Learning how to explain neural networks: PatternNet and PatternAttribution

Kindermans et al. 2017 (Google Brain, TU Berlin)

Florian Kleinicke

Universität Heidelberg
kleinicke@stud.uni-heidelberg.de

June 7, 2018

Motivation

Which area was the most important for the neural network to classify the image?

Trivial approach: look at the weights and the influence of every pixel

Example

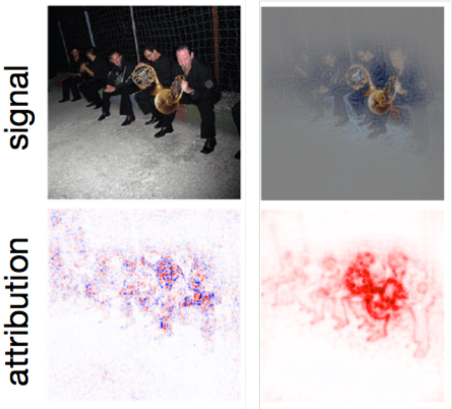


Figure: In first line total data compared to signal. In the second line the attribution of the used signal to the decision.

Overview

- Linear Model
- Signal estimators
- Quality measurements
- Experiments and Results

A Linear Model

$$\boldsymbol{x} = \boldsymbol{s} + \boldsymbol{d} \qquad \boldsymbol{s} = \boldsymbol{a}_s y, \qquad \text{with } \boldsymbol{a}_s = (1,0)^T, \qquad y \in [-1,1]$$

$$\boldsymbol{d} = \boldsymbol{a}_d \epsilon, \qquad \text{with } \boldsymbol{a}_d = (1,1)^T, \qquad \epsilon \sim \mathcal{N}\left(\mu, \sigma^2\right)$$

x is total data
s is the signal
d is the distractor
y is the output (classification)
$\mathbf{a}_s$ and $\mathbf{a}_d$ are directions of spread information.
goal is to extract information y from **x**
multiply **x** with weight vector (filter) $\mathbf{w}=[1,-1]^T$

Dependency of **w** and **d**

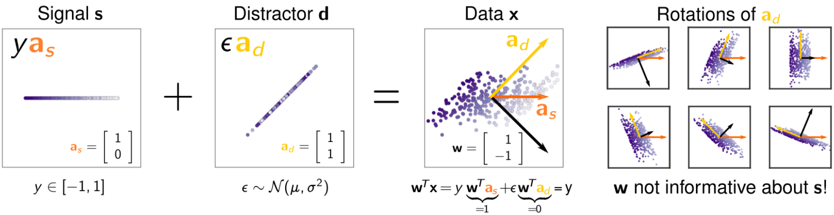


Figure: **w** is dependent on distractor **d**, not the signal **s**

Other approaches take **w** as importance measure.
But it highly depends on the distractor.
Detecting $\mathbf{a_s}$ has to been learned from data.

Signal estimators: $\mathbf{S_x}$ - identity estimator

Signal estimator $S_x(\mathbf{x}) = \mathbf{x}$

Attribution $r = \mathbf{w} \bigodot S_x(\mathbf{x}) = \mathbf{w} \bigodot \mathbf{s} + \mathbf{w} \bigodot \mathbf{d}$

Distractor is present - output noisy

Signal estimators: $\mathbf{S_w}$ - filter based estimator

Assumption: Signal varies in direction of $\mathbf{w}$

Signal estimator $S_w(\mathbf{x}) = \frac{\mathbf{w}}{\mathbf{w}^T\mathbf{w}}\mathbf{w}^T\mathbf{x} = \frac{1}{\mathbf{w}^T\mathbf{w}}\mathbf{w}y$

Attribution $\mathbf{w} \odot S_w(x) = \frac{\mathbf{w} \odot \mathbf{w}}{\mathbf{w}^T\mathbf{w}}y$

Doesn't reconstruct optimal solution for previous linear example.

Signal estimators: $\mathbf{S_a}$ - linear estimator

Distractor $\mathbf{d} = \mathbf{x} - S(\mathbf{x})$ should be 0.
$cov[y, \mathbf{d}] = 0 \Rightarrow cov[\mathbf{x}, y] = cov[S(\mathbf{x}), y]$

Using the linear estimator $S_a(\mathbf{x}) = \mathbf{a}\mathbf{w}^T\mathbf{x} \;\; (= \mathbf{a}_s y)$
$cov[\mathbf{x}, y] = cov[\mathbf{a}\mathbf{w}^T\mathbf{x}, y] = \mathbf{a} \times cov[y, y] \Rightarrow \mathbf{a} = \frac{cov[\mathbf{x},y]}{\sigma_y^2}$

Signal estimators: $\mathbf{S}_{\mathbf{a}+-}$ - two-component estimator

Linear estimator with two cases.

