Interpretable Deep Learning

EE807: Recent Advances in Deep Learning

Lecture 15

Slide made by

Jun Hyun Nam

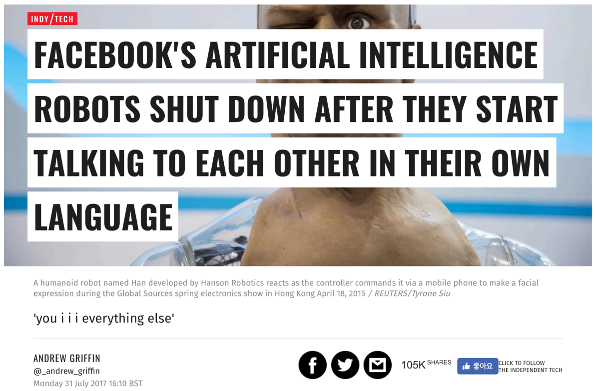
KAIST EE

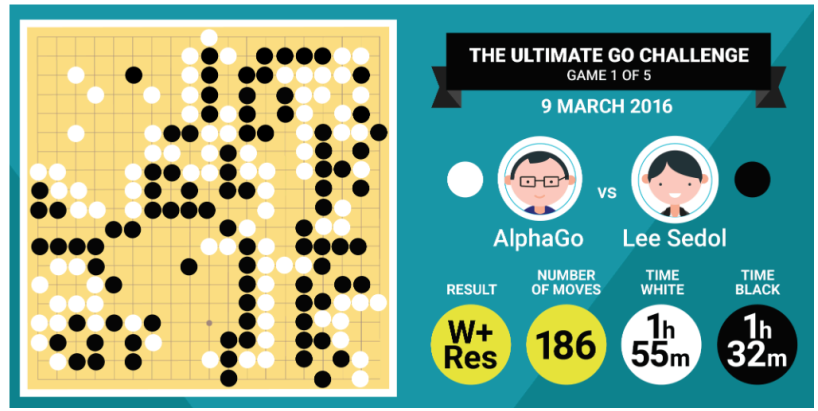
Table of Contents

Table of Contents

Why Interpretability?

- Recently, deep learning shows superior performance in various tasks
- However, we don't know yet why they work so well

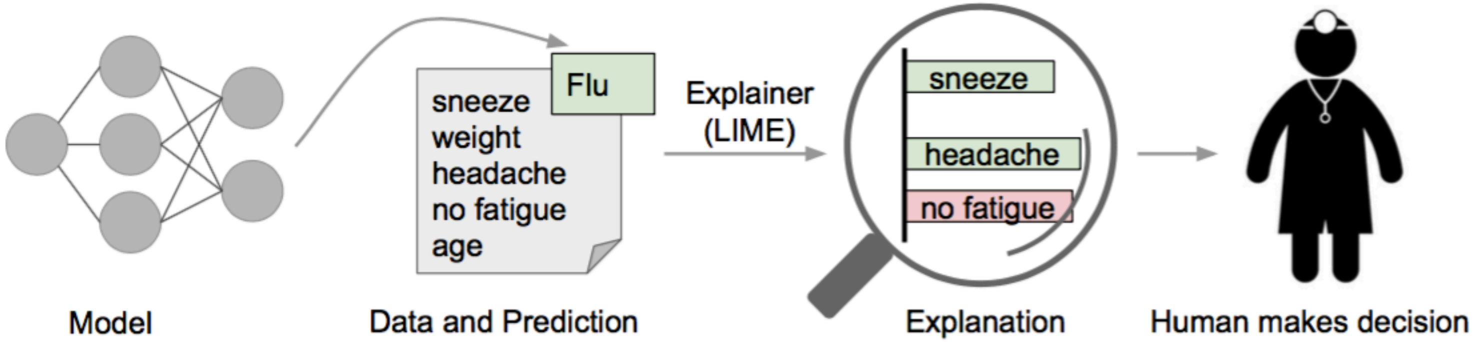




- When it fails, it can cause critical issues

What is Interpretability?

- Interpretation is the process of giving explanations

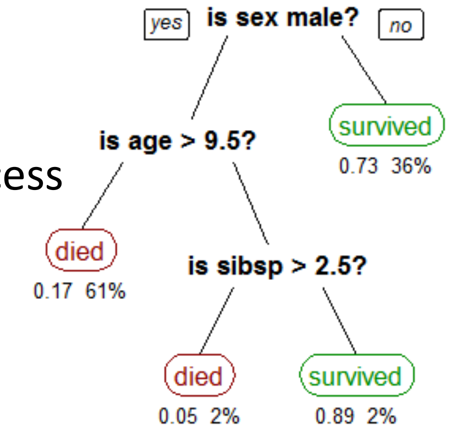


[Ribeiro et al. '16]

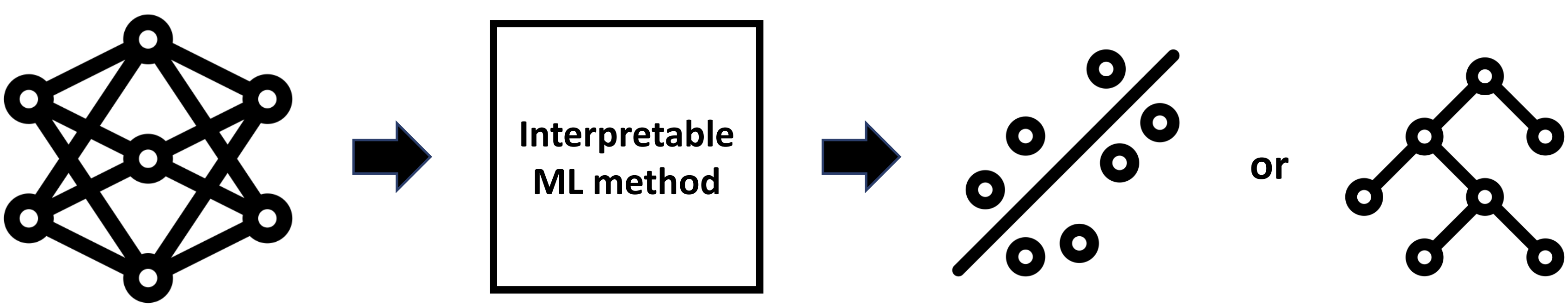
- Situations when ML interpretation can be helped
  - **Safety**: We want to make sure the system is making sound decisions
  - **Debugging**: We want to understand why a system doesn't work
  - **Science**: We want to understand something new
  - **Legal**: We are legally required to provide an explanation
  - **Ethics**: We don't want to discriminate against particular groups

- **Linear model**
  - Consider $y = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$
  - **Question**: How much input feature x_i contributed to (or affected) output y?
  - Answer: β_i

- **Decision tree**
  - **Question**: How much ‘age’ affected probability of survived?
  - Answer: Don’t know
  - Instead of per-feature attribution, we know its decision process



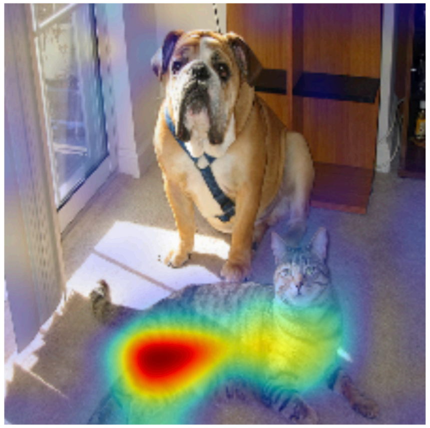
- Many interpretable ML approaches provides explanation of the original model in one of two forms

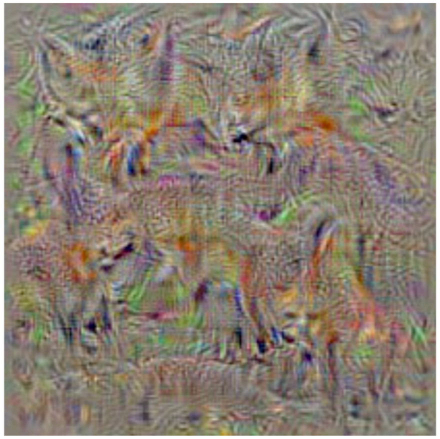


Created by Knut M. Synstad from Noun Project

- **Local explanation**
  - Explain a single prediction
  - e.g. which part of the image affected the prediction most (visual explanation)
  - e.g. find a training data most responsible to the prediction (influence function)

- **Global explanation**
  - Describe the entire model behavior
  - e.g. generate a synthetic image that maximizes certain output (feature visualization)
  - e.g. discover a human-friendly concept related to each neuron (network dissection)
  - e.g. find a training data most responsible to the model (influence function)


(c) Grad-CAM 'Cat'


kit fox

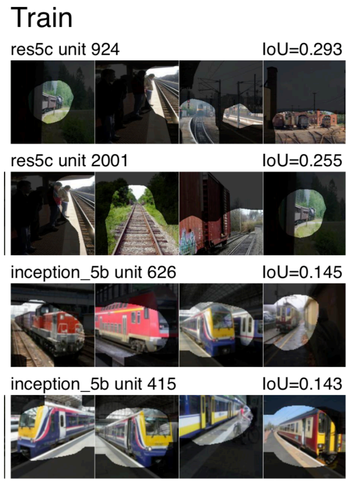
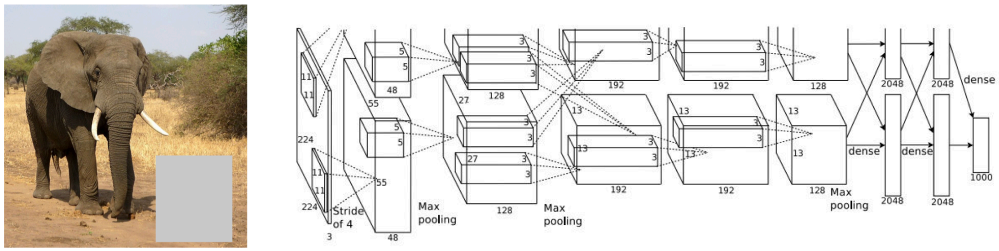


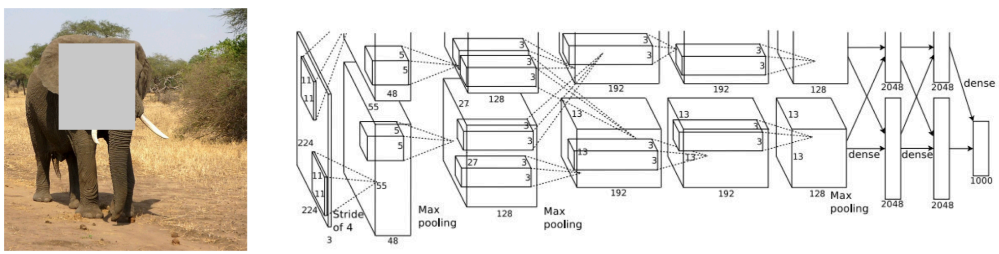
Table of Contents

Occlusion Map [Zeiler et al., 2014]

- **Idea**: Mask part of the image with gray patch before feeding to CNN, and check how much the prediction changes

