UVA CS 6316: Machine Learning : 2019 Fall
Course Project: Deep2Reproduce @
https://github.com/qiyanjun/deep2reproduce/tree/master/2019Fall

Learning how to explain neural networks: PatternNet and PatternAttribution

Reproduced by: Vamshi Garikapati, Dawit Kahsay, Aaron Knife, Chijung Jung

Kindermans, Pieter-Jan, et al. "Learning how to explain neural networks: Patternnet and patternattribution." arXiv preprint arXiv:1705.05598 (2017).

12/6/2019

Motivation

Which parts of a neural network matter the most for image classification?

This is "elephant"(y).

Because,

Background

- Neural network classifiers have become proficient at detecting relevant signals
    - filtering irrelevant and distracting components in the data

- These classifiers are considered to be “black-boxed”
    - various techniques have been proposed to gain insight as to how these models operate

- To understand the classifier’s decisions, most of these techniques assume
    - the output signal can be propagated through the network
    - signal arrives at the original image

Background

- Techniques are mostly tested on
  - renowned deep neural networks using high dimensional real world data
  - further complicates how we can understand how these models operate due to complexity

- PatternNet/PatternAttribution
  - aims to solve this by controlling
    - what input images are placed into a simple neural network
    - what output images are generated from the propagation of said images

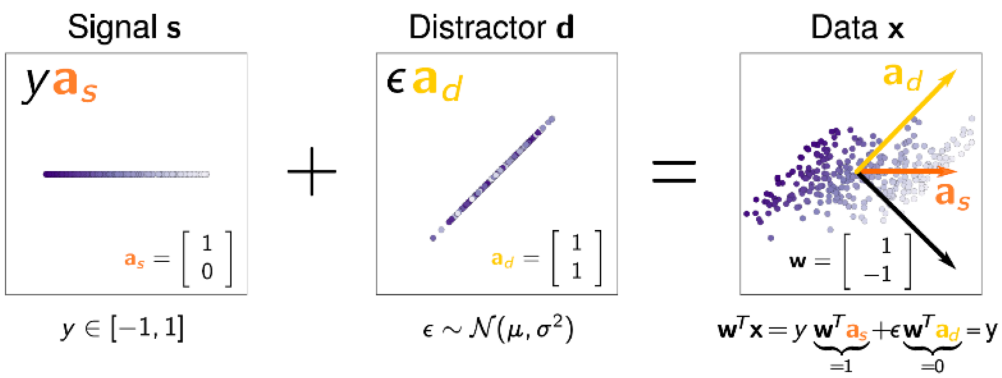
Related Work

- Most of the related work comes from research on other techniques that have tried to understand the decision making process for neural nets via
  - differences in activation function patterns
  - interaction of different layers
  - noise/distraction interference with layers

- One other related work deals with how weight vectors actually operate within neural networks when considering images with multivariate properties

Claim / Target Task

- Claim (Problem definition)
  - Many of the current state-of-the-art interpretability methods are inaccurate even for linear models.
    - e.g.DeConvNet, Guided BackProp, LRP
- Target Task (Approach)
  - Analyze explanation methods including proposed method in the context of the simplest neural network setting.



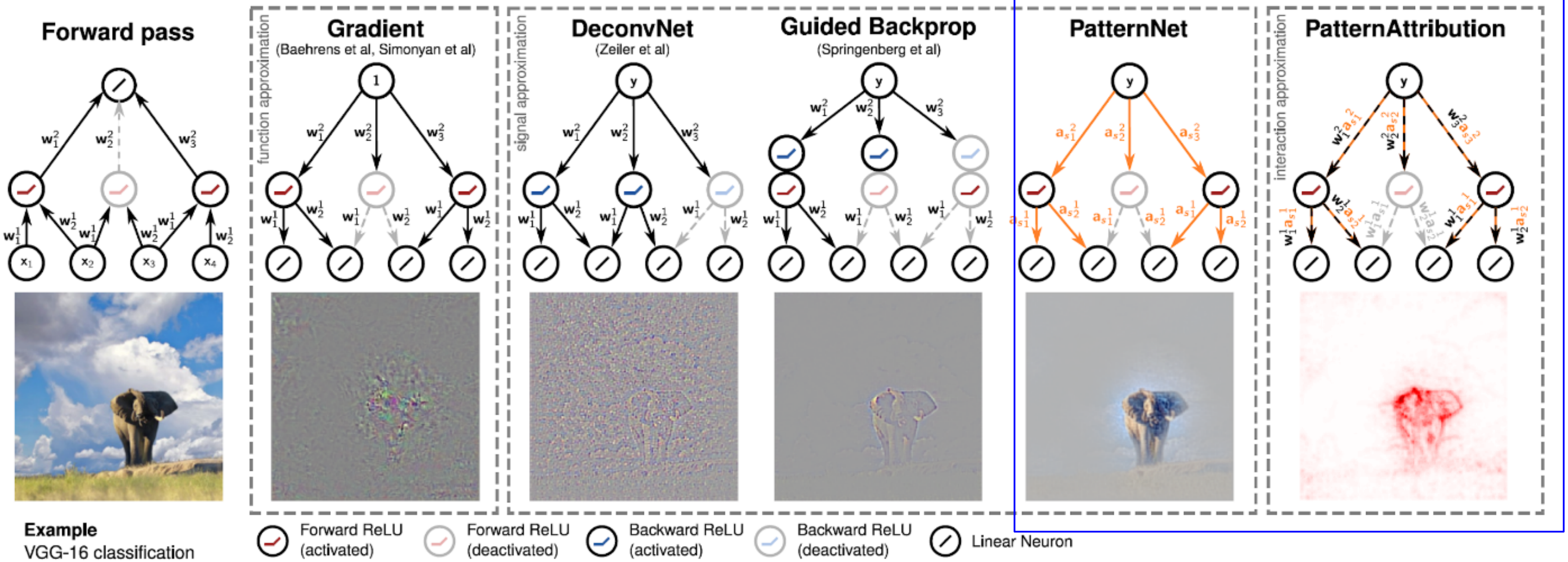
  - Expanded to non-linear models.
    - i.e.VGG-16

