Interpretability

"Interpretability is the degree to which a human can understand the cause of a decision." Tim Miller, 2017

Diane Lingrand

POLYTECH NICE SOPHIA

UNIVERSITÉ CÔTE D'AZUR

SI4

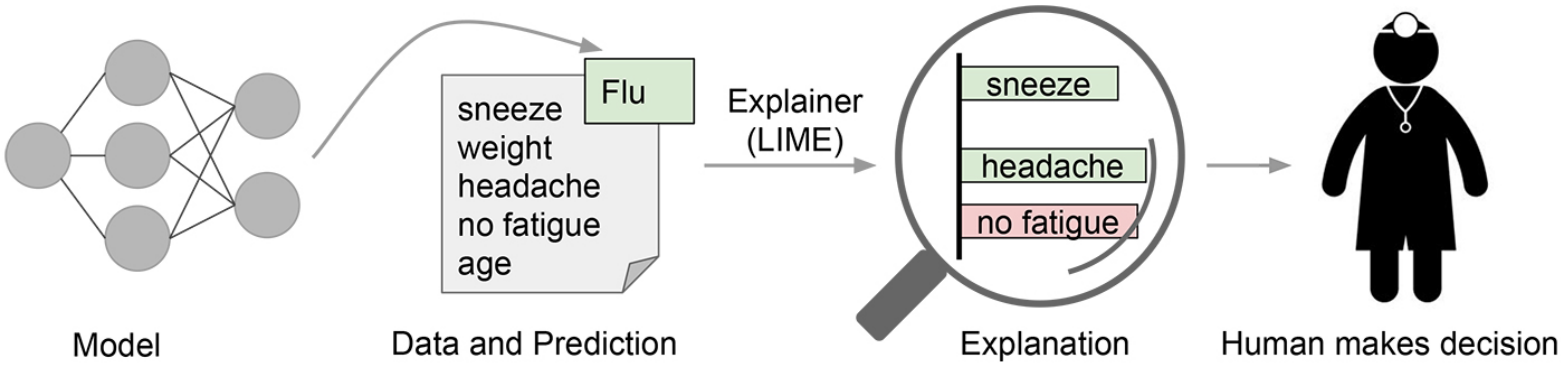
2024 - 2025

Outline

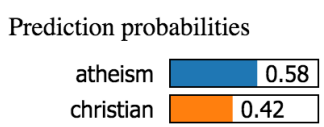
Outline

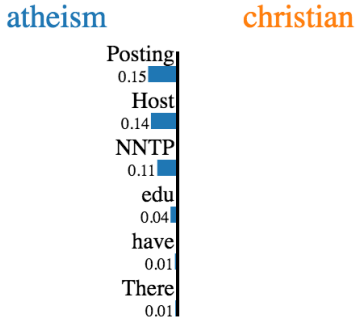
Motivation

- Explaining the decision for decision making



- Trust in the machine learning model
  - Example using the 20newsgroups dataset and highlighting 2 classes : Christianity and Atheism. RF with 500 trees, test acc. 92.4%





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.

- *"Why Should I Trust You ?" Explaining the Predictions of Any Classifier* by Ribeiro, Singh and Guestrin, 2016 `https://arxiv.org/pdf/1602.04938v1.pdf`

Context

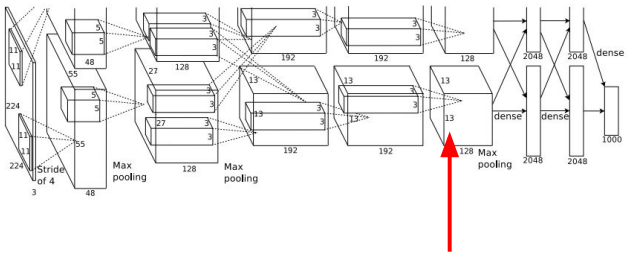
- Agnostic explanation or black-box methods
  - LIME, Shapley based methods
- Explanation of CNN
  - We know the architecture and the weights

Outline

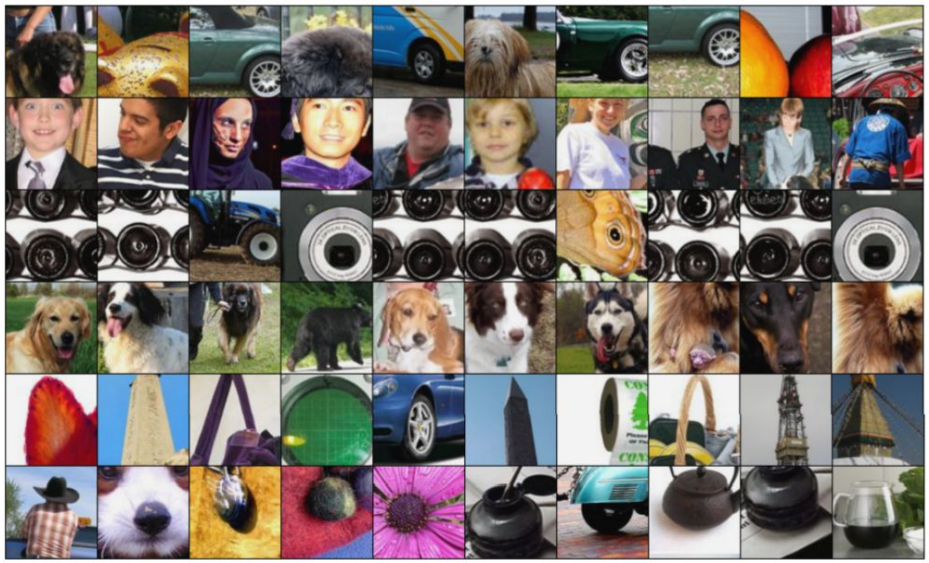
Methods

- Maximally activating patches
- Occlusion
- CAM
- Grad-CAM
- Guided Grad-CAM

Maximally activating patches



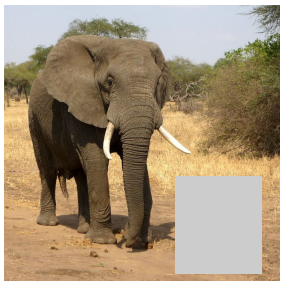
- pick a layer and a channel
  - e.g. conv5 (128x13x13) and channel 17
- run many images through the network
  - record values of chosen channel
- visualize image patches corresponding to max. activations

