

HUMAN CENTERED MACHINE LEARNING

Lecture 6: Neural Network Interpretability

Lecturer: dr. Heysem Kaya



Outline

- Learned Features
 - Feature Visualization
 - Network Dissection
- Feature Attribution Methods
 - Vanilla Gradient
 - DeconvNet
 - Grad-CAM
 - Guided Grad-CAM
 - SmoothGrad

Deep Neural Networks: Data-Hungry Dragons

Daenerys Targaryen: the mother of dragons¹



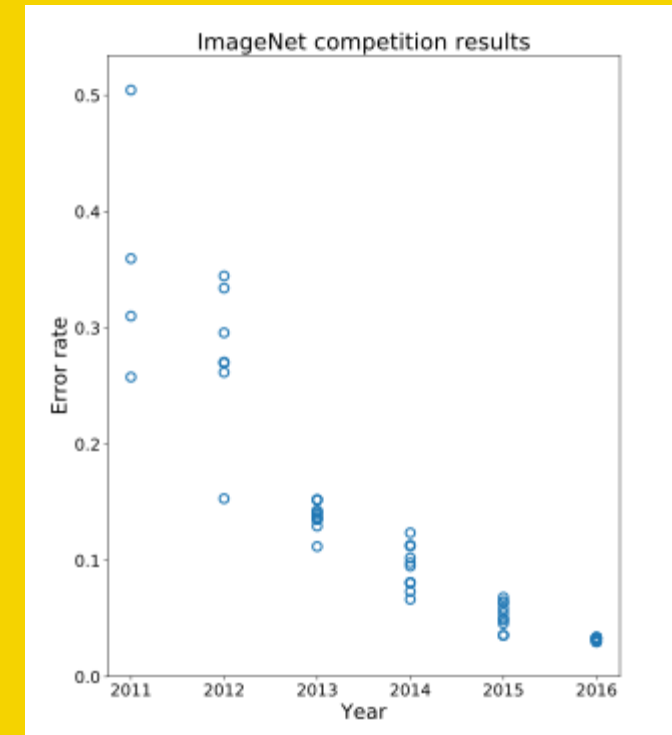
Fei Fei Li*: Mother of DNNs



¹Public domain image, *Image from https://en.wikipedia.org/wiki/Fei-Fei_Li
Suggested: [Fei Fei Li's TedX talk on YouTube](#)

ImageNet

- The project / dataset that created today's ML dragons!
Particularly, thanks to ImageNet Large Scale Visual Recognition Challenge.



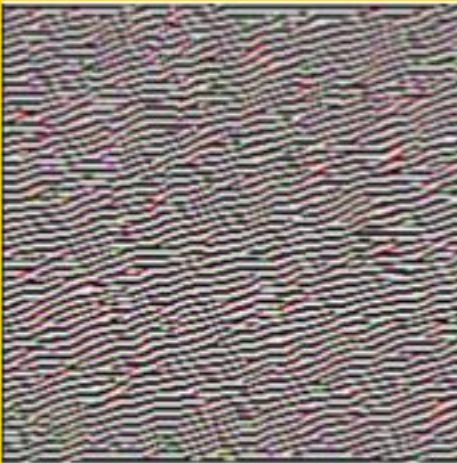
From <https://en.wikipedia.org/wiki/ImageNet>

~1.4 million images, 1000 classes

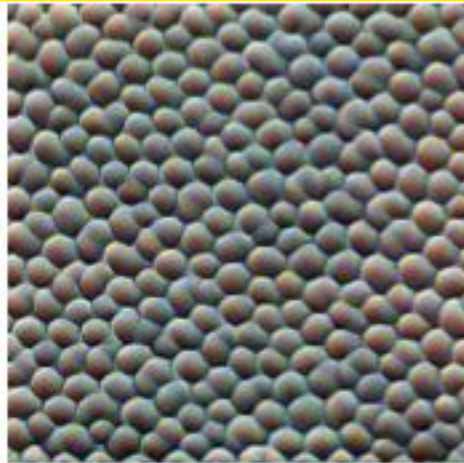
So how do those dragons 'work'?

- One of the main motivations for explainability in the last decade
- Now we know that CNNs learn higher and higher level abstractions

Edges



Textures



Patterns



Parts



Objects

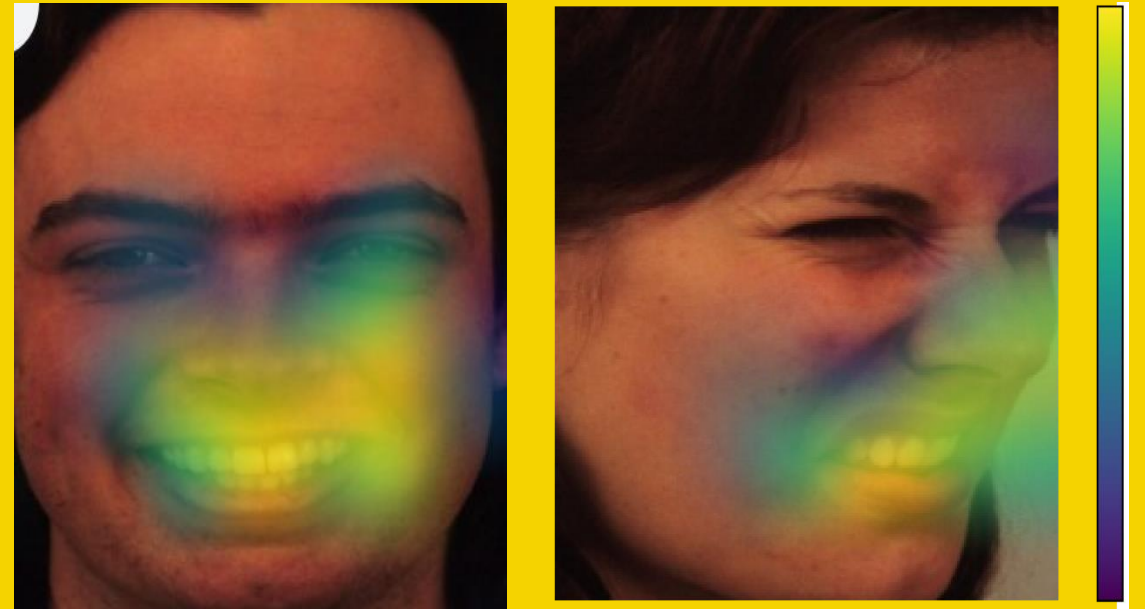


Image from <https://distill.pub/2017/feature-visualization/appendix/>

Interpreting Deep Neural Networks



Feature visualization answers questions about what a network - or parts of a network - are looking for by generating examples.