$$\mathbf{x} = \begin{cases} \mathbf{s}_+ + \mathbf{d}_+ & \text{if } y > 0 \\ \mathbf{s}_- + \mathbf{d}_- & \text{otherwise} \end{cases}$$

$$S_{\mathbf{a}+-}(\mathbf{x}) = \begin{cases} \mathbf{a}_+ \mathbf{w}^T \mathbf{x} & \text{if } \mathbf{w}^T \mathbf{x} > 0 \\ \mathbf{a}_- \mathbf{w}^T \mathbf{x} & \text{otherwise} \end{cases}$$

Signal estimators: $\mathbf{S}_{\mathbf{a}+-}$ - two-component estimator

Another reminder from statistics:
$cov[\mathbf{p}, \mathbf{q}] = E[\mathbf{pq}] - E[\mathbf{p}]E[\mathbf{q}]$

In positive regime: $cov[\mathbf{x}_+, y] = cov[S(\mathbf{x})_+, y]$
$E_+[\mathbf{x}y] - E_+[\mathbf{x}]E_+[y] = E_+[S(\mathbf{x})y] - E_+[S(\mathbf{x})]E_+[y]$
Use $S_{\mathbf{a}+}(\mathbf{x}) = \mathbf{a}_+\mathbf{w}^T\mathbf{x}$

$$\mathbf{a}_+ = \frac{E_+[\mathbf{x}y] - E_+[\mathbf{x}]E_+[y]}{\mathbf{w}^T E_+[\mathbf{x}y] - \mathbf{w}^T E_+[\mathbf{x}]E_+[y]}$$

For $\mathbf{a}_-$ analogous

Attribution

Describes the influence and relevance for the output

For linear model $\mathbf{r}_{input} = \mathbf{w} \bigodot \mathbf{a}_s y = \mathbf{w} \bigodot \mathbf{s}$

For more complicated case Deep Taylor Decomposition

$$r_i^{output} = y, \qquad r_{j \neq i}^{output} = 0, \qquad \mathbf{r}^{l-1,i} = \frac{\mathbf{w} \bigodot (\mathbf{x} - \mathbf{x}_0)}{\mathbf{w}^T \mathbf{x}} r_i^l$$

PatternAttribution is a Deep Taylor Decomposition, extended around distractor with negative attributions determined by ReLUs.

$\mathbf{d} = \mathbf{x}_0 = \mathbf{x} - S(\mathbf{x})_{+-} = \mathbf{x} - \mathbf{a}_+ \mathbf{w}^T \mathbf{x}$

$$\mathbf{r}^{l-1,i} = \frac{\mathbf{w} \bigodot (\mathbf{x} - \mathbf{x} - \mathbf{a}_+ \mathbf{w}^T \mathbf{x})}{\mathbf{w}^T \mathbf{x}} r_i^l = \mathbf{w} \bigodot \mathbf{a}_+ r_i^l$$

Approaches

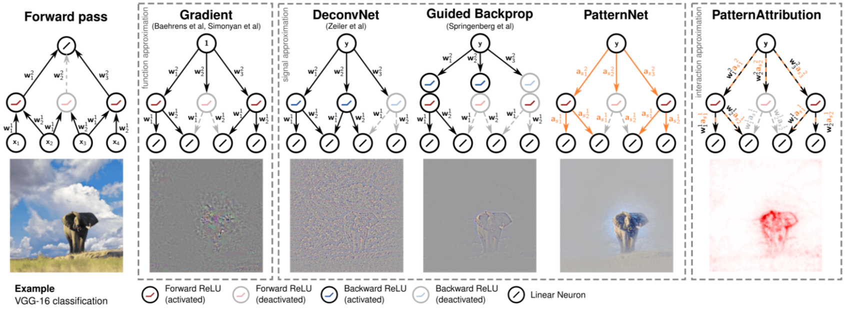


Figure: Illustration of explanation approaches.

Quality

Keep in mind: $\mathbf{x} = \mathbf{s} + \mathbf{d} \qquad \mathbf{w}^T\mathbf{x} = y, \qquad \mathbf{w}^T\mathbf{s} = y, \qquad \mathbf{w}^T\mathbf{d} = 0;$

And a small reminder from statistics:

$$corr(\mathbf{p}, \mathbf{q}) = \frac{cov(\mathbf{p}, \mathbf{q})}{\sqrt{\sigma^2_{\mathbf{p}}\sigma^2_{\mathbf{q}}}}$$

The Quality measure, depending on signal estimator $S(\mathbf{x})$:

$$\rho(S) = 1 - \max_{\mathbf{v}} corr[\mathbf{v}^T(\mathbf{x} - S(\mathbf{x})), \mathbf{w}^T\mathbf{x}] = 1 - \max_{\mathbf{v}} corr[\mathbf{v}^T\mathbf{d}, y]$$

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

Experiments

Implementation with the Lasagne library, trains in Theano.
Data: ImageNet, rescaled and cropped to 224x224 pixels
Used network: pre-trained VGG-16

Signal estimators trained on first half of training set
v used for quality estimator trained on second half.
Official validation set of 50000 samples used for validation

