Methods for Understanding Deep Neural Networks and their Predictions

Grégoire Montavon (TU Berlin)
Joint work with: Wojciech Samek, Klaus-Robert Müller, Sebastian Lapuschkin, Alexander Binder, Leila Arras, etc.

4 December 2017, WIFS, Rennes
Deep Networks Excel at Image Recognition

- 2012: AlexNet 83.6%
- 2015: Googlenet 93.3%
- 2017: Inception-v4 + Res 96.9%

big data  big compute  big models
A Common Machine Learning Problem: Confounding Factors

Desired behavior:
- hidden variable (e.g. medical condition)
- observation (e.g. medical image)

Actual behavior:
- confounding factor (e.g. hospital)

ML
“Big Data” and “Big Confusion”
many (sometimes unsuspected) confounding factors

different qualities of measurement
different subjects

technology evolves over time
different ways of labeling examples
different levels of concentration
different environments
different protocols
time of the day

Can we verify that the model predicts based on the task-relevant features? Yes: Interpretable ML.
Interpretable Machine Learning

- Growing research area in machine learning.
- Aims to present the reasoning of the ML system to the human, so that he can verify it and understand it better.
- **Example:** Heatmap explaining what pixels a deep neural network model uses to classify as “car”.

![Input image](image1.png)  ![Explanation of the DNN classifying it as “car”](image2.png)
Techniques of Interpretability

Active research topic. Many methods have proposed (Symonian’13, Landecker’13, Zeiler’14, Bach’15, Ribeiro’16, Nguyen’16, Montavon’17).

Example: Layer-wise relevance propagation (Bach’15)

\[
R_j = \sum_k \left( \alpha \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} - \beta \frac{a_j w_{jk}^-}{\sum_j a_j w_{jk}^-} \right) R_k
\]
Problems where Interpretability can Help

- **Control Systems**
  Can interpreting a ML classifier help us to identify its incorrect decisions, especially when the latter can endanger peoples life?
  - Automated medical systems, self-driving cars, robots.

- **Complex World**
  Can interpretable ML help us to better understand complex problems?
  - Fan Hui, 2016 about AlphaGo: “It’s not a human move. I’ve never seen a human play this move.”
  - Extracting insights from a deep network trained to predict quantum-chemical electronic properties.
Interpretable ML for Model Validation

Bojarski’17: Self-Driving Cars

Input:

Explanation:

Interpretability is crucial for applications where a single wrong decision can be extremely costly.
Part 1:
Deep Neural Networks Basics
Learning Hierarchical Representations

- large-margin, ridge, ...
- feature map, kernel
- more complex features
- low-level features (e.g. edges)
- pixel values
- output

- neurons, update rule
The Structure of Deep Representations

Multiple neurons with similar structure, but with different weight parameters.

Compose them into a deep layered architecture.

Map the architecture to the problem of interest

\[
y_j = \max(0, \sum_i x_i w_{ij})
\]

\[
h = \max(0, W^{(1)} x)
\]

\[
y = \max(0, W^{(2)} h)
\]
How to Learn in a Neural Network

- **Observation:** Neural network is a function

\[ y = f(x; \theta) \quad \theta = [W^{(1)}, \ldots, W^{(L)}] \]

- **Idea:** define an error function, and minimize the function by gradient descent:

\[ \mathcal{E}(\theta) = \sum_n \left( f(x_n; \theta) - t_n \right)^2 \]

\[ \Delta_n \]

\[ \theta \leftarrow \theta - \gamma \cdot \nabla_{\theta} \mathcal{E}(\theta) \]

\[ \sum_n 2\Delta_n \nabla_{\theta} f(x_n; \theta) \]
Computing the Gradient of a Neural Network

- **Gradient:**
  \[ \nabla_\theta f(x; \theta) \]

- **Example:** 2-layer net
  \[ y = f(x; [W^{(1)}, W^{(2)}]) \]

- **Key idea:** Use chain rule
  \[ \delta(W^{(2)}) = \frac{\partial y}{\partial W^{(2)}} \]
  \[ \delta(W^{(1)}) = \frac{\partial y}{\partial W^{(1)}} = \frac{\partial y}{\partial h} \otimes \frac{\partial h}{\partial W^{(1)}} \]