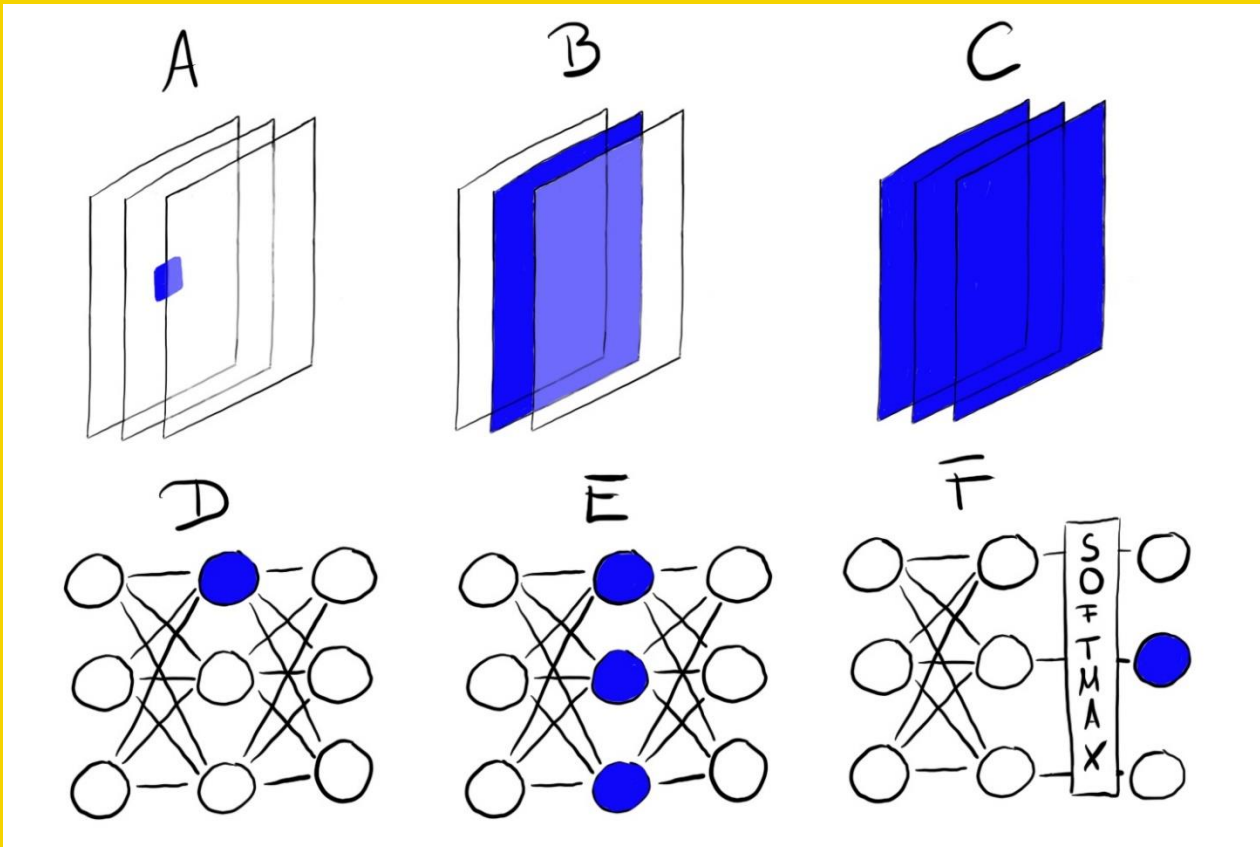


Pixel attribution studies what part of an example is responsible for the network activating a particular way.

Image from <https://distill.pub/2017/feature-visualization/>

Feature Visualization

Units / optimization objectives



- Feature visualization for a unit of a neural network is done by *finding the input* that maximizes the activation of that unit.

A) Convolution neuron,
B) Convolution channel,
C) Convolution layer,
D) Neuron,
E) Hidden layer,
F) Class probability neuron

Feature Visualization by Optimization

Database Search



Optimization



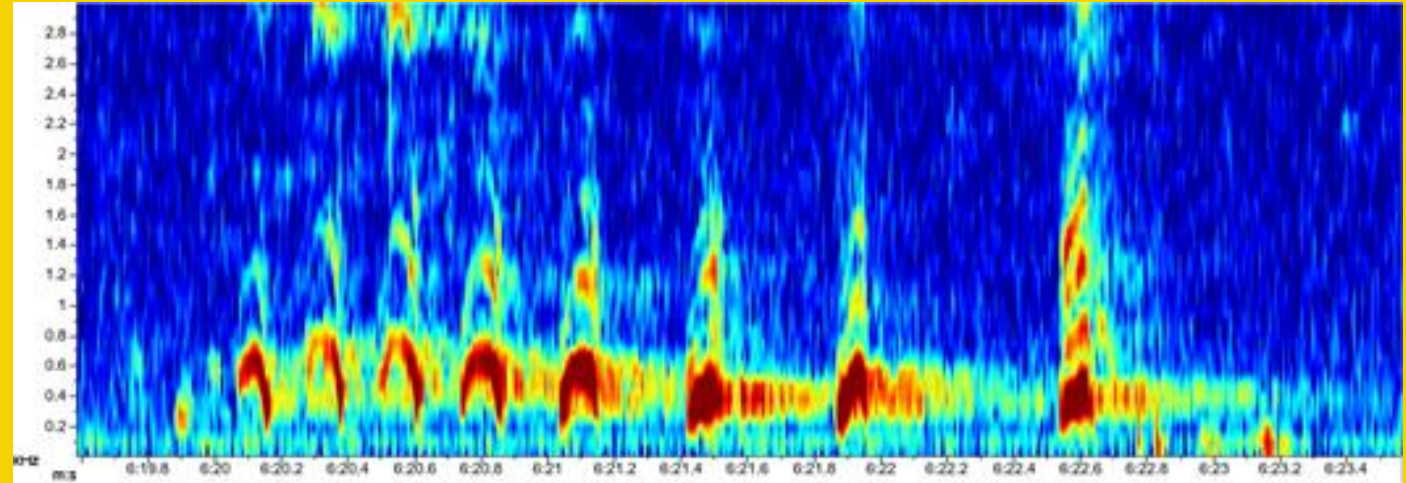
Iterative Optimization for Unit Visualization



1. Initially a random image is fed to the trained network.
2. Activation of the unit of interest is calculated.
3. Difference between the max. activation and the current activation is backpropagated to generate gradient image.
4. The gradient image g_{img} , is combined with the current image c_{img} with a weight w : $n_{img} = c_{img} + w * g_{img}$.
5. Steps 1-4 are repeated with the new image n_{img} until convergence.