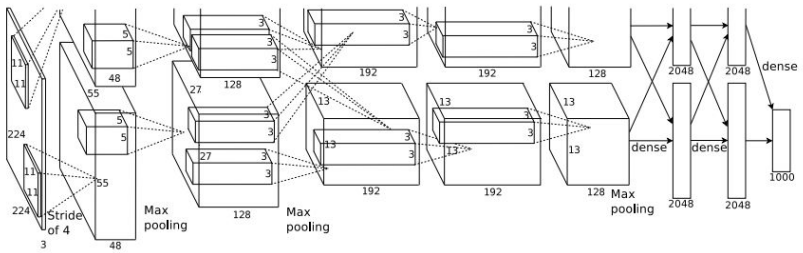


Springenberg et al, "Striving for Simplicity : The All Convolutional Net", ICLR

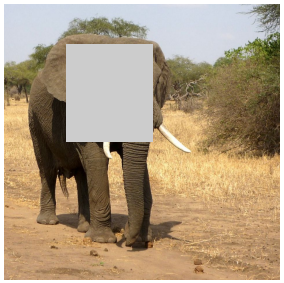
Which are the "important" pixels ?

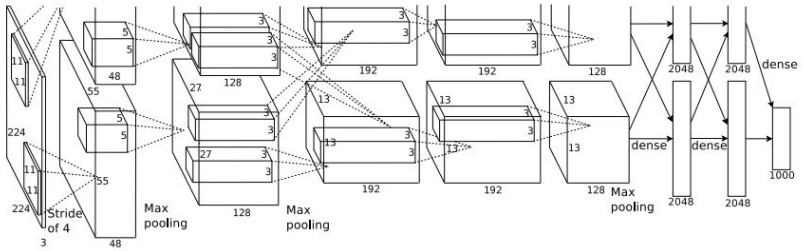
- apply a mask to the image before feed forward to CNN
- check how much predicted probabilities change





P(elephant) = 0.95



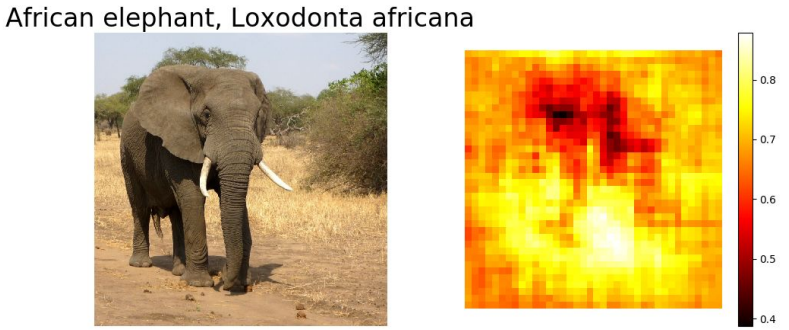


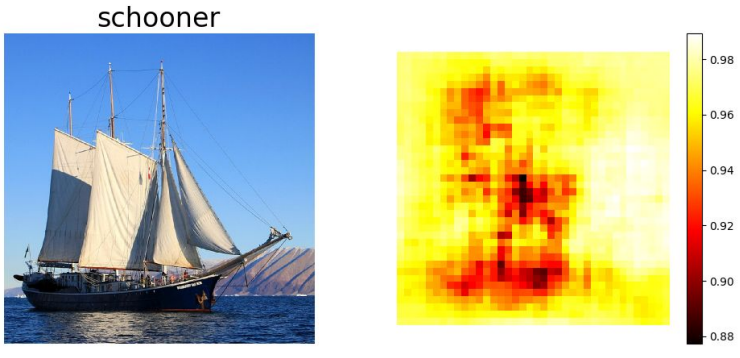
P(elephant) = 0.75

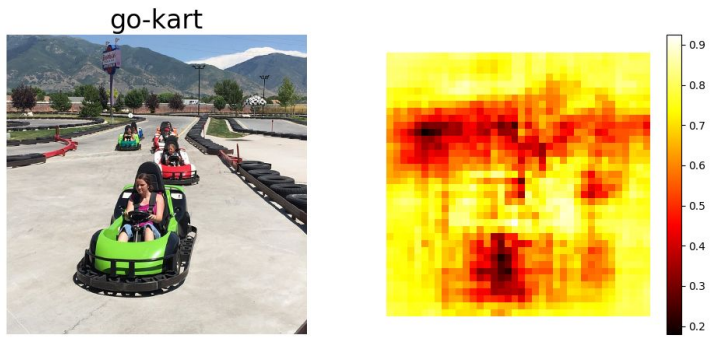
Zeiler and Fergus, "Visualizing and Understanding Convolutional Networks" ECCV 2014

