XAI beyond looking at heatmaps – towards directions for model improvement.

Alexander Binder

September 2, 2022

SINGAPORE INSTITUTE OF TECHNOLOGY

UiO : University of Oslo

Talk Overview

1. The definition of explanation depends on the question
2. Shapley Values + Why for many data types XAI research does not end with them
3. Decompositions II – Linearizations: gradient methods, smoothing, modified gradients including LRP
4. Examples for the value of explanation methods for model improvement

1. As many definitions of explanation as there are different questions

prototype-based models: find closest prototypes

training explainable models

Pertinent Positives

Local Linear Approximation

Finding similar examples in feature space

Understanding the Model

Explaining single decisions

Constrained Decomposition

mining samples maximizing a layer

embed feature map distances

Counterfactuals

Grad x input in feature space

Sensitivity w.r.t. inputs

Authors opinion: no method rules them all

1. As many definitions of explanation as there are different questions

prototype-based models: find closest prototypes

training explainable models

Pertinent Positives

Local Linear Approximation

Finding similar examples in feature space

Understanding the Model

Explaining single decisions

Constrained Decomposition

mining samples maximizing a layer

embed feature map distances

Counterfactuals

Grad x input in feature space

Sensitivity w.r.t. inputs

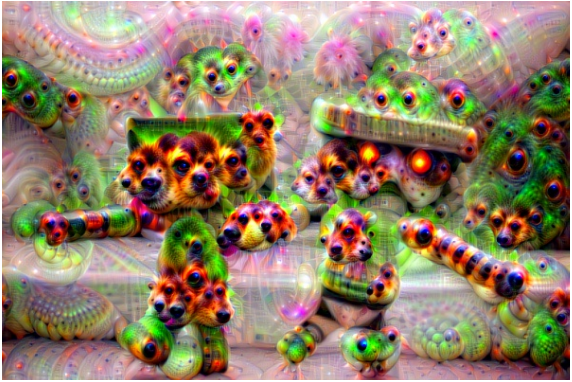
Authors opinion: no method rules them all

Approximate high-dimensional distances of a feature map by low-dimensional embeddings

Popular in deep learning: t-SNE
van der Maaten et al.:
https://lvdmaaten.github.io/tsne/examples/caltech101_tsne.jpg

- PCA projections (K. Pearson), Isomap (Tenenbaum et al. graph defined by k-nearest neighbors and euclidean distances along edges), many others
- CHAL: how to choose a low-dimensional approximation?
- CHAL: parameter sensitivity https://distill.pub/2016/misread-tsne/
- good for exploration with follow up confirmation

DeepDream as an example of Activation Maximization



Credit: https://github.com/gordicaleksa/pytorch-deepdream

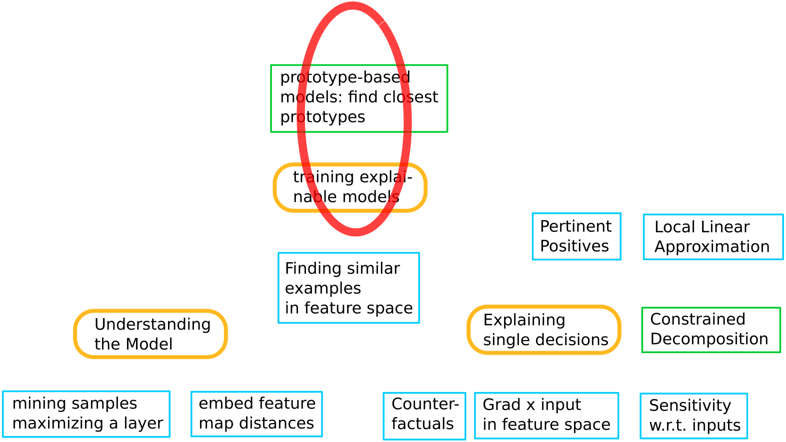
In what ways can one enhance it with more than esthetic value?

Understanding the model: Rank samples which maximize a channel activation

- The top-3 images which maximally activate a particular channel of layer `layer4.2.conv2` of a ResNet-50 after fine-tuning on Pascal VOC.
- The picture also shows an explanation which region in the image is contributing to the activation of the channel (using LRP-max).

Channel has learnt to detect bus views. Selection within the Pascal VOC validation set.

1. The definition of an explanation depends on the question



Authors opinion: no method rules them all

Understanding a *single* prediction: Prototype-based learning

Find most similar samples that were used to arrive at a prediction for a sample x.

- k-nearest neighbors
- Explain a prediction in terms of closest training samples

Explain a prediction in terms of closest training samples

This Looks Like *That*: Deep Learning for Interpretable Image Recognition

Chaofan Chen*
Duke University
cfchen@cs.duke.edu

Oscar Li*
Duke University
oscarli@alumni.duke.edu

Chaofan Tao
Duke University
chaofan.tao@duke.edu

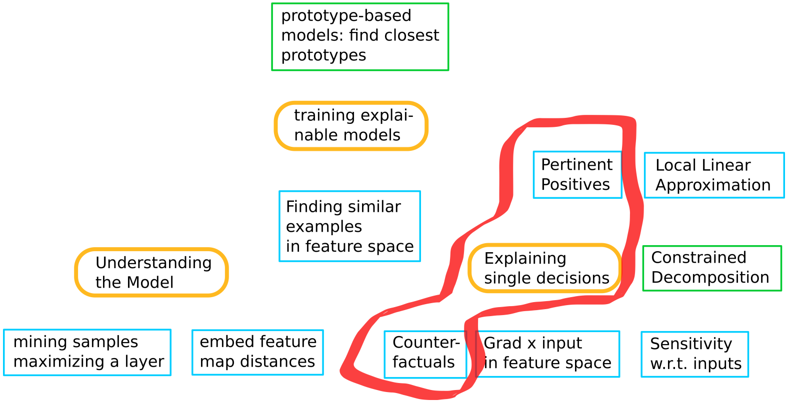
Alina Jade Barnett
Duke University
abarnett@cs.duke.edu

Jonathan Su
MIT Lincoln Laboratory†
su@ll.mit.edu

Cynthia Rudin
Duke University
cynthia@cs.duke.edu

Credit: Chen et al. https://proceedings.neurips.cc/paper/2019/file/adf7ee2dcf142b0e11888e72b43fcb75-Paper.pdf

1. The definition of an explanation depends on the question



Authors opinion: no method rules them all

Understanding a *single* prediction: Pertinent positives and Negatives

Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives

Amit Dhurandhar*
IBM Research
Yorktown Heights, NY 10598
adhuran@us.ibm.com

Pin-Yu Chen*
IBM Research
Yorktown Heights, NY 10598
pin-yu.chen@ibm.com

Ronny Luss
IBM Research
Yorktown Heights, NY 10598
rluss@us.ibm.com

Chun-Chen Tu
University of Michigan
Ann Arbor, MI 48109
timtu@umich.edu

Paishun Ting
University of Michigan
Ann Arbor, MI 48109
paishun@umich.edu

Karthikeyan Shanmugam
IBM Research
Yorktown Heights, NY 10598
karthikeyan.shanmugam2@ibm.com

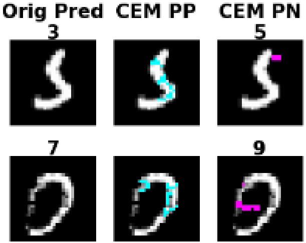
Payel Das
IBM Research
Yorktown Heights, NY 10598
daspa@us.ibm.com

Credit: https://proceedings.neurips.cc/paper/2018/file/c5ff2543b53f4cc0ad3819a36752467b-Paper.pdf

Understanding a *single* prediction: Pertinent positives and Negatives

Pertinent positive: what to retain from a sample?