Feature Visualization in other modalities

- Audio: Generate spectrogram to use as an image
- Tabular: Investigate which feature combinations maximally activate a single unit
- Text: RNN units can be analyzed for respective text patterns that activate them



Red Capped Mangabey "Whoop Gobble" call spectrogram from [Z]

Karpathy et al [K] detected a neuron

- activating after observing '(' and deactivating after ')'
- in an RNN trained to predict the next character.

[Z] Zwerts et al., Introducing a Central African Primate Vocalisation Dataset for Automated Species Classification, arXiv 2021

[K] Karpathy, Andrej, Justin Johnson, and Li Fei-Fei. "Visualizing and understanding recurrent networks." arXiv 2015

Remarks on Feature Visualization



Visualization helps understanding

what images activate a unit

what a unit is looking / responsible for in an image



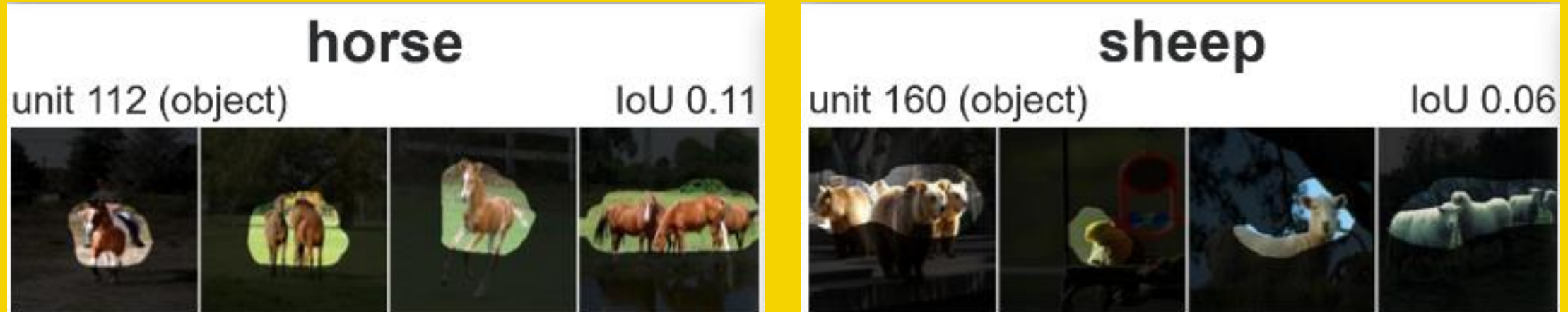
DB search: real factor may not be revealed due to correlating parts



We cannot quantify how well a unit detects a real-world concept

Network Dissection

- Network Dissection [B] quantifies the interpretability of a unit of a convolutional neural network.
- It links highly activated areas of CNN channels with human concepts (objects, parts, textures, colors, ...).



Images from http://netdissect.csail.mit.edu/dissect/vgg16_imagenet/

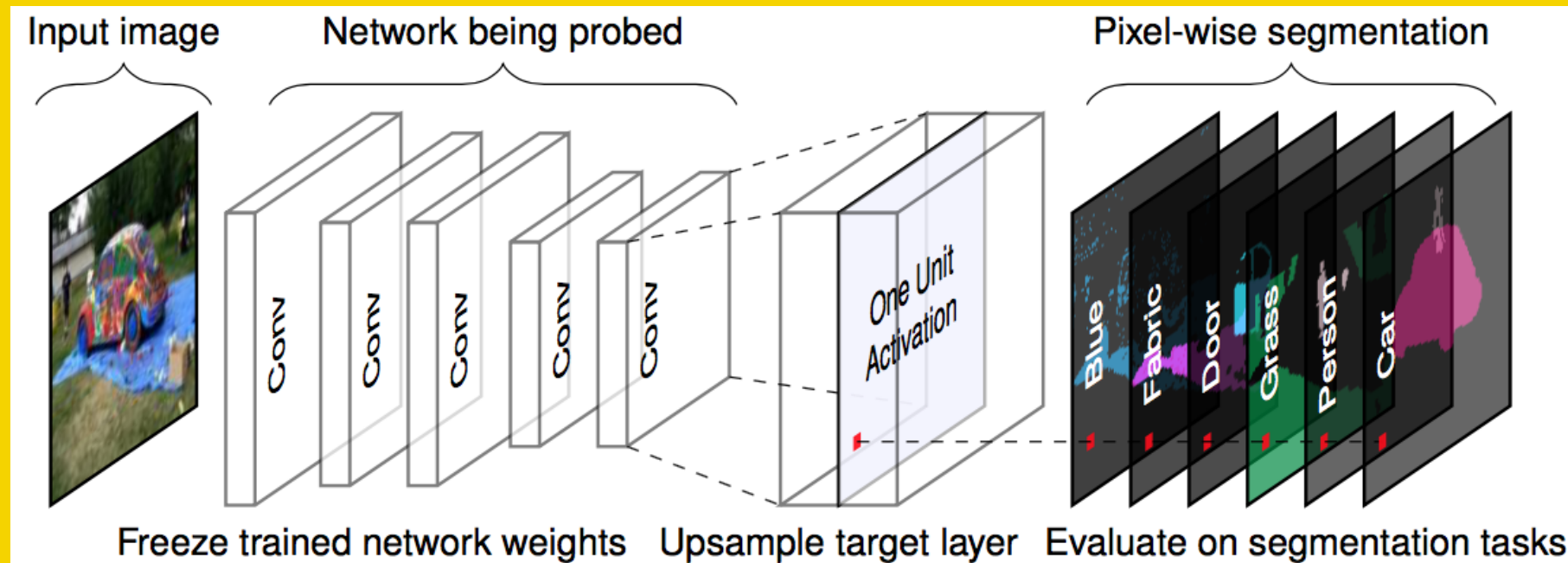
[B] Bau et al. "Network dissection: Quantifying interpretability of deep visual representations." CVPR 2017

Network Dissection

- An open challenge in DNN interpretation is disentanglement [R].
- Do (convolutional) neural networks learn disentangled features?
- Disentangled features mean that individual network units detect specific real-world concepts.
- Disentangled features imply that the network is highly interpretable.