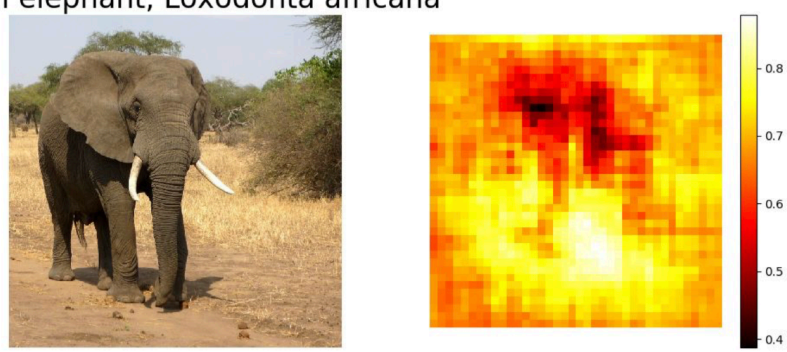


P(elephant) = 0.95

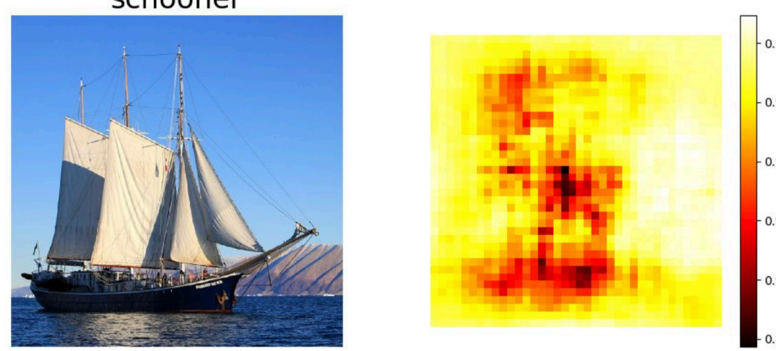


P(elephant) = 0.75

African elephant, Loxodonta africana



schooner



- **Problem**: Removing information with gray patch is too heuristic
- **Idea**: Simulate the absence of a feature by marginalizing the feature

- **Goal**: The attribution of i-th feature for given image and $\mathbf{x}$ and class c

$$p(c|\mathbf{x}) - p(c|\mathbf{x}_{\setminus i})$$

where $\mathbf{x}_{\setminus i}$ represents the absence of x_i in $\mathbf{x}$

$$p(c|\mathbf{x}_{\setminus i}) = \sum_{x_i} p(x_i|\mathbf{x}_{\setminus i})p(c|\mathbf{x}_{\setminus i}, x_i)$$

- Note that $p(x_i|\mathbf{x}_{\setminus i})$ is computationally expensive

- Assume x_i is independent of the other features, i.e., $p(x_i|\mathbf{x}_{\setminus i}) \approx p(x_i)$

$$p(c|\mathbf{x}_{\setminus i}) \approx \sum_{x_i} p(x_i)p(c|\mathbf{x}_{\setminus i}, x_i)$$

- The prior probability $p(x_i)$ is usually approximated by the empirical distribution

Prediction Difference Analysis [Zintgraf et al., 2017]

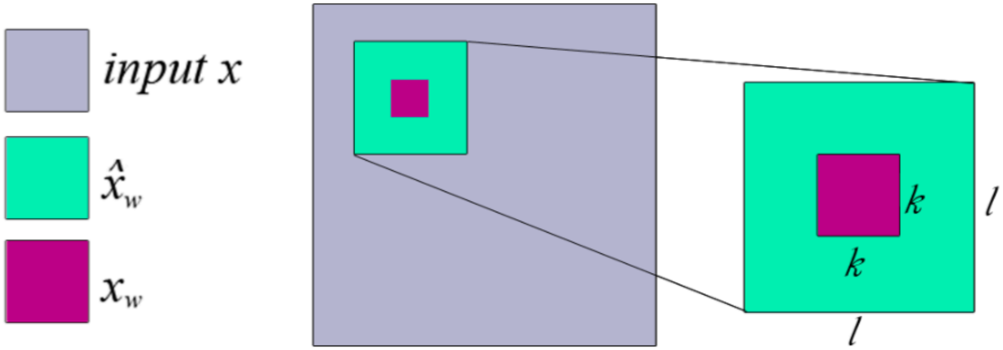
- **Idea**: Simulate the absence of a feature by marginalizing the feature

$$p(c|\mathbf{x}_{\setminus i}) = \sum_{x_i} p(x_i|\mathbf{x}_{\setminus i}) p(c|\mathbf{x}_{\setminus i}, x_i)$$

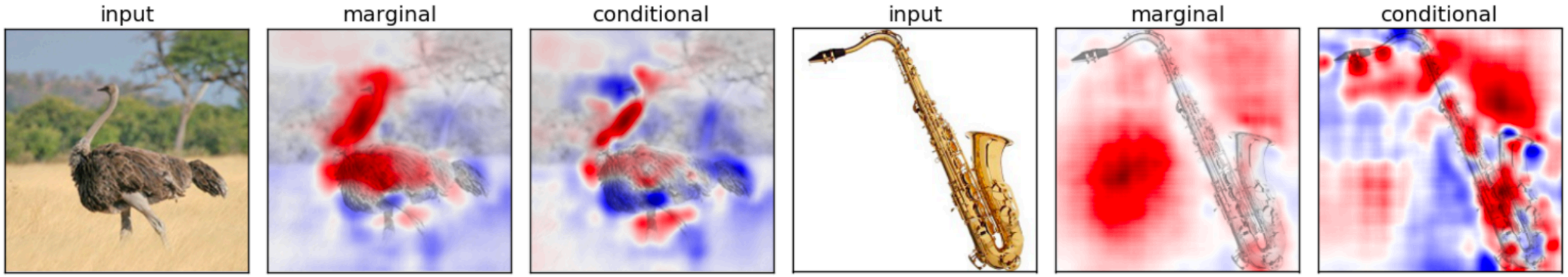
- **Problem**: $p(x_i|\mathbf{x}_{\setminus i}) \approx p(x_i)$ is a very crude approximation
  - e.g. a pixel's value is highly dependent on other pixels

- **Observations**
  - A pixel depends most strongly on a small neighborhood around it
  - The conditional of a pixel given its neighborhood does not depend on the position

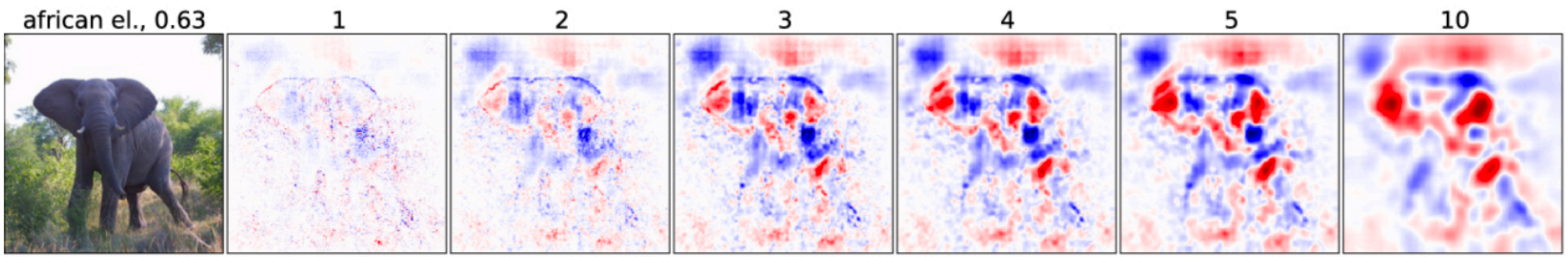
- For a pixel x_i , one can find a patch $\hat{\mathbf{x}}_i$ than contains x_i and $p(x_i|\mathbf{x}_{\setminus i}) \approx p(x_i|\hat{\mathbf{x}}_i)$



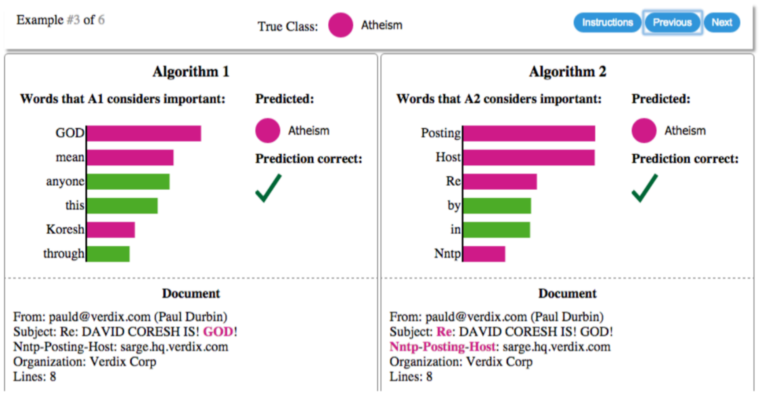
- **Results**
  - Marginal vs. conditional sampling

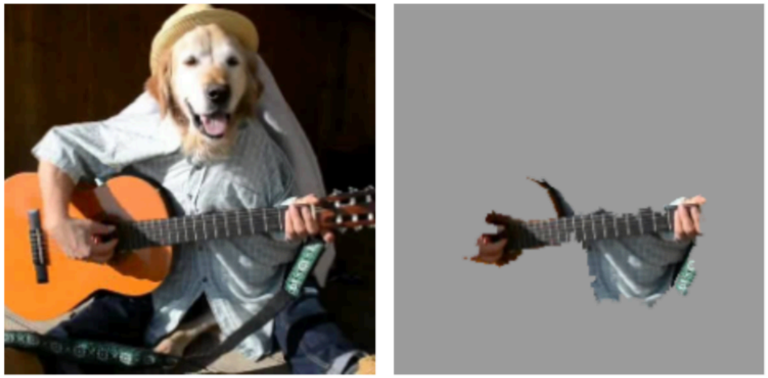


  - Different window sizes

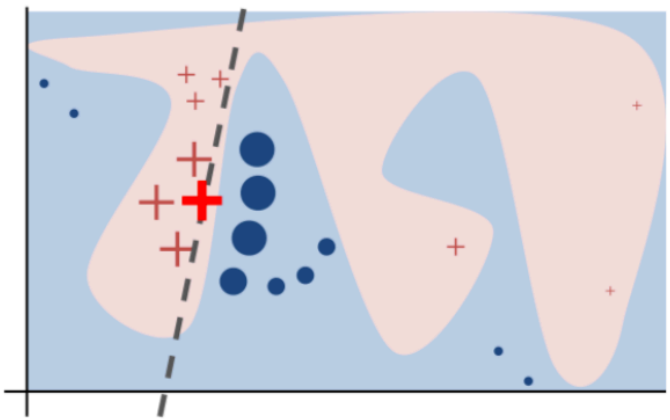


- Remember that a sparse linear model is a good explanation model



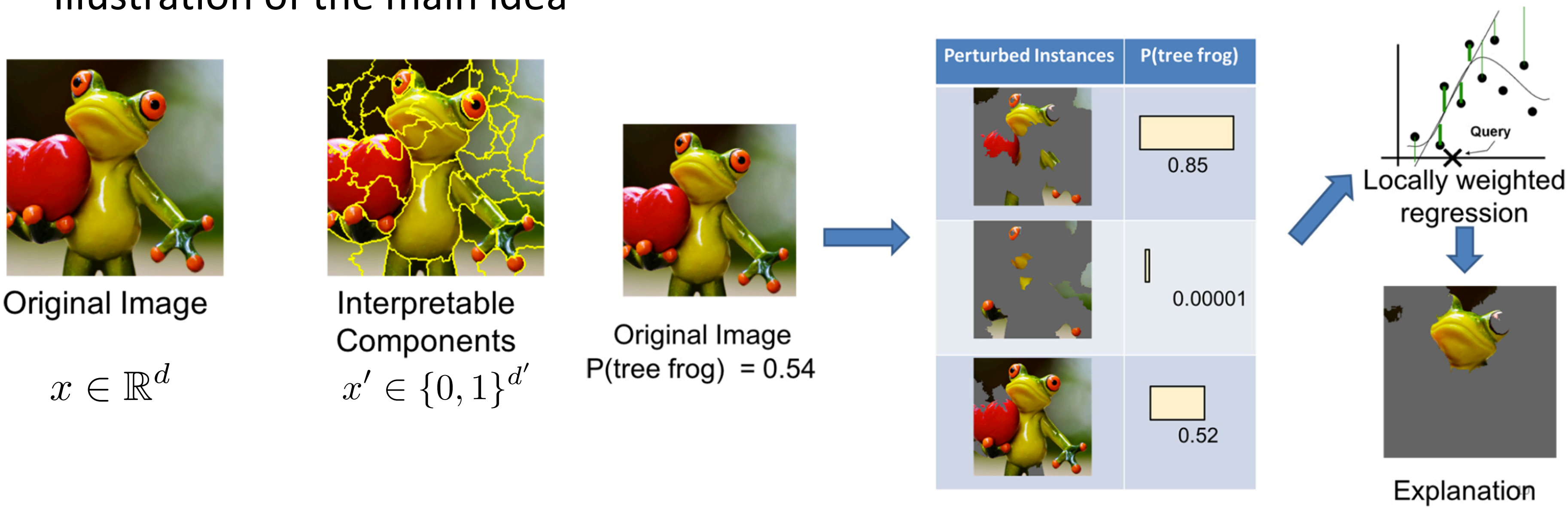

(a) Original Image (b) Explaining *Electric guitar*

- **Idea**: Local linear approximation
  - Explain the entire model is hard, but a single prediction is easier
  - Approximate the model in a local region around the single prediction by a linear classifier