`https://arxiv.org/pdf/1311.2901.pdf`

Occlusion map in the paper



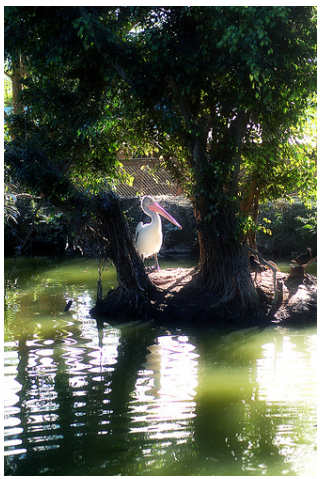


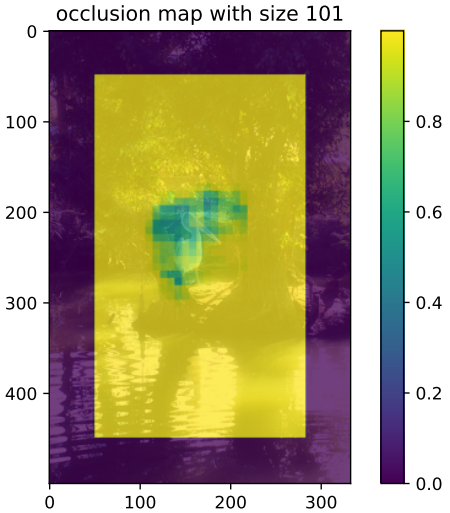


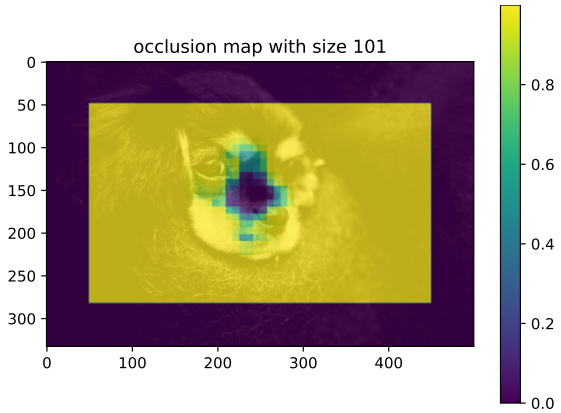
Zeiler and Fergus, "Visualizing and Understanding Convolutional Networks" ECCV 2014

Occlusion map in the lab

- choose a dimension for the hidden part of the image (e.g. 101)
- note the predicted class for the image `predClass`
- for each pixel of the original image (stride can be greater than 1) :
  - center the square of dimension sizexsize and color pixels in medium grey
    - be careful that medium gray after pre-processing is value 0
  - feed forward this modified image in the network and predict the probability of the class `predClass`
- display the occlusion map with the original image using transparency



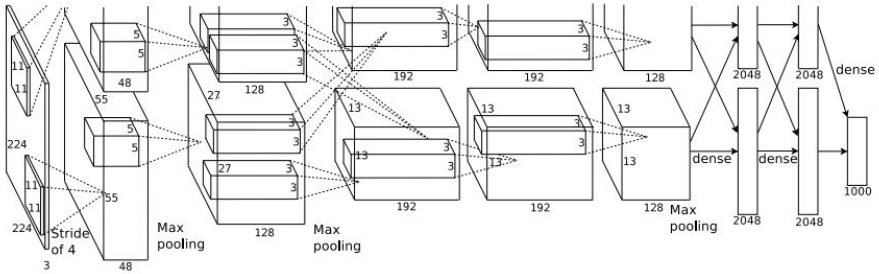


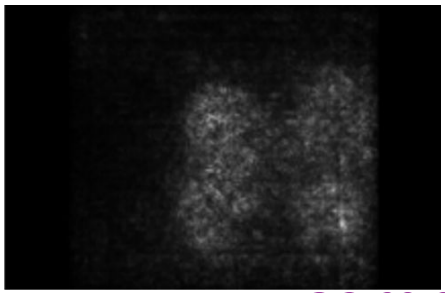


Saliency by back-propagation

Forward pass:
Compute probabilities

Dog



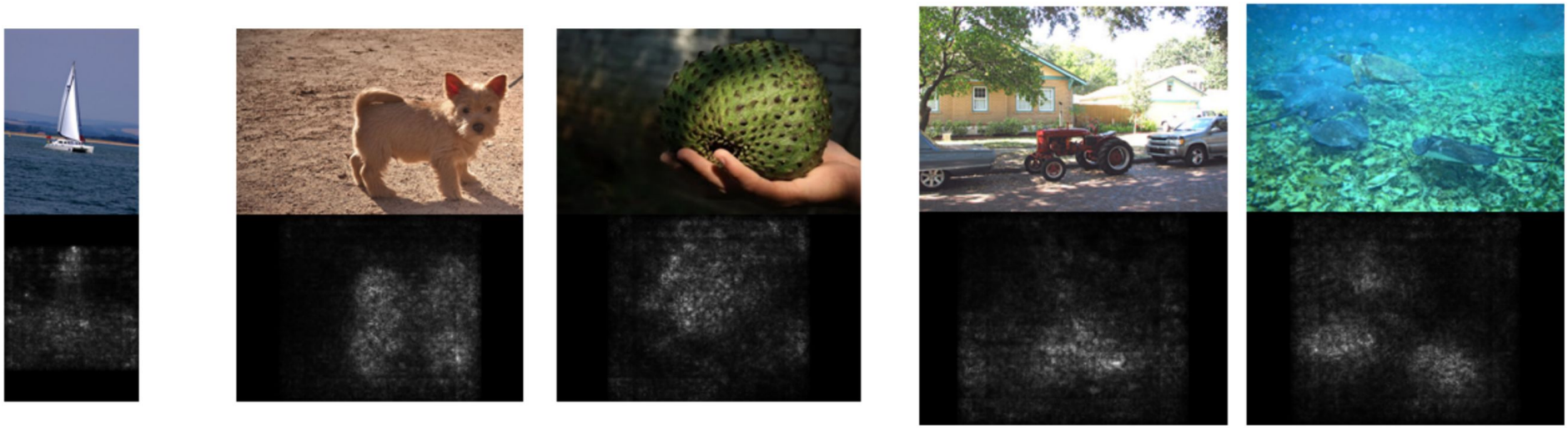


Dog

Back-propagation :
compute gradient of class score w.r.t. pixels
take abs. value and max over RGB channels

Simonyan, Vedaldi, and Zisserman, "Deep Inside Convolutional Networks : Visualising Image Classification Models and Saliency Maps", ICLR Workshop 2014.