Pertinent negative: what to change so that prediction switches?



Pertinent positive: cyan, pertinent negative: pink

Credit: https://proceedings.neurips.cc/paper/2018/file/c5ff2543b53f4cc0ad3819a36752467b-Paper.pdf

- CHAL: how to delete/replace information? Result is plausible/outlier?
- many different PP/PN – how to integrate them?

Is this a complete picture?

Many ways to define an explanation of a prediction or a model.

prototype-based models: find closest prototypes

training explainable models

Pertinent Positives

Local Linear Approximation

Finding similar examples in feature space

Understanding the Model

Explaining single decisions

Constrained Decomposition

mining samples maximizing a layer

embed feature map distances

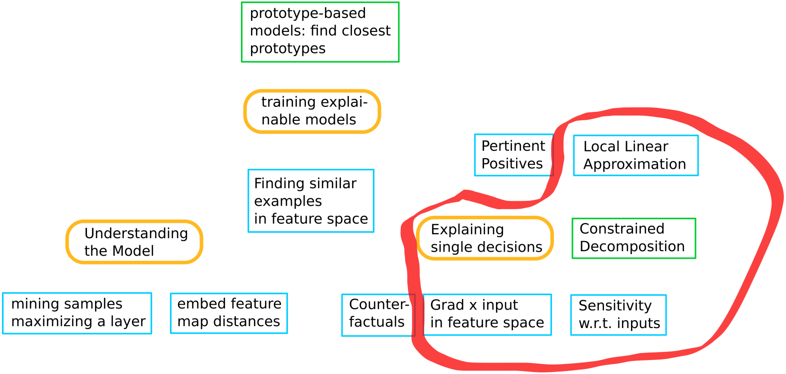
Counterfactuals

Grad x input in feature space

Sensitivity w.r.t. inputs

A more narrow scope: explaining predictions on a single sample by decomposition

Many ways to define an explanation of a prediction or a model.

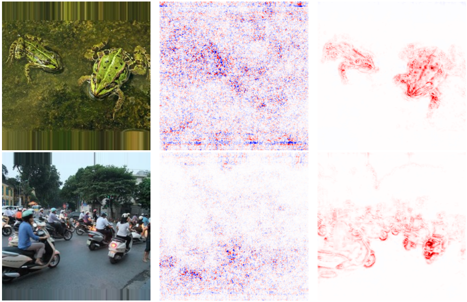


A more narrow scope: explaining predictions on a single sample by decomposition

- case of images: compute a score $r_d(x)$ for every input dimension d of the input sample $x = (x_1, \dots, x_d, \dots, x_D)$

$$\underbrace{f(x) \approx \sum_{d=1}^{D} r_d(x)}_{\textit{decomposition}} \tag{1}$$

- objective function is left open



What is a good explanation within the set of decomposition approaches?

- There a theoretically optimal approach!

2. Shapley values

The Setup:

- Have function f, and a point to be explained $x = (x_1, \ldots, x_d)$.
- We can evaluate f on subsets $x_S = \{x_{i_1}, x_{i_2}, x_{i_3}, \ldots | \forall k : i_k \in S\}$ of features from x.

Shapley value

then the Shapley value is defined as:

$$\phi_j(f, x) = \frac{1}{d} \sum_{S \subseteq \{1,\ldots,d\} \setminus \{j\}} \frac{f(x_{S \cup \{j\}}) - f(x_S)}{\binom{d-1}{|S|}} \tag{2}$$

Game Theory: S is set of players playing a game with outcome $f(x_S)$.

j a member which can join the set with outcome $f(x_{S \cup \{j\}})$

Its interpretation?

- Have function f, and a point to be explained $x = (x_1, \ldots, x_d)$.
- We can evaluate f on subsets $x_S = \{x_{i_1}, x_{i_2}, x_{i_3}, \ldots | \forall k : i_k \in S\}$ of features from x.
- then the Shapley value is defined as:

$$\phi_j(f, x) = \frac{1}{d} \sum_{S \subseteq \{1,\ldots,d\} \setminus \{j\}} \frac{f(x_{S \cup \{j\}}) - f(x_S)}{\binom{d-1}{|S|}} \tag{3}$$

$$= \frac{1}{\text{numsets}} \sum_{k \geq 1} \sum_{\text{sets } S \text{ without } j, |S|=k} \frac{\text{differential contrib of } j \text{ to set } S}{\text{number of sets of same size } |S|} \tag{4}$$

Shapley values

We have interpreted SHAP.

How to categorize this approach?

$$\phi_j(f,x) = \frac{1}{d} \sum_{S \subseteq \{1,\ldots,d\} \setminus \{j\}} \frac{f(x_{S \cup \{j\}}) - f(x_S)}{\binom{d-1}{|S|}} \tag{5}$$

$$= \frac{1}{\text{numsets}} \sum_{k \geq 1} \sum_{\text{sets } S \text{ without } j, |S|=k} \frac{\text{differential contrib of } j \text{ to set } S}{\text{number of sets of same size } |S|} \tag{6}$$

We have interpreted SHAP.
How to categorize this approach?

$$\phi_j(f,x) = \frac{1}{d} \sum_{S \subseteq \{1,\dots,d\} \setminus \{j\}} \frac{f(x_{S\cup\{j\}}) - f(x_S)}{\binom{d-1}{|S|}} \tag{5}$$

$$= \frac{1}{\text{numsets}} \sum_{k \geq 1} \sum_{\text{sets } S \text{ without } j, |S|=k} \frac{\text{differential contrib of } j \text{ to set } S}{\text{number of sets of same size } |S|} \tag{6}$$

Combinatorially exhaustive occlusion differences for j
differential values: sets with player j, sets without j
(cf. Petliuk et al. RISE https://arxiv.org/abs/1806.07421)

Favourable theoretical properties:

- does not make a difference, zero Shapley-value

$$\forall S: \ f(x_{S\cup\{j\}}) = f(x_S) \Rightarrow \phi_j(f,x) = 0 \tag{7}$$

- value-equal pair of features j, k, identical Shapley-value:

$$\forall S: \ f(x_{S\cup\{j\}}) = f(x_{S\cup\{k\}}) \Rightarrow \phi_j(f,x) = \phi_k(f,x) \tag{8}$$

- Efficiency:

$$\sum_{j=1}^{d} \phi_j(f,x) = f(x) - E_X[f(X)] \tag{9}$$

  - decomposition of $f(x)$
  - distributes the difference between function value $f(x)$ and its expectation $E_X[f(X)]$ onto all dimensions in an equal way (not shown here[1])

[1] Grabisch https://www.worldscientific.com/doi/abs/10.1142/S0218488597000440

We can evaluate f on subsets $x_S = \{x_{i_1}, x_{i_2}, x_{i_3}, \ldots | \forall k : i_k \in S\}$ of features from x. – Outside of tabular data types this is a very strong assumption.

- How to remove a region in an image, a language sentence ?
- How to remove an interval of a data point being a time series ?
- is S, $S \cup \{j\}$ plausible or outlier (bonds and molecules) ?

Dimensions x_d of a data sample $x \leftrightarrow$ Players in a game ?

No optimality guarantee when

- applicability assumptions do not hold well
- one has to use approximations (e.g. MC)

What other methods exist for data types where Shapley assumptions do not hold well?

