Indian Institute of Information Technology, Allahabad

CVBL IIIT Allahabad

Explaining CNNs

By

Dr. Satish Kumar Singh & Dr. Shiv Ram Dubey
Computer Vision and Biometrics Lab
Department of Information Technology
Indian Institute of Information Technology, Allahabad

Acknowledgements

- This lecture's slides are based on **Deep Learning for Computer Vision** course taught by Prof. Vineeth Balasubramanian of IIT Hyderabad

- Most of this lecture's slides are based on **Lecture 13** of **CS231n: Convolutional Neural Networks for Visual Recognition** course taught by Fei-Fei Li and others at Stanford University

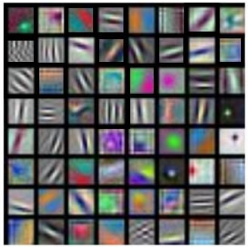
- Some content (AlexNet CNN figures, etc) is also adapted from **Lecture 12** of **CS7015: Deep Learning** course taught by Mitesh Khapra at IIT Madras

Outline: Explaining CNNs

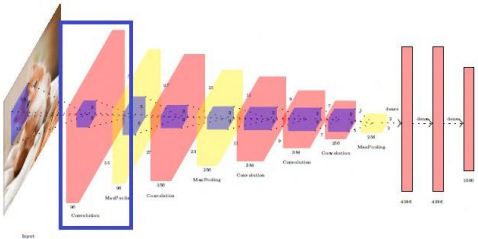
- Visualization Methods
  - Visualize Filters/Kernels (First Layer)
  - Visualize the Representation Space (Last Layer)
  - Visualize Activations
  - Visualize Maximally Activating Image Patches
  - Occlusion Visualization
- Early Methods
  - Backpropagation to Image
  - Visualizing the Data Gradient
  - Image Reconstruction from Latent Representation
  - Guided Backpropagation (Deconvolution method)
- Class Attribution Map Methods
  - Class Activation Maps (CAM)
  - Gradient-weighted CAM (Grad-CAM) and Guided Grad-CAM
  - Grad-CAM++

Explaining CNNs: Visualization Methods

Visualize Filters/Kernels (First Layer)

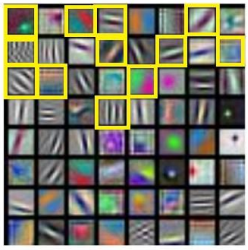


AlexNet:
64 x 3 x 11 x 11

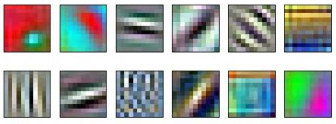


Krizhevsky, One weird trick for parallelizing convolutional neural networks, 2014

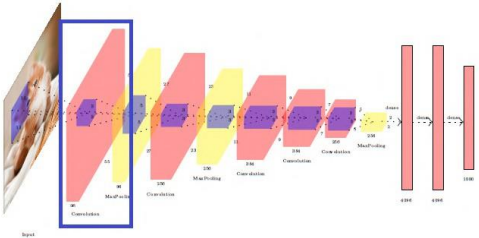
Visualize Filters/Kernels (First Layer)



AlexNet:
64 x 3 x 11 x 11

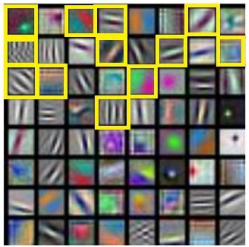


A Closer Look (AlextNet)
Fan et al.(arxiv:1904.05526)

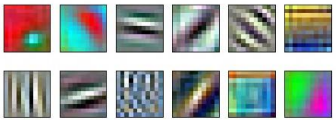


Krizhevsky, One weird trick for parallelizing convolutional neural networks, 2014

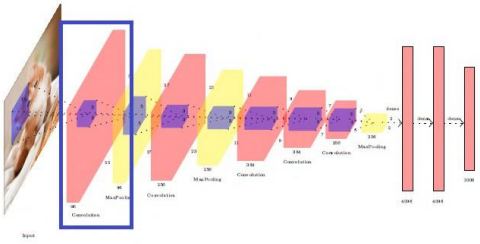
Visualize Filters/Kernels (First Layer)



AlexNet:
64 x 3 x 11 x 11



A Closer Look (AlextNet)
Fan et al.(arxiv:1904.05526)



ResNet-18:
64 x 3 x 7 x 7

ResNet-101:
64 x 3 x 7 x 7

DenseNet-121:
64 x 3 x 7 x 7

Krizhevsky, One weird trick for parallelizing convolutional neural networks, 2014

Visualize Filters/Kernels (First Layer)

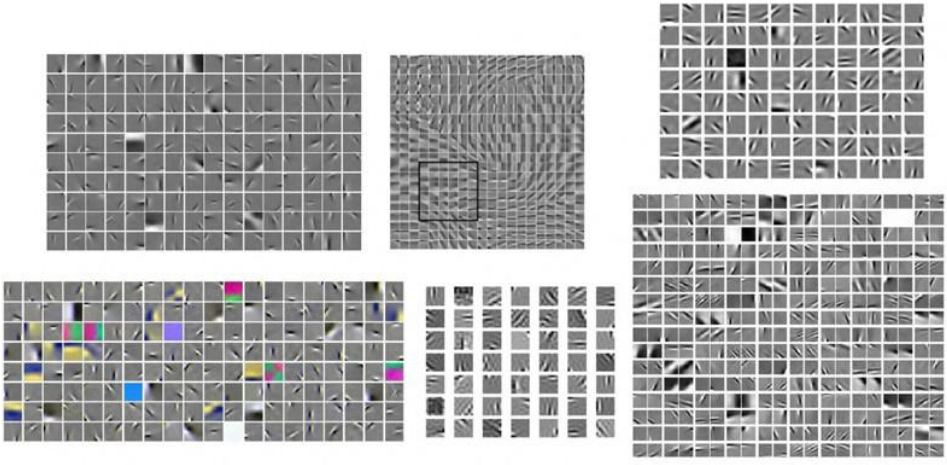
- You can visualize the kernels of higher layers but it is just not interesting

Visualize Filters/Kernels (First Layer)

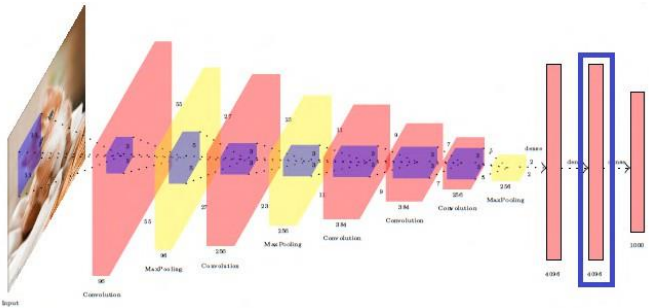
- You can visualize the kernels of higher layers but it is just not interesting
- Input to higher layers is no more the images we know or understand, so becomes difficult to interpret the filters beyond the first layer

Visualize Filters/Kernels (First Layer)

The Gabor-like filters fatigue



Visualize the Representation Space (Last Layer)



- 4096-dimensional feature vector for an image (layer immediately before the classifier)
- Run the network on many images, collect the feature vectors

Visualize the Representation Space (Last Layer)

- Visualize the “space” of FC7 feature vectors by reducing dimensionality of vectors from 4096 to 2 dimensions

Visualize the Representation Space (Last Layer)

- Visualize the “space” of FC7 feature vectors by reducing dimensionality of vectors from 4096 to 2 dimensions
- Use any dimensionality reduction algorithm

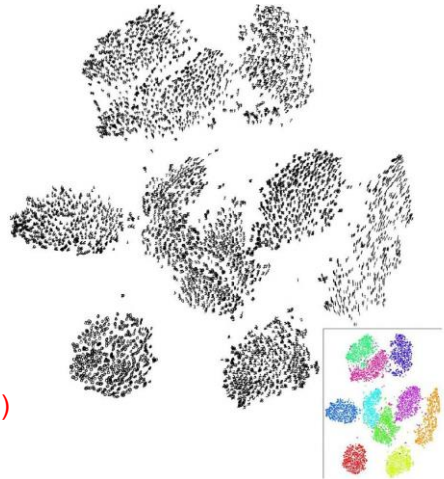
Visualize the Representation Space (Last Layer)

- Visualize the “space” of FC7 feature vectors by reducing dimensionality of vectors from 4096 to 2 dimensions
- Use any dimensionality reduction algorithm
- Simple algorithm: Principal Component Analysis (PCA)

Visualize the Representation Space (Last Layer)

- Visualize the “space” of FC7 feature vectors by reducing dimensionality of vectors from 4096 to 2 dimensions
- Use any dimensionality reduction algorithm
- Simple algorithm: Principal Component Analysis (PCA)
- More complex: t-SNE (right)

  (t-distributed Stochastic Neighbor Embedding)



van der Maaten and Hinton, Visualizing High-Dimensional Data Using t-SNE, Journal of Machine Learning Research, 2008

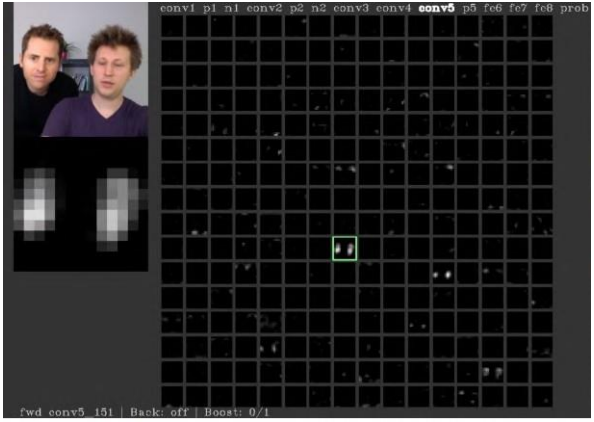
t-SNE Visualization

- Images that are nearby each other are also close in the CNN representation space, which implies that the CNN ”sees” them as being very similar

- Notice that the similarities are more often class-based and semantic, rather than pixel and color-based.