Network Dissection Algorithm

1. Get images with human-labeled visual concepts (stripes, cats etc.).
2. Measure the CNN channel activations for these images.
3. Quantify the alignment of activations and labeled concepts.



Network Dissection Step 1: Broden Dataset

- Network Dissection requires pixel-wise labeled images with concepts of different abstraction levels (from colors to street scenes).
- Bau et. Al [B] combined a couple of datasets with pixel-wise concepts.



60,000 images with over 1,000 visual concepts in different abstraction levels:

- 468 scenes,
- 585 objects,
- 234 parts,
- 32 materials,
- 47 textures and 11 colors.

Network Dissection Step 2: NN activations

- For each convolutional channel k :
 - For each image x in the Broden dataset:
 - Forward propagate image x to the target layer containing channel k .
 - Extract the pixel activations of convolutional channel k : $A_k(x)$
 - Calculate distribution of pixel activations α_k over all images.
 - Determine the top 0.005-quantile level* T_k of activations α_k .
 - For each image x in the Broden dataset:
 - $S_k(x) = \text{upscale}(A_k(x), \text{resolution}(x))$ % upscale activation map to match original
 - $M_k(x) = S_k(x) \geq T_k$ % threshold each pixel and get map mask
- % We will use the masks $M_k(x)$ in the next step

*This corresponds to 0.995 quantile, '%' denotes comment

Network Dissection Step 3: Activation- Concept Alignment

- For each convolutional channel k :
 - For each image x in the Broden dataset:
 - Get corresponding activation mask $M_k(x)$
 - For each concept map $L_C(x)$ associated with image x
 - Calculate Intersection over Union (IoU) score





$$IoU_{k,c} = \frac{\sum |M_k(x) \cap L_C(x)|}{\sum |M_k(x) \cup L_C(x)|}, \text{ where } |.| \text{ denotes set cardinality}$$

$$\text{Ex. } M_k(x) = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \text{ and } L_C(x) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$$

$$M_k(x) \cap L_C(x) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, M_k(x) \cup L_C(x) = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, IoU_{k,c} = 1/3$$

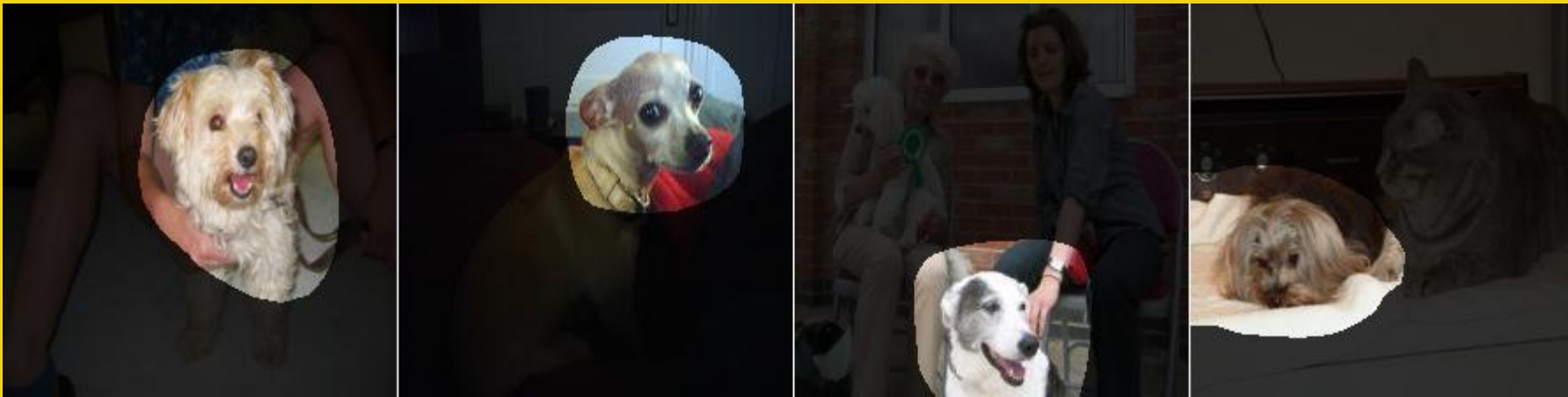
Intersection and Union: Illustration



-  = Human annotated ground truth
-  = Top activated area
-  = Area of Intersection
-  = Area of Union