3. Linearizations

- Starting point was: $f(x) \approx \sum_{i=1}^{d} r_d(x)$ f is non-linear now
- Taylor Expansion (3rd order)

$$
\begin{aligned}
f(x) \approx\ & f(x_0) \\
& + \frac{1}{1!} Df(x_0)[x - x_0] \\
& + \frac{1}{2!} D^2 f(x_0)[x - x_0, x - x_0] \\
& + \frac{1}{3!} D^3 f(x_0)[x - x_0, x - x_0, x - x_0] + \mathcal{O}(\|x - x_0\|^4) \qquad (10)
\end{aligned}
$$

$$
Df(x_0) = \left(\frac{\partial f}{\partial x_d}(x_0), d = 1, \ldots, D \right) = \nabla f(x_0)
$$

$$
D^2 f(x_0) = \left(\frac{\partial^2 f}{\partial x_d x_e}(x_0), d, e = 1, \ldots, D \right)
$$

$$
D^3 f(x_0) = \left(\frac{\partial^3 f}{\partial x_d x_e x_c}(x_0), c, d, e = 1, \ldots, D \right)
$$

- Starting point was: $f(x) = \sum_{i=1}^{d} r_d(x)$ f is non-linear now
- Taylor Expansion up to first order:

$$
\begin{aligned}
f(x) &\approx f(x_0) + \nabla f(x_0) \cdot (x - x_0) && (11) \\
&= f(x_0) + \sum_d \frac{\partial f}{\partial x_d}|_{x_0}(x_d - x_{0,d}) && (12)
\end{aligned}
$$

- use as explanation:

$$
r_d(x) = \frac{\partial f}{\partial x_d}|_{x_0}(x_d - x_{0,d}) \tag{13}
$$

Linearizations

- Gradient $\times$ Input
- Integrated Gradient
- LIME
- Grad-CAM
- LRP

Use as explanation:

$$r_d(x) = \frac{\partial f}{x_d}(x)\, x_d, \;\; \mathbf{R} = (r_d, d = 1, \dots, D) = \nabla f(x) \cdot x$$

(+) derivation via global Taylor decomposition for a point x_0 orthogonal to the gradient ($\nabla f(x) \cdot x_0 = 0$) in the point x to be explained.

$$\begin{aligned} f(x_0) &= f(x) + \nabla f(x) \cdot (x_0 - x) + \mathcal{O}(\|x - x_0\|^2) \\ \Rightarrow f(x) &\approx f(x_0) + \nabla f(x) \cdot (x - x_0) \\ &= f(x_0) + \nabla f(x) \cdot x - \underbrace{\nabla f(x) \cdot x_0}_{=0} \end{aligned}$$

$f(x_0)$ is a bias term, independent of any dimension

(−) can be noisy for deep ReLU-networks due to gradient shattering:

The noiseness of gradient × input and related methods for deep ReLU-networks:

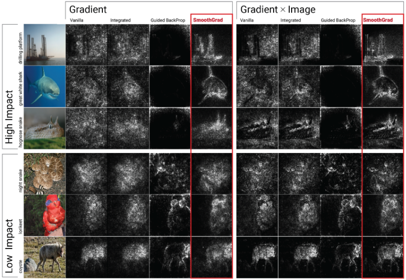


Figure 5. Qualitative evaluation of different methods. First three (last three) rows show examples where applying SMOOTHGRAD had high (low) impact on the quality of sensitivity map.

Credit: https://arxiv.org/pdf/1706.03825.pdf

Gradient Shattering Effect

- Montufar et al. 2014 https://papers.nips.cc/paper/5422-on-the-number-of-linear-regions-of-deep-neural-networks.pdf.
- Balduzzi et al. 2017 http://proceedings.mlr.press/v70/balduzzi17b/balduzzi17b.pdf.

Explanation Methods: Integrated Gradient

Sundararajan et al., *ICML 2017*,
https://dl.acm.org/doi/10.5555/3305890.3306024

A heuristic very similar to the gradient $\times$ input:

$$r_d(x) = (x_d - x_d^{(0)}) \frac{1}{R} \sum_{r=1}^{R} \frac{\partial f}{\partial x_d} \Big|_{z = x^{(0)} + \frac{r}{R}(x - x^{(0)})}$$

- Averages over partial derivatives along multiple points $x^{(0)} + \frac{r}{R}(x - x^{(0)})$ along a path from $x^{(0)}$ to x.
- Noisy heatmaps in ReLU networks due to gradient shattering. Averaging gradients to smoothe the noise.
- IG gets better with many roots used (+ slows down).

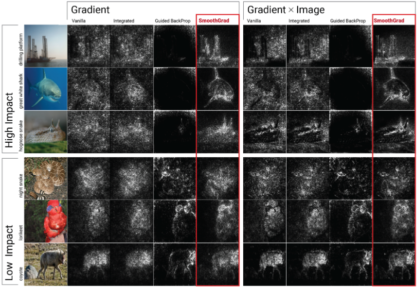


Figure 5. Qualitative evaluation of different methods. First three (last three) rows show examples where applying SmoothGrad had high (low) impact on the quality of sensitivity map.

Credit: https://arxiv.org/pdf/1706.03825.pdf

Explanation Methods: Grad-CAM

Selvaraju et al. https://arxiv.org/abs/1610.02391

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{14}$$

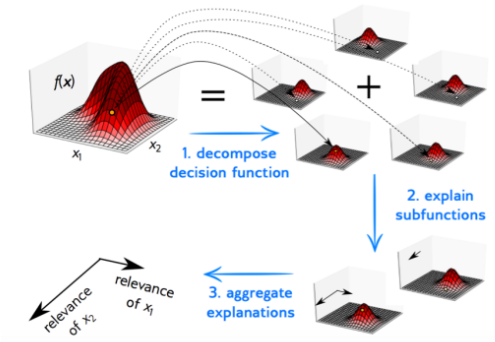
$$G_{ij}^c = \sum_k \alpha_k^c A_{ij}^k \tag{15}$$

$$L_{ij}^c = \mathrm{ReLU}(G_{ij}^c) \tag{16}$$

It is almost gradient $\times$ input in feature space. Three differences:

- smoothing of the gradient in feature space $\frac{\partial y^c}{\partial A_{ij}^k}$ by spatial averaging (cf. Integrated Gradients)
- average over all channels $\sum_k \alpha_k^c A_{ij}^k$
- retain positive part only (cf. Guided Backpropagation)

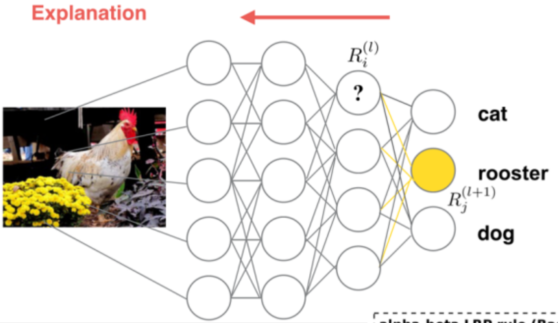
- Divide and conquer: decompose network in layers



credit: W. Samek

- Taylor approximation per layer/neuron
- easier to find roots for one layer
- robustness to gradient shattering

Explanation Methods: LRP



Theoretical interpretation
Deep Taylor Decomposition
(Montavon et al., 2017)

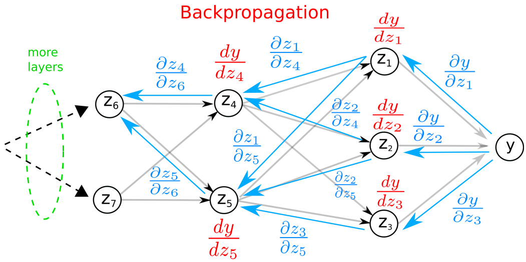
alpha-beta LRP rule (Bach et al. 2015)

$$R_i^{(l)} = \sum_j \left(\alpha \cdot \frac{(x_i \cdot w_{ij})^+}{\sum_{i'}(x_{i'} \cdot w_{i'j})^+} + \beta \cdot \frac{(x_i \cdot w_{ij})^-}{\sum_{i'}(x_{i'} \cdot w_{i'j})^-}\right) R_j^{(l+1)}$$