[00] https://openreview.net/forum?id=Hkn7CBaTW

An Intuitive Figure Showing WHY Claim

- Different types of explanation methods can be divided into 3 parts of visualization:

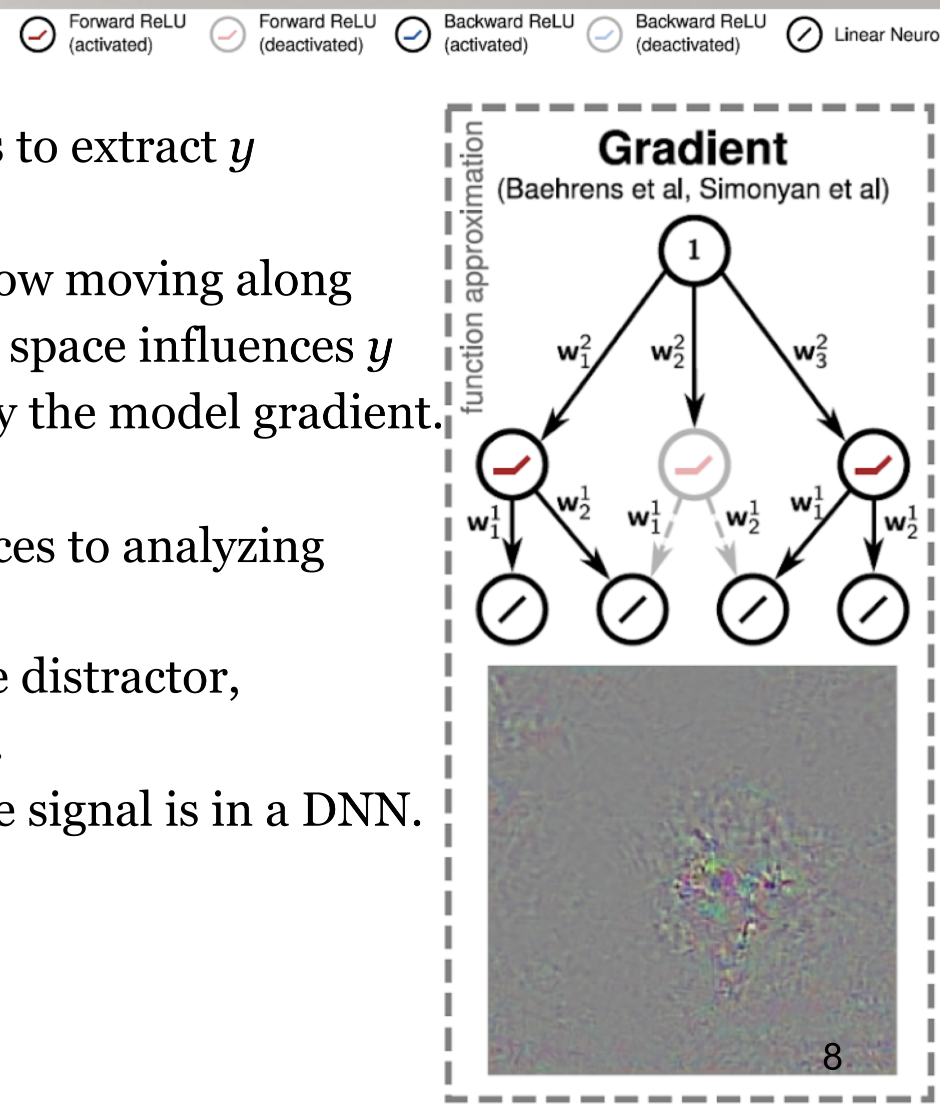
Function | Signal | Attribution



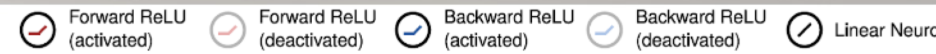
proposed methods

An Intuitive Figure Showing WHY Claim

- Function
  - the operations the model uses to extract y from $\boldsymbol{x}$.
  - The saliency map estimates how moving along a particular direction in input space influences y where the direction is given by the model gradient.
  - In the linear model, this reduces to analyzing the weights $\boldsymbol{w}$.
    - mostly determined by the distractor, not presenting the signal.
    - We cannot know what the signal is in a DNN.