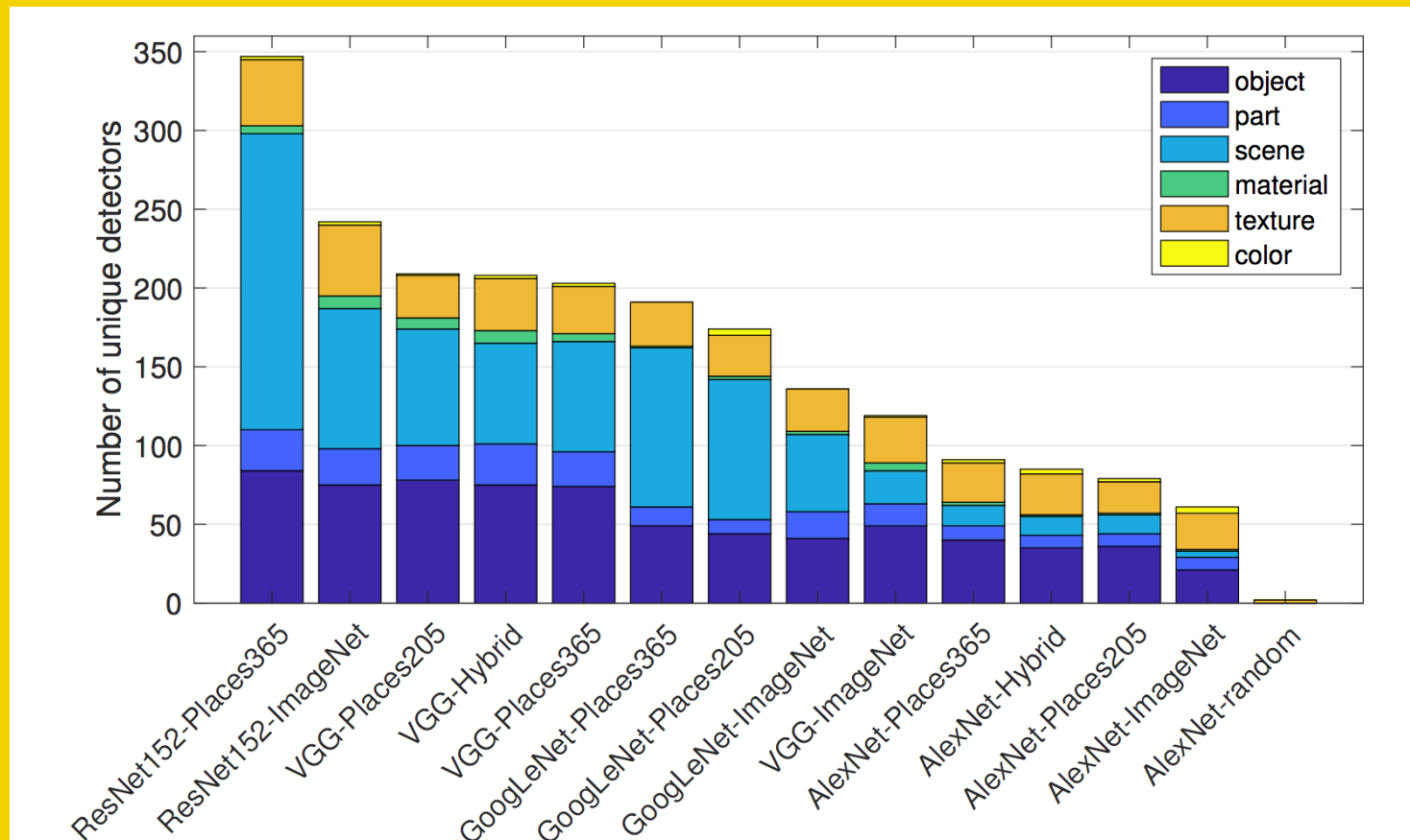
Network Dissection: Findings

- The networks detect lower-level concepts (colors, textures) at lower layers and higher-level concepts (parts, objects) at higher layers.
- Batch normalization reduces the number of unique concept detectors.
- Increasing the number of channels in a layer increases the number of interpretable units.
- Many units detect the same concept. (e.g. there are 95 dog channels in VGG trained on ImageNet when using $\text{IoU} \geq 0.04$ as detection cutoff.)



Network Dissection: Findings

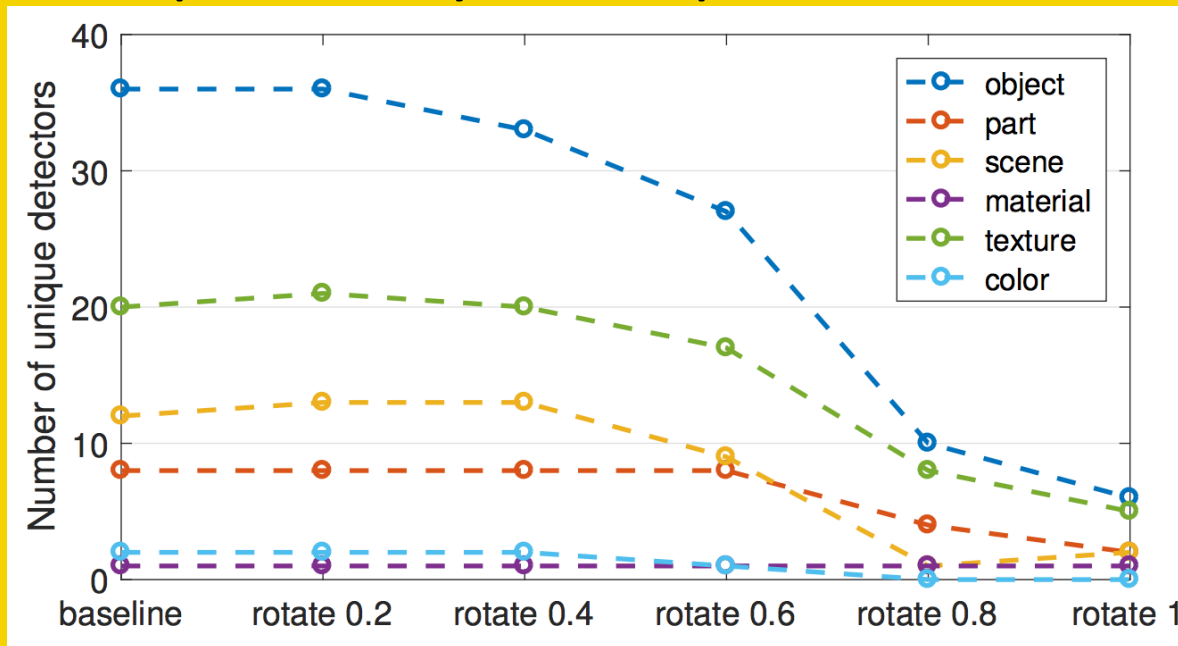
- Number of Unique concept detectors learned by a Network architecture.



ResNet is the network architecture with the largest number of unique detectors, followed by VGG, GoogLeNet and AlexNet

Network Dissection: Findings

- The number of unique concept detectors increases with the number of training iterations.
- In transfer learning, the concept of a channel can change.
- Interpretability is independent of discriminative power.



Random orthogonal rotation applied to channels reduce concept detectors (interpretability) without affecting predictions / performance.



Remarks on Feature Visualization

- Network dissection allows us to automatically link units to concepts, which is very convenient.
- Feature visualization is a great tool to communicate in a non-technical way how neural networks work.
- Feature visualization can be combined with feature attribution methods, which explain which pixels were important for the classification.
- Many unit visualizations are not interpretable at all.
- The feature visualizations can convey the illusion that we understand what the neural network is doing.
- There are too many units to look at, even if we visualize only the channel activations.