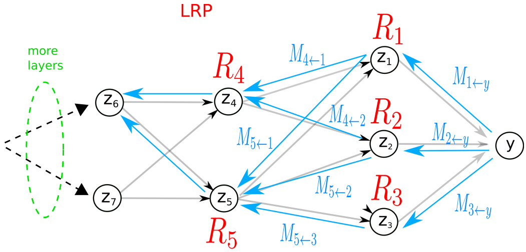
where $\alpha + \beta = 1$

credit: W. Samek

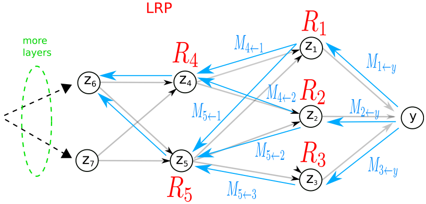
LRP has the same flow along graphs as the gradient.



$$\frac{dy}{dz_6} = \frac{\partial z_4}{\partial z_6}\frac{dy}{dz_4} + \frac{\partial z_5}{\partial z_6}\frac{dy}{dz_5}$$ partial derivatives flow along the edges.



Relevance distribution for one neuron: example β-rule



$$\text{forward pass: } y_k = g\left(\sum_i w_{ik}x_i + b\right)$$

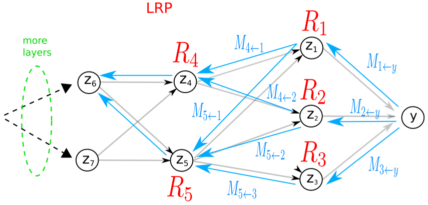
backward: have computed already the relevance R_k for the neuron output y_k

$$\text{LRP backward: } R_{i\leftarrow k}(\mathbf{x}) = R_k M_{i\leftarrow k}(w, x)$$

$$\beta = 0: \; M_{i\leftarrow k} = \overbrace{\frac{(w_{ik}x_i)_+}{\sum_{i'}(w_{i'k}x_{i'})_+}}^{\text{fraction of positive part of input dim } x_i \text{ relative to all inputs } x_{i'}}$$

$$\beta > 0: \; M_{i\leftarrow k} = (1+\beta)\underbrace{\frac{(w_{ik}x_i)_+}{\sum_{i'}(w_{i'k}x_{i'})_+}}_{\text{positive contributions}} - \beta\underbrace{\frac{(w_{ik}x_i)_-}{\sum_{i'}(w_{i'k}x_{i'})_-}}_{\text{negative contributions}}$$

Relevance distribution for one neuron: example β-rule



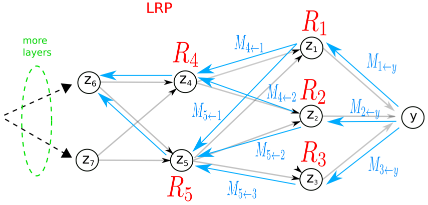
forward pass: $y_k = g\left(\sum_i w_{ik}x_i + b\right)$

backward: have computed already the relevance R_k for the neuron output y_k

LRP backward: $R_{i\leftarrow k}(\mathbf{x}) = R_k M_{i\leftarrow k}(w,x)$

$$\beta = 0: \quad M_{i\leftarrow k} = \overbrace{\frac{(w_{ik}x_i)_+}{\sum_{i'}(w_{i'k}x_{i'})_+}}^{\text{fraction of positive part of input dim } x_i \text{ relative to all inputs } x_{i'}}$$

$$\beta > 0: M_{i\leftarrow k} = (1+\beta)\underbrace{\frac{(w_{ik}x_i)_+}{\sum_{i'}(w_{i'k}x_{i'})_+}}_{\text{positive contributions}} - \beta\underbrace{\frac{(w_{ik}x_i)_-}{\sum_{i'}(w_{i'k}x_{i'})_-}}_{\text{negative contributions}}$$



forward pass: $y_k = g\left(\sum_i w_{ik} x_i + b\right)$

backward: have computed already the relevance R_k for the neuron output y_k

LRP backward: $R_{i\leftarrow k}(\mathbf{x}) = R_k M_{i\leftarrow k}(w, x)$

$$\beta = 0: \quad M_{i\leftarrow k} = \overbrace{\frac{(w_{ik}x_i)_+}{\sum_{i'}(w_{i'k}x_{i'})_+}}^{\text{fraction of positive part of input dim } x_i \text{ relative to all inputs } x_{i'}}$$

$$\beta > 0: \quad M_{i\leftarrow k} = (1+\beta)\underbrace{\frac{(w_{ik}x_i)_+}{\sum_{i'}(w_{i'k}x_{i'})_+}}_{\text{positive contributions}} - \beta \underbrace{\frac{(w_{ik}x_i)_-}{\sum_{i'}(w_{i'k}x_{i'})_-}}_{\text{negative contributions}}$$

given: have computed already R_k as relevance of neuron output

$$
\begin{aligned}
y_k &= g\left(\sum_i w_{ik} x_i + b\right) \\
R_{i \leftarrow k}(\mathbf{x}) &= R_k M_{i \leftarrow k}(w, x) \\
M_{i \leftarrow k} &= (1+\beta) \underbrace{\frac{(w_{ik} x_i)_+}{\sum_{i'} (w_{i'k} x_{i'})_+}}_{\text{positive contributions}} - \beta \underbrace{\frac{(w_{ik} x_i)_-}{\sum_{i'} (w_{i'k} x_{i'})_-}}_{\text{negative contributions}}
\end{aligned}
$$

- β controls ratio of negative to total relevance $= \frac{\beta}{1+2\beta}$.
- negative to total relevance – fixed independent of neuron inputs x_i (!).
- bounded relevance scale: $|R_{i \leftarrow k}| \leq (1+\beta)|R_k|$ for smoothness of explanations
- compare to gradient clipping for batchnorm-free training, e.g. Brock et al https://arxiv.org/pdf/2102.06171.pdf

My method is the best!! Am I one of them?

Credit: Hanabusa Itcho

A. Image captioning: LRP detects words induced by textual correlations unsupported by image content[2]
B. Improving test phase model accuracy in small-sample size training tasks (few-shot learning):
LRP-guided training to improve cross-domain few-shot learning[3]
C. Identifying and Improving what universal counterfeit image detectors use:
Discovering Transferable Forensic Features for CNN-generated Images Detection[4]

[2] J Sun, S Lapuschkin, W Samek, A Binder, Information Fusion, 2021
[3] J Sun, S Lapuschkin, W Samek, Y Zhao, NM Cheung, A Binder, ICPR 2020
[4] K Chandrasegaran, NT Tran, A Binder, NM Cheung, ECCV 2022

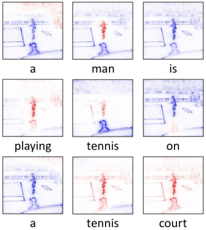
Case A: Image captioning: LRP detects words induced by textual correlations unsupported by image content[5]

- starting point: wanted to compare attention vs backward explanation
- relevance: e.g. medical image to text model: does it look at the X-ray to make a prediction?