Visualize Activations

Conv5 feature map is 128 × 13 × 13; Visualize as 128 13 × 13 grayscale images

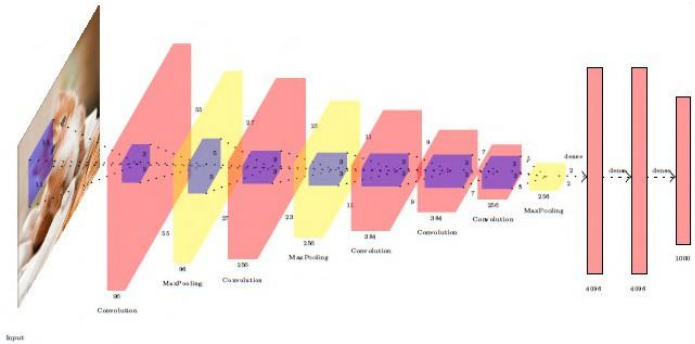


Yosinski et al, "Understanding Neural Networks Through Deep Visualization", ICML DL Workshop 2014.

Figure copyright: Jason Yosinski, 2014.

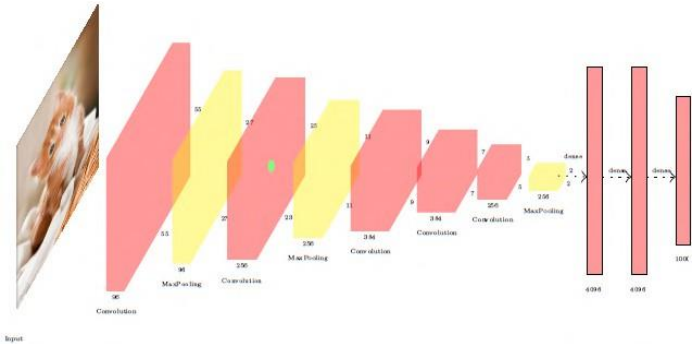
Visualize Maximally Activating Image Patches

- Consider a CNN, and a single neuron in any of its intermediate layers



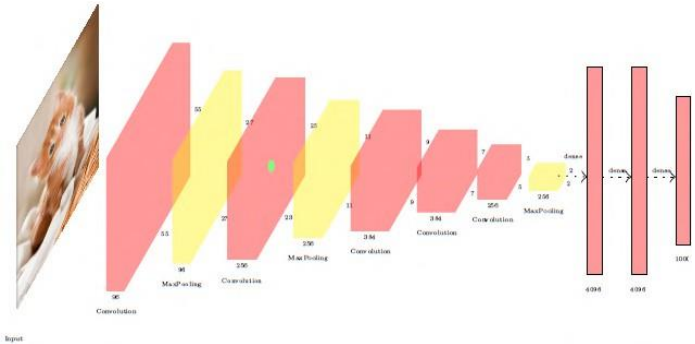
Visualize Maximally Activating Image Patches

- Consider a CNN, and a single neuron in any of its intermediate layers



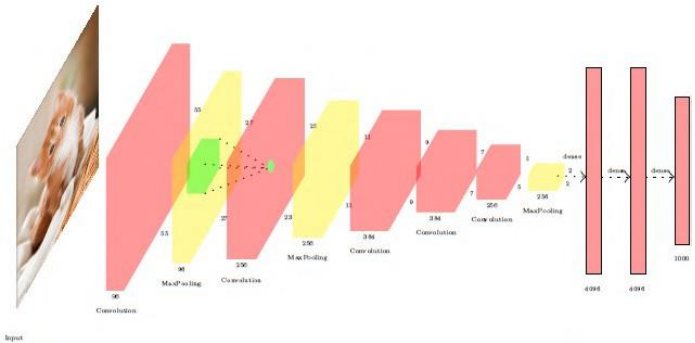
Visualize Maximally Activating Image Patches

- Consider a CNN, and a single neuron in any of its intermediate layers
- Feed images to the CNN and identify all images which cause that particular neuron to fire



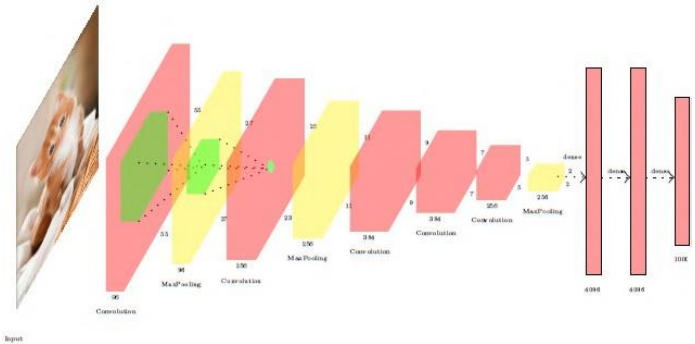
Visualize Maximally Activating Image Patches

- Consider a CNN, and a single neuron in any of its intermediate layers
- Feed images to the CNN and identify all images which cause that particular neuron to fire
- We can then easily trace back to the patch in the image which causes that neuron to fire



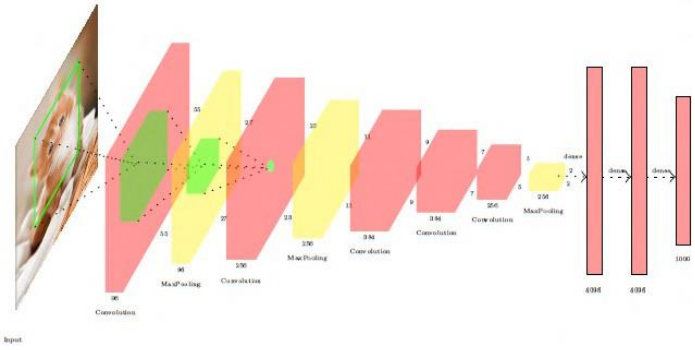
Visualize Maximally Activating Image Patches

- Consider a CNN, and a single neuron in any of its intermediate layers
- Feed images to the CNN and identify all images which cause that particular neuron to fire
- We can then easily trace back to the patch in the image which causes that neuron to fire



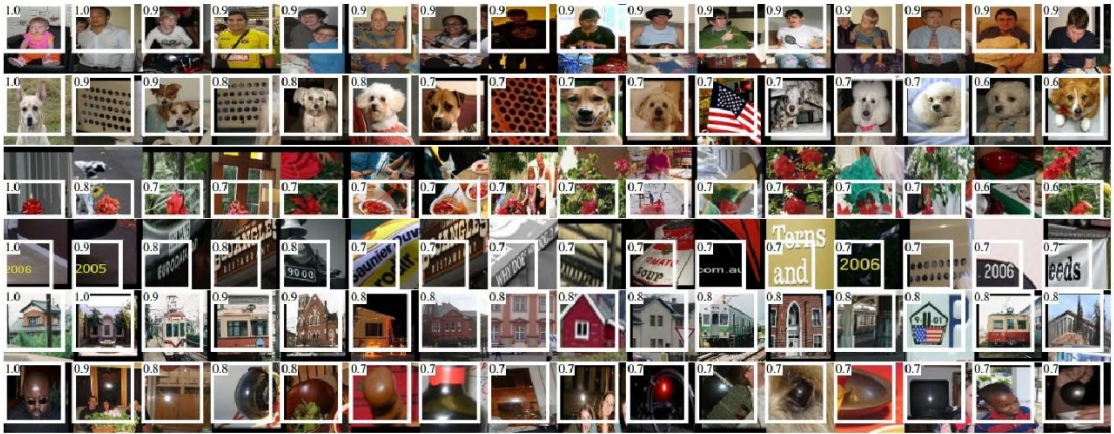
Visualize Maximally Activating Image Patches

- Consider a CNN, and a single neuron in any of its intermediate layers
- Feed images to the CNN and identify all images which cause that particular neuron to fire
- We can then easily trace back to the patch in the image which causes that neuron to fire



Visualize Maximally Activating Image Patches

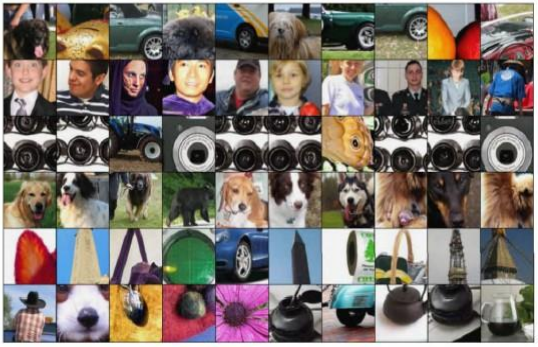
- Repeating this for others neurons in the CNN shows us a pattern



Rich feature hierarchies for accurate object detection and semantic segmentation by Ross Girshick et al.