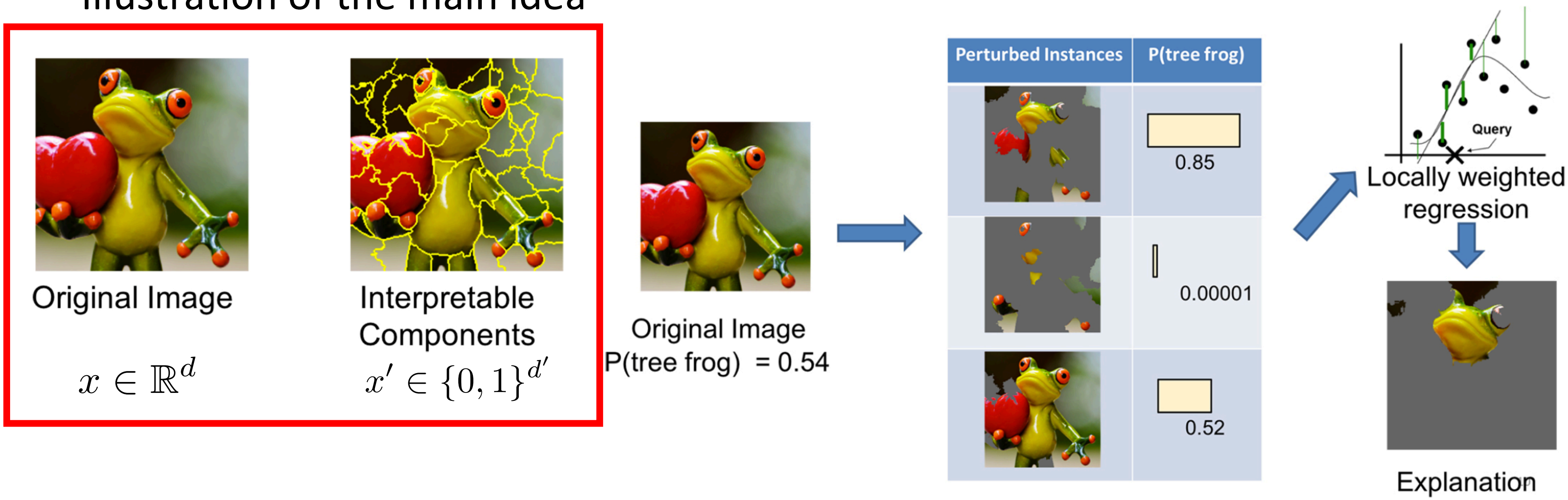
Local Interpretable Model-agnostic Explanations (LIME) [Ribeiro et al., 2016]

- Illustration of the main idea



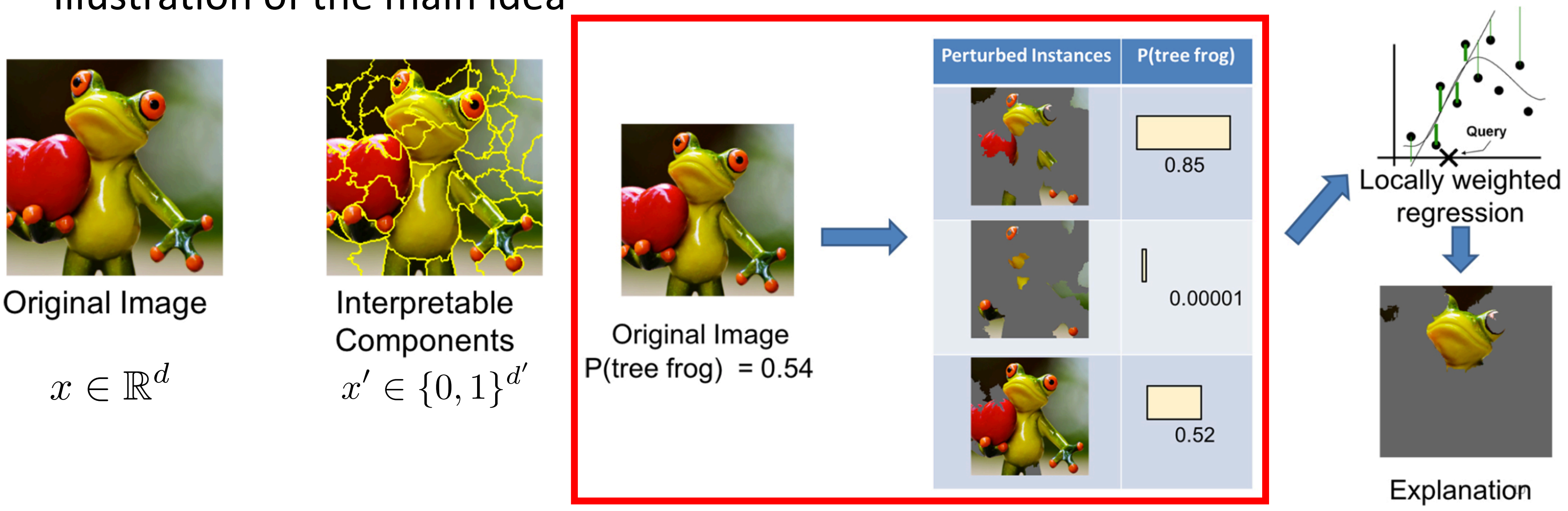
- Overall Procedure
    1. Decompose original input to interpretable representation
    2. Model local region around given input by sampling
    3. Approximate original model as a linear classifier

- Illustration of the main idea



- **Step 1**: Interpretable representation
  - Understandable to humans
  - For text classification, a binary vector indicating the presence or absence of a word
  - For image classification, a binary vector indicating the presence or absence of a contiguous patch of similar pixels
  - $x \in \mathbb{R}^d$: original representation / $x' \in \{0,1\}^{d'}$: its interpretable representation

- Illustration of the main idea



- **Step 2**: Model local region around given input
  - Sample instances around x by drawing nonzero elements of $x' \in \{0,1\}^{d'}$ uniformly at random
  - Given a perturbed sample $z' \in \{0,1\}^{d'}$, recover the original representation $z \in \mathbb{R}^d$
  - Compute $f(z)$: the prediction of model for each perturbed output

Local Interpretable Model-agnostic Explanations (LIME) [Ribeiro et al., 2016]

- Illustration of the main idea

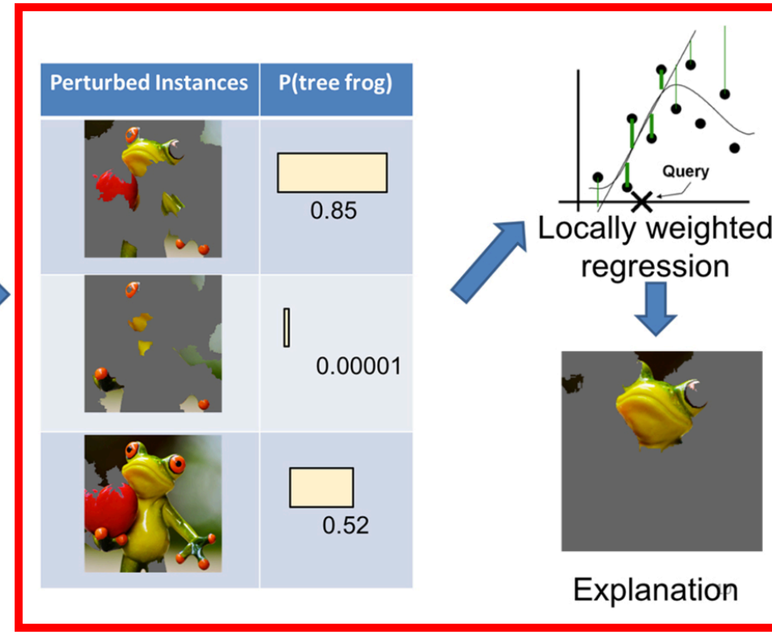
Original Image

$x \in \mathbb{R}^d$

Interpretable Components

$x' \in \{0,1\}^{d'}$

Original Image
P(tree frog) = 0.54



- **Step 3**: Approximate original model as a linear classifier
  - Fit a linear classifier $g(z') = w_g \cdot z'$ and use it as an explanation model

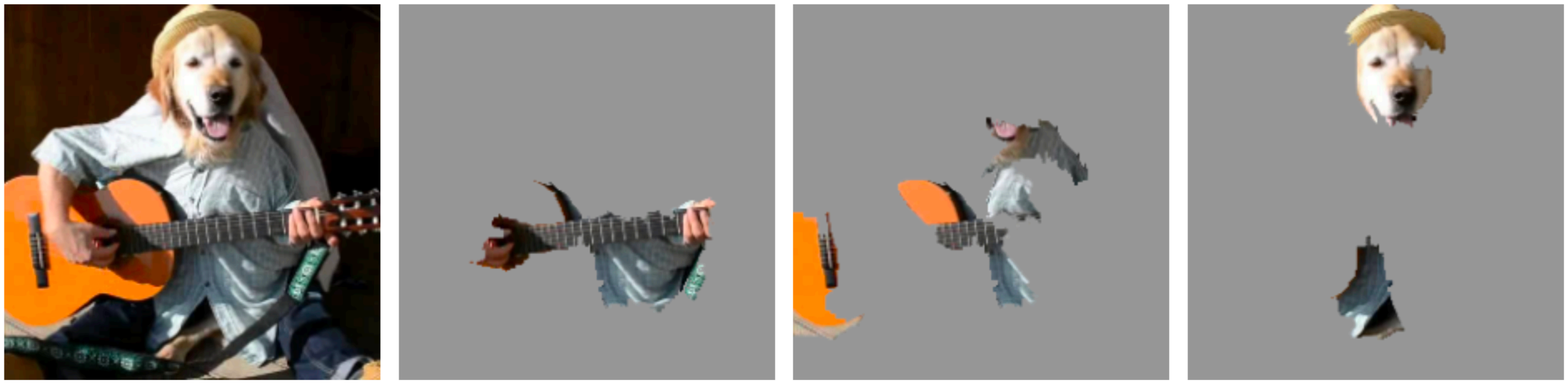
$$\mathcal{L}(f, g, \Pi_x) = \sum_{z, z' \in \mathcal{Z}} \Pi_x(z)(f(z) - g(z'))^2$$

  - $\Pi_x(z)$ defines locality (e.g. $\Pi_x(z) = \exp(-\|x - z\|_2^2/0.1)$)
  - Final objective

$$\xi(x) = \arg\min_{g \in G} \underbrace{\mathcal{L}(f, g, \Pi_x)}_{\text{local fidelity}} + \underbrace{\Omega(g)}_{\text{measure of complexity}}$$

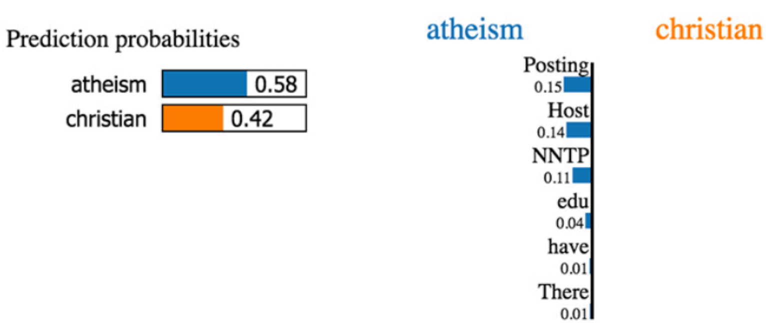
Local Interpretable Model-agnostic Explanations (LIME) [Ribeiro et al., 2016]

- **Results**: Can be applied to any model
  - Top 3 predictions of Inception-v3 for ImageNet dataset



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

  - Random forest prediction for the 20 newsgroups dataset



Text with highlighted words

From: johnchad@triton.unm.edu (jchadwic)
Subject: Another request for Darwin Fish
Organization: University of New Mexico, Albuquerque
Lines: 11
NNTP-Posting-Host: triton.unm.edu

Hello Gang,

There have been some notes recently asking where to obtain the
DARWIN fish.
This is the same question I have and I have not seen an answer on
the
net. If anyone has a contact please post on the net or email me.