[5] J Sun, S Lapuschkin, W Samek, A Binder, Information Fusion, 2021

Detecting structure-induced predictions – image captioning case

- Words are generated often by recurrent neural networks: $\text{word}_{n+1} = \mathit{RNN\_Attention}(\text{Image}, \text{word}_1, \text{word}_2, \ldots, \text{word}_n)$
- Models use attention usually



2.Man: a
3.Is: a man
4.Playing: a man is
5.Tennis: a man is playing
6.On: a man is playing tennis
7.A: a man is playing tennis on
8.Tennis: a man is playing tennis on a
9.Court: a man is playing tennis on a tennis

0.75,1	0.5,0.75	0.25,0.5	0,0.25
-0.25,0	-0.5,-0.25	-0.75,-0.5	-1,-0.75

Detecting structure-induced predictions – image captioning case

Many image captioning models have a principled structure:

take a word embedding $\boldsymbol{E_m}(w_{t-1})$ and a CNN feature map $\boldsymbol{I}_g$ of an image as inputs, process them by an RNN:

$$\boldsymbol{x}_t = [\boldsymbol{E_m}(w_{t-1}), \boldsymbol{I}]$$
$$\boldsymbol{h}_t, \boldsymbol{m}_t = LSTM(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}, \boldsymbol{m}_{t-1})$$

An attention mechanism $Att(\cdot)$ uses $\boldsymbol{h}_t$ and $\boldsymbol{I}$ to obtain a spatially reweighted context $\boldsymbol{c}_t$ for word prediction.

$$\boldsymbol{c}_t = Att(\boldsymbol{h}_t, \boldsymbol{m}_t, \boldsymbol{I})$$
$$\boldsymbol{y}_t = Predictor(\boldsymbol{h}_t, \boldsymbol{c}_t)$$

How may $Att(\cdot)$ look like?

For *adaptive attention:* Let $\boldsymbol{m}_t$ be the LSTM memory cell, then:

$$
\begin{aligned}
\boldsymbol{s}_t &= \sigma(\boldsymbol{W}_x \boldsymbol{x}_t + \boldsymbol{W}_h \boldsymbol{h}_{t-1}) \odot \tanh(\boldsymbol{m}_t) \\
\boldsymbol{a} &= \boldsymbol{w}_a \tanh(\boldsymbol{I}\boldsymbol{W}_I + \boldsymbol{W}_g \boldsymbol{h}_t) \\
b &= \boldsymbol{w}_a \tanh(\boldsymbol{W}_s \boldsymbol{s}_t + \boldsymbol{W}_g \boldsymbol{h}_t)) \\
\boldsymbol{\alpha}_t &= \mathrm{softmax}(\boldsymbol{a}) \in \mathbb{R}^{n_v} \\
\beta_t &= \mathrm{softmax}([\boldsymbol{a}, b])_{(n_v+1)} \in \mathbb{R}^1 \\
\boldsymbol{c}_t &= (1 - \beta_t) \sum_{k=1}^{n_v} \alpha_t^{(k)} \boldsymbol{I}^{(k)} + \beta_t \boldsymbol{s}_t \\
\boldsymbol{c}_t &= Att(\boldsymbol{h}_t, \boldsymbol{m}_t, \boldsymbol{I})
\end{aligned}
$$

How may $Att(\cdot)$ look like?

For *multi-head transformer attention:*

$$\hat{\boldsymbol{V}}^{(i)} = \mathrm{softmax}\left(\frac{\boldsymbol{Q}^{(i)}\boldsymbol{K}^{(i)\boldsymbol{T}}}{\sqrt{d_k}}\right)\boldsymbol{V}^{(i)} + \boldsymbol{V}^{(i)}$$

$$\boldsymbol{Q} := \boldsymbol{h}_t, \boldsymbol{K} := \boldsymbol{I}\boldsymbol{W}_k, \boldsymbol{V} := \boldsymbol{I}\boldsymbol{W}_v$$

$$\boldsymbol{c}_t = Att(\boldsymbol{h}_t, \boldsymbol{I}) = (\hat{\boldsymbol{V}}^{(1)}, \ldots, \hat{\boldsymbol{V}}^{(K)})$$

Observation of a common structure: weight (depends on features) times concatenated features

Detecting structure-induced predictions – image captioning case

Attention-weighted features share a common structure

$$f = \sum_i w_i(v) v_i$$

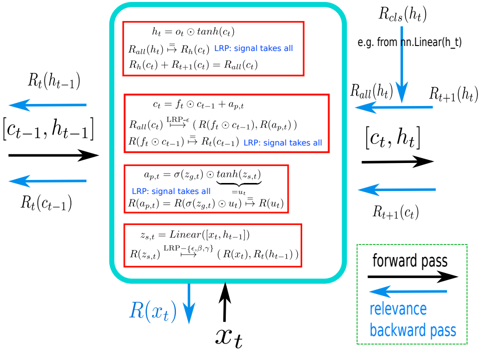
apply signal takes all idea (L Arras et al. ACL 2019):

- do not propagate relevance through weights $w_i(v)$ to v
- propagate relevance only to v_i directly, by interpreting it as a weighted sum of v_i with static weights w_i:

$$R(f) \overset{LRP-\epsilon}{\longmapsto} \{R(v_i)\}$$

Combine LSTM-explanation idea and this idea – have explanations for image captioning

Detecting structure-induced predictions – image captioning case



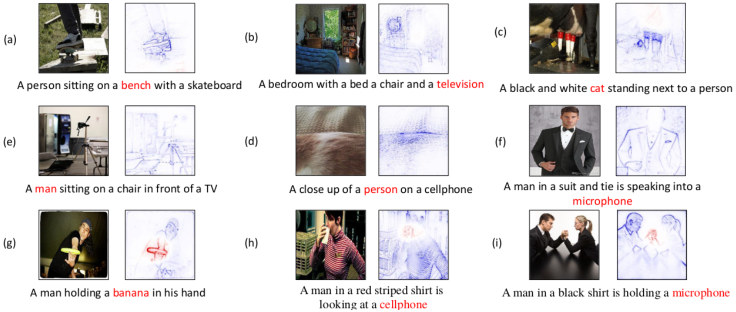
- Words are generated often by recurrent neural networks: $\text{word}_{n+1} = f(\text{Image}, \text{word}_1, \text{word}_2, \ldots, \text{word}_n)$
- Two principles when using LRP for RNNs:

(1) signal takes all in terms like $w = \sigma(z_{g,t}) \odot \tanh(z_{s,t})$ do not distribute relevance on gates $z_{g,t}$. Only onto signal terms $z_{s,t}$:

$$R(w) \mapsto (\,R(z_{g,t}), R(z_{s,t})\,) := (0, R(z_{s,t}))$$

(2) $+$: use LRP-ϵ, other linear operations: use LRP-ϵ, β, γ

Detecting structure-induced predictions – image captioning case



- Forward pass spatial attention cannot perform this task

Detecting structure-induced predictions – image captioning case

- Debias by explaining object words. Use explanations to reweight CNN features in training.
- If the prediction for one step is made by an fc layer using c_t, h_t as

$$p_u = f_c(c_t + h_t),$$

  then during training compute the normalized relevances $\hat{R}(c_t) \in [0, 2]$, $\hat{R}(h_t) \in [0, 2]$
- use them as element-weise weighting and optimize:

$$p_w = f_c(\underbrace{\hat{R}(c_t) \odot c_t}_{\text{weighted feat}} + \underbrace{\hat{R}(h_t) \odot h_t}_{\text{weighted feat}})$$

$$L = \lambda \underbrace{L_{ce}(p_u, y)}_{\text{usual loss}} + (1 - \lambda) L_{ce}(p_w, y)$$

  wher L_{ce} is the usual cross entropy loss, and y are the ground truth labels
- Able to measure the quality of debiasing

During training: reweight CNN features using explanation scores. Improves prediction on most frequent object words – by reducing hallucinating them.