Visualize Maximally Activating Image Patches

- Repeating this for others neurons in the CNN shows us a pattern



Springenberg et al, "Striving for Simplicity: The All Convolutional Net", ICLR Workshop 2015.
Figure copyright Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin Riedmiller

Occlusion Experiments

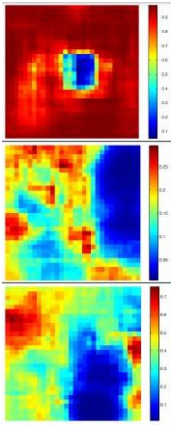
- Typically, we are interested in understanding which portions of an image are responsible for maximizing probability of a certain class

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

Occlusion Experiments

- Typically, we are interested in understanding which portions of an image are responsible for maximizing probability of a certain class
- Occlude (gray out) different patches in the image (centered on each pixel), and see effect on predicted probability of the correct class ⇒ gives you a probability for each pixel

Input image



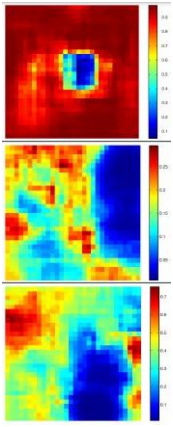
Probability of correct class

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

Occlusion Experiments

- Typically, we are interested in understanding which portions of an image are responsible for maximizing probability of a certain class
- Occlude (gray out) different patches in the image (centered on each pixel), and see effect on predicted probability of the correct class ⇒ gives you a probability for each pixel
- For example, the first heat map (top) shows that occluding the face of the dog causes a maximum drop in the prediction probability

Input image



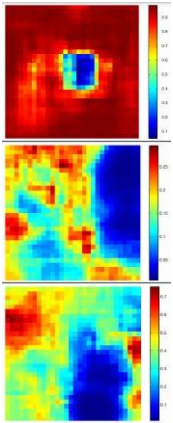
Probability of correct class

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

Occlusion Experiments

- Typically, we are interested in understanding which portions of an image are responsible for maximizing probability of a certain class
- Occlude (gray out) different patches in the image (centered on each pixel), and see effect on predicted probability of the correct class ⇒ gives you a probability for each pixel
- For example, the first heat map (top) shows that occluding the face of the dog causes a maximum drop in the prediction probability
- Similar observations for other images

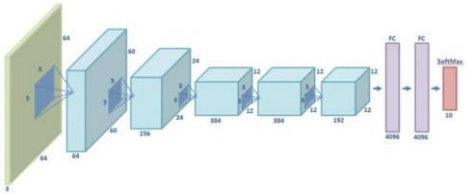
Input image



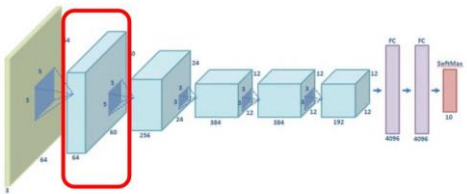
Probability of correct class

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

Summary: ‘Don’t-disturb-the-model’ methods

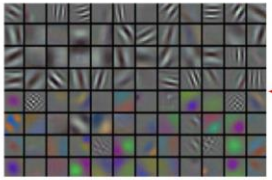


Summary: ‘Don’t-disturb-the-model’ methods

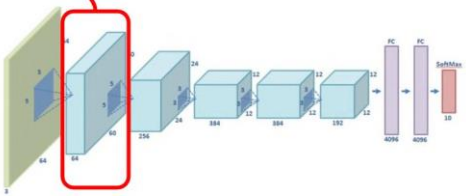


Conv

Summary: ‘Don’t-disturb-the-model’ methods

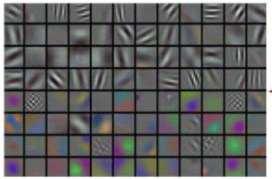


Visualizing the filter/kernels
(raw weights)



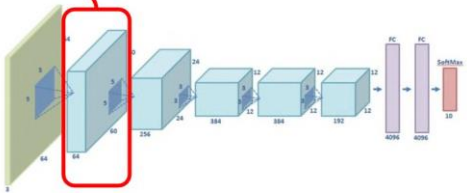
Conv

Summary: ‘Don’t-disturb-the-model’ methods



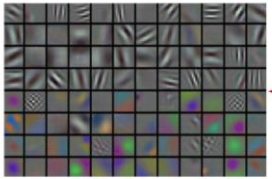
Visualizing the filter/kernels
(raw weights)

Only interpretable at the first layer!
Not interesting enough for higher layers!



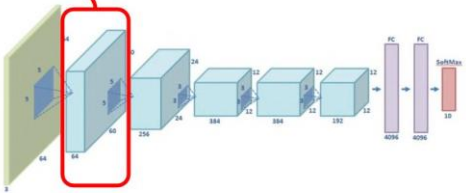
Conv

Summary: ‘Don’t-disturb-the-model’ methods

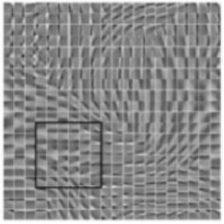


Visualizing the filter/kernels
(raw weights)

Only interpretable at the first layer!
Not interesting enough for higher layers!

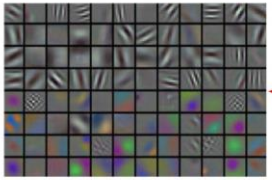


Conv

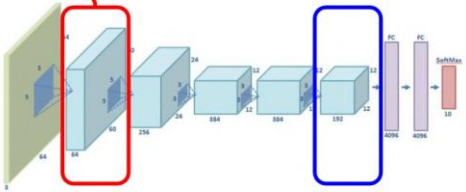


The Gabor-like filter fatigue

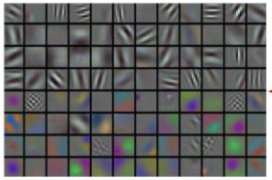
Summary: ‘Don’t-disturb-the-model’ methods



Visualizing the filter/kernels
(raw weights)

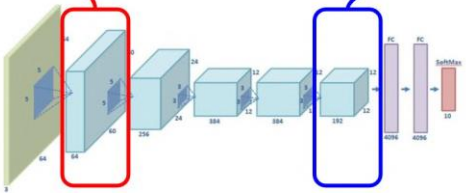


Summary: ‘Don’t-disturb-the-model’ methods



Visualizing the filter/kernels
(raw weights)

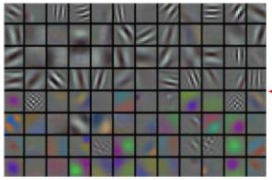
Visualize patches that
maximally activate neuron



Conv

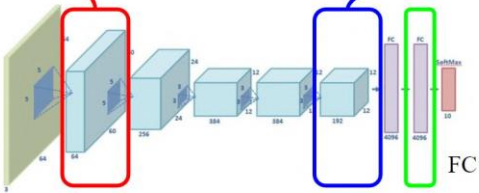
Pool

Summary: ‘Don’t-disturb-the-model’ methods

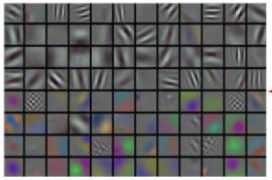


Visualizing the filter/kernels
(raw weights)

Visualize patches that
maximally activate neuron

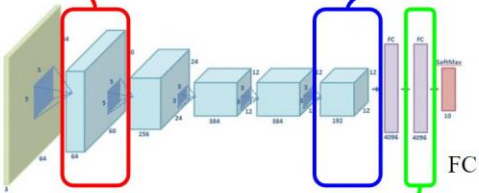


Summary: ‘Don’t-disturb-the-model’ methods



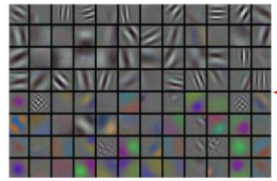
Visualizing the filter/kernels
(raw weights)

Visualize patches that
maximally activate neuron



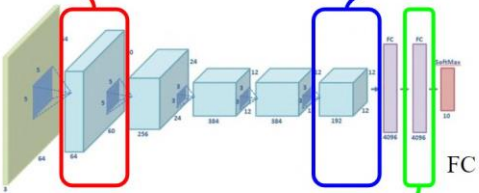
Visualizing the representation

Summary: ‘Don’t-disturb-the-model’ methods



Visualizing the filter/kernels
(raw weights)

