$

where y is generated by a trained inpainting network G as

$$y := x \odot s + G(x, s, n) \odot (1 - s).$$

General Approach (Heiß, Levie, Resnick, K, Bruna; '20)

Optimization Problem:

We consider the following minimization problem:

$$\min_{s\in\{0,1\}^d} \mathbb{E}_{y\sim\Upsilon_s}\left[\frac{1}{2}(\Phi(x)-\Phi(y))^2\right]+\lambda\|s\|_1,$$

where y is generated by a trained inpainting network G as

$$y := x \odot s + G(x,s,n)\odot(1-s).$$

Requirements of Different Modalities: Can be applied ...

- ... to images, but also audio data, etc.
- ... after a transform (e.g. a dictionary) to allow more complex features.

Conceptually general and flexible interpretability approach!

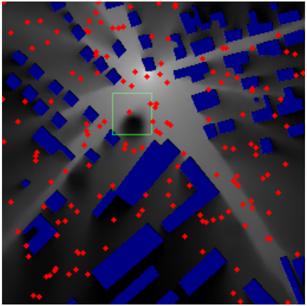
Audio Processing

NSynth Dataset:

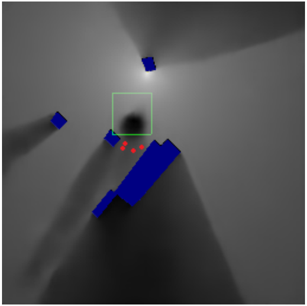
Instrument	Magnitude Importance	Phase Importance
Organ	0.829	1.0
Guitar	0.0	0.999
Flute	0.092	1.0
Bass	1.0	1.0
Reed	0.136	1.0
Vocal	1.0	1.0
Mallet	0.005	0.217
Brass	0.999	1.0
Keyboard	0.003	1.0
String	1.0	0.0

Telecommunication

RadioUNet (Levie, Cagkan, K, Caire; 2020):



Estimated map



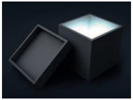
Explanation

Conclusions

What to take Home...?

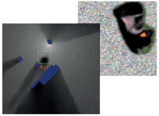
Deep Learning:

- A theoretical foundation of neural networks is largely missing: *Expressivity, Learning, Generalization, and Interpretability*.
- Deep neural networks act still as a *black box*.



Interpretability:

- Determining which input features are *most relevant* for a decision.
- We provide a precise mathematical notion for *relevance* based on *rate-distortion theory*.
- Computing the *minimal rate* is *hard*.
- We introduce a *general and flexible interpretability approach* for various modalities, based on a *relaxed version*.
- On classical examples, *outperforms current methods* for smaller rates.



THANK YOU!

References available at:

`www.ai.math.lmu.de/kutyniok`

Check related information on Twitter at:

`@GittaKutyniok`

Upcoming Book:

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