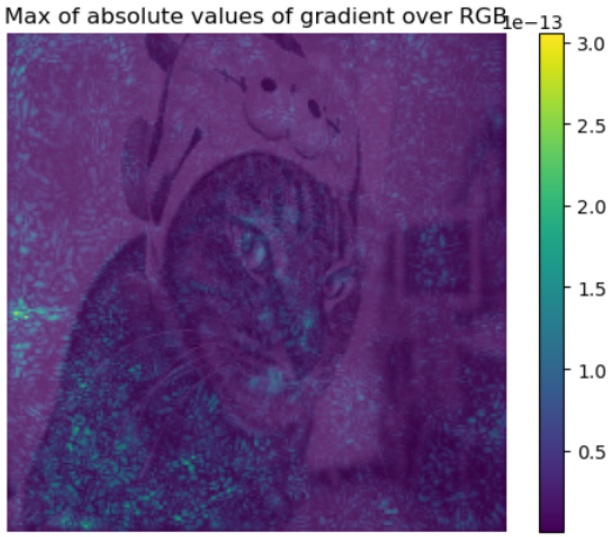
Saliency : examples from the paper



Simonyan, Vedaldi, and Zisserman, "Deep Inside Convolutional Networks : Visualising Image Classification Models and Saliency Maps", ICLR Workshop 2014.

Saliency in the lab

- Compute the gradient of class score w.r.t. pixels.
  - The variables are the pixels :
    - `requires_grad` is `True` for the pixels
  - forward pass : compute the probability of `predClass` for a image
  - backward pass : computation of the gradient of `predClass` with respect to pixels
- Take absolute value and max over RGB channels.



CAM : Class Activation Maps (2015)

Discriminative region used by CNN to identify the class.

Learning Deep Features for Discriminative Localization

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba
Computer Science and Artificial Intelligence Laboratory, MIT

https://arxiv.org/pdf/1512.04150.pdf

inside CAM

- CNN network for image classification without FC layer(e.g. Googlenet Inception v1)
- use a GAP (Global Average Pooling) layer before the softmax layer

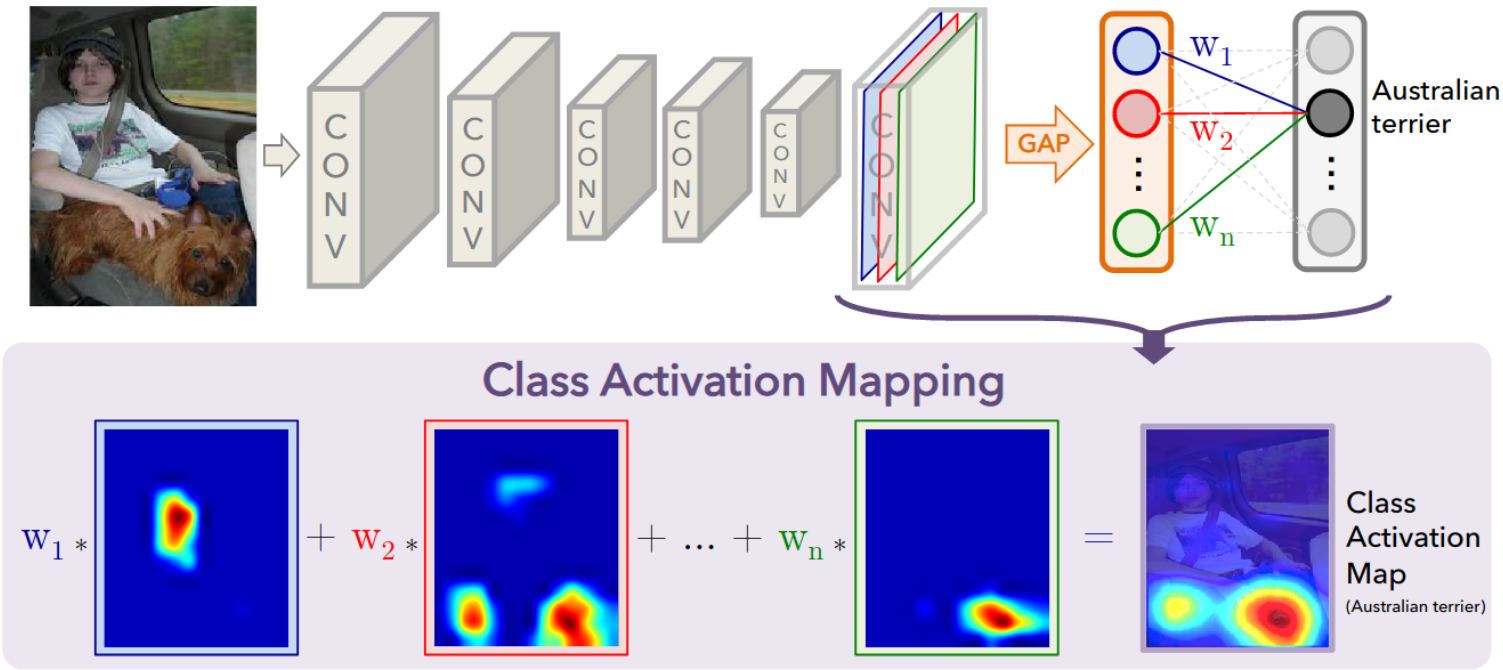


illustration from `https://arxiv.org/pdf/1512.04150.pdf`

- upsampling

Implementation of CAM

- needs a learned CNN
  - remove the top (fc layers)
  - add a global average pooling (GAP)
  - add a dense layer for classification with a softmax activation
  - learn this last layer
- Identify :
  - the detected class

```
predClasses = model.predict(xTest)
index = 800 # one of the images
cl = np.argmax(predClasses[index])
```