Table of Contents

Saliency Map [Simonyan et al., 2014]

- **Problem**: Perturbation-based methods are too slow
- **Idea:** Use gradient of output with respect to the input as the attribution

- **Goal**: Find the influence on the score $S_c(I_0)$ for given image I_0
  - Consider the linear score model for class c

$$S_c(I) = w_c^\top I + b_c$$

where I : image, w_c, b_c : the weight vector and the bias of the model
  - w_c defines the importance of the corresponding pixels of I for the class c

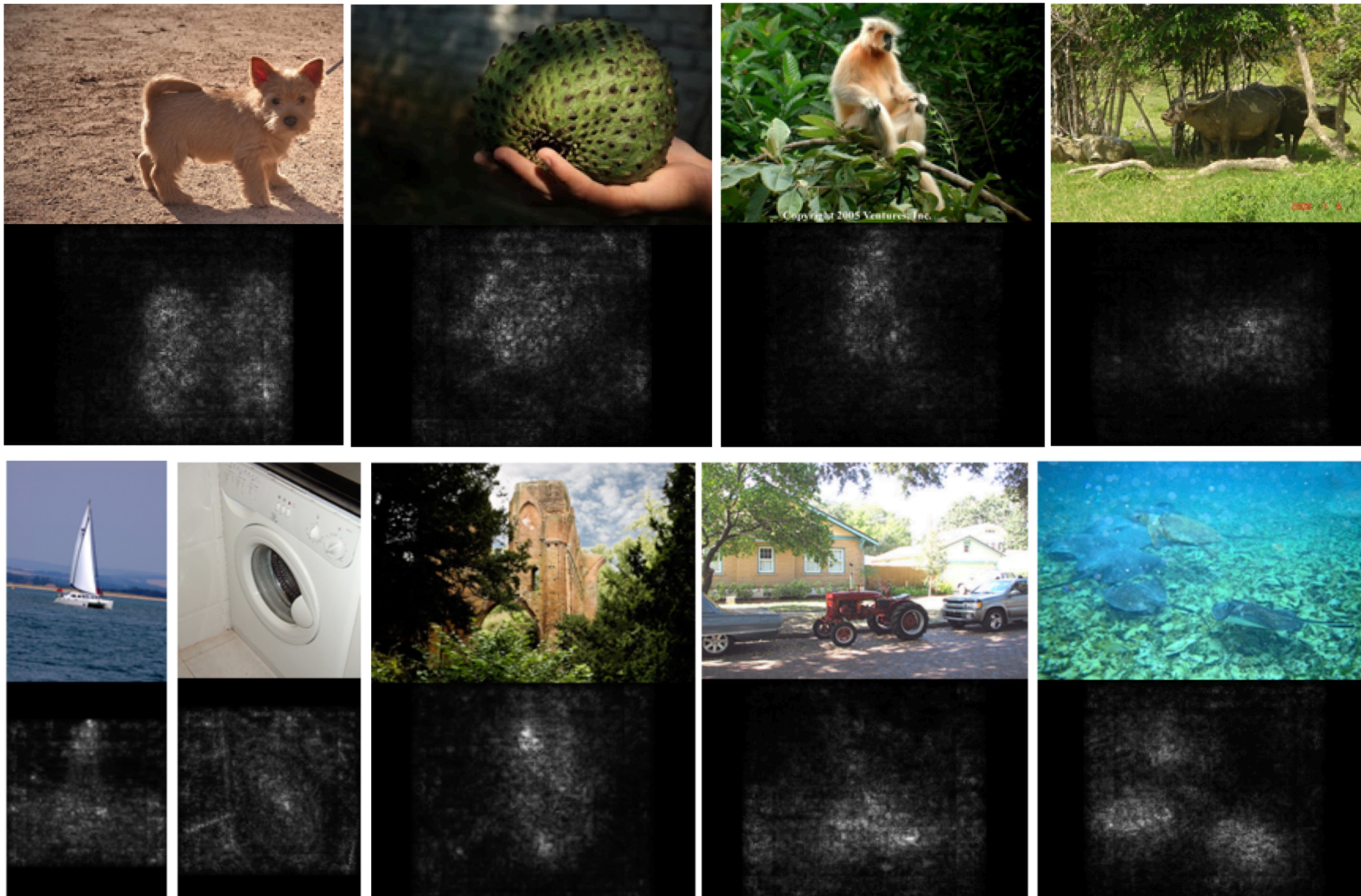
  - In case of non-linear/complex models, approximate $S_c(I)$ by the first-order Taylor expansion

$$S_c(I) \approx w^\top I + b$$

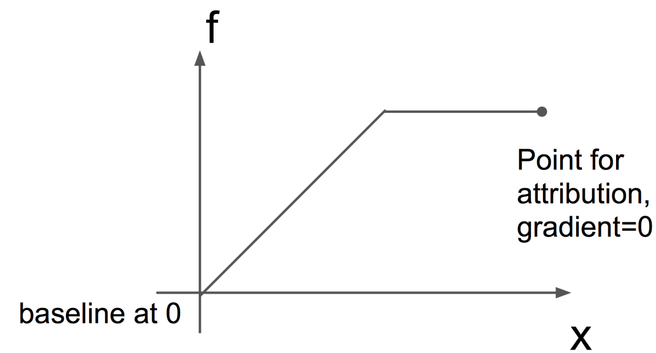
where $w = \left.\frac{\partial S_c}{\partial I}\right|_{I=I_0}$

Saliency Map [Simonyan et al., 2014]

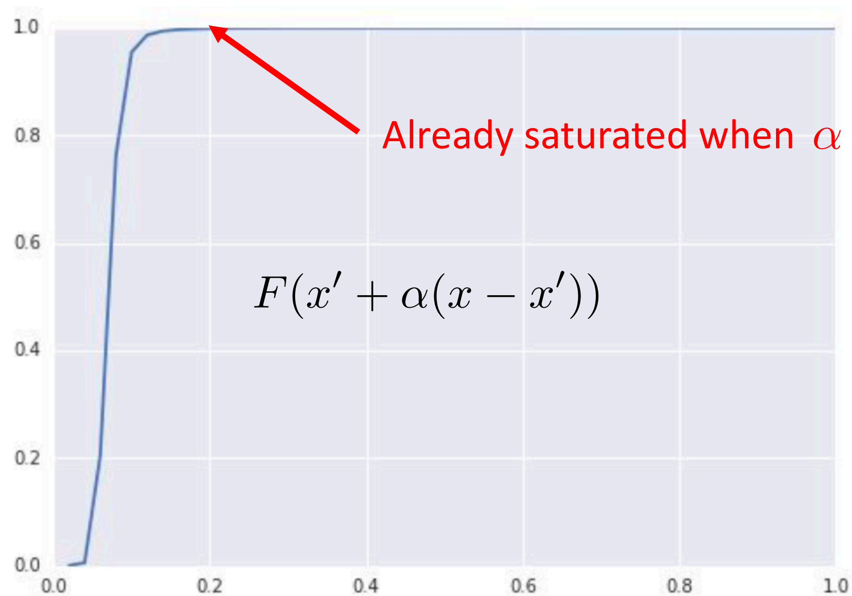
- **Results:** Without any additional annotation, gradient can localize the object



- **Problem**: Prediction score might saturate
  - For high confidence prediction, small perturbation in input does not change the prediction value



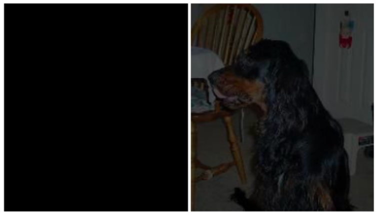
Prediction score



Already saturated when $\alpha = 0.2$

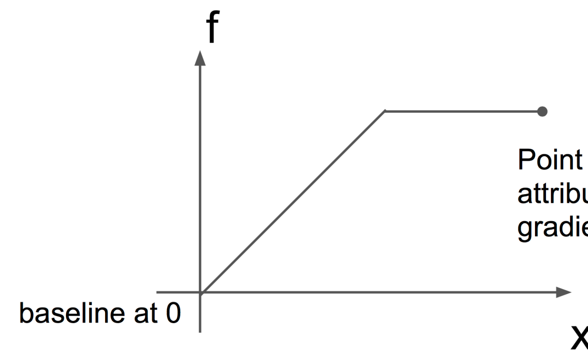
F: prediction score
x : original image
x': baseline image

intensity α

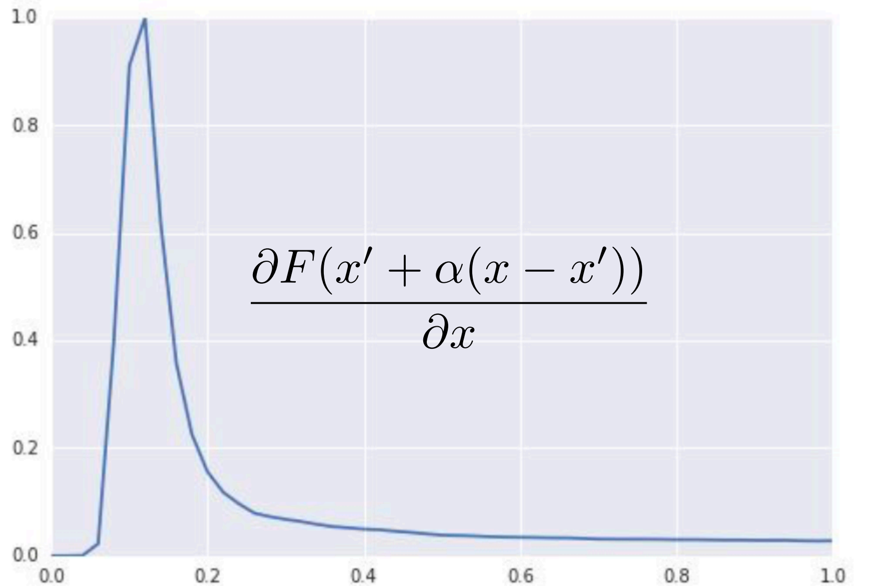


Integrated Gradients [Sundararajan et al., 2017]

- **Problem**: Prediction score might saturate
  - For high confidence prediction, small perturbation in input does not change the prediction value



Average pixel gradient (normalized)



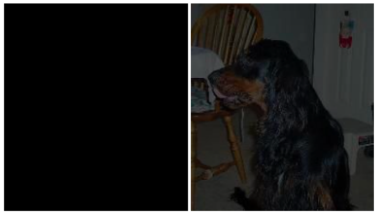
$$\frac{\partial F(x' + \alpha(x - x'))}{\partial x}$$