Table: (mAP) of the predicted 25 most frequent object words. (*ce*): models are trained only with cross-entropy loss. The other models are finetuned with SCST for the non-differentiable CIDEr score. *BU* and *CNN* denote Faster-RCNN features and CNN features. Higher mAP means less object hallucination.

dataset	Flickr30K		MSCOCO2017	
mAP	baseline	LRP-IFT	baseline	LRP-IFT
Ada-LSTM-CNN	52.95	**54.47**	72.29	**73.85**
Ada-LSTM-BU	63.84	**64.61**	78.57	**80.55**
MH-FC-CNN	55.98	**57.71**	**73.74**	73.42
MH-FC-BU	64.46	**64.98**	**78.10**	77.71
Ada-LSTM-CNN (ce)	58.53	**60.80**	73.65	**74.00**
Ada-LSTM-BU (ce)	60.70	**65.01**	79.06	**79.80**
MH-FC-CNN (ce)	55.50	**59.23**	**77.15**	76.87
MH-FC-BU (ce)	64.08	**66.10**	81.02	**81.16**

Little change on the set of all words:

Table: The performance of the Ada-LSTM model and MH-FC model with or without LRP-IFT on the test set of Flickr30K and MSCOCO2017 datasets. L. denotes LRP-inference fine-tuned models. *BU* and *CNN* denote bottom-up features and CNN features. Measures: F_B: F_{BERT} S: SPICE.

dataset	Flickr30K		MSCOCO2017	
	F_B	S	F_B	S
Ada-LSTM-CNN	90.6	13.9	91.7	19.5
L.Ada-LSTM-CNN	90.6	14.0	91.2	19.2
Ada-LSTM-BU	90.0	16.4	91.0	19.2
L.Ada-LSTM-CNN	90.0	16.5	91.0	19.3
MH-FC-CNN	89.9	14.5	91.1	20.1
L.MH-FC-CNN	89.7	14.2	91.0	20.1
MH-FC-BU	90.1	17.1	91.3	21.8
L.MH-FC-BU	90.1	17.0	91.3	21.9

Why there is no global improvement ?

Consider for a test sentence the minimal frequency of non-stop words counted over the training set:

dataset	Flickr30K		MSCOCO2017	
average counts	LRP-IFT-improved	LRP-IFT-degraded	LRP-IFT-improved	LRP-IFT-degraded
Ada-LSTM-CNN	**26.1**	35.2	**123.7**	134.0
Ada-LSTM-BU	**30.1**	31.4	**130.8**	134.7
MH-FC-CNN	**29.3**	31.4	**124.4**	132.8
MH-FC-BU	**29.3**	29.7	**118.7**	139.0
Ada-LSTM-CNN (ce)	34.4	**28.5**	**124.4**	137.0
Ada-LSTM-BU (ce)	31.7	**28.1**	**119.0**	150.6
MH-FC-CNN (ce)	**29.4**	30.6	**128.0**	142.6
MH-FC-BU (ce)	**22.6**	35.9	**124.7**	148.5

Tradeoff: LRP-finetuning improves on sentences with more rare words!

Case B: Improving test phase model accuracy in small-sample size training tasks (few-shot learning): Explanation-Guided Training for Cross-Domain Few-Shot Classification[6]

https://arxiv.org/abs/2007.08790

[6] J Sun, S Lapuschkin, W Samek, Y Zhao, NM Cheung, A Binder, ICPR 2020

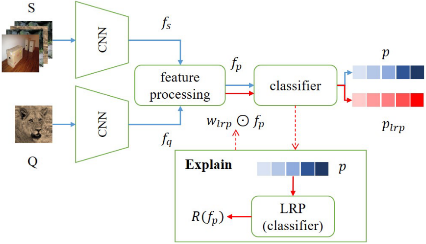
Explanation-Guided Training for Cross-Domain Few-Shot Classification

Motivation:

- Improve model accuracy at test time by explanation-guided interventin during the training phase.
- Choose a low sample size setup with a somewhat challenging task.

Few-shot classification

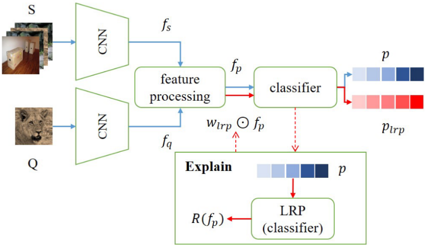
- classify a query image into a set of support classes with few samples only
- difference to vanilla classification: no fixed set of test classes
- test classes given by example images from support classes
- support set classes are variable in the test/training setup
- classifier is a class-transferable similarity



Steps:

- compute prediction with original model $p(f)$ based on feature maps f
- compute explanation scores $R(\cdot)$ for selected feature maps $f \longmapsto \boldsymbol{R}(f) \in [-1,+1]^d$
- re-weight selected feature maps:

$$f_{lrp} = (1 + \boldsymbol{R}(f)) \odot f$$



- train: optimize sum of two losses: original features and reweighted features

$$L = L(y, p(f)) + \lambda L(y, p(f_{lrp}))$$

- prediction time: use unweighted features $p(f)$

– observation: consistent improvement (3 models, several datasets).

LRP-: explanation-guided training using LRP. *T*: transductive inference.

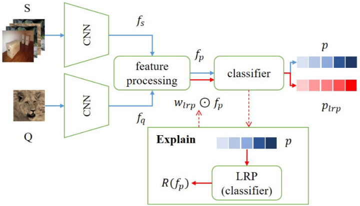
miniImagenet	1-shot	1-shot-T	5-shot	5-shot-T
RN	58.31±0.47%	61.52±0.58%	72.72±0.37%	73.64±0.40%
LRP-RN	**60.06±0.47%**	**62.65±0.56%**	**73.63±0.37%**	**74.67±0.39%**
CAN	**64.66±0.48%**	67.74±0.54%	79.61±0.33%	80.34±0.35%
LRP-CAN	64.65±0.46%	**69.10±0.53%**	**80.89±0.32%**	**82.56±0.33%**
mini-CUB	1-shot	1-shot-T	5-shot	5-shot-T
RN	41.98±0.41%	42.52±0.48%	58.75±0.36%	59.10±0.42%
LRP-RN	**42.44±0.41%**	**42.88±0.48%**	**59.30±0.40%**	**59.22±0.42%**
CAN	44.91±0.41%	46.63±0.50%	63.09±0.39%	62.09±0.43%
LRP-CAN	**46.23±0.42%**	**48.35±0.52%**	**66.58±0.39%**	**66.57±0.43%**
mini-Cars	1-shot	1-shot-T	5-shot	5-shot-T
RN	29.32±0.34%	28.56±0.37%	38.91±0.38%	37.45±0.40%
LRP-RN	**29.65±0.33%**	**29.61±0.37%**	**39.19±0.38%**	**38.31±0.39%**
CAN	31.44±0.35%	30.06±0.42%	41.46±0.37%	40.17±0.40%
LRP-CAN	**32.66±0.46%**	**32.35±0.42%**	**43.86±0.38%**	**42.57±0.42%**
mini-Places	1-shot	1-shot-T	5-shot	5-shot-T
RN	**50.87±0.48%**	**53.63±0.58%**	66.47±0.41%	67.43±0.43%
LRP-RN	50.59±0.46%	53.07±0.57%	**66.90±0.40%**	**68.25±0.43%**
CAN	56.90±0.49%	60.70±0.58%	72.94±0.38%	74.44±0.41%
LRP-CAN	**56.96±0.48%**	**61.60±0.58%**	**74.91±0.37%**	**76.90±0.39%**
mini-Plantae	1-shot	1-shot-T	5-shot	5-shot-T
RN	33.53±0.36%	33.69±0.42%	47.40±0.36%	46.51±0.40%
LRP-RN	**34.80±0.37%**	**34.54±0.42%**	**48.09±0.35%**	**47.67±0.39%**
CAN	36.57±0.37%	36.69±0.42%	50.45±0.36%	48.67±0.40%
LRP-CAN	**38.23±0.45%**	**38.48±0.43%**	**53.25±0.36%**	**51.63±0.41%**