**Diagram:**
- \[ h = f_1(x; W^{(1)}) \]
- \[ y = f_2(h; W^{(2)}) \]
Example: Gradient of a ReLU layer

\[ h = \max(0, z) \]

\[ \frac{\partial y}{\partial z} = \frac{\partial h}{\partial z} \odot \frac{\partial y}{\partial h} \]

\[ \text{step}(z) \quad \text{received from higher layer} \]

```python
class ReLU:
    def forward(self, Z):
        self.Z = Z
        return numpy.maximum(0, Z)
    def gradprop(self, DH):
        return (self.Z > 0) * DH
```

14/63
Stochastic Gradient Descent (SGD)

\[ \mathcal{E}(\theta) = \frac{1}{N} \left[ \mathcal{E}_1(\theta) + \cdots + \mathcal{E}_N(\theta) \right] \]

Advantages:

Faster convergence (adds noise but no need to evaluate error on the whole training data at each iteration).

Noise is actually good (improves generalization error).

\[ i \leftarrow \text{random}(1, N) \]
\[ \theta \leftarrow \theta - \gamma \frac{\partial E_i}{\partial \theta} \]
GD vs. SGD Convergence

$\log \mathcal{E}_{\text{train}}(\theta)$

generalization error starts increasing (overfitting)

$\text{SGD}$

$\text{GD}$

$t$
Improving Generalization Error

- **Noise in training**: Use SGD instead of GD
- **Additive penalty**: Add a term to the error function that punishes complex models (e.g. L2 penalty on model’s weights)
- **Noise in model**: Randomly turn off half of the neurons at training time (→ learn an exponential ensemble). Method is called “Dropout” (Srivastava’14)
- **Transfer learning**: e.g. multi-task learning (Caruana’98), generative pretraining (Hinton’06), pretraining on a task with many labels and fine-tuning on a task with fewer labels.
- **Hard-coding domain knowledge into the network**: Restricted connectivity, shared weights, pooling layers.
Neural Networks with Various Structures

Fully connected:

Convolutional:
Part 2:

Techniques of Interpretation and Explanation
Understanding Deep Nets: Two Views

**mechanistic understanding**

Understanding what mechanism the network uses to solve a problem or implement a function.

**functional understanding**

Understanding how the networks relates the input to the output variables.

\[ f: \mathbb{R}^d \rightarrow \mathbb{R} \]
Understanding Deep Nets: Two Problems

**model analysis**

possible approach
- build prototypes of "typical" examples of a certain class.

**decision analysis**

possible approach
- identify which input variables contribute to the prediction.
Model Analysis: Generating Prototypes

Example: Understanding how a deep net models the ImageNet class “Junco” by:

**Approach 1:** Looking which image in the dataset maximally activates that class.

**Approach 2:** Synthesizing an input image with maximum activation for that class.

Images from *Nguyen’16*
Decision Analysis: Identifying Relevant Variables

Example: Understanding what makes given images belong to their CIFAR-10 image class.

CIFAR-10 Images of birds

Pixel-wise explanation

Observation: each image has a different explanation.

Question: how to find which pixels are relevant for a given image?
Explaining by Decomposing

Importance of a variable is the share of the function score that can be attributed to it.

Decomposition property: \( f(x_1, \ldots, x_d) = \sum_{i=1}^{d} R_i \)
Sensitivity Analysis

We compute for each pixel:

\[ R_i = \left( \frac{\partial f}{\partial x_i} \right)^2 \]
Sensitivity Analysis and Decomposition

Question: If sensitivity analysis computes a decomposition of something: Then, what does it decompose?

\[ R_i = \left( \frac{\partial f}{\partial x_i} \right)^2 \]

\[ \sum_{i=1}^{d} R_i = \| \nabla f(x_1, \ldots, x_d) \|^2 \]
Sensitivity Analysis and Decomposition

Sensitivity analysis explain a variation of the function, not the function value itself.