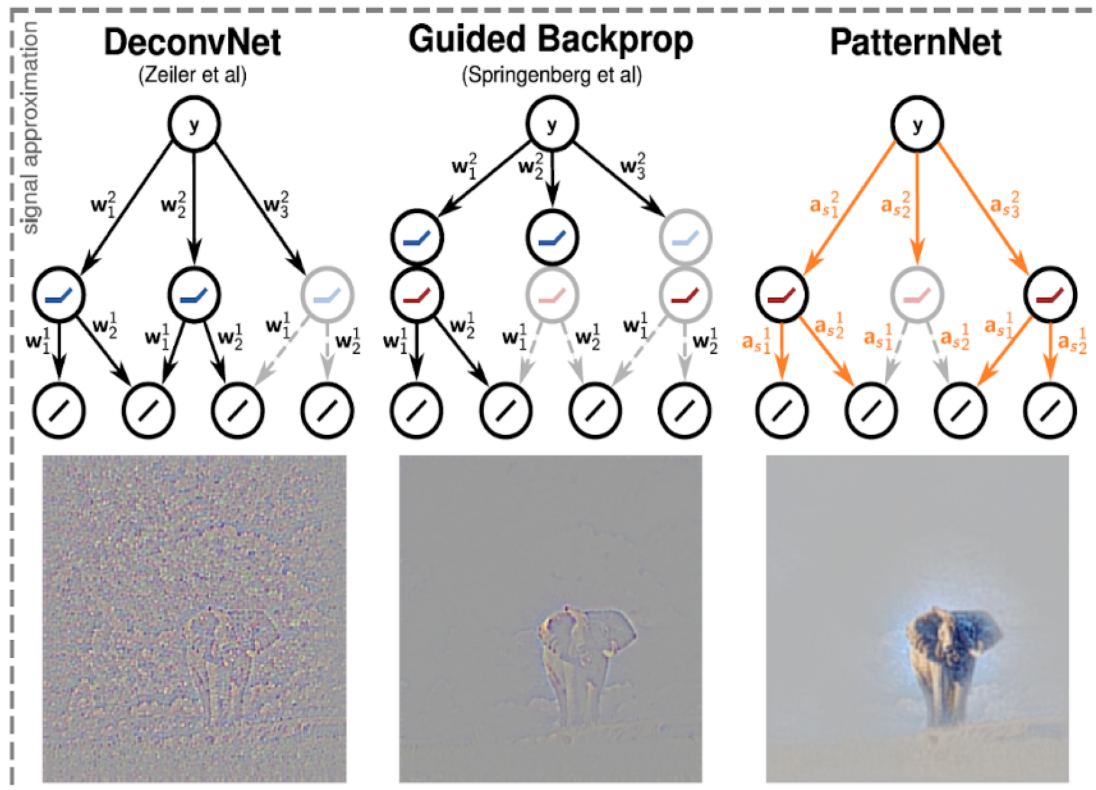


An Intuitive Figure Showing WHY Claim



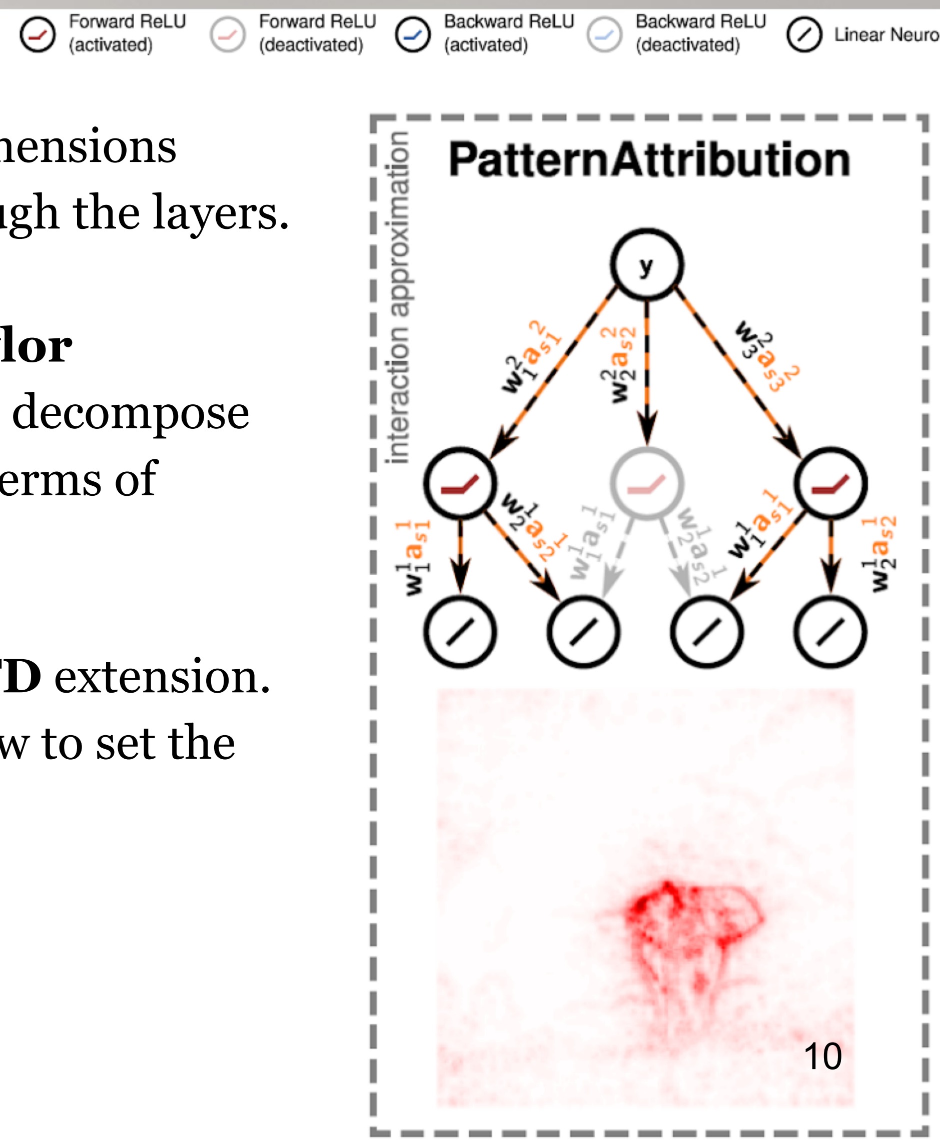
- Signal
  - The component of the data that caused the networks activations.
  - Tells which input pattern originally caused a given activation in the feature maps.