F: prediction score

x : original image

x': baseline image

intensity α



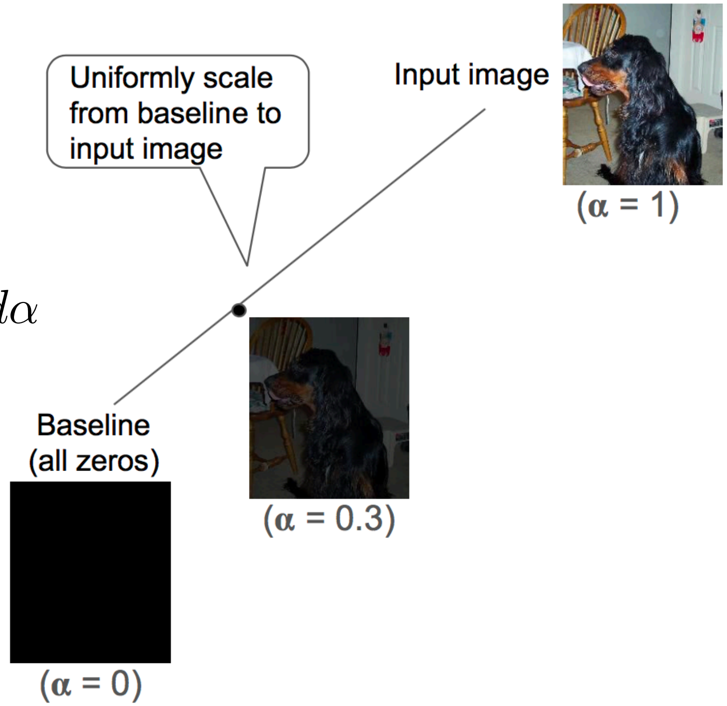
Integrated Gradients [Sundararajan et al., 2017]

- **Idea**: Compute all the gradients for images from baseline to actual image

- Construct a sequence of images interpolating from a baseline (black) to the actual image
- Average the gradients across these images

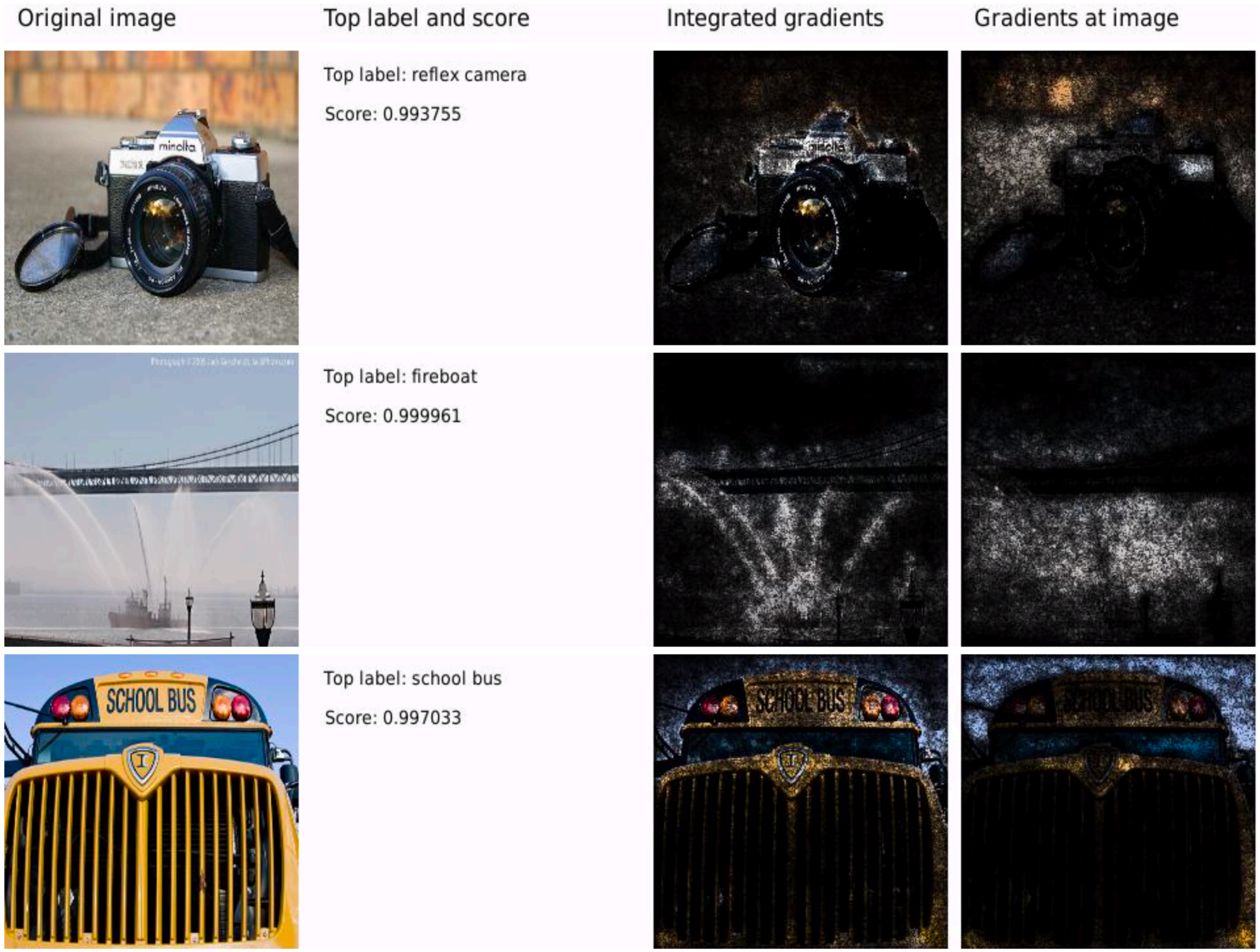
$$\mathrm{IG}_i(x) = (x_i - x'_i) \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha(x - x'))}{\partial x_i} d\alpha$$

  - F is the prediction function for the label
  - x_i is the intensity of ith pixel
  - $\mathrm{IG}_i(x)$ is the integrated gradient w.r.t. the ith pixel

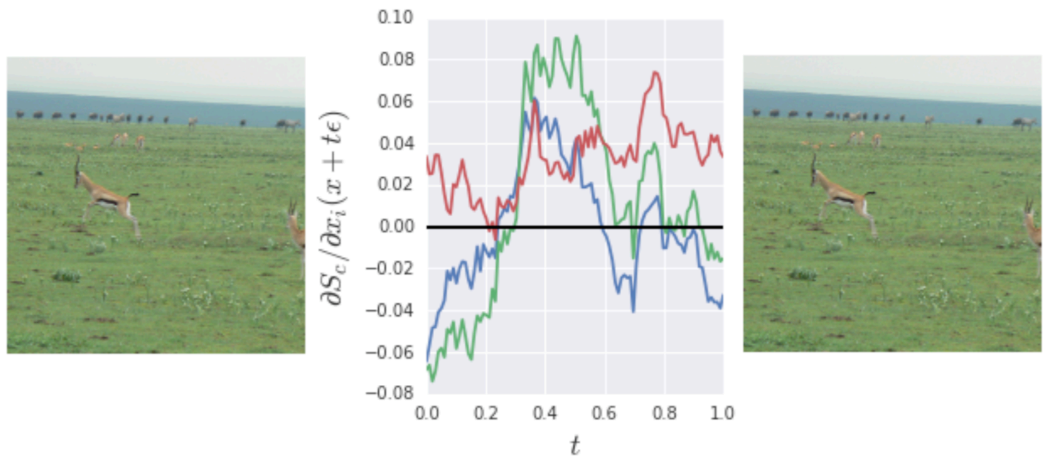


- Properties
  - **Sensitivity**: A variable changes output, then the variable should get an attribution
  - **Insensitivity**: A variable has no effect on the output gets no attribution
  - **Completeness**: $\sum_{i=1}^{n} \mathrm{IG}_i(x) = F(x) - F(x')$

- **Results:** For high confidence predictions,
  integrated gradients provide discriminative region



- **Problem**: Gradients strongly fluctuate!
  - Given image x, and an image pixel x_i, plots values of $\max_i \frac{\partial S_c}{\partial x_i}(x + t\epsilon)$ for a short line segment $x + t\epsilon$



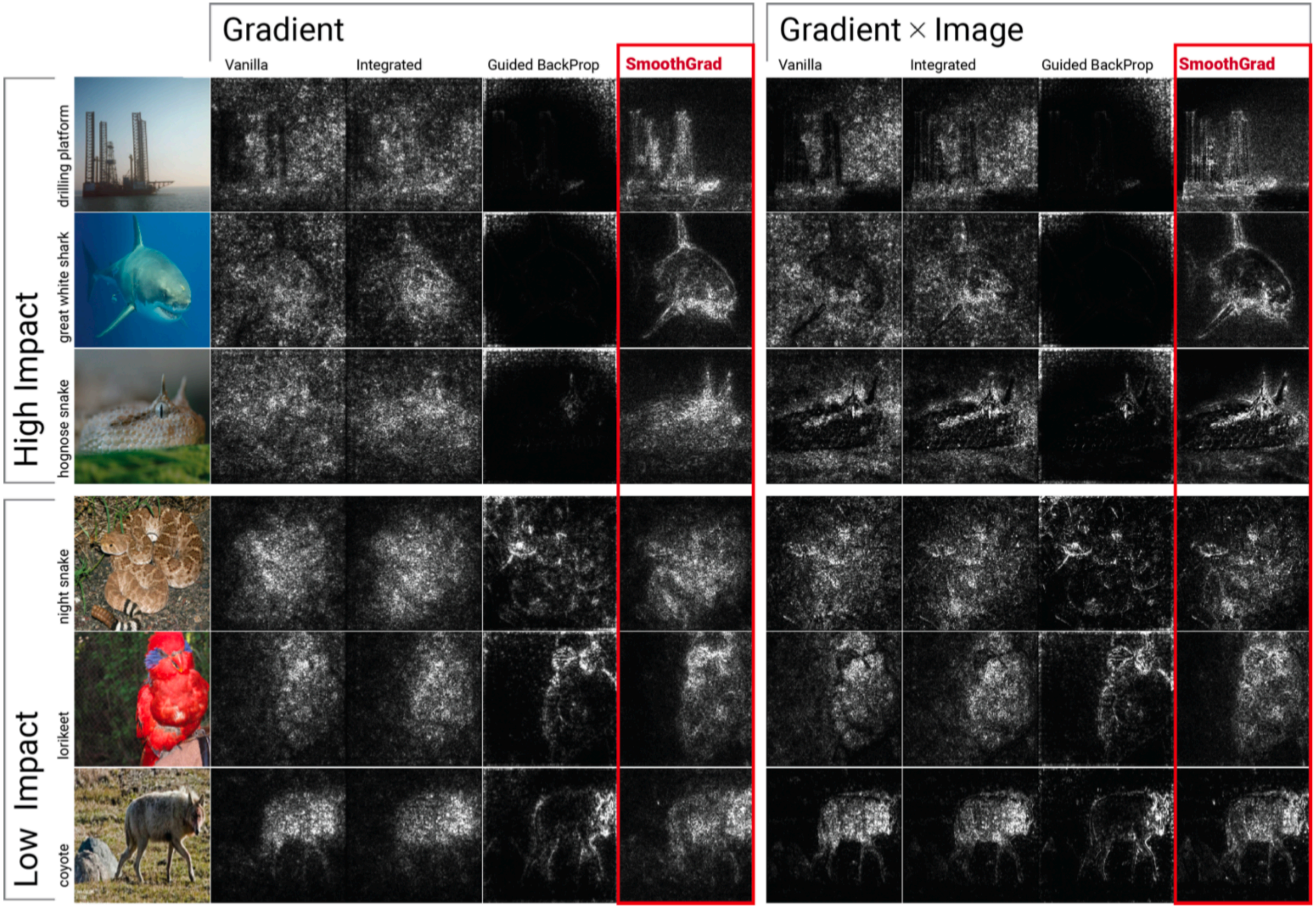
  - Even x and $x + \epsilon$ are indistinguishable, the partial derivative rapidly fluctuate
- **Idea**: Use a local average of gradient values

$$\mathrm{SG}(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial S_c}{\partial x}(x + g_i)$$