\[ \sum_{i=1}^{d} R_i = \| \nabla f(x_1, \ldots, x_d) \|^2 \]
Simple Taylor Decomposition

\[ f(x) = f(\tilde{x}) + \langle \nabla f \bigg|_{x=\tilde{x}}, x - \tilde{x} \rangle + \epsilon \]

write it as a sum and identify first-order terms

\[ \sum_i \frac{\partial f}{\partial x_i} \bigg|_{x=\tilde{x}} (x_i - \tilde{x}_i) \]

\( \tilde{x} \) is the root point (similar to the data point but without the evidence)
Simple Taylor Decomposition

Special case: unbiased Deep ReLU nets

\[ f(tx) = tf(x) \quad \Rightarrow \quad f(0) = 0 \]

choose:

\[ \tilde{x} = \lim_{t \to 0} tx \]

then:

\[ R_i = \frac{\partial f}{\partial x_i} \bigg|_x \cdot x_i \]
Sensitivity vs. Simple Taylor Decomposition

\[ R_i = \left( \frac{\partial f}{\partial x_i} \bigg|_x \right)^2 \]

\[ R_i = \frac{\partial f}{\partial x_i} \bigg|_x \cdot x_i \]
Gradient Shattering [Montufar’14, Balduzzi’17]

Explanation methods should not rely on the gradient.
Beyond Gradient-Based Techniques

Gradient propagation (→ Simple Taylor)

\[ \delta_j = 1_{a_j > 0} \sum_k w_{jk} \delta_k \]

Bach'15 Layer-wise relevance propagation (LRP)

\[ R_j = \sum_k \left( \alpha \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} - \beta \frac{a_j w_{jk}^-}{\sum_j a_j w_{jk}^-} \right) R_k \]
Understanding LRP

\[ R_j = \sum_k \left( \alpha \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} - \beta \frac{a_j w_{jk}^-}{\sum_j a_j w_{jk}^-} \right) R_k \]

\[ \alpha = 1 \quad \beta = 0 \]

Let’s consider a simplified version (no negative relevance, less sparse).

\[ R_j = \sum_k \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} R_k \]

Application of this rule can be understood as a “Deep Taylor Decomposition” Montavon’17
Question: Suppose that we have propagated the relevance until a given layer. How should the relevance be propagated to the previous layer?

Idea: Perform a Taylor decomposition of the function $R_k((a_j)_j)$
Proposition: relevance is (approximately) a multiple of the neuron activation.

\[ R_k \approx a_k \cdot \text{const} \]

Inductive reasoning:
Assume that it holds for \( R_k \), then show that it also holds for \( R_j \)

\[
R_j = \sum_k \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} R_k
\]

\[
R_j \approx a_j \cdot \sum_k w_{jk}^+ \left( \frac{\sum_j a_j w_{jk} + b_j}{\sum_j a_j w_{jk}^+} \right)^+ \cdot \text{const}
\]

\[ R_j \approx a_j \cdot \text{const} \]
Relevance as a Function [Montavon’17]

We can build a relevance function

\[ R_k((a_j)_j) = \max(0, \sum_j a_j w_{jk} + b_k) \cdot \text{const.} \]

\[ = \max(0, \sum_j a_j w'_{jk} + b'_k) \]

that we can use to decompose on the layer before.
Decomposing on the Lower-Layer [Montavon’17]

Relevance function:

\[ R_k((a_j)_j) = \max(0, \sum_j a_j w'_{jk} + b'_k) \]

Taylor expansion:

\[ R_k((a_j)_j) = \sum_j \left. \frac{\partial R_k}{\partial a_j} \right|_{(\tilde{a}_j)_j} \cdot (a_j - \tilde{a}_j^{(k)}) \]

Summands have the closed-form solution

\[ R_{j\leftarrow k} = \frac{v_{jk} w_{jk}}{\sum_j v_{jk} w_{jk}} R_k \]

where \( v_{jk} \propto a_j - \tilde{a}_j^{(k)} \) is the root search direction.
## Interpreting Search Direction

<table>
<thead>
<tr>
<th>root point</th>
<th>search direction</th>
<th>case 1</th>
<th>case 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$v_{jk} = w'_{jk}$ nearest root</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$v_{jk} = a_j$ origin (simple Taylor)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>blue</td>
<td>$v_{jk} = a_j 1_{w'_{jk} &gt; 0}$ reduces to LRP-$\alpha_1\beta_0$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram](image1.png)  
![Diagram](image2.png)
Building Rules for Specific Domains [Montavon’17]