Feature Attribution (Saliency Maps)

Vanilla Gradient

DeconvNet

Grad-CAM

Guided Grad-CAM

SmoothGrad

Pixel Attribution

- Pixel attribution methods highlight the pixels that were relevant for a certain image prediction by a neural network.
- Consider a classification prediction vector of length C for $I \in \mathbb{R}^p$:
$$S(I) = [S_1(I), S_2(I), \dots, S_C(I)]$$
- The output of pixel attribution for each class prediction $S_c(I)$ is a pixel-wise relevance map $R^c(I) = [R_1^c(I), R_2^c(I), \dots, R_p^c(I)]$
- **Occlusion- or perturbation-based:** Methods like SHAP and LIME manipulate parts of the image to generate explanations (model-agnostic).
- **Gradient-based:** Many methods compute the gradient of the prediction (or classification score) with respect to the input features.

Vanilla Gradient

- We calculate the gradient of the class score for the class we are interested in with respect to the input pixels.
- The obtained map is of the size of the input and can be overlaid.
- Steps for calculating the saliency map:
 1. Perform a forward pass of the image of interest.
 2. Compute the gradient of class score of interest with respect to the input pixels. (Here we set all other classes to 0).
 3. Visualize the gradients. You can either show the absolute values or highlight negative and positive contributions separately.

Vanilla Gradient: a bit more technical

- Calculating the gradient of $S_C(I) \approx w^T I + b$ (first order Taylor exp.):

$$w = \frac{\partial S_C(I)}{\partial I} \Big|_{I_0}$$

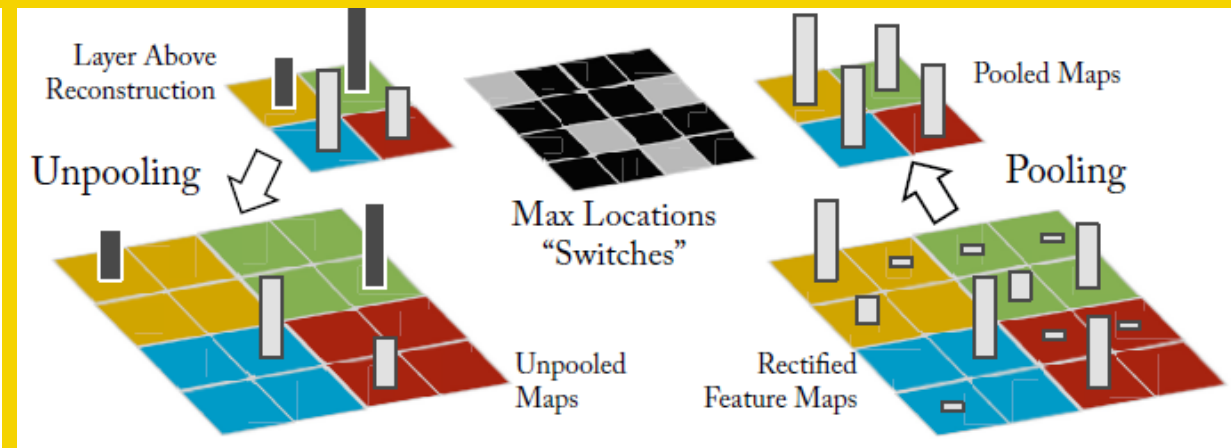
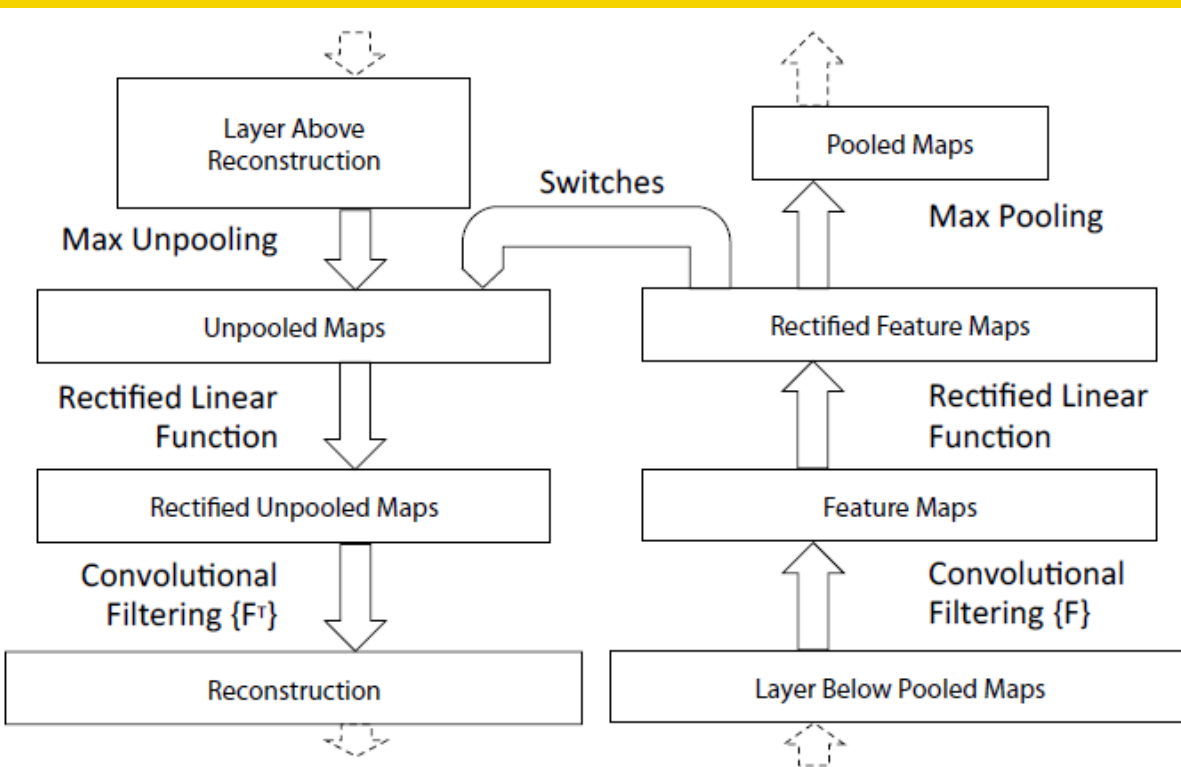
- The gradient of $S_C(I)$ is passed backwards layer wise.
- How to handle 'ReLU(x)= max(0, x)' that ignores negative values?
- Vanilla approach: $\frac{\partial f}{\partial X_n} = \frac{\partial f}{\partial X_{n+1}} \mathbf{1}(X_n > 0)$, where $\mathbf{1}$ is the indicator function

$$\text{If } X_n = \begin{bmatrix} 1 & -10 \\ 0 & 0.1 \end{bmatrix} \text{ and } \frac{\partial f}{\partial X_{n+1}} = \begin{bmatrix} 1 & -1 \\ 0.5 & 3 \end{bmatrix}, \text{ then } \frac{\partial f}{\partial X_n} = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}$$

- This approach is reported to cause *a saturation problem* [A]

DeconvNet

Reverse the NN: DeconvNet [Z] proposes operations that are reversals of the filtering, pooling and activation layers.