  - In the linear model, **DeConvNet** and **Guided BackProp** don't guarantee to produce the detected signal.
    - They show the filter $\boldsymbol{w}$ only.



An Intuitive Figure Showing WHY Claim

- Attribution
  - Tells how much the signal dimensions contribute to the output through the layers.
  - The key idea of the **deep Taylor decomposition** (**DTD**) is to decompose the activation of a neuron in terms of contributions from its inputs.
  - **PatternAttribution** is a **DTD** extension.
    - It can learn from data how to set the root point.



Proposed Solution (Approaches)

- Functions
  - A way to extract output y from data x. ex) gradients, saliency map
  - Differentiate y by x, and look how output changes as input changes
  - This is using the model's gradients = weight.

$$\rightarrow \begin{cases} y = w^T x \\ \partial y / \partial x = w \end{cases}$$

- Signal
  - Signal: the component that activate model's neuron
  - Look the gradient using backpropagation from output to input space
  - In case of DeConvNet, Guided BackProp, they focus on weights.

→ PatternNet

- Attribution
  - The indicator of how specific signal contribute to output.
  - In linear model, it is element-wise multiplication of signal and weight vector

→ PatternAttribution

Proposed Solution (Quality criterion for signal estimator)

- Derivation

$$w^T x = y$$

$$w^T(s+d) = y \quad \longleftarrow \quad (x = s + d)$$

$$w^T s + w^T d = y$$

$$w^T s = y \quad \longleftarrow \quad (w^T d = 0)$$

$$(w^T)^{-1} w^T s = (w^T)^{-1} y$$

$$\hat{s} = u u^{-1} (w^T)^{-1} y \quad \longleftarrow \quad \text{u = random vector}$$

$$\hat{s} = u(w^T u)^{-1} y \qquad (w^T u \neq 0)$$

Illposed problem.
We need another way.

- Quality measure ρ

$$S(x) = \hat{s} \;,\quad \hat{d} = x - S(x) \;,\quad y = w^T x$$

$$\rho(S) = 1 - max_v corr(w^T x, v^T(x - S(x)))$$

$$= 1 - max_v \frac{v^T cov[y, \hat{d}]}{\sqrt{\sigma^2_{v^T \hat{d}} \sigma^2_y}}$$

  - Good signal estimator makes correlation 0 → big ρ
  - We assume that w is weight from well-trained model.
  - As correlation is invariant to scale, We can add constraints: variance of $v^T \hat{d}$ = variance of y
  - the training method in a process that we fix S(x) and find optival v is Leaset-squares regression.

Proposed Solution
(Detour – existing signal estimator)

- The identity estimator

$$S_x(x) = x$$

  - Assumption: data is consist of signal without distractor
  - When data is image, signal = image
  - When simple linear model, attribution can be calculated from
    (even if distractors exist, it belongs to attribution)

$$r = w \odot x = w \odot s + w \odot d$$

  - However, there's distractor in real data.
    Though it is removed in forward pass, but it is maintained in backward pass by element wise multiplication
  - In case of visualization, a lot of noise can be found.(LRP)

- The filter based estimator

$$S_{\boldsymbol{w}}(\boldsymbol{x}) = \frac{\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{w}}\boldsymbol{w}^T\boldsymbol{x}.$$

  - Assumption: observed signal belongs to the direction of weight.
    e.g. DeConvNet, Guided BackProp
  - Weight should be normalized
  - When linear model, attribution can be calculated from
    (it cannot reconstruct the signal well)

$$r = \frac{w \odot w}{w^T w} y$$

Proposed Solution (PatternNet & PatternAttribution)

- Training method and assumption
  - optimize ρ criterion
  - When correlation between y and d is 0 with all possible vector V, signal estimator S is optimal

  - When the model is linear model, covariance between y and d = 0

$$cov[y, \hat{d}] = 0$$

$$cov[y, x] - cov[y, S(x)] = 0$$

$$cov[y, x] = cov[y, S(x)]$$

- Quality measure

$$\rho(S) = 1 - max_v corr(w^T x, v^T(x - S(x))) \\ = 1 - max_v \frac{v^T cov[y, \hat{d}]}{\sqrt{\sigma^2_{v^T\hat{d}} \sigma^2_y}}$$