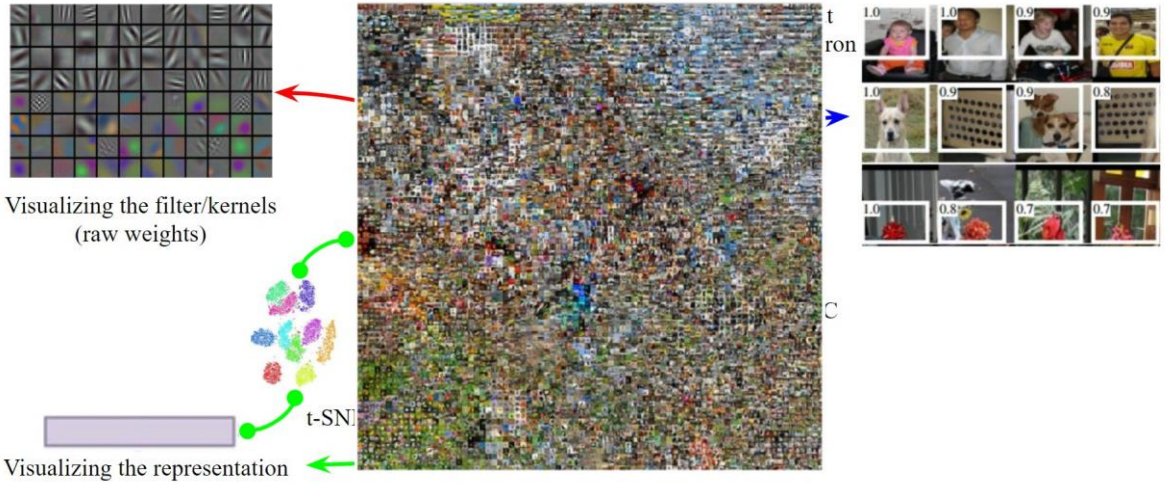
Visualize patches that
maximally activate neuron



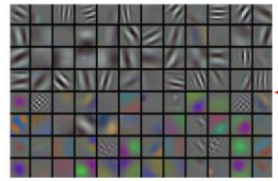
t-SNE

Visualizing the representation

Summary: ‘Don’t-disturb-the-model’ methods

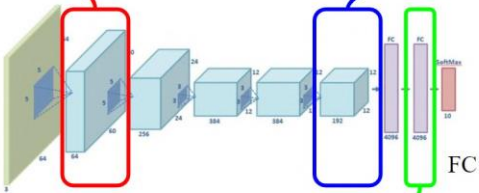


Summary: ‘Don’t-disturb-the-model’ methods



Visualizing the filter/kernels
(raw weights)

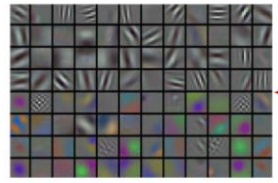
Visualize patches that
maximally activate neuron



t-SNE

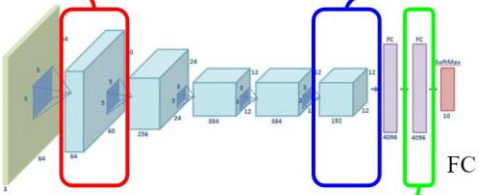
Visualizing the representation

Summary: ‘Don’t-disturb-the-model’ methods



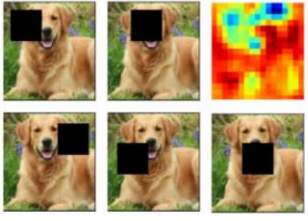
Visualizing the filter/kernels
(raw weights)

Visualize patches that
maximally activate neuron



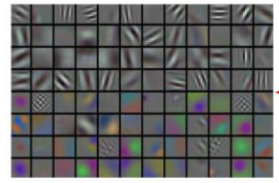
t-SNE

Visualizing the representation



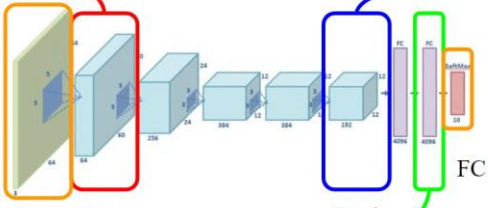
Occlusion Experiment

Summary: ‘Don’t-disturb-the-model’ methods



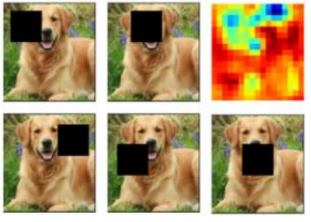
Visualizing the filter/kernels
(raw weights)

Visualize patches that
maximally activate neuron



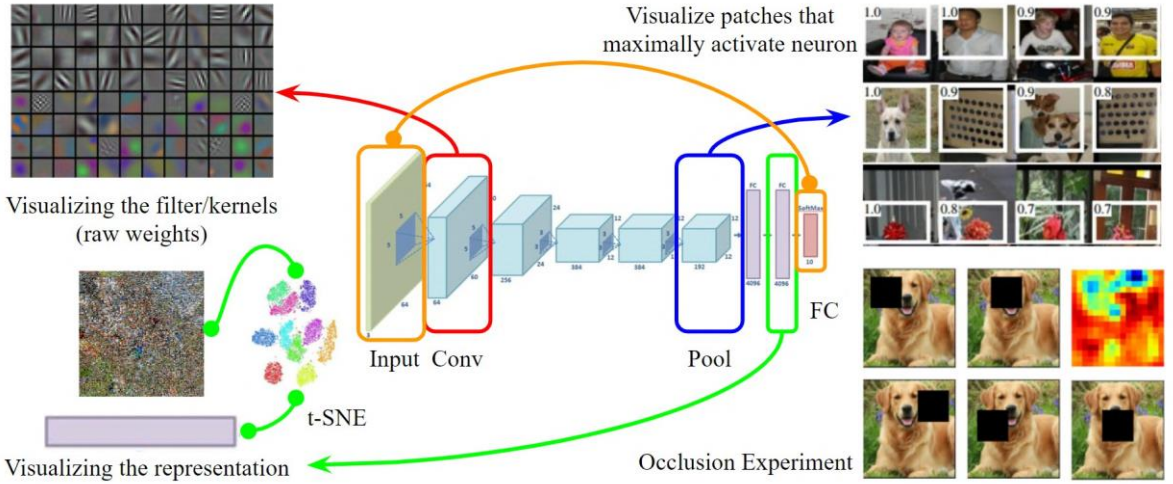
t-SNE

Visualizing the representation

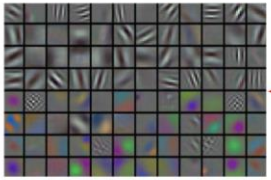


Occlusion Experiment

Summary: ‘Don’t-disturb-the-model’ methods

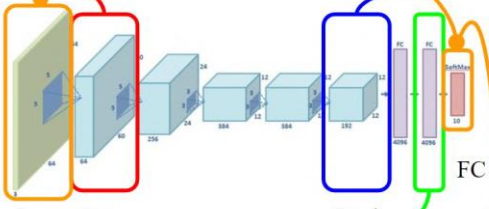


Summary: ‘Don’t-disturb-the-model’ methods



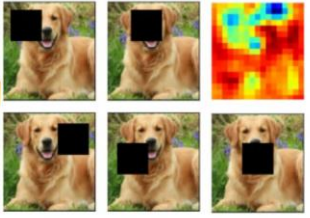
Visualizing the filter/kernels
(raw weights)

Visualize patches that
maximally activate neuron



t-SNE

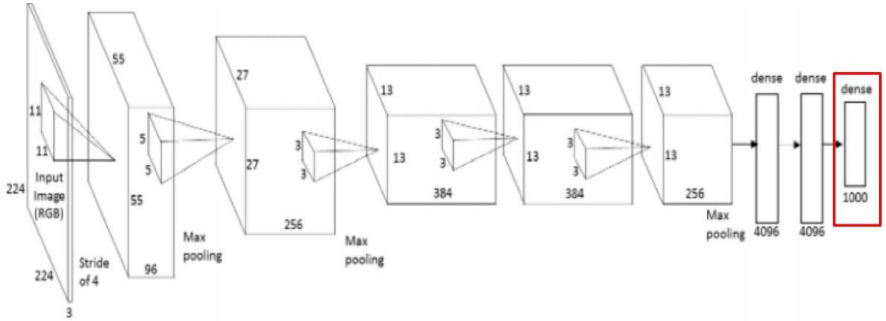
Visualizing the representation



Occlusion Experiment

Explaining CNNs: Early Methods

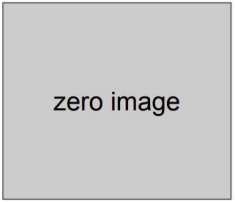
Backpropagation to Image

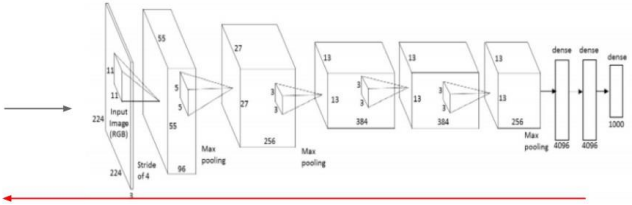


- Question: Can we find an image that maximizes some class score?

Backpropagation to Image

1. Feed zeros as input.

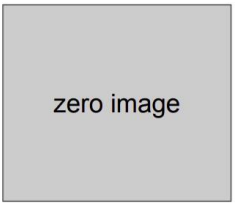


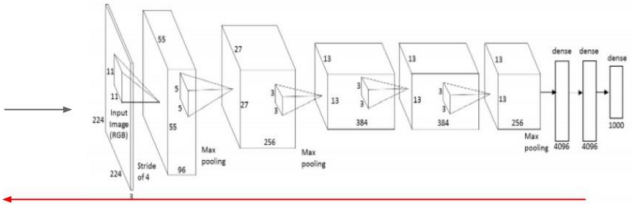


2. Set the gradient of the scores vector to be [0, 0 , .. , 1 , ..., 0]. Then backprop to image.
3. Do a small “image update”
4. Forward pass the image through the network.
5. Go back to step 2.

