NN Interpretability

Aleksandar Aleksandrov
Hans Hao-Hsun Hsu

Outline

- Introduction

- Model-based Methods
  - Activation Maximization

- Decision-based Methods
  - Deconvolutional Network
  - Occlusion Sensitivity
  - Saliency Maps
  - Layer-wise Relevance Propagation

Introduction

- What is NN interpretability?
    - NN interpretability refers to the process of **mapping of abstract concepts in a human-understandable domain**. A collection of features in the human-interpretable domain allows us to **provide possible explanations for the decisions of a model**.

- What types of NN interpretability methods are there?
    - **Model-based methods** (e.g. Activation Maximization) try to explain what does the concepts learned from a model look like. (How does a “dog” typically look like?)

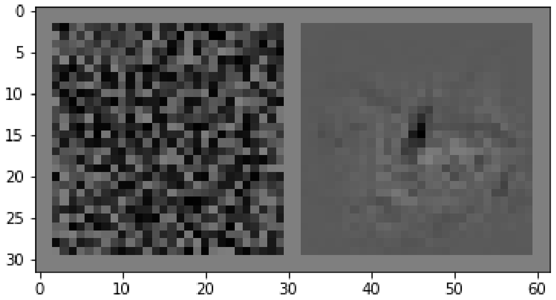
    - **Decision-based methods** (e.g. Layerwise Relevance Propagation) try to explain why did the model assign a certain concept to a premeditated input. (Why is this example classified as “dog”?)

Activation Maximization (AM)

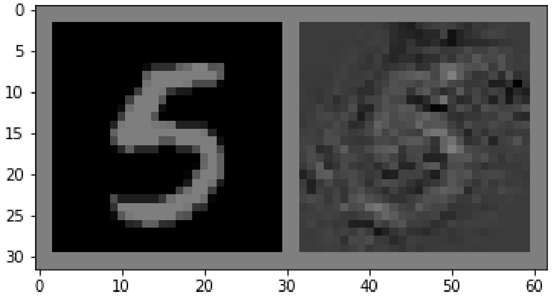
- AM is a **model-based approach** that searches for an **input pattern which elicits a maximum model response** for a class of interest.

- Variations:
  - General AM $\max_{\boldsymbol{x}} \ \log p(\omega_c|\boldsymbol{x}) - \lambda\|\boldsymbol{x}\|^2.$
  - AM with an Expert $\max_{\boldsymbol{x}} \ \log p(\omega_c|\boldsymbol{x}) + \log p(\boldsymbol{x}).$
  - AM in Code Space $\max_{\boldsymbol{z}\in Z} \ \log p(\omega_c \,|\, g(\boldsymbol{z})) - \lambda\|\boldsymbol{z}\|^2,$

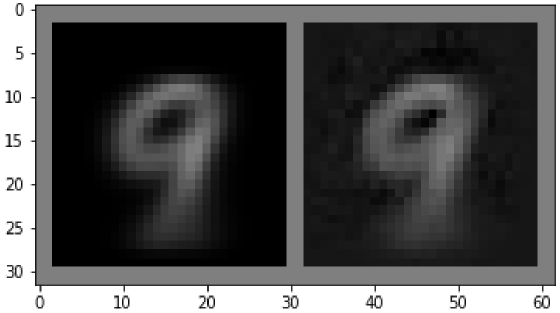
- Random noise (Class 6)



- Random image (Class 5)



- Mean image for class (Class 9)



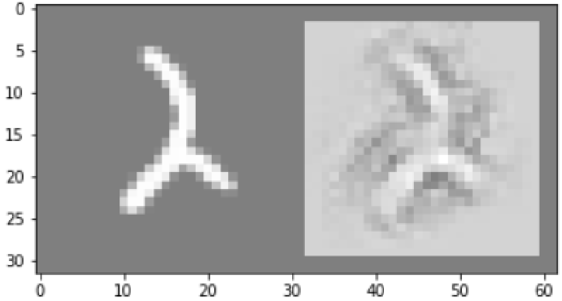
Deconvolutional Network (DeConvNet)

- DeConvNet is a **decision-based approach** for mapping feature activities back to the input pixel space.

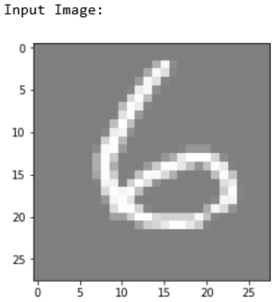
- DeConvNet has the **reversed structure** of a concrete CNN model and **reuses the initially learned weights**.