Proposed Solution (PatternNet & PatternAttribution)

- The linear estimator

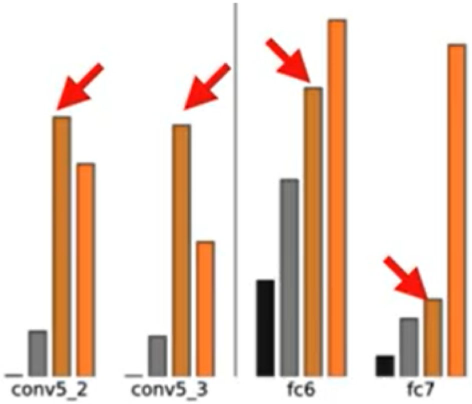
$$S_a(x) = aw^T x = ay$$

  - Linear neuron can be extracted from linear signal on data x
  - Like above formular, we can get signal from linear equation of y
  - When the model is linear model, as covariance between y and d =0,

$$\begin{aligned} & cov[x, y] \\ = & cov[S(x), y] \\ = & cov[aw^T x, y] \\ = & a \cdot cov[y, y] \end{aligned}$$

$$a = \frac{cov[x, y]}{\sigma_y^2}$$

  - Well performed in Convolution layer.
  - As the correlation in the part that Relu is connected to FC layer cannot be erased well, criterion value is low like this.



Proposed Solution (PatternNet & PatternAttribution)

- The two-component(Non-linear) estimator

$$S_{a+-}(x) = \begin{cases} a_+ w^\intercal x & \textbf{if } w^\intercal x > 0 \\ a_- w^\intercal x & \textbf{otherwise} \end{cases}$$

  - This is differ based on the sign of y
  - The information of whether the neuron is activated or not exist in distractor as well.
    This is the reason why the negative y should be considered.
  - Because of ReLu, only positive domain is updated locally, this estimator adjusts like this

$$x = \begin{cases} s_+ + d_+ & \textbf{if } y > 0 \\ s_- + d_- & \textbf{otherwise} \end{cases}$$

  - Covariance between x and y

$$cov(x,y) = \mathbb{E}[xy] - \mathbb{E}[x]\mathbb{E}[y]$$

  - According to the sign, weighted sum:

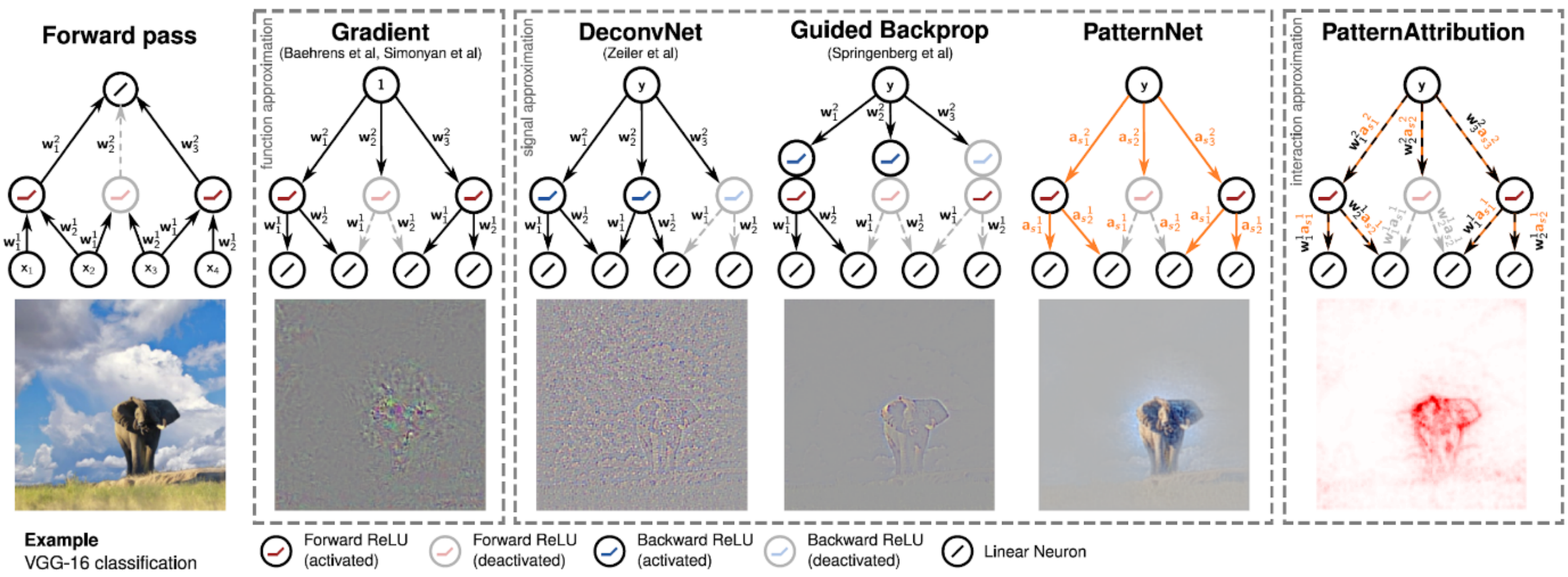
$$\begin{aligned} cov(x,y) &= \pi_+(\mathbb{E}_+[xy] - \mathbb{E}_+[x]\mathbb{E}[y]) \\ &\quad +(1-\pi_+)(\mathbb{E}_-[xy] - \mathbb{E}_-[x]\mathbb{E}[y]) \\ cov(s,y) &= \pi_+(\mathbb{E}_+[sy] - \mathbb{E}_+[s]\mathbb{E}[y]) \\ &\quad +(1-\pi_+)(\mathbb{E}_-[sy] - \mathbb{E}_-[s]\mathbb{E}[y]) \end{aligned}$$