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

Backpropagation to Image





- Formally, this optimization can be written as:

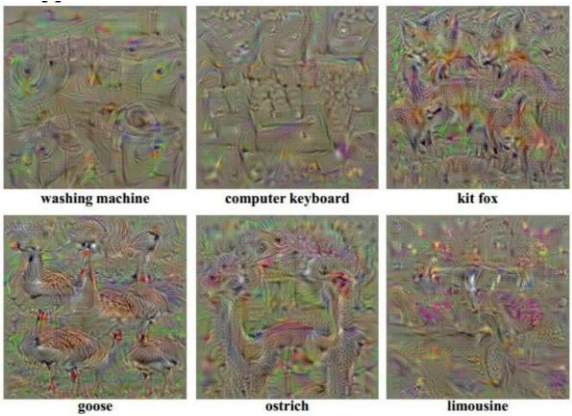
$$\arg\max_I S_c(I) - \lambda \|I\|_2$$

- Here, *Sc* is the scores vector for class *c*, before applying softmax.

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

Backpropagation to Image

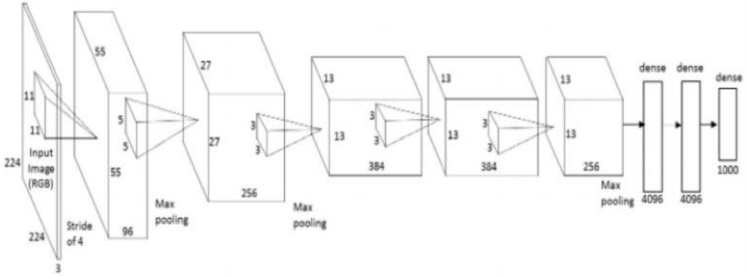
- Finding images that maximize some class score:



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

Backpropagation to Image

- Such optimization can in fact be done for arbitrary neurons in the network



- Repeat:
  1. Forward an image
  2. Set activations in a layer of interest to all zero, except 1.0 for a neuron of interest Backprop to image
  3. Do an “image update”

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

Visualizing the Data Gradient

- Since the gradient on image data has three channels, visualise M such that:

$$M_{ij} = \max_c |\nabla_I S_c(I)|_{(i,j,c)}$$

- At each pixel, take absolute value and pick maximum across channels

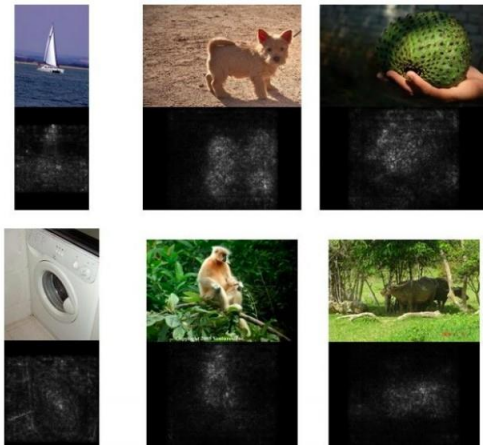
Simonyan, Vedaldi, and Zisserman, Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, ICLR Workshop 2014

Visualizing the Data Gradient

- Since the gradient on image data has three channels, visualise M such that:

$$M_{ij} = \max_c |\nabla_I S_c(I)|_{(i,j,c)}$$

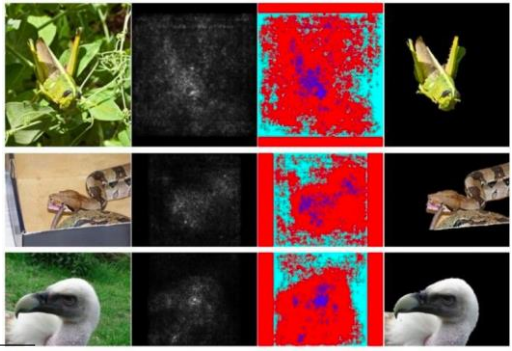
- At each pixel, take absolute value and pick maximum across channels



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

Visualizing the Data Gradient

- GrabCut[1], a segmentation method, can be applied to obtain the object mask from the data gradient
  - Recall Graph-Cut segmentation - GrabCut is an extension/adaptation



[1] https://docs.opencv.org/3.4/d8/d83/tutorial_py_grabcut.html

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

Image Reconstruction from Latent Representation

Given a CNN code (latent representation from a layer, say, FC7), is it possible to reconstruct the original image?

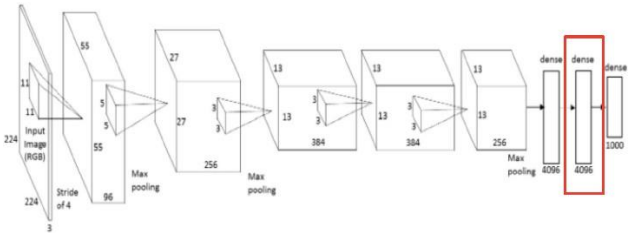
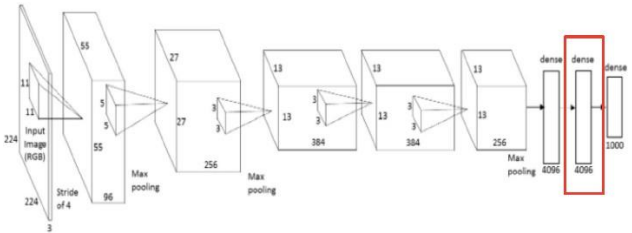


Image Reconstruction from Latent Representation

Given a CNN code (latent representation from a layer, say, FC7), is it possible to reconstruct the original image?



Yes, solve an optimization problem such that:

- The image's code is similar to a given code
- It "looks natural" (image prior regularization)

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathbb{R}^{H \times W \times C}}{\arg\min} \|\mathbf{\Phi}(x) - \mathbf{\Phi}_0\|^2 + \lambda \mathcal{R}(\mathbf{x})$$

Image Reconstruction from Latent Representation

On AlexNet model

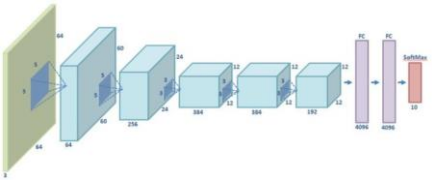
original image

reconstructions from the 1000 log probabilities for ImageNet (ILSVRC) classes

Image Reconstruction from Latent Representation

Reconstructions from representations after last pooling layer
(before first FC layer) in AlexNet

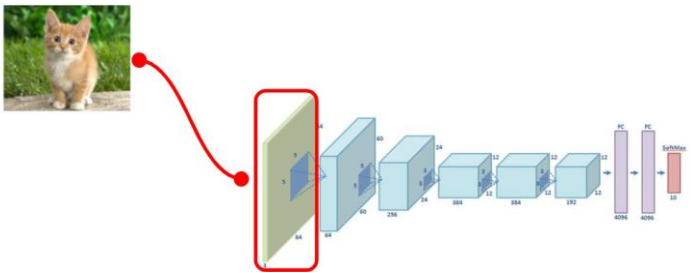
Guided Backpropagation (also known as Deconvolution method)



Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

Guided Backpropagation (also known as Deconvolution method)

a) Feed image into net.

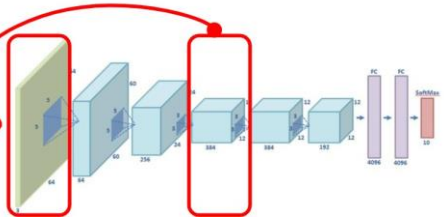


Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

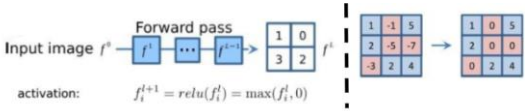
Guided Backpropagation (also known as Deconvolution method)

a) Feed image into net.

b) Pick a layer, set the gradient there to zero except for the neuron of interest.



b)

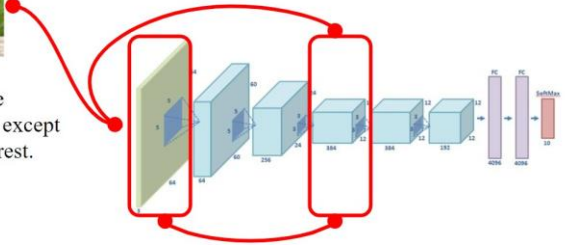


Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

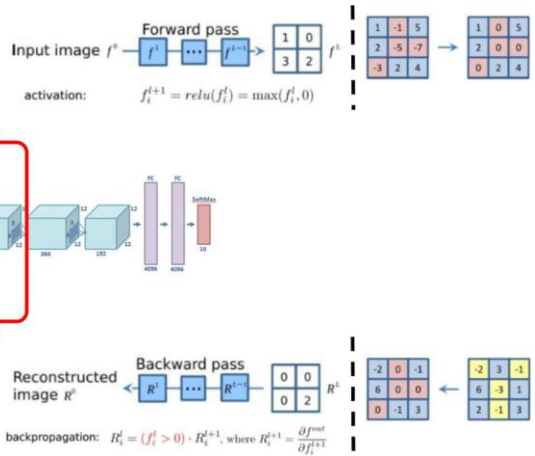
Guided Backpropagation (also known as Deconvolution method)

a) Feed image into net.

b) Pick a layer, set the gradient there to zero except for the neuron of interest.