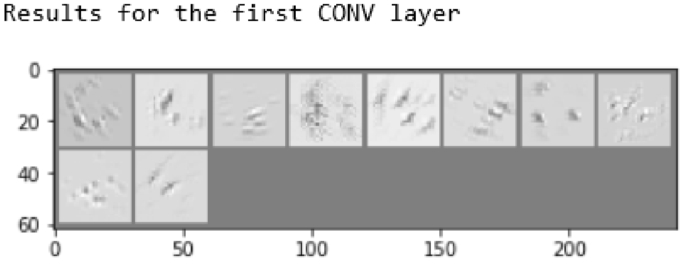
- Variations
    - DeConvNet with all filters
    - DeConvNet with a single filter in a given layer

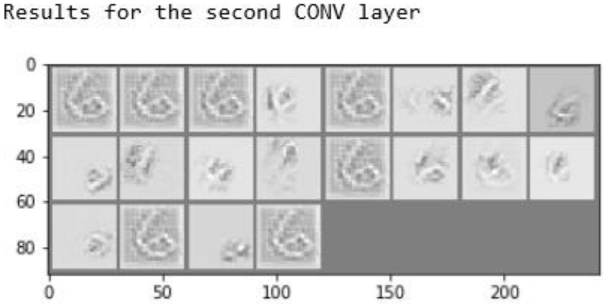
- **DeConvNet with all filters**



- **DeConvNet with a single filter in a given layer**

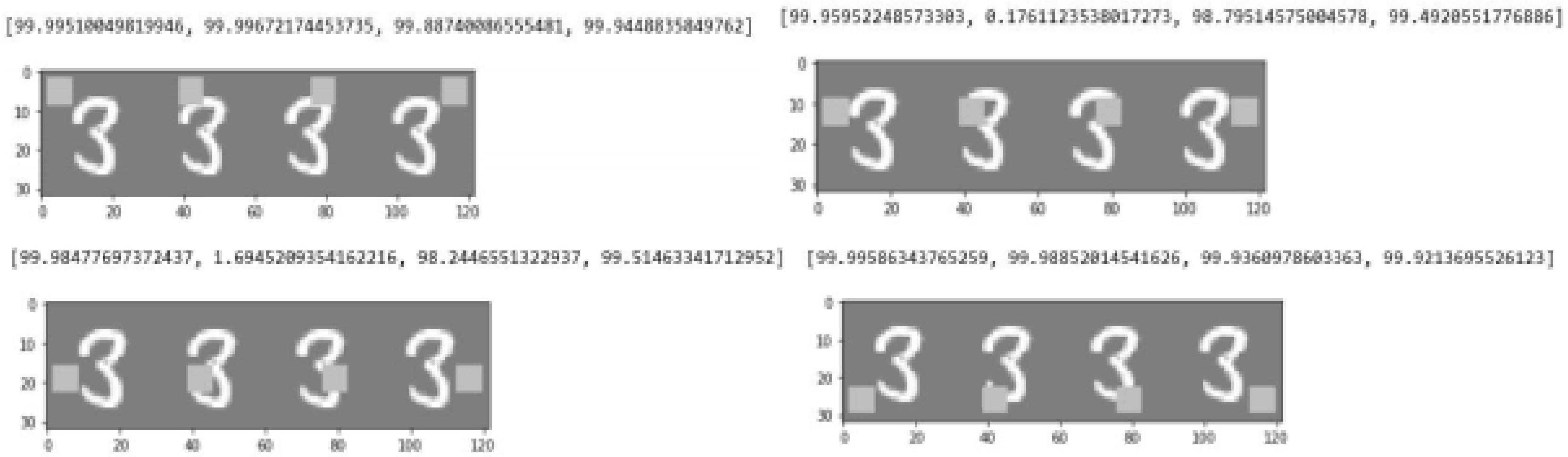






Occlusion Sensitivity

- Occlusion Sensitivity is a **decision-based approach** in which parts of the input are **deliberately obstructed to mislead** the decision of the model
-

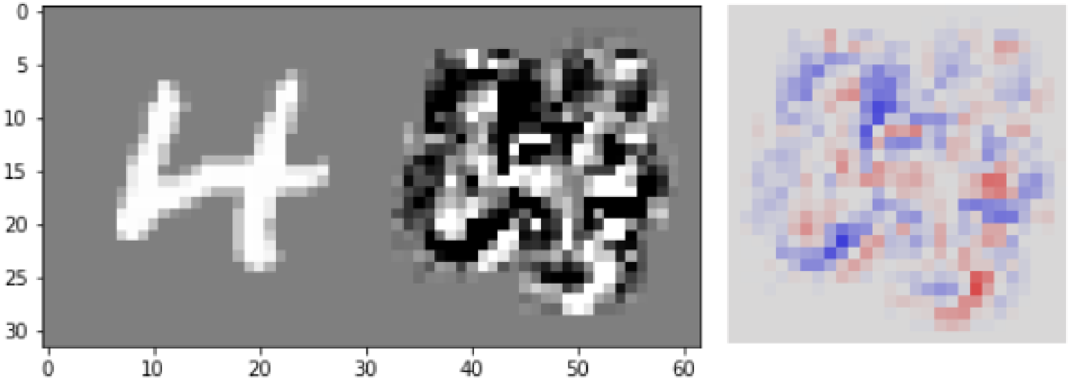


Saliency Maps

- Saliency Map is a **decision-based approach** that indicates which pixels need to be changed the least to affect the class score the most