  - Where cov(x,y) = cov(s,y)

$$a_+ = \frac{\mathbb{E}_+[xy] - \mathbb{E}_+[x]\mathbb{E}[y]}{w^\intercal\mathbb{E}_+[xy] - w^\intercal\mathbb{E}_+[x]\mathbb{E}[y]}$$

Proposed Solution (PatternNet & PatternAttribution)

- PatternNet and PatternAttribution



Proposed Solution (PatternNet & PatternAttribution)

- PatternNet and PatternAttribution
  - PatternNet, Linear
    - Because cov(x,y) = cov(s,y)
    - a can be calculated with only x and y

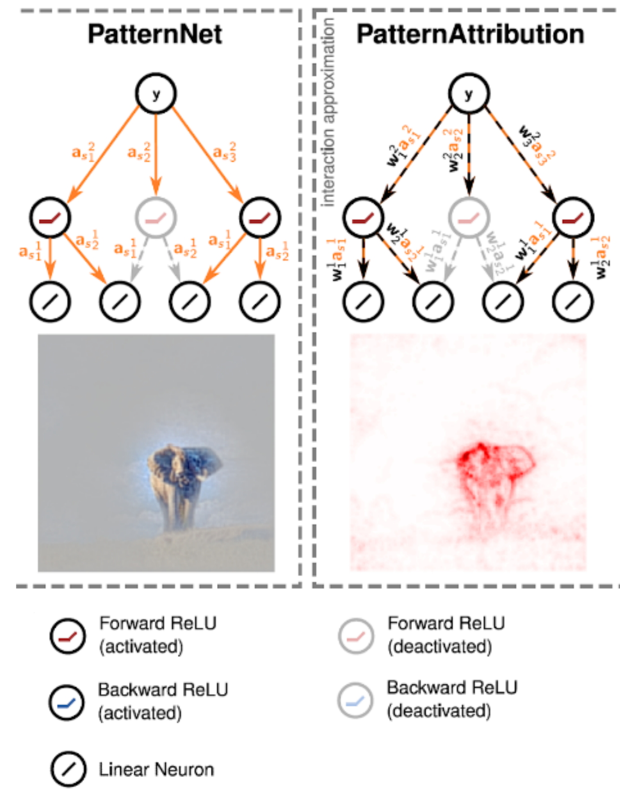
$$a = \frac{cov[x, y]}{\sigma_y^2}$$

  - PatternNet, Non-Linear
    - ReLU activation is considered
    - a is calculated based on the sign
    - it performs well in Non-linear model

$$a_+ = \frac{\mathbb{E}_+[xy] - \mathbb{E}_+[x]\mathbb{E}[y]}{w^\intercal \mathbb{E}_+[xy] - w^\intercal \mathbb{E}_+[x]\mathbb{E}[y]}$$

  - PatternAttribution
    - the result of element-wise multiplication of a and w
    - this can make clearer heat map

$$r = w \odot a_+$$



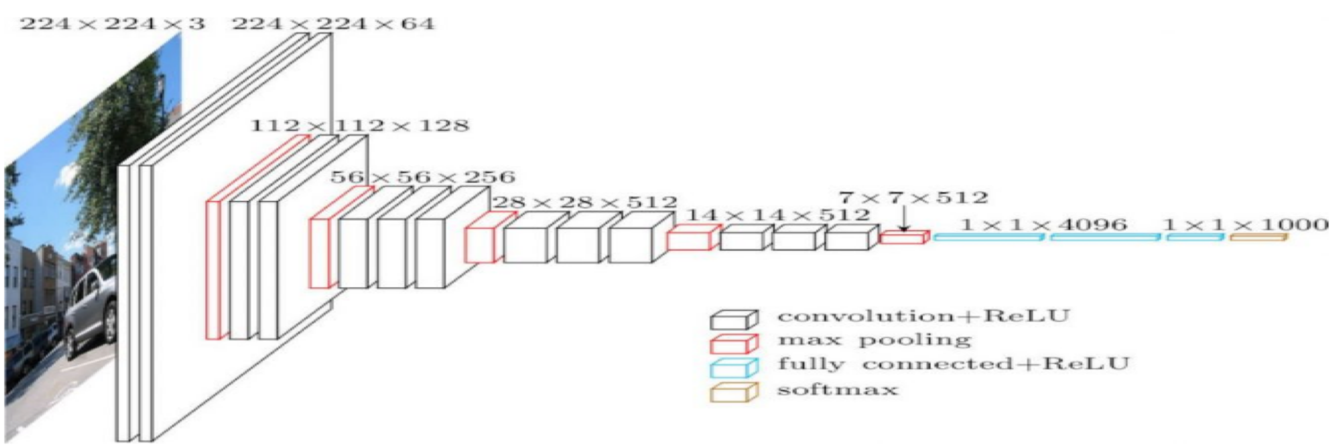
Proposed Solution

- PatternNet and PatternAttribution
  - PatternNet yields a layer-wise back-projection of the estimated signal to input space.
  - The signal estimator is approximated as a superposition of neuron-wise, nonlinear signal estimators Sa+– in each layer
  - PatternAttribution exposes the attribution w a+ and improves upon the layer-wise relevance propagation (LRP) framework
  - By ignoring the distractor, PatternAttribution can reduce the noise and produces much clearer heat maps

Implementation

The experiment focused on the image classification.