  - the weights w_i, $1 \leq i \leq n$

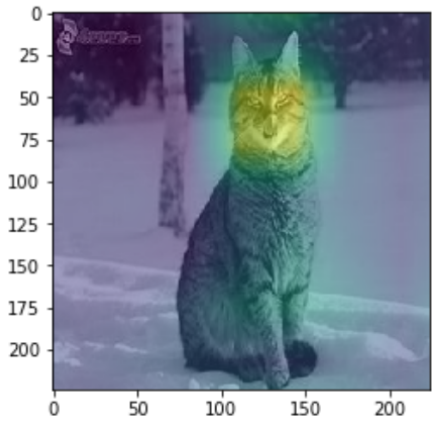
```
layer = model.get_layer('dense')
we = layer.get_weights()[0]
w = we[:,cl]
```

  - the activation maps M_i, $1 \leq i \leq n$

```
allFeatures = VGGmodel.predict(xTest)
maps = allFeatures[index]
```

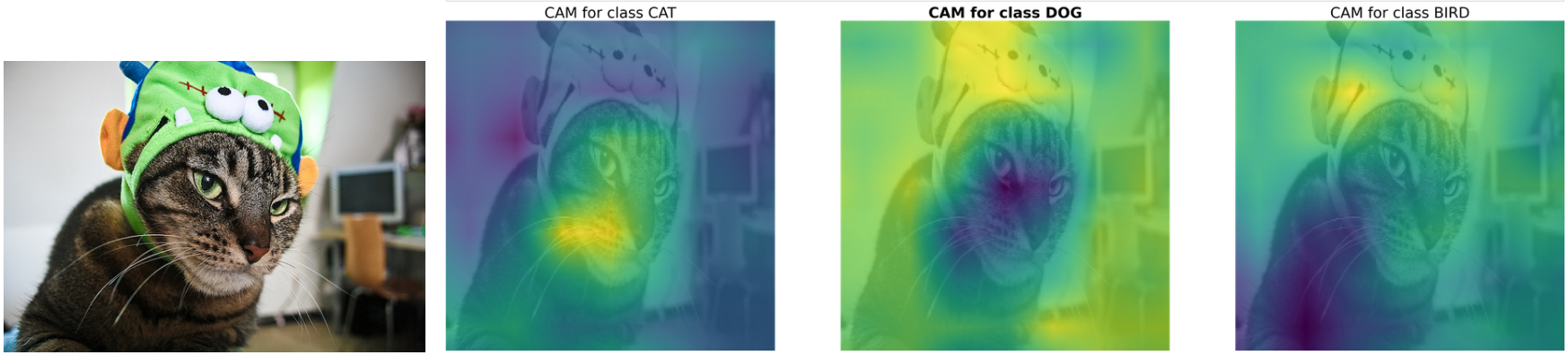
- Compute :
  - the weighted sum : $\sum_{i=1}^{n} w_i M_i$: `cam = np.inner(maps,w)`
  - upscaling

```
from skimage.transform import resize
plt.imshow(resize(camimg, (224,224)))
plt.imshow(image[:,:,1],cmap=plt.cm.gray, alpha=0.5)
plt.show()
```

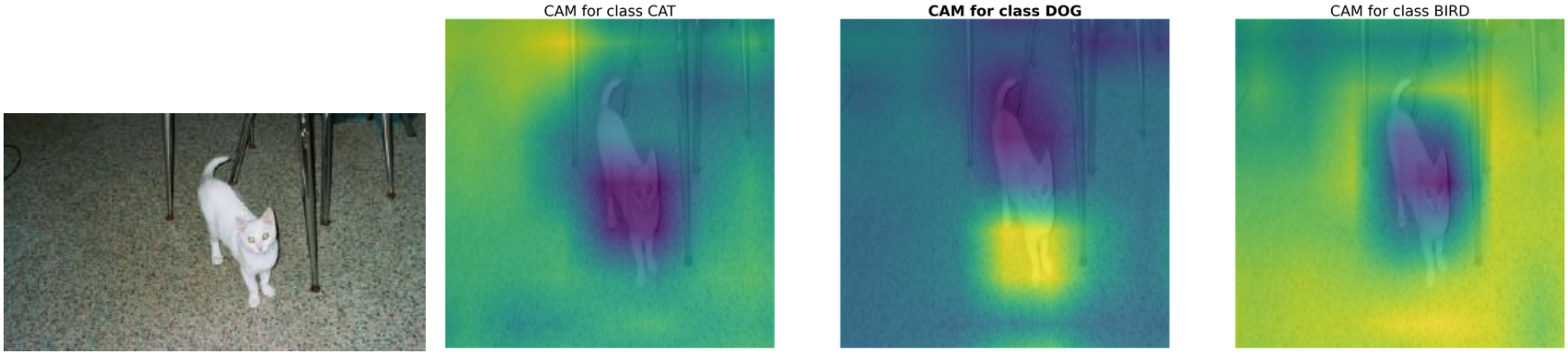


Some CAM displays

- image 2009_003867.jpg predicted as cat ($[1, 2.10^{-19}, 5.10^{-11}]$)



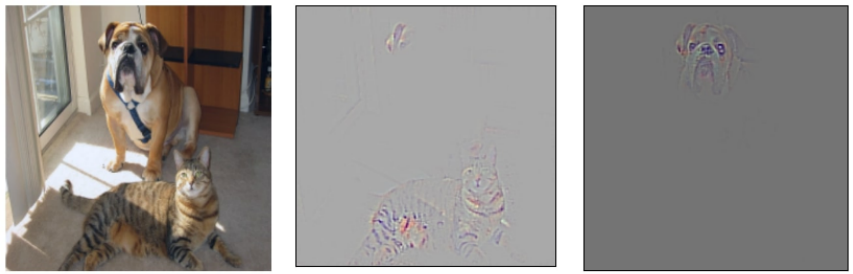
- image 2008_007589.jpg predicted as a dog ($[0.02, 0.97, 0.01]$)