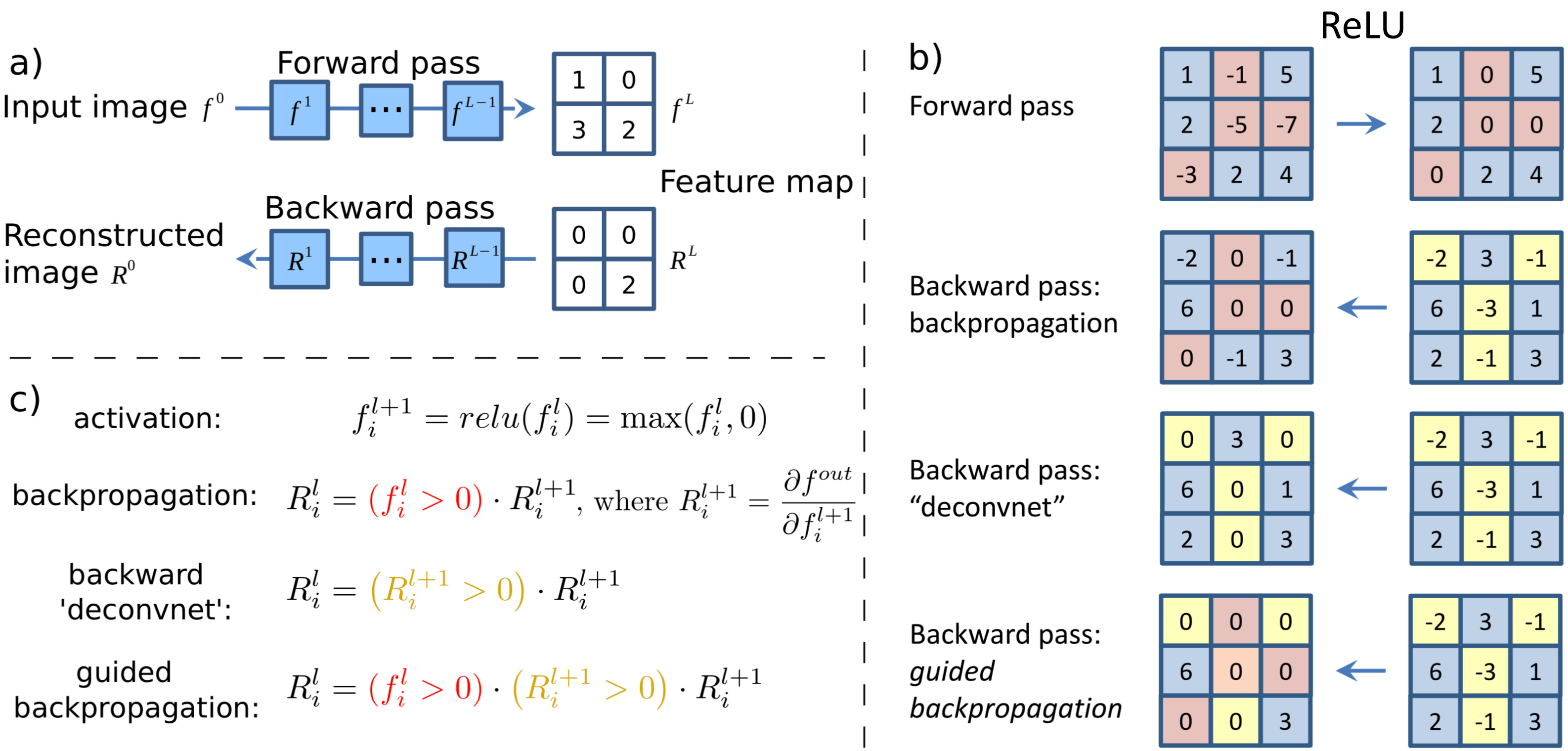
where noise vectors $g_i \sim \mathcal{N}(0, \sigma^2)$ are drawn i.i.d. from a normal distribution

SmoothGrad [Smilkov et al., 2017]

- **Results:** Simple noise-adding method can dramatically improve the quality of saliency map

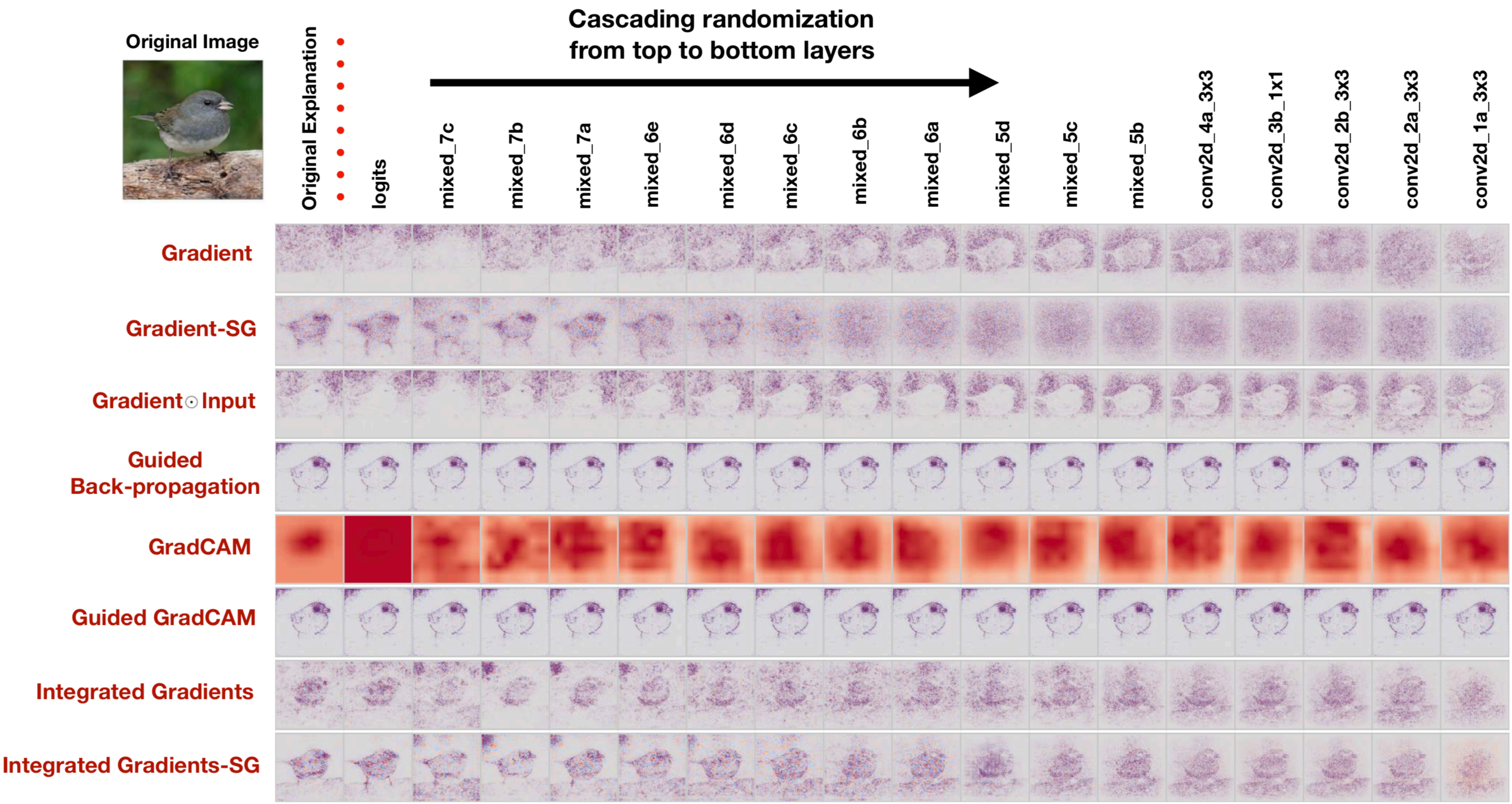


- **Deconvolution** [Zeiler et al., 2014]
  - Reverse operation of convolution
- **Guided Backpropagation** [Springenberg et al., 2015]
  - Backpropagate only positive gradients through each ReLU
- Both methods visualize the activations of high layer neurons (also the prediction)



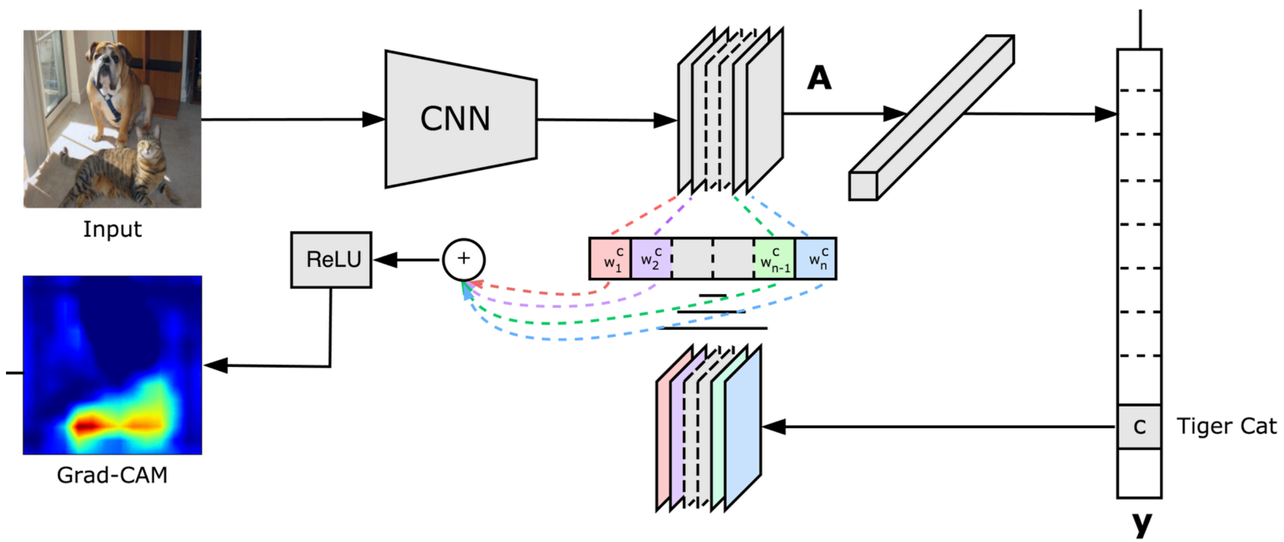
Grad-CAM [Selvararaju et al., 2017]

- **Problem**: Many pixel-level attribution methods insensitive to model parameter [Adebayo et al., 2018]



- **Idea**: Activation-level attribution instead of pixel-level attribution
- Gradient-based extension of CAM [Zhou et al., 2015]
- Can be applied to any CNN based model
  - Image classification, image captioning or visual question answering
- Use GAP of gradients instead of weights after GAP layer
  - y^c : the score for class c, A^k : feature map of the last convolutional layer

$$\alpha_k^c = \sum_{i,j} \frac{\partial y^c}{\partial A_{ij}^k} \qquad L_{\text{Grad-CAM}}^c = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right)$$

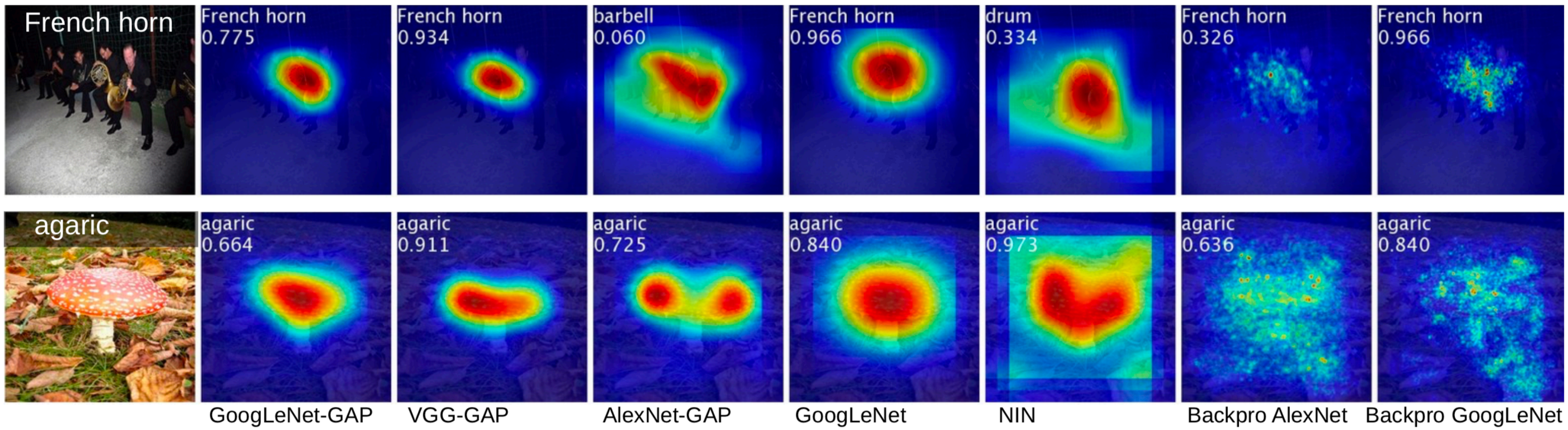


- **Idea**: Activation-level attribution instead of pixel-level attribution
- Gradient-based extension of CAM [Zhou et al., 2015]
- Can be applied to any CNN based model
  - Image classification, image captioning or visual question answering
- Use GAP of gradients instead of weights after GAP layer
  - y^c : the score for class c, A^k : feature map of the last convolutional layer