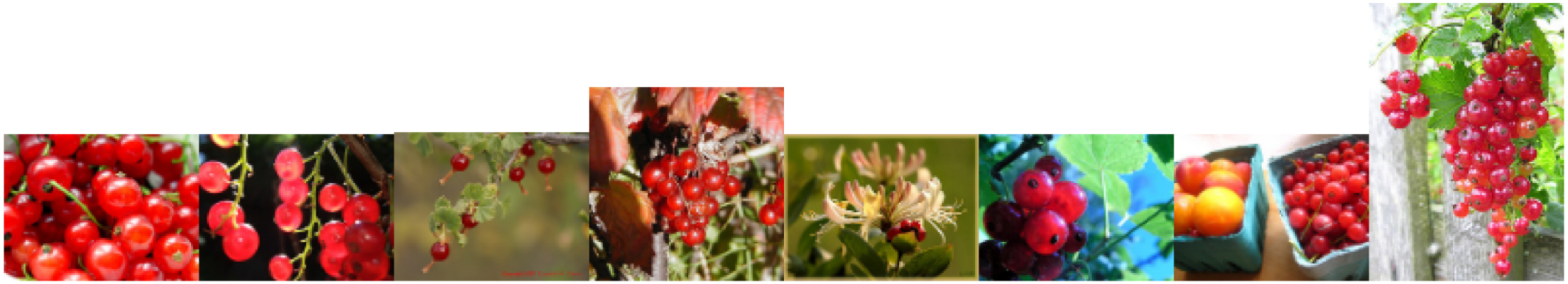
- Keras library has been used on top of tensorflow.
- ImageNet dataset with the pre-trained VGG-16 model was used.
- We run our experiment on 4 GPUs and 2.3 GHz Intel core 9 Macbook Pro laptop.
- DeconvNet, Guided Backprop, Gradient, Pattern Attribution and, Patternnet algorithms are implemented.



VGG16 Architecture

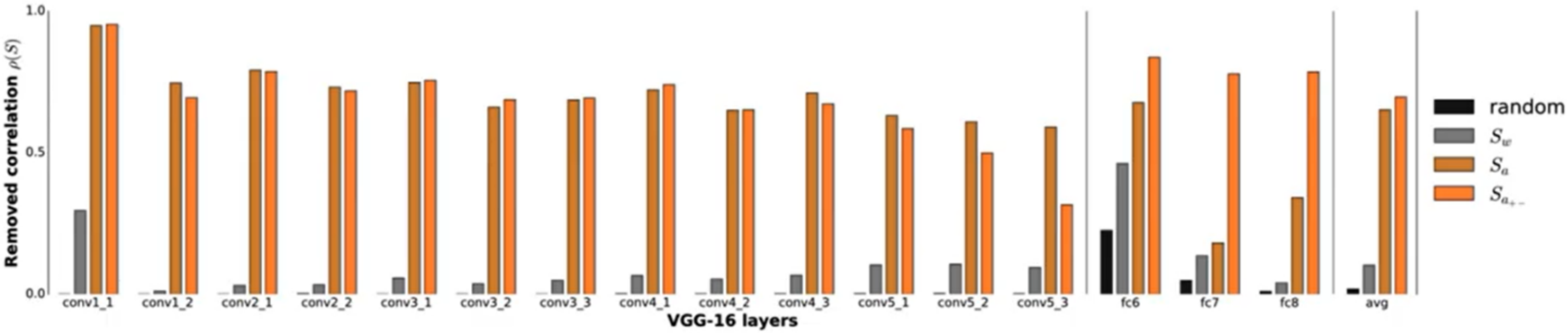
Data Summary

- Used imageNet dataset
- Images were rescaled and cropped to 224x224 pixels
- 50,000 validation images



Experimental Result & Analysis

- $\rho(S)$ values in VGG16 on ImageNet



  - Convolution Layer

    Linear estimator is good in most cases.

    Also, non-linear estimator is better than filter-based, random.

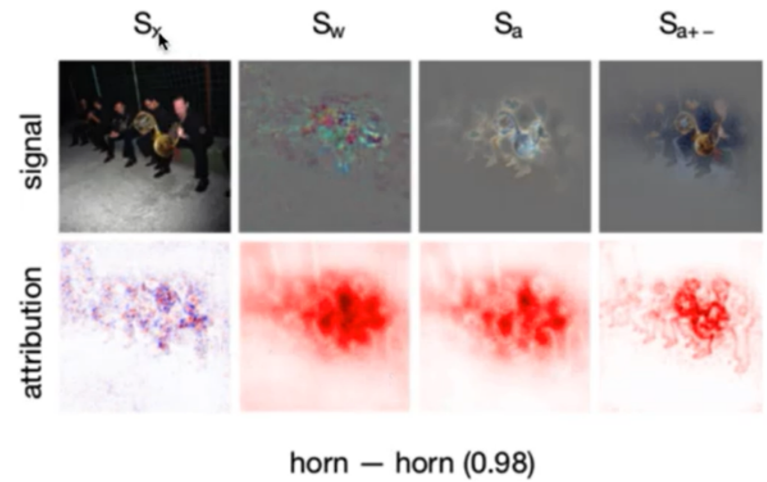
  - FC layer with ReLu

    Non-linear estimator > linear estimator

    Non-learn estimator maintains it's performance level.