- observation: consistent improvement (3 models, several datasets)

5-way 1-shot	miniImagenet	Cars	Places	CUB	Plantae
GNN	64.47±0.55%	30.97±0.37%	54.64±0.56%	46.76±0.50%	37.39±0.43%
LRP-GNN	**65.03±0.54%**	**32.78±0.39%**	**54.83±0.56%**	**48.29±0.51%**	**37.49±0.43%**

5-way 5-shot	miniImagenet	Cars	Places	CUB	Plantae
GNN	80.74±0.41%	42.59±0.42%	72.14±0.45%	63.91±0.47%	**54.52±0.44%**
LRP-GNN	**82.03±0.40%**	**46.20±0.46%**	**74.45±0.47%**	**64.44±0.48%**	54.46±0.46%

- combined with the feature transform from: (Cross-domain few-shot classification via learned feature-wise transformation, HY Tseng, HY Lee, JB Huang, MH Yang, ICLR 2020), it improves synergistically:



5-way 1-shot	Cars	Places	CUB	Plantae
RN	29.40±0.33%	48.05±0.46%	44.33±0.43%	34.57±0.38%
FT-RN	30.09±0.36%	48.12±0.45%	44.87±0.44%	35.53±0.39%
LRP-RN	30.00±0.32%	48.74±0.45%	45.64±0.42%	36.04±0.38%
LFT-RN	30.27±0.34%	48.07±0.46%	47.35±0.44%	35.54±0.38%
LFT-LRP-RN	**30.68±0.34%**	**50.19±0.47%**	**47.78±0.43%**	**36.58±0.40%**
5-way 5-shot	**Cars**	**Places**	**CUB**	**Plantae**
RN	40.01±0.37%	64.56±0.40%	62.50±0.39%	47.58±0.37%
FT-RN	40.52±0.40%	64.92±0.40%	61.87±0.39%	48.54±0.38%
LRP-RN	41.05±0.37%	66.08±0.40%	62.71±0.39%	48.78±0.37%
LFT-RN	41.51±0.39%	65.35±0.40%	64.11±0.39%	49.29±0.38%
LFT-LRP-RN	**42.38±0.40%**	**66.23±0.40%**	**64.62±0.39%**	**50.50±0.39%**

RelationNet. *FT* and *LFT* indicate the feature-wise transformation layer with fixed or trainable parameters.

- also works with other XAI methods such as gradient times input, see Table 2 / page 19 in https://arxiv.org/pdf/2203.08008.pdf

Case C: Identifying and Improving what universal counterfeit image detectors use:
Discovering Transferable Forensic Features for CNN-generated Images Detection[7]

[7] K Chandrasegaran, NT Tran, A Binder, NM Cheung, ECCV 2022, https://keshik6.github.io/transferable-forensic-features/

- find relevant feature space channels
- How to validate the findings from explainability methods in feature spaces? Eyeballing heatmaps is not informative anymore.
- Analyze what do universal detectors for counterfeit images learn?

Identifying what drives universal counterfeit image detectors

- obtain trained universal counterfeit image detector models (ResNet-50, EfficientNet-B0)
- Discover relevant feature channels

Identifying what drives universal counterfeit image detectors

- compute LRP score for every feature map $R[i, c, h, w]$ for image x_i
- aggregate it into a measure for a channel:

$$R_i[c] = \frac{\sum_{h,w}(R[i, c, h, w])_+}{\sum_{c,h,w} |R[i, c, h, w]|}$$

- average it over images x_i

$$R[c] = \frac{1}{n}\sum_{i=1}^{n} R_i[c]$$

- select top-k feature channels ($k = 114$ for ResNet, $k = 27$ for EffNet-B0) according to $R[c]$

- Validate the discovered top-k relevant feature channels

Identifying what drives universal counterfeit image detectors

Validate the discovered top-k relevant feature channels:

- measure accuracy drop when performing dropout of top-k relevant feature channels
- measure accuracy drop when performing dropout k randomly selected feature channels (perform 5 times, average accuracies)
- measure when performing dropout of bottom-k relevant feature channels
- compare accuracies

Identifying what drives universal counterfeit image detectors

ResNet-50	ProGAN			StyleGAN2			StyleGAN			BigGAN			CycleGAN			StarGAN			GauGAN		
$k = 114$	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**
baseline	100.	100.0	100.	99.1	95.5	95.0	99.3	96.0	95.6	90.4	83.9	85.1	97.9	93.4	92.6	97.5	94.0	89.3	98.8	93.9	96.4
top-k	**69.8**	**99.4**	**3.2**	**55.3**	**89.4**	**11.3**	**56.6**	**90.6**	**13.7**	**55.4**	**86.3**	**18.3**	**61.2**	**91.4**	**17.4**	**72.6**	**89.4**	**35.9**	**71.0**	**95.0**	**18.8**
random-k	100.	99.9	96.1	98.6	89.4	96.9	98.7	91.4	96.1	88.0	79.4	85.0	96.6	81.0	96.2	97.0	88.0	91.7	98.7	91.9	97.1
low-k	100.	100.	100.	99.1	95.6	95.0	99.3	96.0	95.6	90.4	83.9	85.1	97.9	93.4	92.6	97.5	94.0	89.3	98.8	93.9	96.4

EfficientNet-B0	ProGAN			StyleGAN2			StyleGAN			BigGAN			CycleGAN			StarGAN			GauGAN		
$k = 27$	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**
baseline	100.	100.	100.	95.9	95.2	85.4	99.0	96.1	94.3	84.4	79.7	75.9	97.3	89.6	93.0	96.0	92.8	85.5	98.3	94.1	94.4
top-k	**50.0**	**100.**	**0.0**	**54.5**	**94.3**	**7.0**	**52.1**	**97.3**	**2.6**	**53.5**	**97.4**	**3.8**	**47.5**	**100.**	**0.0**	**50.0**	**100.**	**0.0**	**46.2**	**100.**	**0.0**
random-k	100.	99.9	100.	96.5	91.9	89.8	99.2	91.2	97.5	84.5	59.4	89.1	96.9	82.6	95.8	96.7	82.5	93.3	98.1	87.8	96.2
low-k	100.	100.	100.	95.3	88.7	88.3	98.9	90.8	96.1	83.5	70.8	80.8	96.6	85.2	94.1	95.4	91.0	85.4	98.1	91.2	96.4

- validation: top-k feature maps seem to be important

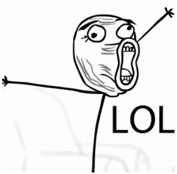
- Visualize the discovered relevant feature channels

Visualize the discovered relevant feature channels

- choose channels c belonging to top-k feature maps
- find $(h^*, w^*) = \text{argmax}_{h,w} R_i[0, c, h, w]$. Identify regions in input space corresponding to this (h^*, w^*)

What do we observe ???

Identifying what drives universal counterfeit image detectors



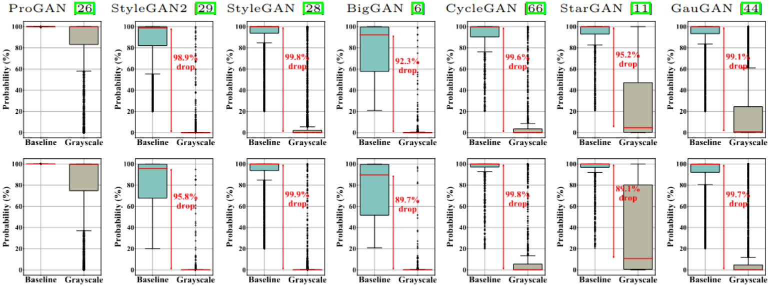
Colorful!