Experiments

VGG16-network - several days training of 4 GPUs
Linear and two-component estimators - 4 hours training
Quality estimator - 1 day training signal estimator on Tesla K40

afterwards individual explanations are computationally cheap

Results

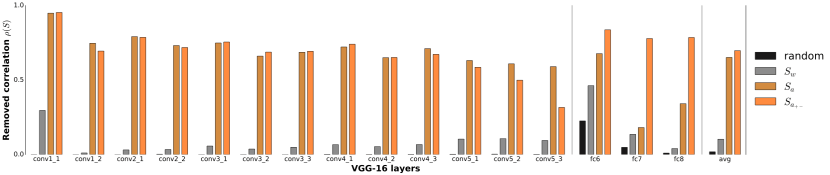


Figure: Comparing different signal estimators in each layer. Higher is better

Results

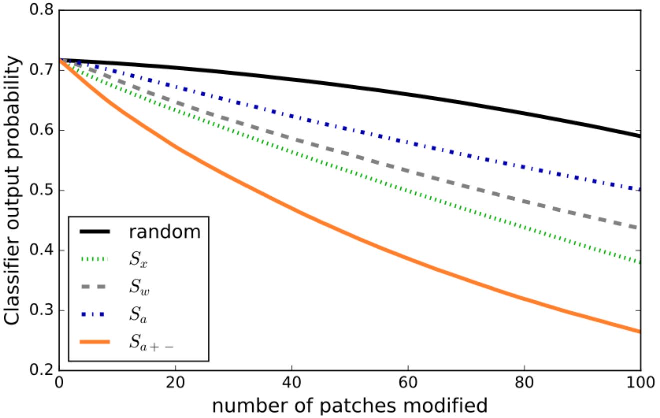


Figure: Averaged most relevant image patches. Higher decay is better.

Results

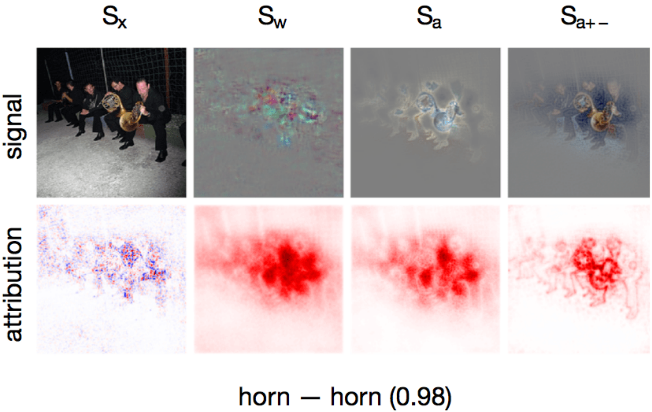
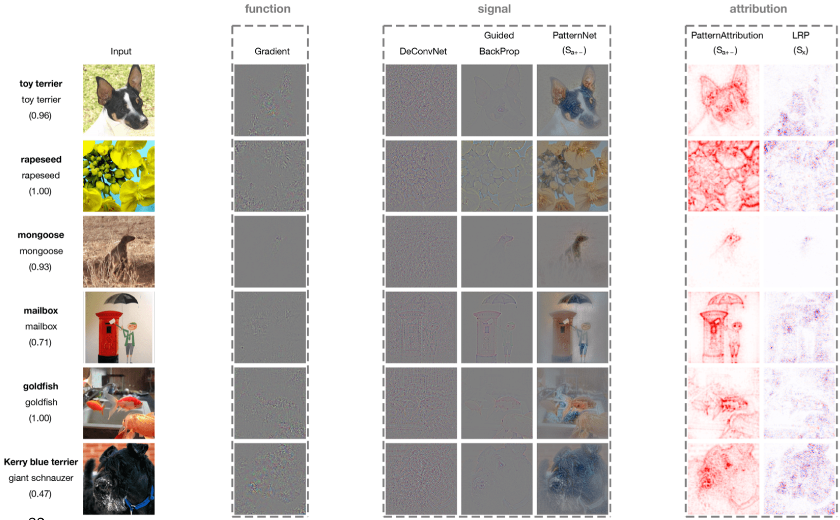


Figure: Compare different signal estimators and it's attribution

Results


function
signal
attribution
Input
Gradient
DeConvNet
Guided BackProp
PatternNet (S_{a+-})
PatternAttribution (S_{a+-})
LRP (S_x)
toy terrier
toy terrier
(0.96)
rapeseed
rapeseed
(1.00)
mongoose
mongoose
(0.93)
mailbox
mailbox
(0.71)
goldfish
goldfish
(1.00)
Kerry blue terrier
giant schnauzer
(0.47)

Summary

- Showed a interesting approach to learn what areas are of interest
- PatternNet works perfectly for linear model and good for real images
- Requires additional time for training, but is computationally cheap for individual explanations

Discussion

- a few formulas were unnecessarily complicated
- It's hard to tell if comparison is to other methods fair or there is something better around
- easy to build a minimal model that self proposed method is optimized for

Learning how to explain neural networks: PatternNet and PatternAttribution

Kindermans et al. 2017 (Google Brain, TU Berlin)

Florian Kleinicke

Universität Heidelberg
kleinicke@stud.uni-heidelberg.de

June 7, 2018