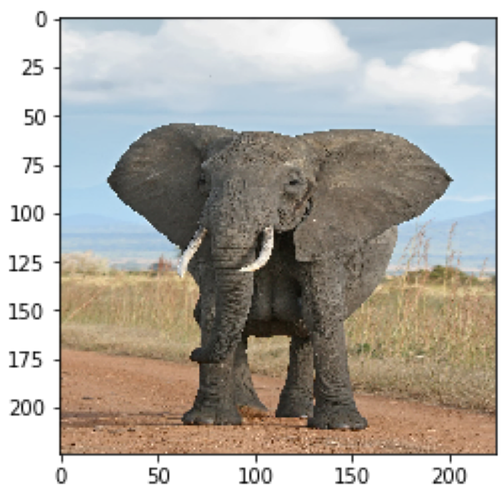
Experimental Result & Analysis

- The optimized estimators remove much more of the correlations across the board.
- For convolutional layers, S_a and S_{a+-} perform comparably in all but one layer.
- The two component estimator S_{a+-} is best in the dense layers
- Quality of the signal estimators of individual neurons are measured and the higher values are better.

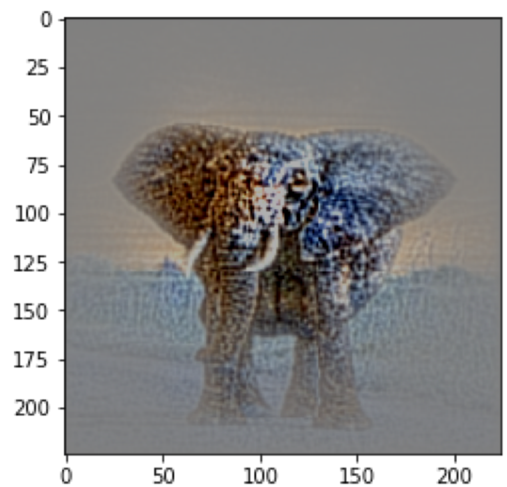


- Sx : Identity estimator
- Sw : DeConvNet, Guided BackProp
- Sa : Linear
- Sa+- : Non-linear

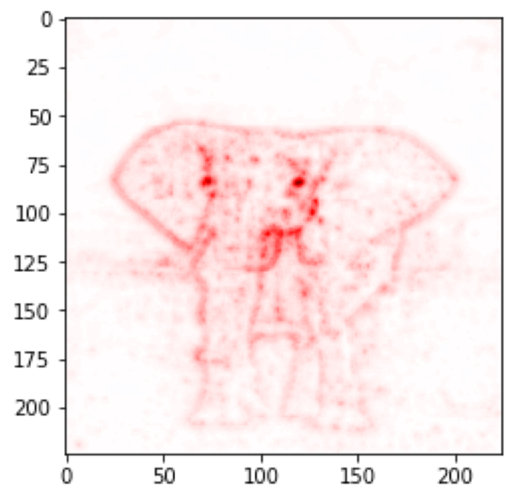
Experimental Result & Analysis



Original Input Image



PatternNet



PatternAttribution

Conclusion

- The direction of the model gradient does not necessarily provide an estimate for the signal in the data. Instead it reflects the relation between the signal direction and the distracting noise contributions.

- The popular explanation approaches for neural networks (DeConvNet, Guided BackProp) do not provide the correct explanation for linear and nonlinear models.

- PatternNet and PatternAttribution provide a theoretical, qualitative and quantitative improvement for understanding deep neural networks.

Each member’s job split

- Vamshi Garikapati
  - Modifying the code
  - Editing the presentation slides
- Dawit Kahsay
  - Modifying the code
  - Editing the presentation slides
- Aaron Knife
  - Coding
  - Editing the presentation slides
- Chijung Jung
  - Modifying the code
  - Making the presentation slides

References

- Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. In ICLR, 2017
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211–252, 2015.
- Sebastian Bach, Alexander Binder, Gregoire Montavon, Frederick Klauschen, Klaus-Robert M ´ uller, ¨ and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one, 10(7):e0130140, 2015.
- Gregoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert ´ Muller. Explaining nonlinear classification decisions with deep taylor decomposition. ¨ Pattern Recognition, 65:211–222, 2017
- David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert M¨uller. How to explain individual classification decisions. Journal of Machine Learning Research, 11(Jun):1803–1831, 2010.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.
- Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In ICLR, 2014.
- Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In European Conference on Computer Vision, pp. 818–833. Springer, 2014.
- Gyubin Son, Youtube movie clip: https://www.youtube.com/watch?v=W7YBfr6EQT4&list=WL&index=5&t=1616s