- **Problem:** Doesn't highlight which pixel causes the prediction of "4"

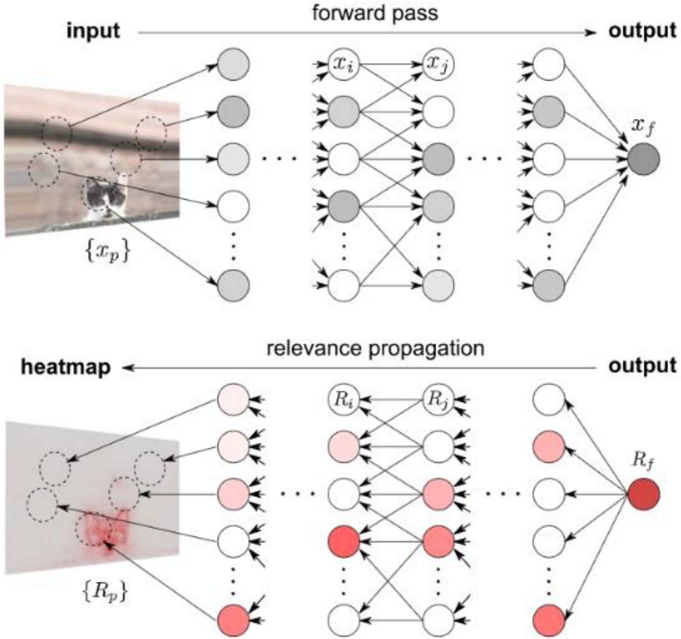
  Not conservative $f(x) = \sum_{i=1}^{V} R(x_i)$



Layer-wise Relevance Propagation

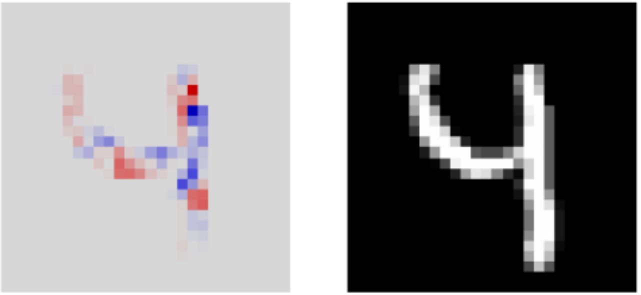
- Layer-wise Relevance Propagation(LRP) is a **decision-based approach** by propagating **relevance scores backward** using a set of purposely designed rules

- Variations:
    - Naive LRP Rule $$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k$$
    - LRP-ε Rule $$R_j = \sum_k \frac{a_j w_{jk}}{\epsilon + \sum_{0,j} a_j w_{jk}} R_k$$
    - LRP-γ Rule $$R_j = \sum_k \frac{a_j \cdot (w_{jk} + \gamma w_{jk}^+)}{\sum_{0,j} a_j \cdot (w_{jk} + \gamma w_{jk}^+)} R_k$$

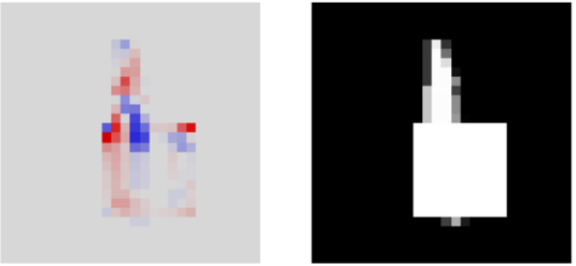


Layer-wise Relevance Propagation

- LRP-ε Rule



- Red pixels: Raise the probability for the class “4”
- Blue pixels: Lower the probability for the class "4"
- Combine with occlusion:



Label 1; Predicted 6

Next Step

- Implementation of further methods
    - Guided Backpropagation
    - Class Activation Maps
    - AM in CodeSpace (e.g. with GAN)
    - DeepDream

- Introduction of uncertainty

Sources

1) Montavon, Grégoire, Wojciech Samek, and Klaus-Robert Müller. “Methods for Interpreting and Understanding Deep Neural Networks.” Digital Signal Processing 73 (2018): 1–15. Crossref. Web.
2) Matthew D. Zeiler and Rob Fergus (2013). Visualizing and Understanding Convolutional NetworksCoRR, abs/1311.2901.
3) Layer-Wise Relevance Propagation: An Overview
4) Olah, et al., "Feature Visualization", Distill, 2017.
5) Olah, et al., "The Building Blocks of Interpretability", Distill, 2018.
6) Karen Simonyan, Andrea Vedaldi, & Andrew Zisserman. (2013). Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps.