Unpooling using switches that record the location of the local max in each pooling region during pooling in the convnet. From [Z]

A deconvnet layer (left) attached to a convnet layer (right)

DeconvNet

- The math is slightly different, but process is like Vanilla Gradient
- Main difference is in handling ReLU: $R_n = R_{n+1} \mathbf{1}(R_{n+1} > 0)$, where R_n is layer n reconstruction.
- R_{n+1} also considers activations $X_{n+1} > 0$ from the forward pass

$$\text{If } R_{n+1} = \begin{bmatrix} 1 & -1 \\ 0.5 & 3 \end{bmatrix}, \text{ then } R_n = \begin{bmatrix} 1 & 0 \\ 0.5 & 3 \end{bmatrix}$$

Gradient-weighted Class Activation Map

- GradCAM provides visual explanations for CNN decisions.
- Decision may be a class prediction or any other layer.
- The goal of Grad-CAM is to understand at which parts of an image a convolutional layer 'looks' for a certain classification.
- The gradient is not backpropagated all the way back to the image,
 - but usually to the last convolutional layer to produce a coarse localization map that highlights important regions of the image.

GradCAM Algorithm

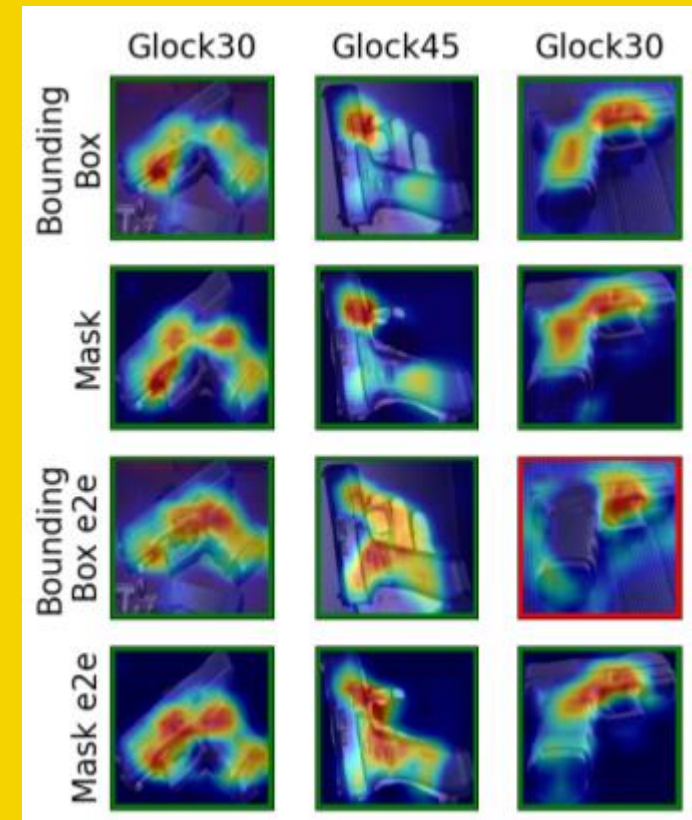
1. Forward propagate the input image through the convolutional neural network.
2. Obtain the raw score for the class of interest (right before softmax), setting all other class activations to zero.
3. Backpropagate the gradient of the class of interest to the last convolutional layer before the fully connected layers $\frac{\partial y^c}{\partial A^k}$
4. Weight each feature map 'pixel' by the gradient for the class:

$$\alpha_k^c = \frac{1}{Z} \sum_{i,j} \frac{\partial y^c}{\partial A_{ij}^k},$$

where k indexes the feature map, i and j index the pixel location in it.

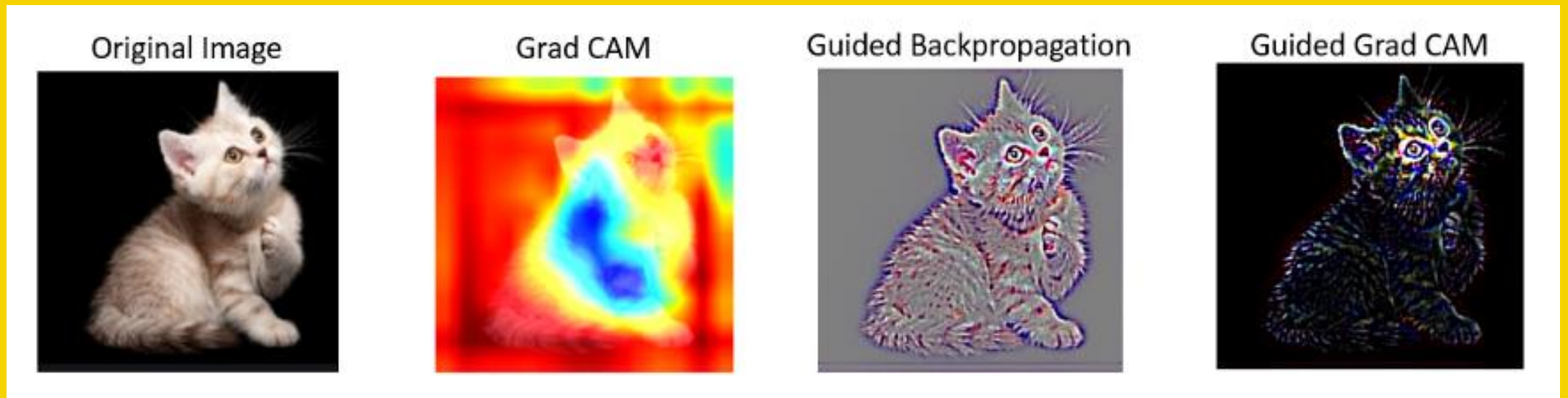
GradCAM Algorithm - Continued

5. Calculate a weighted average of the feature maps: $\sum_k \alpha_k^c A^k$
6. Apply ReLU on the averaged feature map.
7. Transform the attributions into [0,1]
8. Upscale to match the original image
 - Coarser approach: regional gradient, less noisy, faster
 - Can we benefit from coarse and fine approaches?