Guided Grad-CAM

Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization

Ramprasaath R. Selvaraju · Michael Cogswell · Abhishek Das · Ramakrishna Vedantam · Devi Parikh · Dhruv Batra

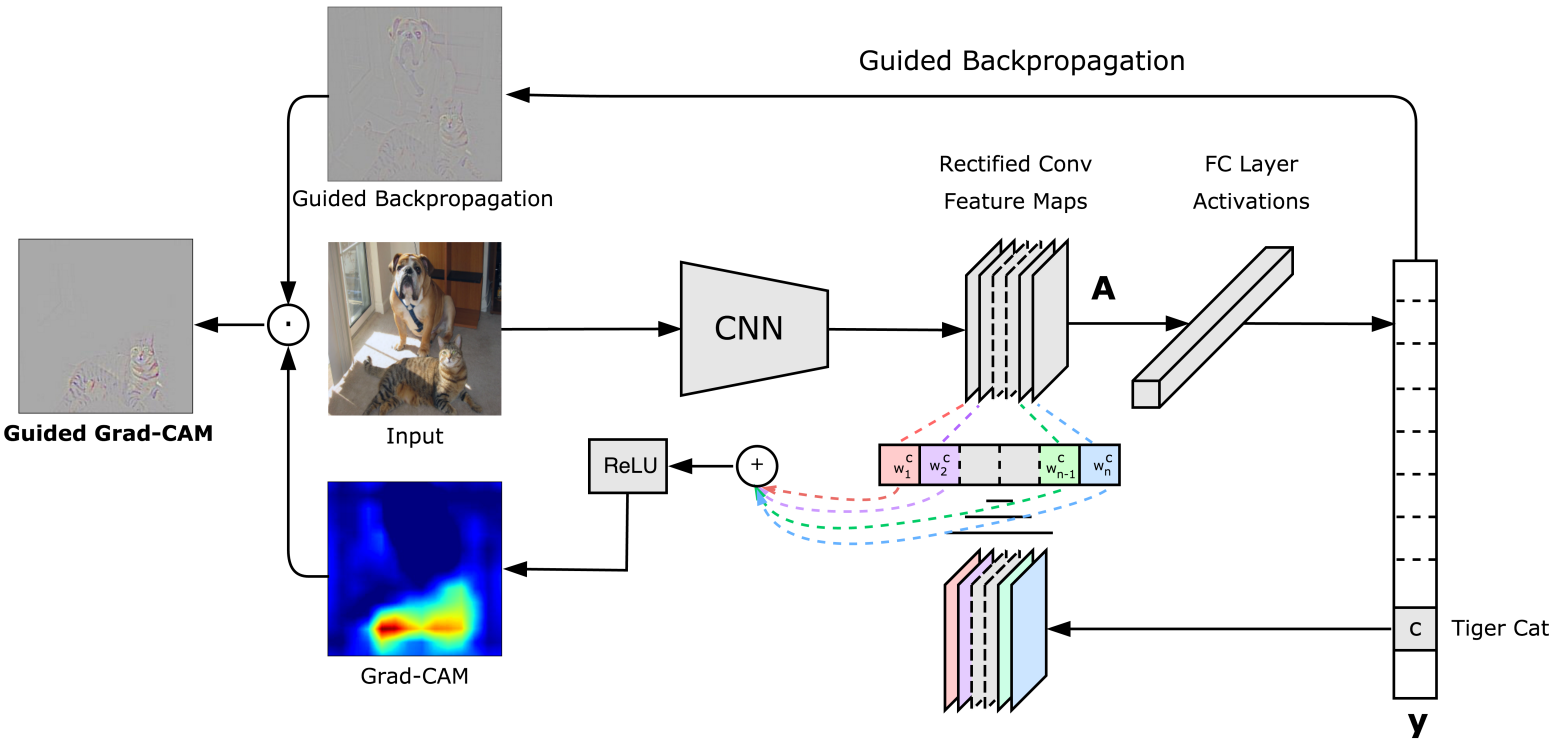


Original Image | Guided Grad-CAM 'Cat' | Guided Grad-CAM 'Dog'

`https://arxiv.org/pdf/1610.02391.pdf`

- Improvement of CAM
  - CAM needs to add GAP between feature map and softmax layer : usually performs worse that other CNNs
- Guided Grad-CAM
  - don't modify the CNN
  - may be use with any CNN (including Image Captioning and VQA)
  - both **class discriminative** (localisation of the target class pixels) and **high-resolution** (e.g. stripes on the tiger cat)

Guided Grad-CAM network



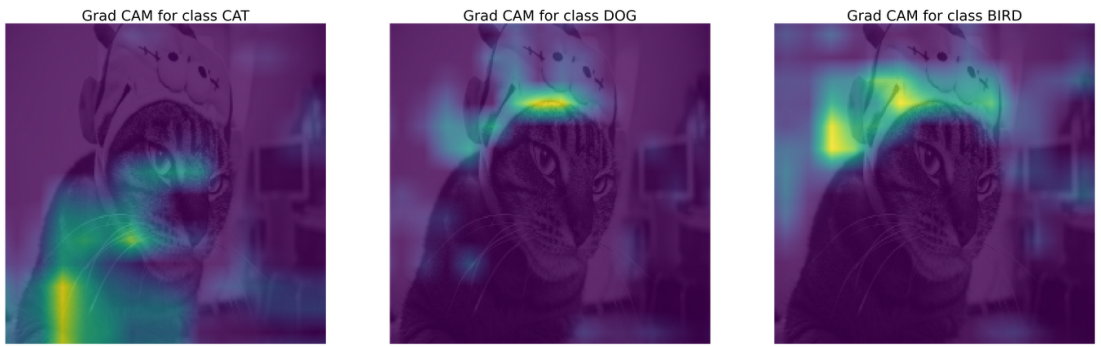
- Guided Backpropagation
- Grad CAM

Grad CAM

- Grad-CAM is the ReLU activation of the weighted sum of feature maps :

$$L^c = \text{ReLU}(\sum_k \alpha_c^k A^k) \text{ with } \alpha_c^k = \overbrace{\frac{1}{N}\sum_{u,v}}^{GAP} \overbrace{\frac{\partial y^c}{\partial A_{uv}^k}}^{\text{gradient via backprop}}$$