<table>
<thead>
<tr>
<th>Input domain</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLU activations ((a_j \geq 0))</td>
<td>(R_j = \sum_k \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} R_k)</td>
</tr>
<tr>
<td>Pixel intensities ((x_i \in [l_i, h_i],) (l_i \leq 0 \leq h_i))</td>
<td>(R_i = \sum_j \frac{x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}{\sum_i x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-} R_j)</td>
</tr>
<tr>
<td>Real values ((x_i \in \mathbb{R}))</td>
<td>(R_i = \sum_j \frac{w_{ij}^2}{\sum_i w_{ij}^2} R_j)</td>
</tr>
</tbody>
</table>

Rules can also be designed to not only fulfill domain constraints but also to match the statistics of the input distribution [Kindermans’17]
Comparing Explanation Methods

sensitivity analysis           simple Taylor           LRP [Bach’15]

Visual intuition suggests that LRP produces better explanations.

But can we *measure* heatmap quality?
Explanation Continuity [Montavon’17b]

Idea: If two examples are almost the same (and their prediction too), then, the explanations should also be almost the same.
Explanation Continuity [Montavon’17b]

LRP explanations are more continuous. Follow a path on the data manifold.
**Explanation Selectivity** [Bach’15, Samek’16]

**Idea:** Removing input variables with high relevance should make the function value drop.
Explanation Selectivity [Bach’15, Samek’16]
Part 3:

Applications
LRP for Comparing Classifiers [Binder’16]
LRP for Comparing Classifiers [Binder’16]

Googlenet focuses on the face of animals (suppresses background noise)
LRP for Comparing Classifiers [Arras’17]

<table>
<thead>
<tr>
<th>ML Model</th>
<th>Test Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoW/SVM ($V = 70631$ words)</td>
<td>80.10</td>
</tr>
<tr>
<td>CNN1 ($H = 1, F = 600$)</td>
<td>79.79</td>
</tr>
<tr>
<td>CNN2 ($H = 2, F = 800$)</td>
<td><strong>80.19</strong></td>
</tr>
<tr>
<td>CNN3 ($H = 3, F = 600$)</td>
<td>79.75</td>
</tr>
</tbody>
</table>

- **SVM/BoW classifier**

  on a roller coaster ride than others. The mental part is usually induced by a lack of clear indication of which way is up or down, i.e: the Shuttle is normally oriented with its cargo bay pointed towards Earth, so the Earth (or ground) is "above" the head of the astronauts. About 50% of the astronauts experience some form of motion sickness, and NASA has done numerous tests in

- **CNN/word2vec classifier**

  on a roller coaster ride than others. The mental part is usually induced by a lack of clear indication of which way is up or down, i.e: the Shuttle is normally oriented with its cargo bay pointed towards Earth, so the Earth (or ground) is "above" the head of the astronauts. About 50% of the astronauts experience some form of motion sickness, and NASA has done numerous tests in
LRP for Comparing Classifiers [Lapushkin’16]

Comparing Performance on Pascal VOC 2007
(Fisher Vector Classifier vs. DeepNet pretrained on ImageNet)

<table>
<thead>
<tr>
<th></th>
<th>aeroplane</th>
<th>bicycle</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fisher</td>
<td>79.08%</td>
<td>66.44%</td>
<td>45.90%</td>
<td>70.88%</td>
<td>27.64%</td>
<td>69.67%</td>
<td>80.96%</td>
</tr>
<tr>
<td>DeepNet</td>
<td>88.08%</td>
<td>79.69%</td>
<td>80.77%</td>
<td>77.20%</td>
<td>35.48%</td>
<td>72.71%</td>
<td>86.30%</td>
</tr>
<tr>
<td></td>
<td>cat</td>
<td>chair</td>
<td>cow</td>
<td>diningtable</td>
<td>dog</td>
<td>horse</td>
<td>motorbike</td>
</tr>
<tr>
<td>Fisher</td>
<td>59.92%</td>
<td>51.92%</td>
<td>47.60%</td>
<td>58.06%</td>
<td>42.28%</td>
<td>80.45%</td>
<td>69.34%</td>
</tr>
<tr>
<td>DeepNet</td>
<td>81.10%</td>
<td>51.04%</td>
<td>61.10%</td>
<td>64.62%</td>
<td>76.17%</td>
<td>81.60%</td>
<td>79.33%</td>
</tr>
<tr>
<td></td>
<td>person</td>
<td>pottedplant</td>
<td>sheep</td>
<td>sofa</td>
<td>train</td>
<td>tvmonitor</td>
<td>mAP</td>
</tr>
<tr>
<td>Fisher</td>
<td>85.10%</td>
<td>28.62%</td>
<td>49.58%</td>
<td>49.31%</td>
<td>82.71%</td>
<td>54.33%</td>
<td>59.99%</td>
</tr>
<tr>
<td>DeepNet</td>
<td>92.43%</td>
<td>49.99%</td>
<td>74.04%</td>
<td>49.48%</td>
<td>87.07%</td>
<td>67.08%</td>
<td>72.12%</td>
</tr>
</tbody>
</table>