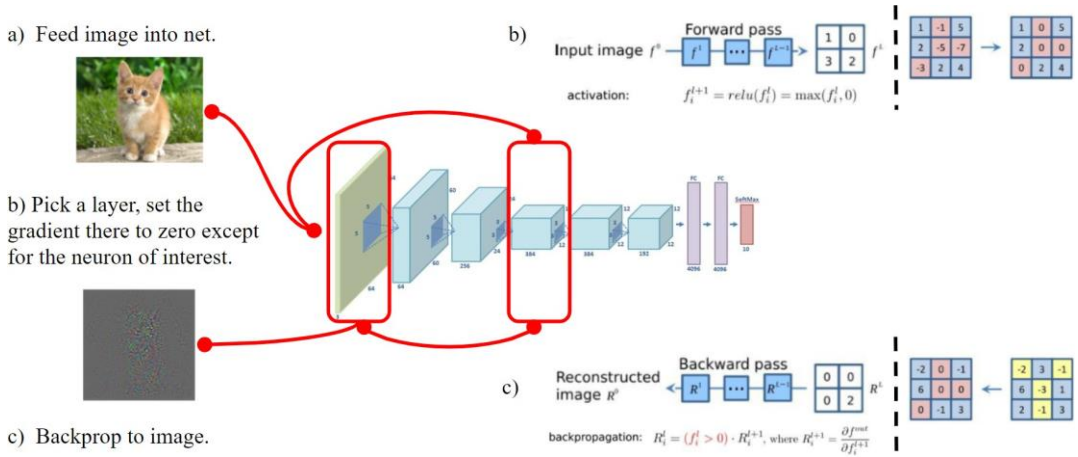


c) Backprop to image.



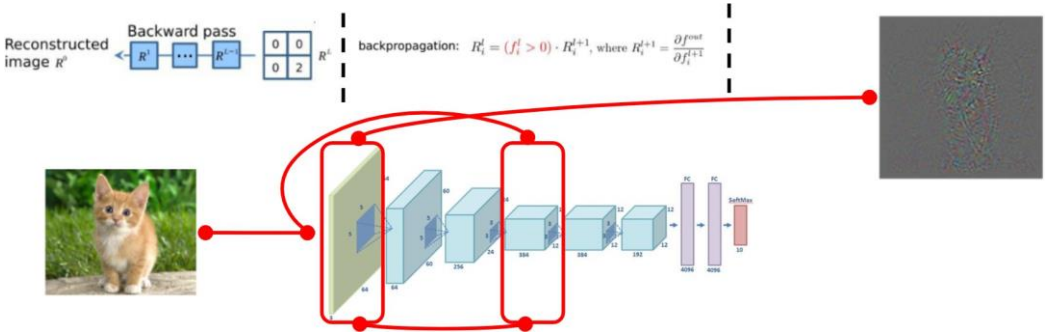
Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

Guided Backpropagation (also known as Deconvolution method)



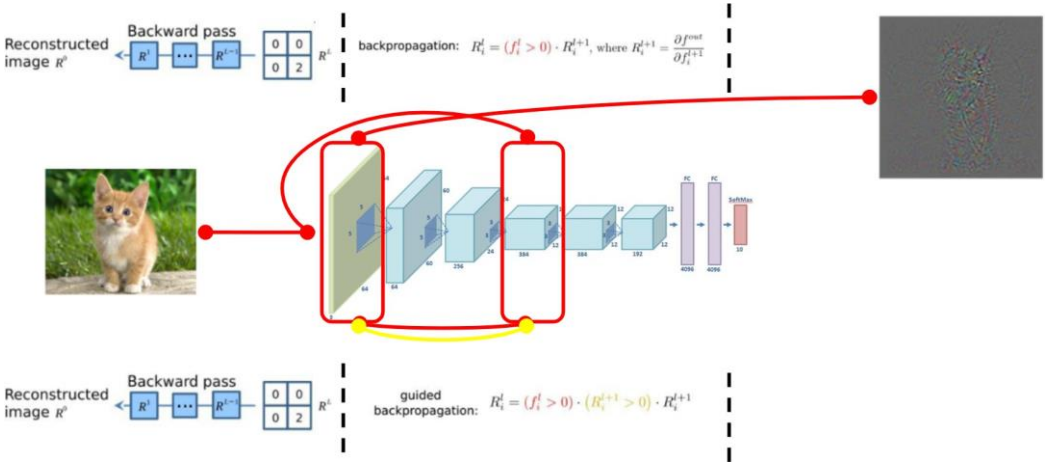
Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

Guided Backpropagation (also known as Deconvolution method)



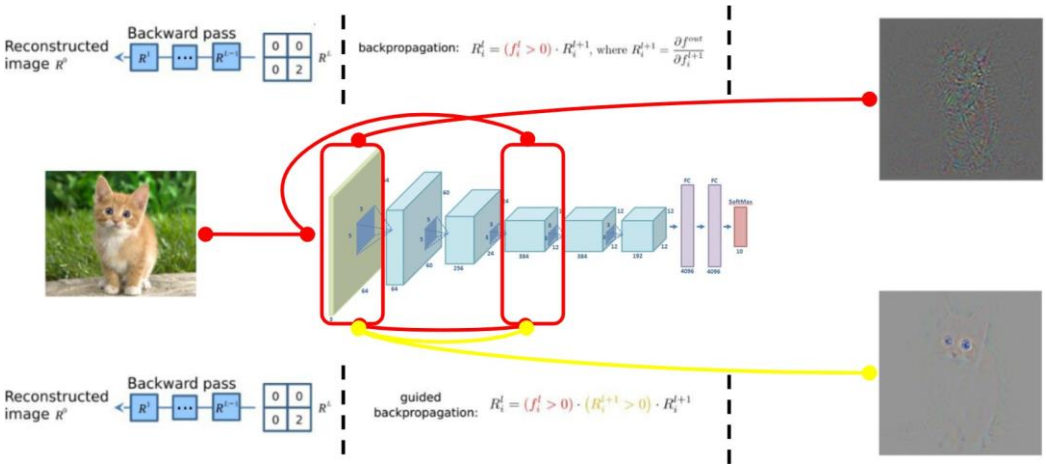
Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

Guided Backpropagation (also known as Deconvolution method)



Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

Guided Backpropagation (also known as Deconvolution method)



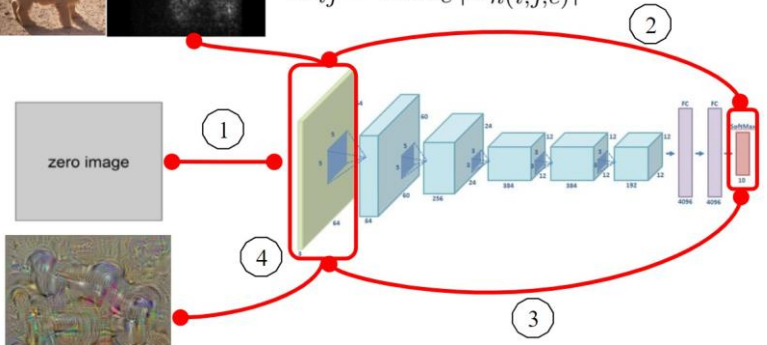
Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015

Explaining CNNs: Class Attribution Map Methods

Going Beyond Optimization-to-Image Methods

Visualize the data gradient. (Note that the data has 3 channels)

$$M_{ij} = \max_c |w_{h(i,j,c)}|$$



Find images that maximize some class score.

1) Feed in zeros.

2) Forward the image through the network.

3) Set the gradient of the score vector to be [0,0,..,1,..0] and backprop to image.

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

[Score for class c (before softmax)]

4) Do a small image update.

5) Go back to 2.

Going Beyond Optimization-to-Image Methods

Visualize the data gradient. (Note that the data has 3 channels)

$$M_{ij} = \max_c |w_{h(i,j,c)}|$$

1) Feed in zeros.

Question

Can we know what a network was looking at, while predicting a class?

zero image

Find images that maximize some class score.

score vector to be [0,0,..,1,..0] and backprop to image.