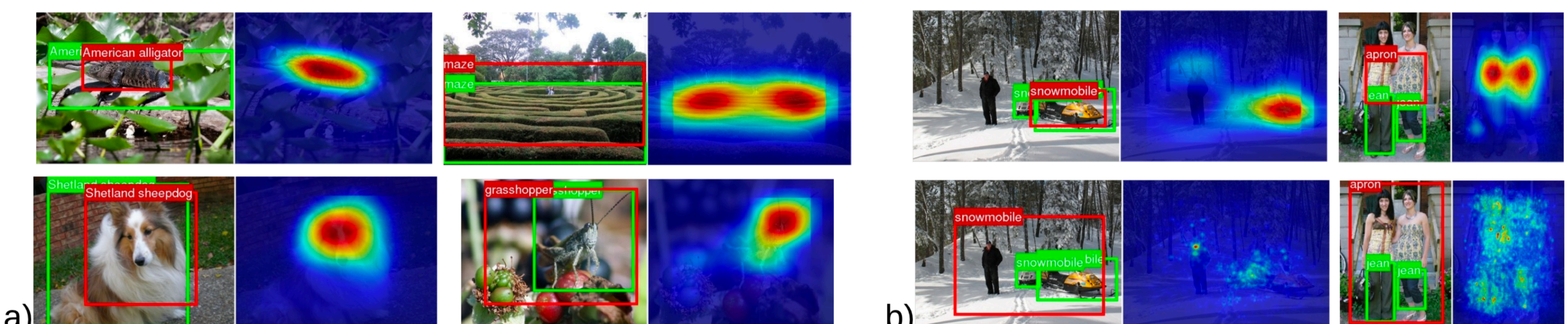
$$\alpha_k^c = \sum_{i,j} \frac{\partial y^c}{\partial A_{ij}^k} \qquad L_{\text{Grad-CAM}}^c = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right)$$

- Typically, the conv activation has low-resolution → low resolution explanation
- Less affected by CNN architecture prior → more sensitive to model parameter

- **Results**
  - CAM vs. Saliency map

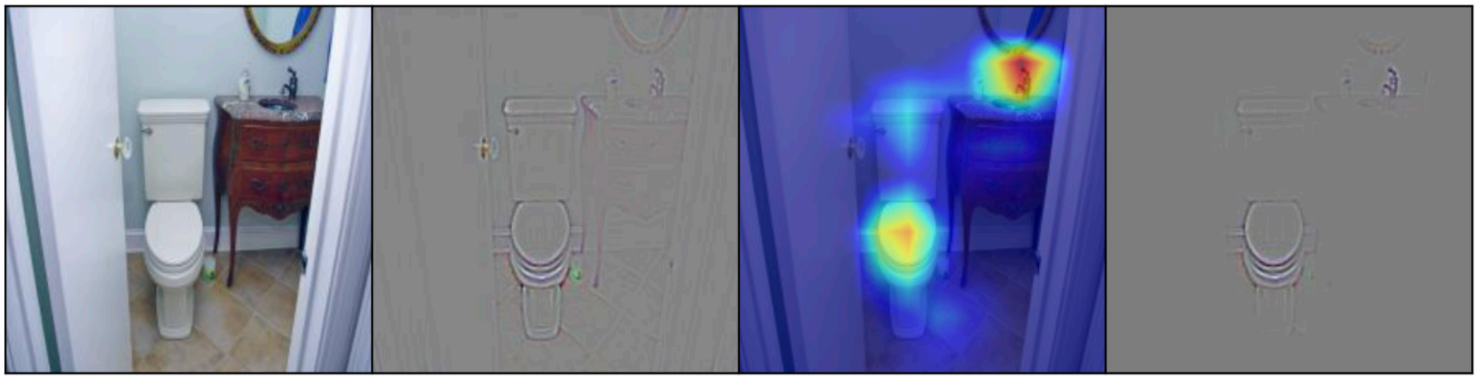


  - Examples of localization (green: ground truth / red: predicted)

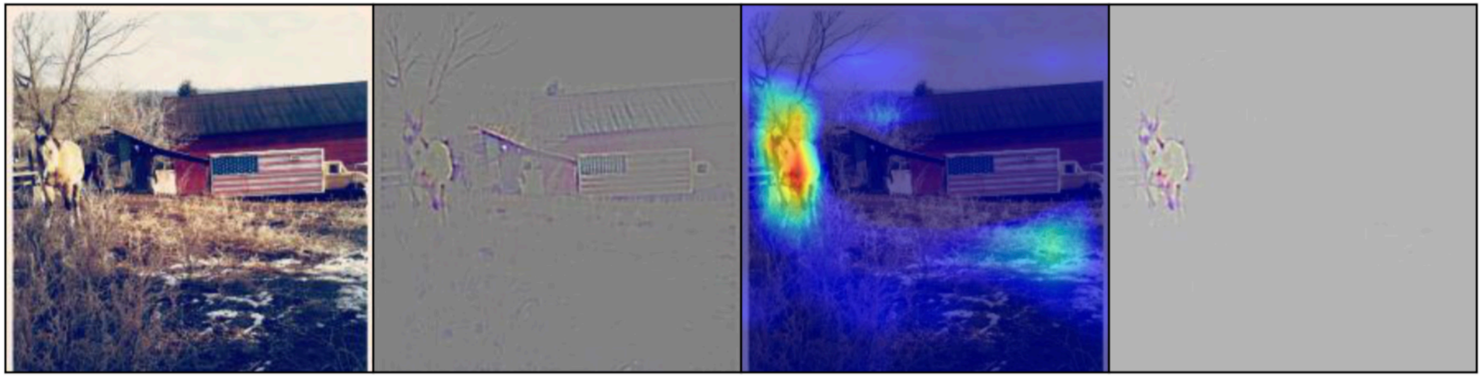


- **Results**: focus on right place without any attention module
  - Visual explanations for captioning

Guided Backprop | **Grad-CAM** | Guided Grad-CAM



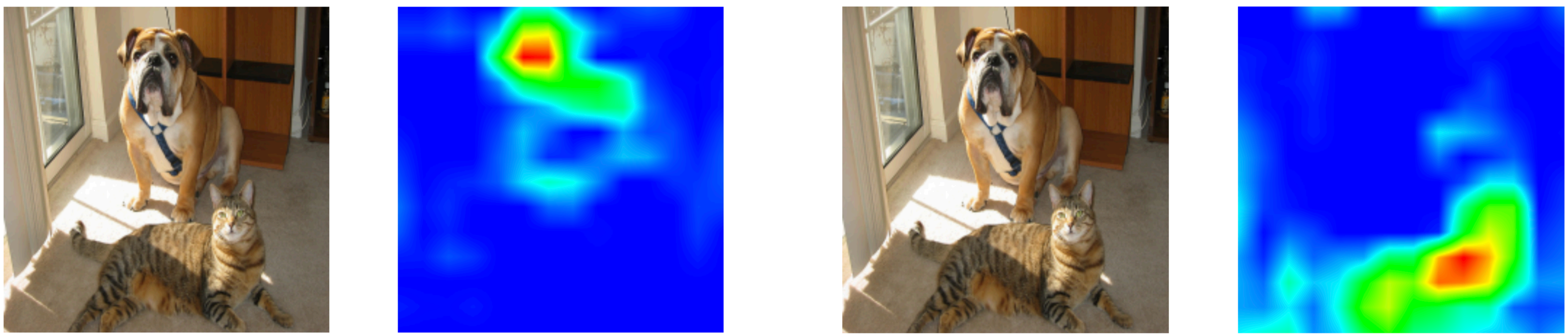
A bathroom with a toilet and a sink



A horse is standing in a field with a fence in the background

- **Results**: can discriminate different objects
  - Visual explanations for VQA

What animal is in this picture? (left) Answer: dog / (right) Answer: cat



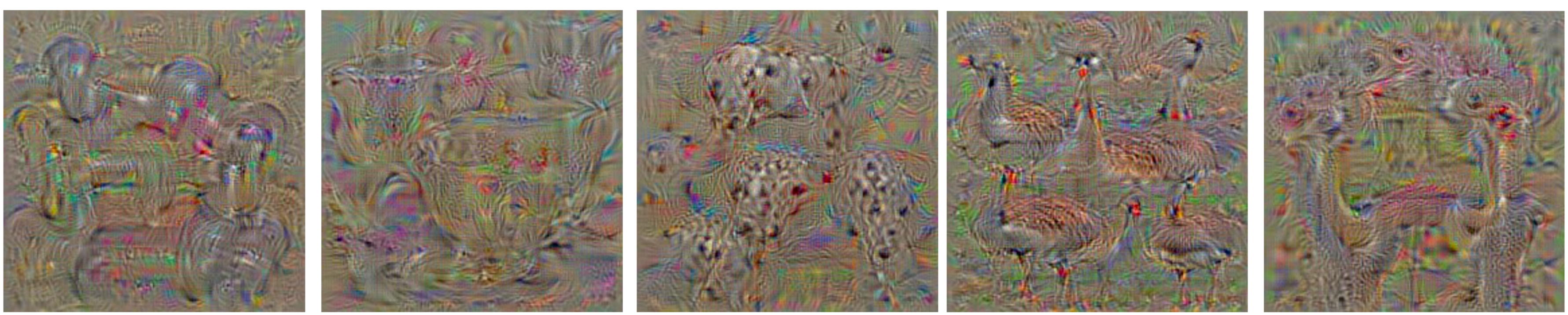
What color is the hydrant? (left) Answer: yellow / (right) Answer: green

Table of Contents

- **Goal**: Generate a synthetic image that maximally activates a neuron
  - So far, we have focused on finding which part of an input that a neuron (or output) responds to
  - Can observe the models behavior when classify image to certain class
- **Idea**: Solve the following optimization

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2$$

  - Initialized image to zeros
  - Forward image to compute current class scores
  - Backprop to get gradient of neuron value w.r.t. image pixels
  - Make a small update to the image
- **Results**: Different aspects of class appearance are captured



dumbbell cup dalmatian goose ostrich

Network Dissection [Bau et al., 2017]

- **Goal**: Interpreting deep visual representation and quantifying their interpretability

- **Idea**: Network dissection
  1. Identify a broad set of human-labeled visual concepts
  2. Gather hidden variables' response to known concepts
  3. Quantify alignment of hidden variable – concept pairs

- **Step 1**: Use the broadly and densely labeled (Broden) dataset
  - Gather images from various dataset
  - Total 63,305 pixel-level annotated images, 1,197 visual concepts

Table 1. Statistics of each label type included in the data set.