  - activation maps are computed as prediction of the description part of the network
  - the gradient of the predicted class with respect to the features maps is computed using :
    - a model with activation maps as input and predictions as input corresponding to the end of the original network
    - feature maps with `requires_grad` set to `True`
    - GAP on the gradients to obtain the alphas parameters
    - ReLU on the weighted sum of feature map with alphas

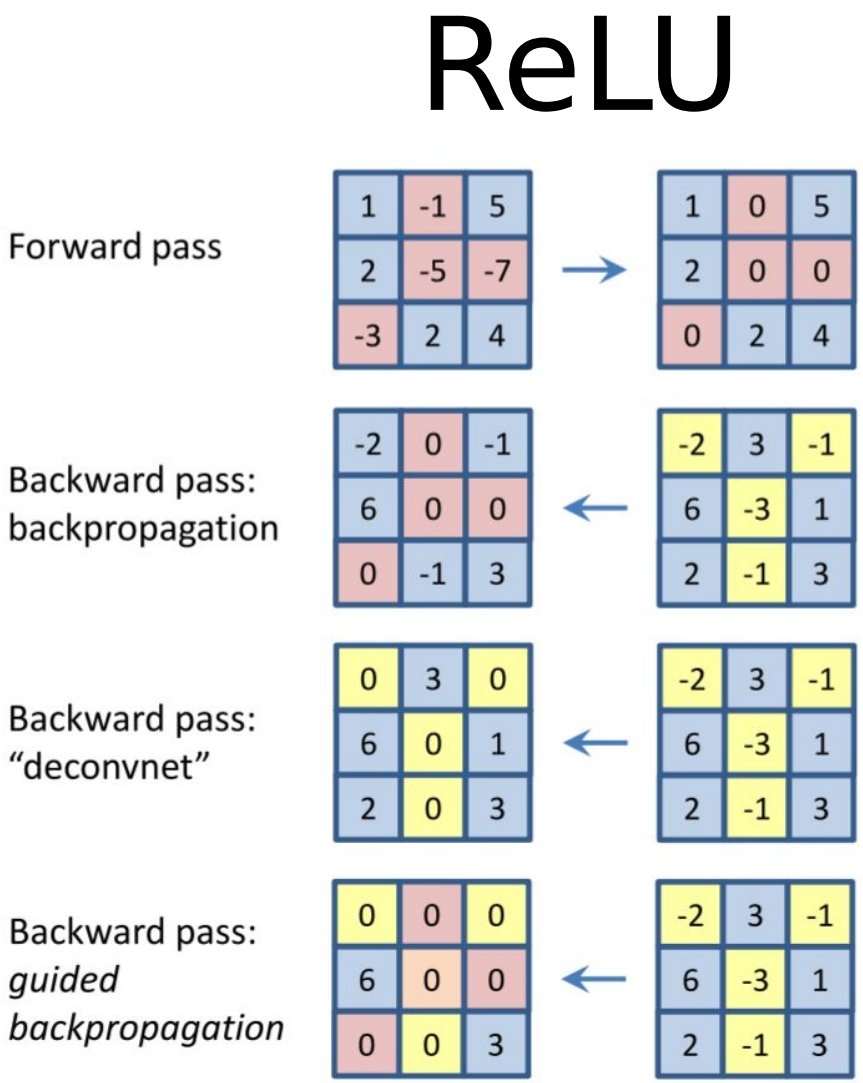


Grad-CAM in the lab

```
VGGmodel = VGG16(weights='imagenet', include_top=True)
modelA = Model(inputs=VGGmodel.input,
               outputs=VGGmodel.get_layer('block5_conv3').output)
modelP = Model(inputs=VGGmodel.get_layer('block5_pool').input,
               outputs=VGGmodel.output)
featureMaps = modelA(image.unsqueeze(0)).requires_grad_(True)
probas = modelP(featureMaps)
probas[0,predClass].backward()
grads2 = featureMaps.grad
# result with shape torch.Size([1, 14, 14, 512])
alphas = torch.mean(grads2.squeeze(), dim=(0,1))
#...
```

Guided Back-propagation

Images come out nicer if you only backpropagate positive gradients through each ReLU.



Guided Back-propagation in the lab

- Except the ReLU, it is similar to the saliency map by backpropagation
- Every ReLU activation in the network should be modified for the backpropagation part
  - easy for pytorch models

```
def relu_hook_function(module, grad_in, grad_out):
  if isinstance(module, torch.nn.ReLU):
     return (torch.clamp(grad_in[0], min=0.),)

for block in network.modules():
  if isinstance(block,nn.Sequential):
     for layer in block.modules():
        if isinstance(layer,nn.ReLU):
            layer.register_backward_hook(relu_hook_function)
```

  - easy for tensorflow models
  - for the lab of today
    - simplification with only considering the positive gradients (not the real guided back-prop)
    - exact back-propagation by building the same network in pytorch and copying the weights

Guided Grad-CAM example (from the paper)