- Is color an important feature? Or still texture?

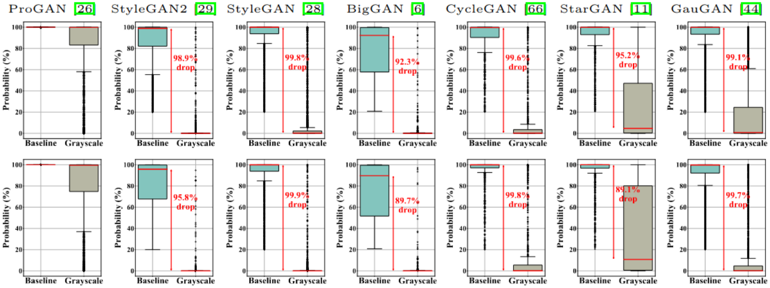
Predictive accuracy from the forward pass:

- Measure cross-GAN detector accuracy with color-ablated counterfeits:
- measure accuracy on colored and gray-scaled counterfeits.

Measure cross-GAN detector accuracy with gray-scaled counterfeits

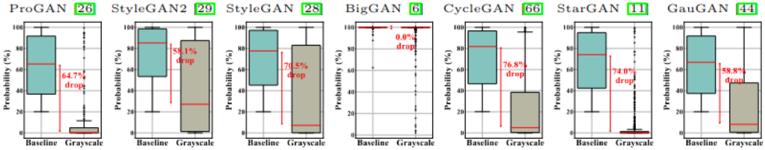


Identifying what drives universal counterfeit image detectors



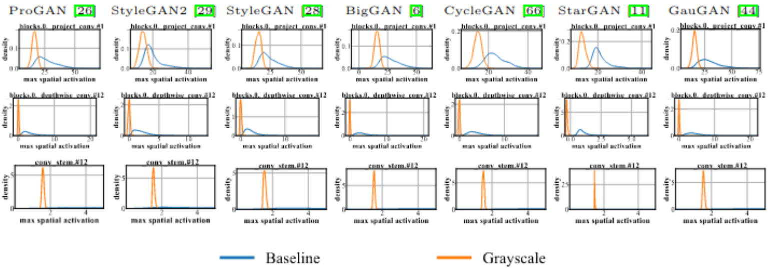
- small drop when using color ablation and the same GAN used for training (leftmost)
- big drop when using gray-scaling and unseen GANs
- color is important for cross-GAN generalization ?!

Measure cross-GAN detector accuracy with gray-scaled counterfeits: other dataset: BigGAN-real/fake, Effnet-B0



– same observation!! Not limited to one dataset

- compare forward pass activation statistics: original vs gray-scaled images for relevant channels (Effnet-B0)

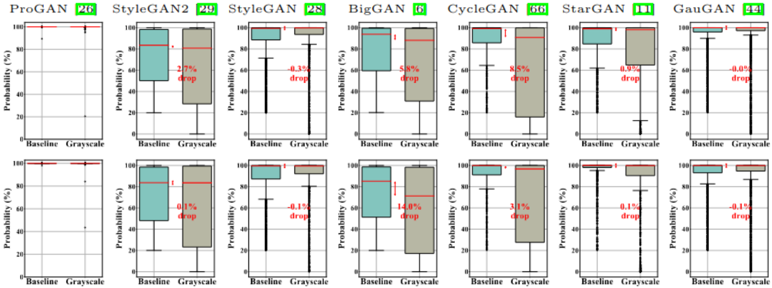


Identifying what drives universal counterfeit image detectors

- Improve cross-GAN detector performance

Improve cross-GAN detector performance

- retrain with 50% grayscaled counterfeits
- measure accuracy for colored vs grayscaled counterfeits



- Improvement!

The end of the talk

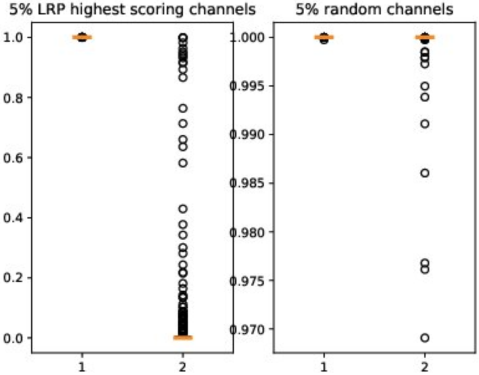
There is no one optimal explanation. LRP works in practice if used properly. Other methods are useful, too.

Credit: Hanabusa Itcho

Can we ablate channels in the generator to fool the detector?

Identifying what drives universal counterfeit image detectors

- Backproject LRP scores from the detector into the image, then into the GAN code which I had.
- yes, but resulting images look absurd!



different top-k values

ResNet-50	ProGAN [26]			StyleGAN2 [29]			StyleGAN [28]			BigGAN [6]			CycleGAN [66]			StarGAN [11]			GauGAN [44]		
	AP	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**
baseline	100.0	100.0	100.0	99.3	95.5	95.0	99.3	96.0	95.6	90.4	83.9	85.1	97.9	93.4	92.6	97.5	94.0	89.3	98.8	93.9	96.4
top-29	**98.6**	99.9	**40.7**	**84.9**	89.2	**62.3**	**84.9**	92.9	**52.4**	**66.8**	85.1	**35.4**	**76.9**	89.4	**42.2**	**87.7**	98.2	**30.4**	**85.6**	94.0	**45.6**
top-57	**96.8**	99.9	**26.3**	**84.0**	91.1	**54.9**	**84.0**	92.4	**50.6**	**63.2**	83.3	**30.9**	**71.4**	88.9	**30.6**	**86.0**	98.1	**29.0**	**82.4**	92.7	**41.2**
top-114	**69.8**	99.4	**3.2**	**56.6**	89.4	**11.3**	**56.6**	90.6	**13.7**	**55.4**	86.3	**18.3**	**61.2**	91.4	**17.4**	**72.6**	89.4	**35.9**	**71.0**	95.0	**18.8**
top-228	**58.6**	99.3	**2.3**	49.2	29.2	76.6	49.2	24.5	76.2	51.6	48.1	50.6	50.2	83.0	16.2	59.3	46.7	66.4	60.7	65.5	52.5

EfficientNet-B0	ProGAN [26]			StyleGAN2 [29]			StyleGAN [28]			BigGAN [6]			CycleGAN [66]			StarGAN [11]			GauGAN [44]		
	AP	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**	**AP**	**Real**	**GAN**
baseline	100.	100.	100.	99.0	95.2	85.4	99.0	96.1	94.3	84.4	79.7	75.9	97.3	89.6	93.0	96.0	92.8	85.5	98.3	94.1	94.4
top-5	**91.8**	99.9	**14.5**	**68.9**	75.1	**53.7**	**68.9**	74.6	**38.3**	**57.4**	74.6	**38.3**	**78.9**	85.5	**54.4**	**82.4**	94.2	**40.8**	**70.7**	97.4	**13.9**
top-27	**50.0**	100.	**0.0**	**52.1**	94.3	**7.0**	**52.1**	97.3	**2.6**	**53.5**	97.4	**3.8**	**47.5**	100.0	**0.0**	**50.0**	100.	**0.0**	**46.2**	100.	**0.0**
top-49	**50.0**	100.	**0.0**	**50.0**	100.	**0.0**	**50.0**	100.	**0.0**	**50.0**	100.	**0.0**	**50.0**	100.	**0.0**	**50.0**	100.	**0.0**	**50.0**	100.	**0.0**