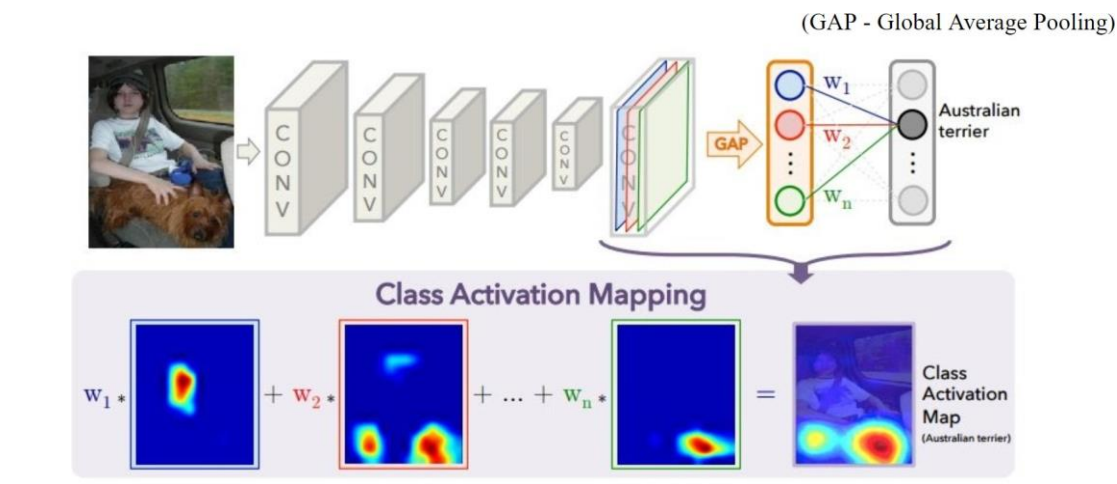
$$\arg\max_I S_c(I) - \lambda \|I\|_2^2$$

[Score for class c (before softmax)]

4) Do a small image update.

5) Go back to 2.

Class Activation Maps (CAM)

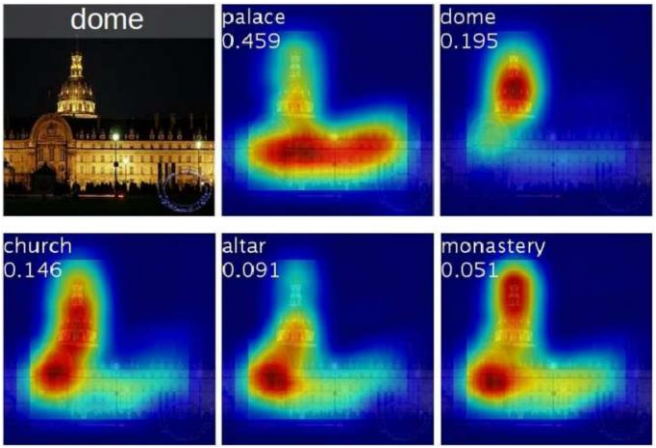


Zhou et al, Learning Deep Features for Discriminative Localization, CVPR 2016

CAM: Examples

- Discriminative image regions used for classification of “Briard” and “Barbells” classes.
- In the first set, the model is using the dog’s face to make the decision and
- In the second set, it is using the weight plates.

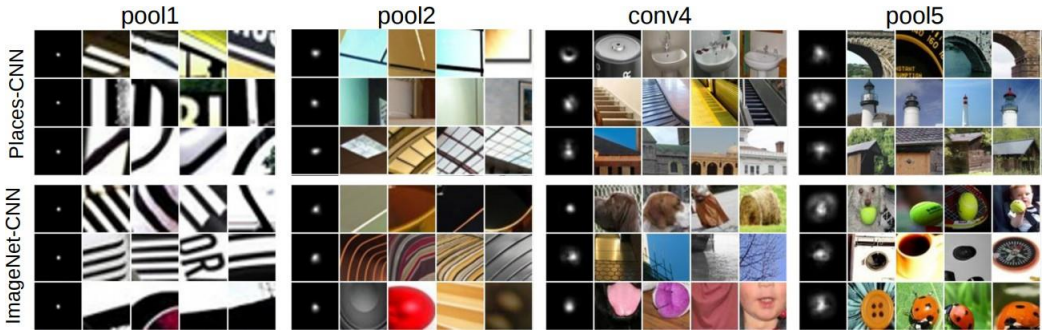
CAM: Examples



- Top 5 predicted classes and their corresponding CAMs.
- Notice how the same activation maps produce different CAMs based on weights connecting features to individual classes

CAM: Intuition

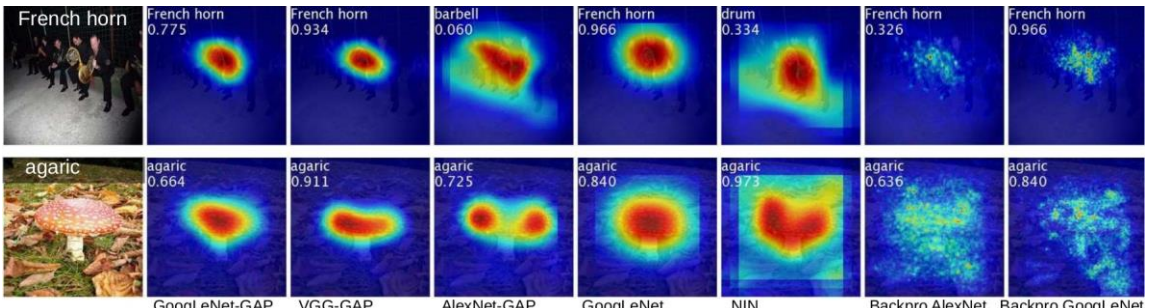
Convolutional units behave as object localizers even without supervision over objects' location; this capability is lost if FC layers are used for classification



Receptive fields of convolutional units and their maximally activating image patch examples

Zhou et al, Object Detectors emerge in Deep Scene CNNs, ICLR 2015

CAM: Comparison



CAM: Pros and Cons

Advantages

- Is class discriminative (can localize objects without positional supervision).
- Doesn't require a backward pass unlike guided backprop or deconvolution

CAM: Pros and Cons

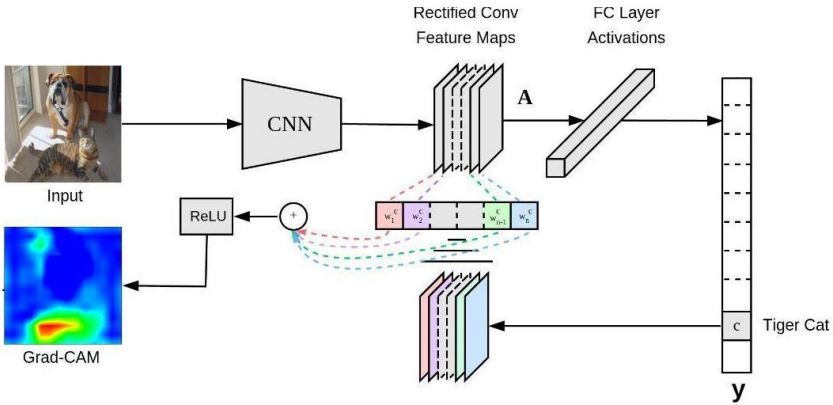
Advantages

- Is class discriminative (can localize objects without positional supervision).
- Doesn't require a backward pass unlike guided backprop or deconvolution

Disadvantages

- Constraint on architecture is restrictive; may not be useful to explain complex tasks like image captioning or visual question answering (VQA)
- Model may trade off accuracy for interpretability
- Need for retraining to explain trained models

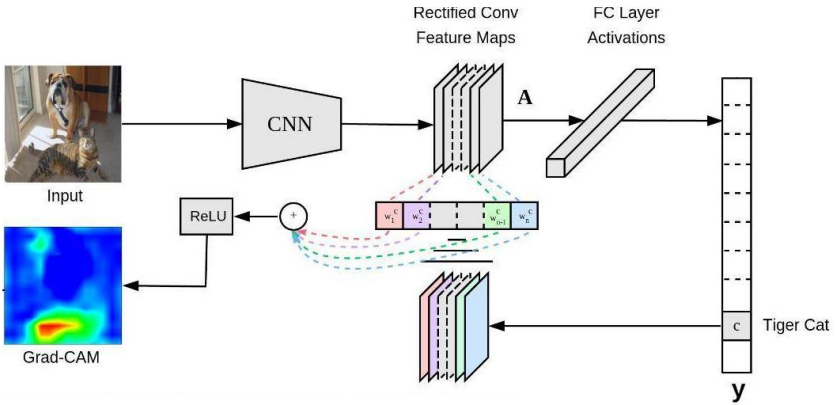
Gradient-weighted CAM (Grad-CAM)



Selvaraju et al, Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, ICCV 2017

Gradient-weighted CAM (Grad-CAM)

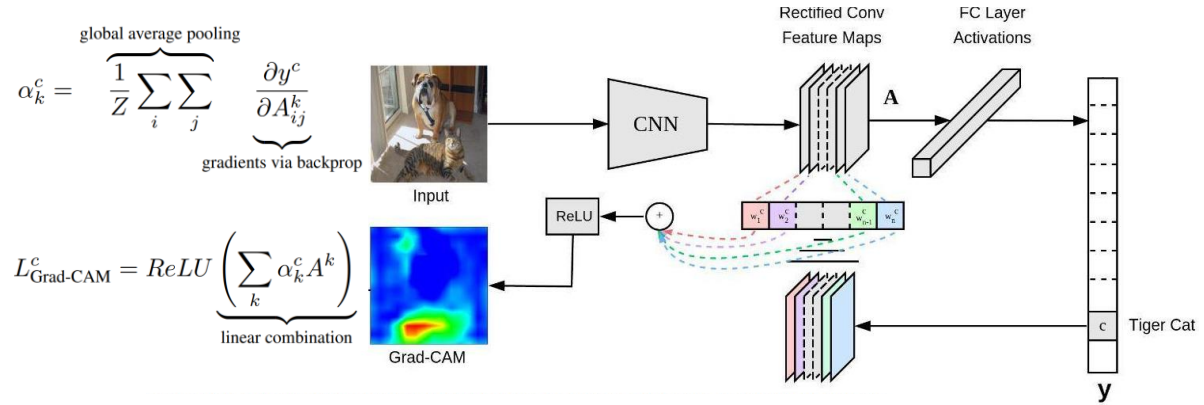
Instead of the GAP, Grad CAM get the gradient value for the weight.