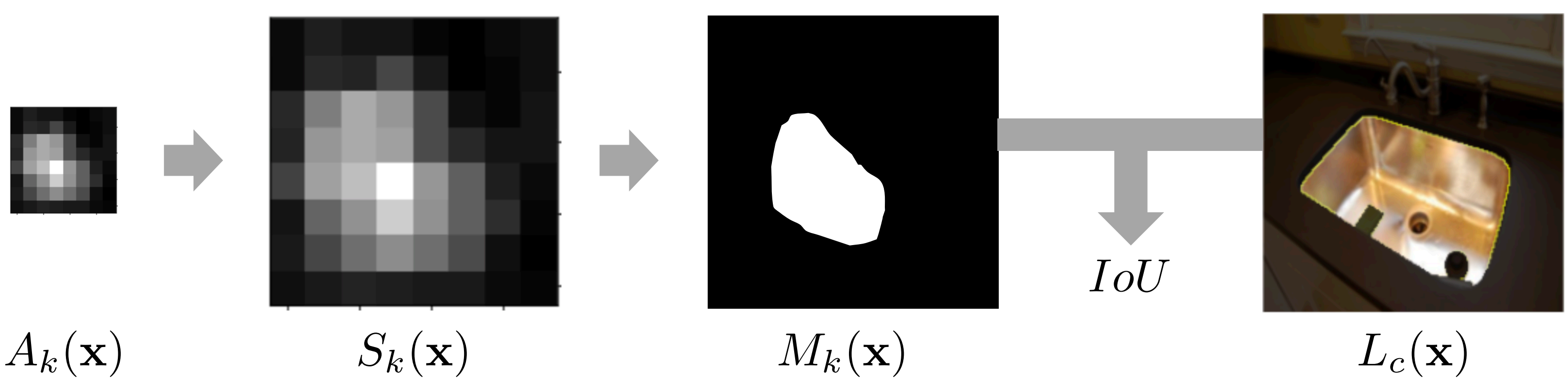
Category	Classes	Sources	Avg sample
scene	468	ADE [43]	38
object	584	ADE [43], Pascal-Context [19]	491
part	234	ADE [43], Pascal-Part [6]	854
material	32	OpenSurfaces [4]	1,703
texture	47	DTD [7]	140
color	11	Generated	59,250

- **Step 2**: Gather hidden variables' response
  - For every input image $\mathbf{x}$ in the Broden dataset, collect the activation map $A_k(\mathbf{x})$ of every convolutional unit k
  - Define the binary segmentation $M_k(\mathbf{x}) = \mathbf{1}\{S_k(\mathbf{x}) \geq T_k\}$
  - $S_k(\mathbf{x})$: scaled up activation map of $A_k(\mathbf{x})$ (same size as the image)
  - T_k : some threshold value

- **Step 3**: Scoring unit interpretability
  - The score of unit k for concept c is reported as a dataset-wide IoU score

$$IoU_{k,c} = \frac{\sum_{\mathbf{x}} |M_k(\mathbf{x}) \cap L_c(\mathbf{x})|}{\sum_{\mathbf{x}} |M_k(\mathbf{x}) \cup L_c(\mathbf{x})|}$$

  - $L_c(\mathbf{x})$: ground truth mask of image $\mathbf{x}$ for concept c

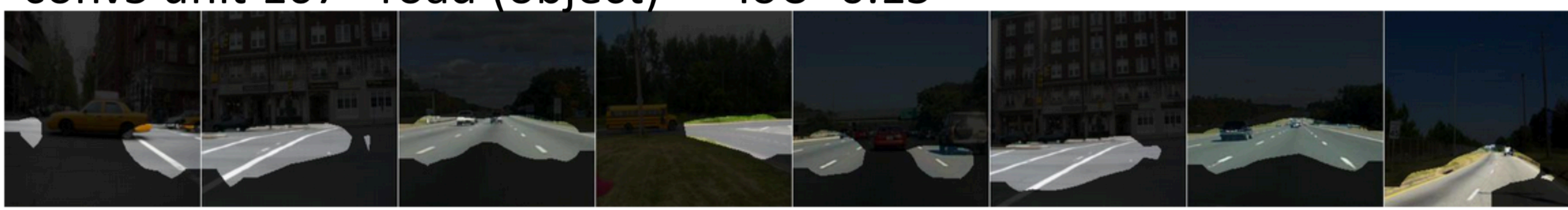


Network Dissection [Bau et al., 2017]

- **Results**: Object detector emerges even when the model trained on scene dataset
  - High-scored (interpretable) convolutional units

conv5 unit 79 car (object) IoU=0.13

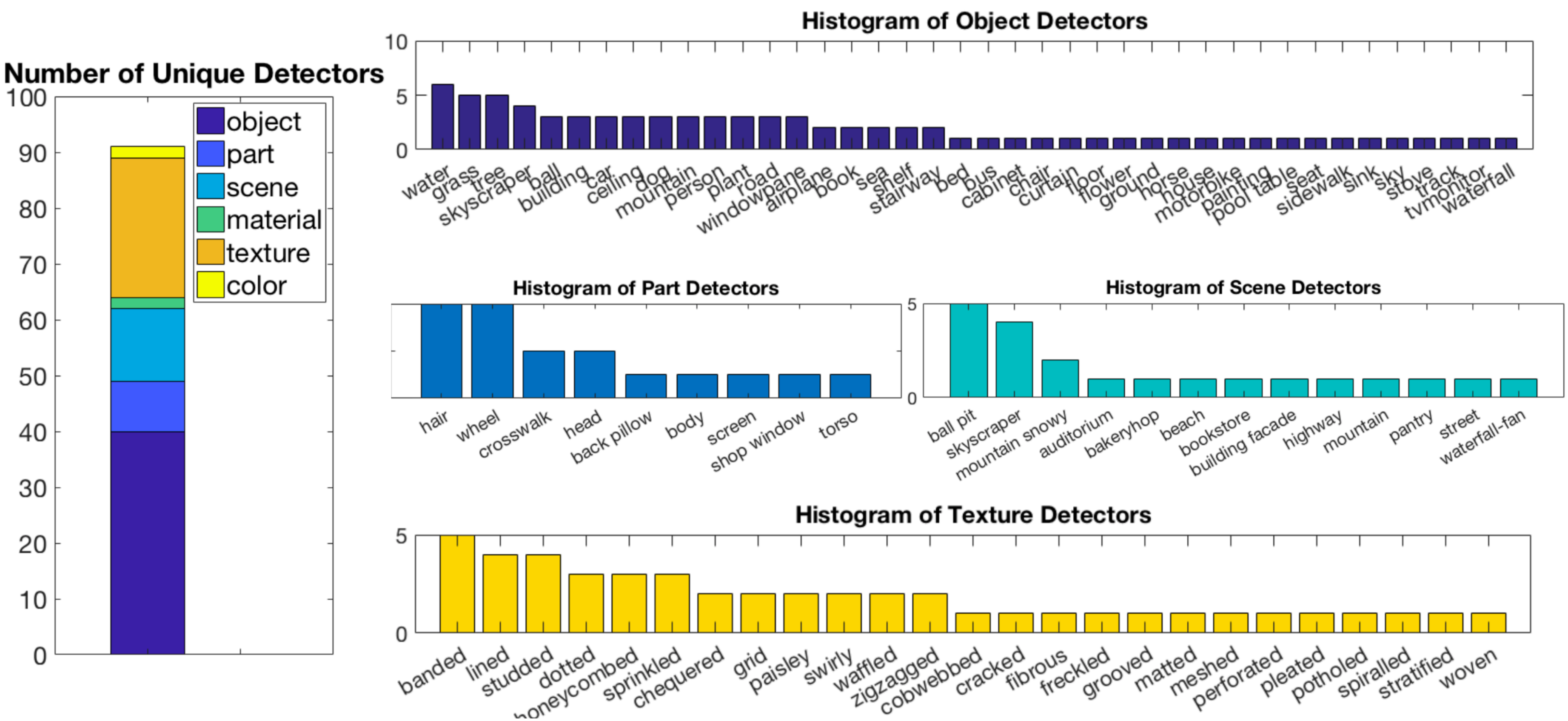
conv5 unit 107 road (object) IoU=0.15



conv5 unit 144 mountain (object) IoU=0.13

Network Dissection [Bau et al., 2017]

- **Results**
  - Dissection report (AlexNet / trained on places 365)



Influence Function [Koh et al., 2017]

- **Goal**: Identify most responsible training point for a given prediction
  - Retraining the model can be prohibitively expensive!

- **Idea**: Find $\hat{\theta}_{-z} = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{z_i \neq z}^{n} L(z_i, \theta)$ using influence function
  - Training points $z_1, \cdots, z_n$ are given
  - The empirical risk minimizer is given by $\hat{\theta} = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$

- Measure influence of $L(z, \theta)$ on parameter $\hat{\theta}$ to approximate $\hat{\theta}_{-z}$

- Influence function

$$I(T) = \lim_{t \to 0^+} \frac{T(tG + (1-t)F) - T(F)}{t}$$

where T : an estimator, F, G: distribution

Influence Function [Koh et al., 2017]

- Approximate $\hat{\theta}_{-z}$ in terms of perturbation ϵ

$$\hat{\theta}_{-z} \approx \hat{\theta} - \frac{1}{n} \frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\bigg|_{\epsilon=0} \qquad \text{where} \qquad \hat{\theta}_{\epsilon,z} = \arg\min_{\theta\in\Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i,\theta) + \epsilon L(z,\theta)$$

  - From a classic result [Cook et al., 1982],

$$\mathcal{I}_{\text{up,params}}(z) = \frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\bigg|_{\epsilon=0} = -H_{\hat{\theta}}^{-1} \nabla_\theta L(z,\hat{\theta})$$

- Influence of z on the loss function at z_{test}

$$\begin{aligned}
\mathcal{I}_{\text{up,loss}}(z, z_{\text{test}}) &= \frac{dL(z_{\text{test}}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\bigg|_{\epsilon=0} \\
&= \nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top \frac{d\theta_{\epsilon,z}}{d\epsilon}\bigg|_{\epsilon=0} \\
&= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z,\hat{\theta})
\end{aligned}$$

  - Helpful images implies negative influence on the loss function

- **Results**
  - Understanding model behavior (discriminate fish vs. dog)
    - Helpful images implies negative influence on loss function
    - For Inception network, most helpful image was actually a dog

Test image

Inception RBF SVM

Helpful images

Helpful train dog image (Inception)

Conclusion

- **Interpretability method** is about **giving explanation** to human
  - Form of explanation is various

- In this lecture, we covered some of interpretability methods
  - Visual explanation (saliency map / class activation map)
  - Network dissection
  - Influence function

- There are still many research directions
  - Lots of interpretability methods not covered in this slide

References

[Zeiler et al., 2014] Visualizing and Understanding Convolutional Networks. ECCV 2014.
https://arxiv.org/abs/1610.02136

[Zintgraf et al., 2017] Visualizing Deep Neural Network Decisions: Prediction Difference Analysis. ICLR 2017.
https://arxiv.org/abs/1702.04595

[Ribeiro et al., 2016] "Why Should I Trust You?": Explaining the Predictions of Any Classifier, KDD 2016.
https://arxiv.org/abs/1602.04938

[Simonyan et al., 2014] Deep Inside Convolutional Networks: Visualising Image Classificiation ... ICLR Workshop 2014.
https://arxiv.org/abs/1312.6034

[Sundararajan et al., 2017] Axiomatic Attribution for Deep Networks. ICML 2017.
https://arxiv.org/abs/1703.01365

[Smilkov et al., 2017] SmoothGrad: removing noise by adding noise.
https://arxiv.org/abs/1706.03825

[Selvararaju et al., 2017] Grad-CAM: Visual Explanations from Deep Networks via Gradient-based ... ICCV 2017.
https://arxiv.org/abs/1610.02391

[Zhou et al., 2015] Learning Deep Features for Discriminative Localization. ICCV 2015.
https://arxiv.org/abs/1512.04150

[Adebayo et al., 2017] Sanity Checks for Saliency Maps. NIPS 2018.
https://arxiv.org/abs/1810.03292

[Bau et al., 2017] Network Dissection: Quantifying Interpretability of Deep Visual Representations. CVPR 2017.
https://arxiv.org/abs/1704.05796

[Koh et al., 2017] Understanding Black-box Predictions via Influence Functions. ICML 2017.
https://arxiv.org/abs/1703.04730