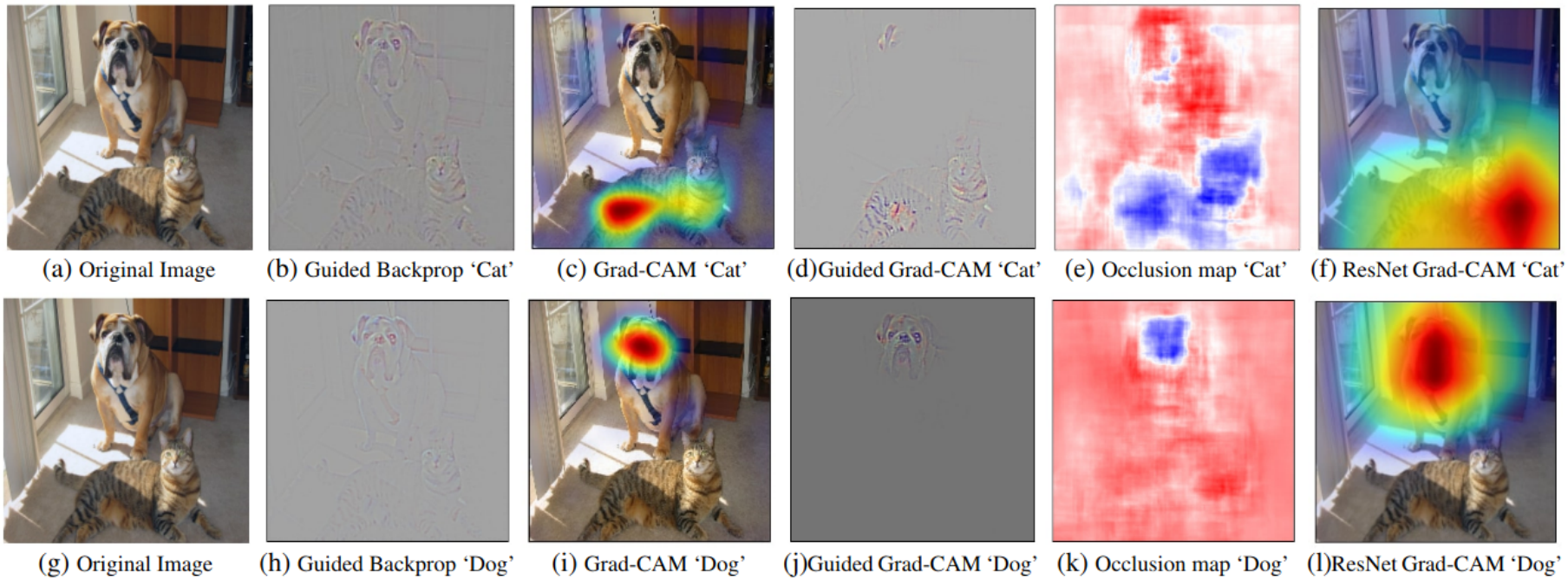
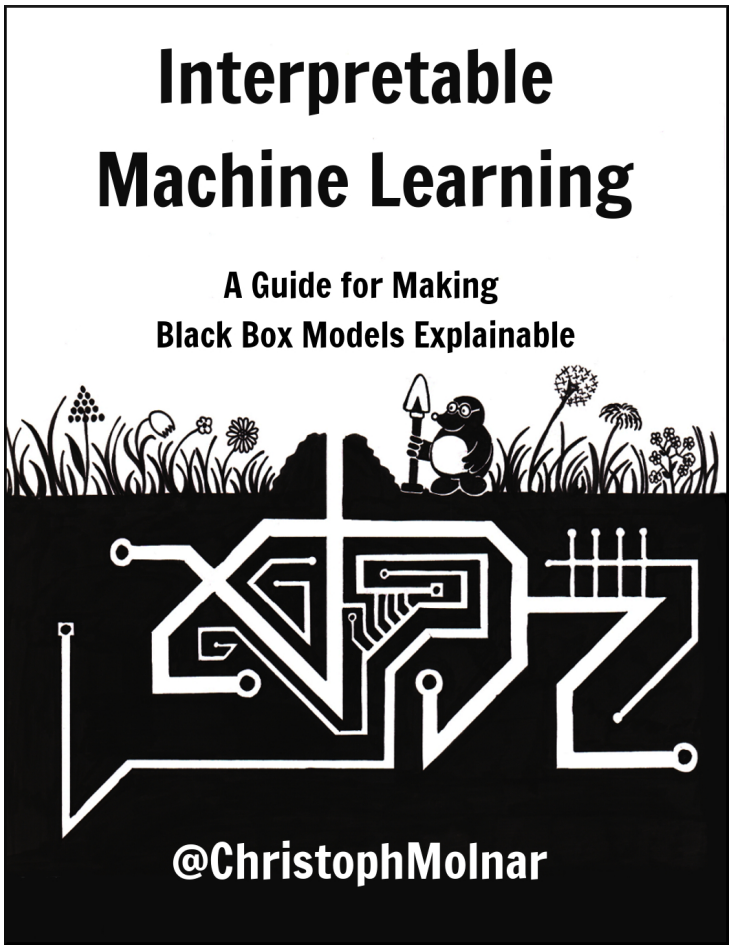


Fig. 1: (a) Original image with a cat and a dog. (b-f) Support for the cat category according to various visualizations for VGG-16 and ResNet. (b) Guided Backpropagation [53]: highlights all contributing features. (c, f) Grad-CAM (Ours): localizes class-discriminative regions, (d) Combining (b) and (c) gives Guided Grad-CAM, which gives high-resolution class-discriminative visualizations. Interestingly, the localizations achieved by our Grad-CAM technique, (c) are very similar to results from occlusion sensitivity (e), while being orders of magnitude cheaper to compute. (f, l) are Grad-CAM visualizations for ResNet-18 layer. Note that in (c, f, i, l), red regions corresponds to high score for class, while in (e, k), blue corresponds to evidence for the class. Figure best viewed in color.

Reading



https://christophm.github.io/interpretable-ml-book/