Example of a horse image next to the LRP analysis of the Fisher’s decision:
LRP for Comparing Classifiers [Lapushkin'16]
LRP and Region-Based Analysis

\[ f(x) = \sum_{\mathcal{I}} R_{\mathcal{I}}(x) \]

\[ f(x) = \sum_i R_i(x) \]

\[ R_{\mathcal{I}}(x) = \sum_{i \in \mathcal{I}} R_i(x) \]
Comparing Reliance of Different Classifiers on Context (PASCAL VOC 2007)

[Lapuschkin'16]
Interpretable ML for the Complex World

Can we use interpretable ML to make sense of complex systems for which we have little understanding or intuition?

Winning strategies in board games

Quantum chemistry

\[ i\hbar \frac{\partial}{\partial t} \Psi = H\Psi \]

Deep Blue, Alpha Go

DFT-based solvers, numerical simulations

Data, experiments

Neurosciences, biology
Approximating Optimal Strategies with ML

- **Optimal Strategy**: (min-max search, brute-force)

- **Neural Network**: (sequence of layers, interpretable with e.g. sensitivity analysis, or LRP)
Interpreting a DNN Playing “Breakout”

Insight: DNN player keeps track of the ball to position its cursor.
Approximating the Schrödinger Equation

molecular structure (e.g. atoms positions)

DFT calculation of the stationary Schrödinger Equation

\[ H\Phi = E\Phi \]

PBE0, Pedrew’86

molecular electronic properties (e.g. atomization energy)

DTNN, Schütt’17
Interpreting the Schrödinger Equation

Schütt’17 Quantum-Chemical Insights from Deep Tensor Neural Networks

Example: Perturbation analysis of the neural network for various molecules.
Interpreting Human “Left/Right” Decisions

Sturm’16 Understanding in terms of EEG data why a person thinks “left hand” or “right hand”
EEG Data and Pooling Explanations

Relevance is first redistributed on channels and time. It can then be pooled either in time, or in space.

Temporal pooling

\[ f(x) = \sum_{I} R_{I}(x) \]

\[ f(x) = \sum_{i} R_{i}(x) \]

Image from Sturm’16
Summary

- In presence of data heterogeneity, it can be tricky to ensure that the classifier uses the right features.
- Interpretability can be used to validate a trained model, or to learn something from the model.
- Sensitivity analysis explains a variation of the prediction, not the actual prediction.
- Gradient-based methods (e.g. sensitivity and simple Taylor) perform poorly when applied to deep models.
- The LRP technique can identify reliably the input features used by a complex nonlinear deep classifier.
- For deep ReLU nets, LRP can be understood as a deep Taylor decomposition of the neural network function.
Check our webpage with interactive demos, software, tutorials, ...
References


- [Binder’16] Binder, A., Samek, W., Montavon, G., Bach, S., Müller, K.-R. Analyzing and Validating Neural Networks Predictions ICML Workshop on Visualization for Deep Learning, 2016


References


- [Samek’17] Samek, W., Binder, A., Montavon, G., Lapuschkin, S., Müller, K.-R. Evaluating the visualization of what a deep neural network has learned. IEEE Transactions on Neural Networks and Learning Systems, 2017