Guided GradCam

- Combines a fine pixel attribution (e.g. Vanilla) and GradCAM
- Pixel-wise attributions are multiplied.
- GradCAM works as regional magnifier.

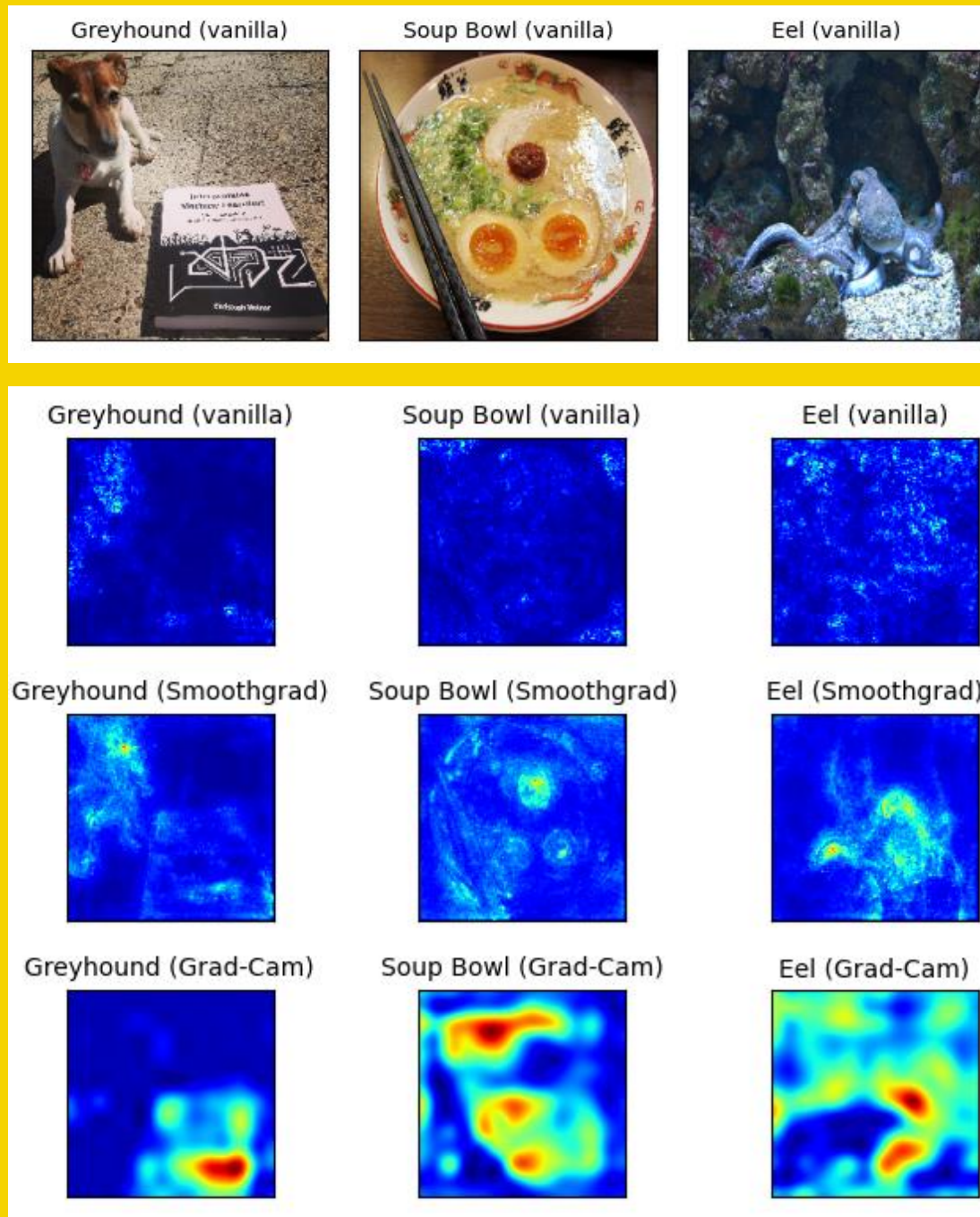


SmoothGrad

- It is not a gradient based method by itself
- It proposes to use preprocessing (e.g. noise addition) and ensemble averaging to alleviate gradient instabilities:
 1. Generate multiple versions of the image of interest by adding noise to it.
 2. Create pixel attribution maps for all images.
 3. Average the pixel attribution maps.
- This idea is similar in spirit to Random Forests robustness by diverse generation of Decision Trees that are sensitive to perturbations



Examples





Remarks on Saliency Maps

- Visual explanations are easy understand.
- Gradient based methods are faster to compute than perturbation-based methods.
- No way to calculate fidelity.
 - We cannot be sure whether explanation is correct.
- Pixel attribution methods may be fragile
 - Small perturbations, pixel shift that yield the same prediction but very different explanations.
- There is yet no consensus on what qualifies as explanation.

Literature for today

- Required Reading

- C. Molnar's book, [Chapter 7](#)
- Grad-Cam: Visual explanations from deep networks via gradient-based localization, Selvaraju et al. ICCV 2017.
https://openaccess.thecvf.com/content_ICCV_2017/papers/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.pdf

- Suggested Reading

- Layer-Wise Relevance Propagation: An Overview, Montavon et al. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning 2019
https://link.springer.com/chapter/10.1007/978-3-030-28954-6_10
- Feature Visualization, Olah et al., Distill, 2017.
<https://distill.pub/2017/feature-visualization/>

Practise for today



A quiz for this lecture is available on course Blackboard page.



An exemplar notebook that includes GradCam implementation will be provided.



The deadline for submitting the quiz is Wednesday May 28, 17:00!



You will find the answers after you submit.

Other Announcements



Tomorrow 17:00 is the deadline for project proposals.



Next Thursday (June 3rd) we will provide some practice questions for midterm.