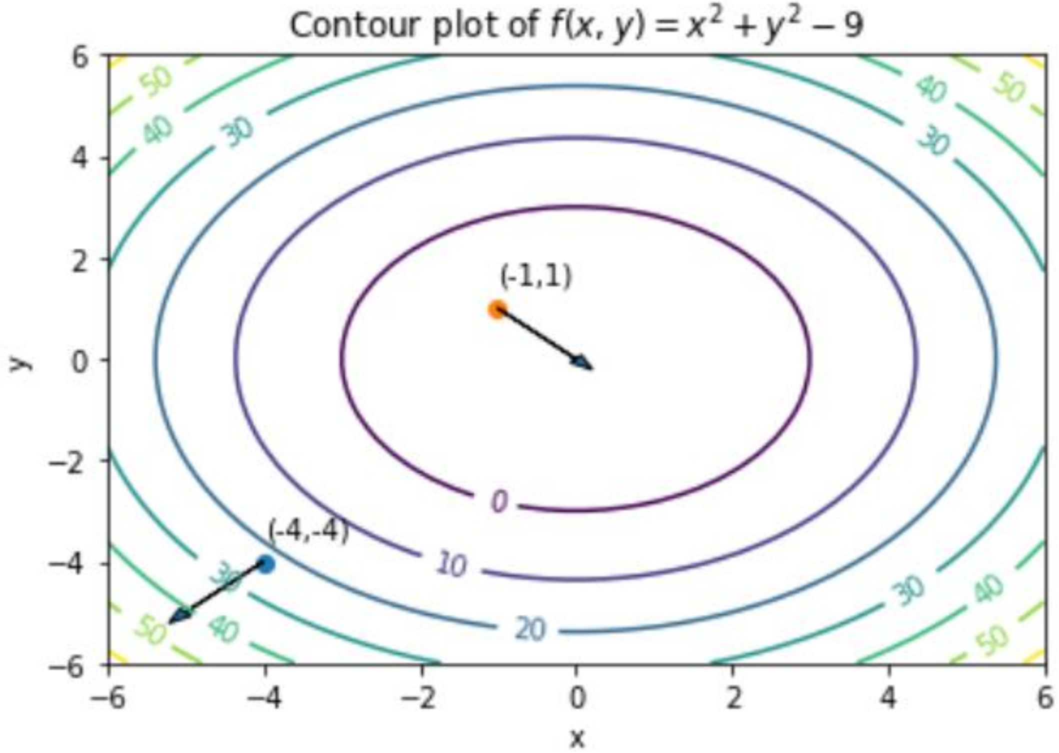
Backup Slides

Saliency Map

$$S_c(I) \approx w^T I + b,$$

$$w = \left. \frac{\partial S_c}{\partial I} \right|_{I_0}$$

Saliency Map



$$f(x,y) = \begin{cases} x^2 + y^2 - 9 < 0 & \text{for class N(egative)} \\ x^2 + y^2 - 9 \geq 0 & \text{for class P(ositive)} \end{cases}$$

$$\nabla f(x,y) = [\,\partial f/\partial x \quad \partial f/\partial y\,] = [\,2x \quad 2y\,]$$

LRP Calculation

element-wise	vector form
$z_k \leftarrow \sum_j a_j w_{jk}^+$	$\boldsymbol{z} \leftarrow W_+^\top \cdot \boldsymbol{a}$
$s_k \leftarrow R_k / z_k$	$\boldsymbol{s} \leftarrow \boldsymbol{R} \oslash \boldsymbol{z}$
$c_j \leftarrow \sum_k w_{jk}^+ s_k$	$\boldsymbol{c} \leftarrow W_+ \cdot \boldsymbol{s}$
$R_j \leftarrow a_j c_j$	$\boldsymbol{R} \leftarrow \boldsymbol{a} \odot \boldsymbol{c}$

```
def lrp(layer,a,R):

    clone = layer.clone()
    clone.W = maximum(0,layer.W)
    clone.B = 0

    z = clone.forward(a)
    s = R / z
    c = clone.backward(s)

    return a * c
```

$$f(x) = \cdots = \sum_{d=1}^{V(l+1)} R_d^{(l+1)} = \sum_{d=1}^{V(l)} R_d^{(l)} = \cdots = \sum_{i=1}^{V(1)} R_d^{(1)}$$

$$c_j = \left[\nabla\left(\sum_k z_k(\boldsymbol{a}) \cdot s_k\right)\right]_j$$

Deconvolutional Network