Selvaraju et al, Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, ICCV 2017

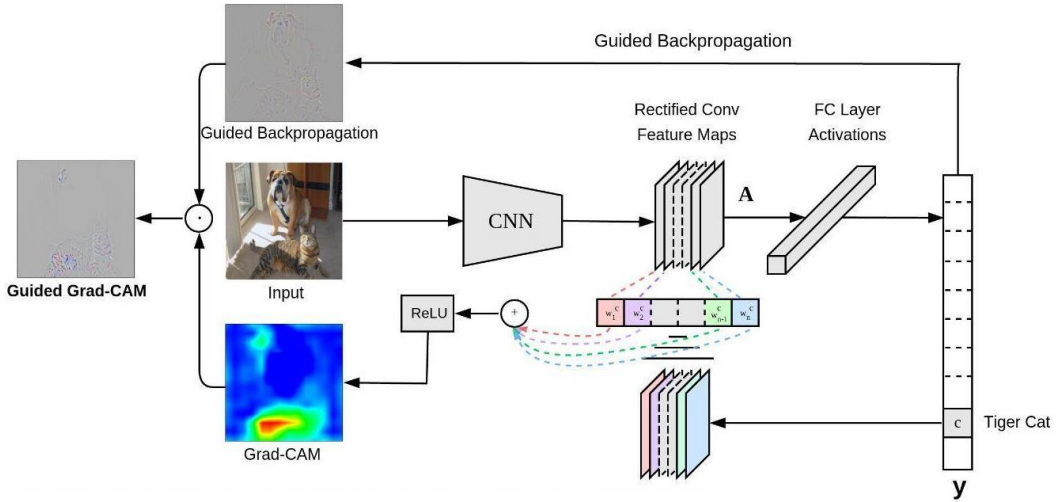
Gradient-weighted CAM (Grad-CAM)

Instead of the GAP, Grad CAM get the gradient value for the weight.



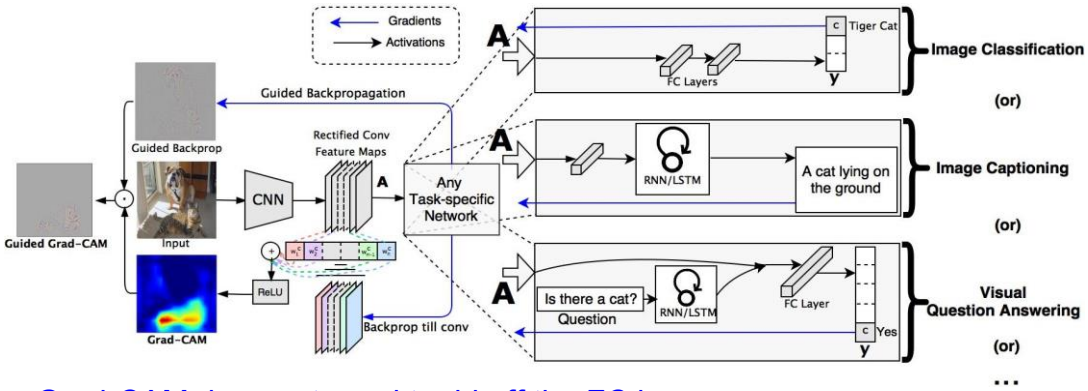
Selvaraju et al, Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, ICCV 2017

Guided Gradient-weighted CAM (Guided Grad-CAM)



Selvaraju et al, Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, ICCV 2017

Gradient-weighted CAM (Grad-CAM)

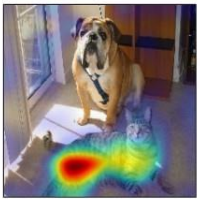


- Grad-CAM does not need to rid off the FC layer.
- It can be applied for every task where CAM could be only applied on the Classification.

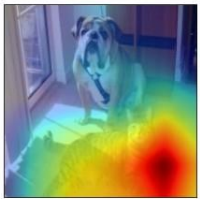
Selvaraju et al, Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, ICCV 2017

Grad-CAM: Example

Original Image

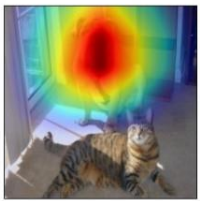


Grad-CAM ‘Cat’



ResNet Grad-CAM ‘Cat’

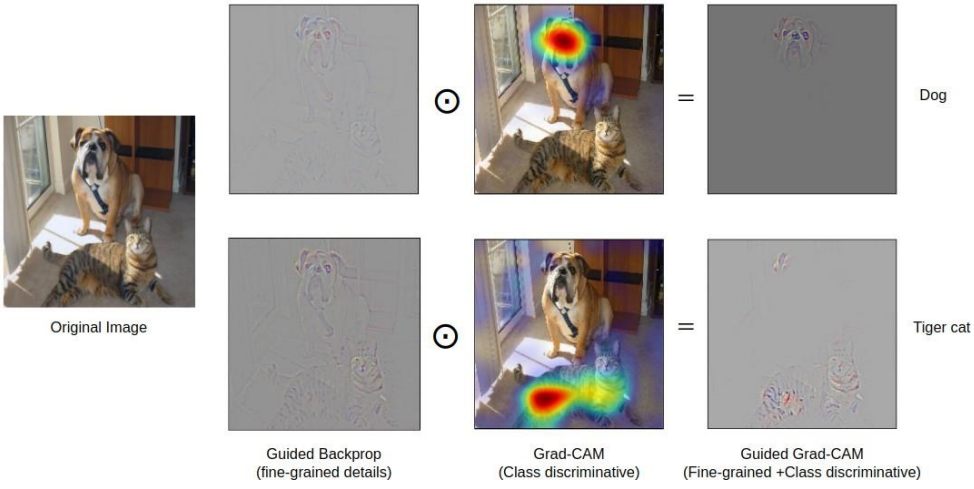
Grad-CAM ‘Dog’



ResNet Grad-CAM ‘Dog’

- Grad-CAM maps are class-discriminative
- However, it is **unclear** from this heat-map why the network predicts this particular instance as **‘tiger cat’**
- Can we do something about this?

Guided Grad-CAM



Grad-CAM: Counterfactual Explanations

- Negating the value of gradients used for calculation of importance weights (w_k^c) causes localization maps to show image patches that adversarially affect classification output

$$w_k^c = \frac{1}{Z} \sum_i \sum_j -\frac{\partial Y^c}{\partial A_{ij}^k}$$

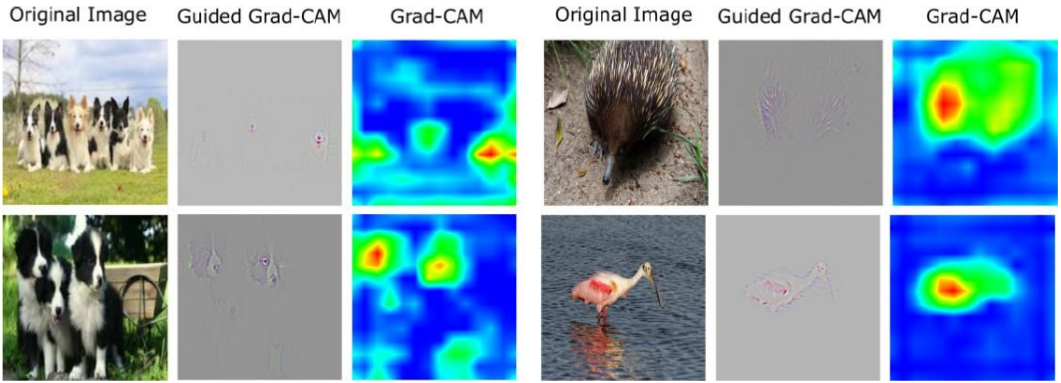
- Removing/suppressing features occurring in such patches can improve model confidence

(a) Original Image (b) Grad-CAM for Cat (c) Grad-CAM negative explanation for Cat

(d) Original Image (e) Grad-CAM for Dog (f) Grad-CAM negative explanation for Dog

Grad-CAM: Limitations

- Inability to identify multiple instances of objects
- Unsatisfactory localization performance, especially under occlusion



Chattopadhay et al, Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks, WACV 2018

Grad-CAM++: Motivation

- Grad-CAM considers all pixel gradients equally when computing importance weights of activation maps

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

Grad-CAM++: Motivation

- Grad-CAM considers all pixel gradients equally when computing importance weights of activation maps

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

- This can suppress activation maps with comparatively lesser spatial footprint

Grad-CAM++: Motivation

- Grad-CAM considers all pixel gradients equally when computing importance weights of activation maps

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

- This can suppress activation maps with comparatively lesser spatial footprint
- Since instances of objects in an image tend to have different shapes and orientations, some of them can fade away

Grad-CAM++: Motivation

- Grad-CAM considers all pixel gradients equally when computing importance weights of activation maps

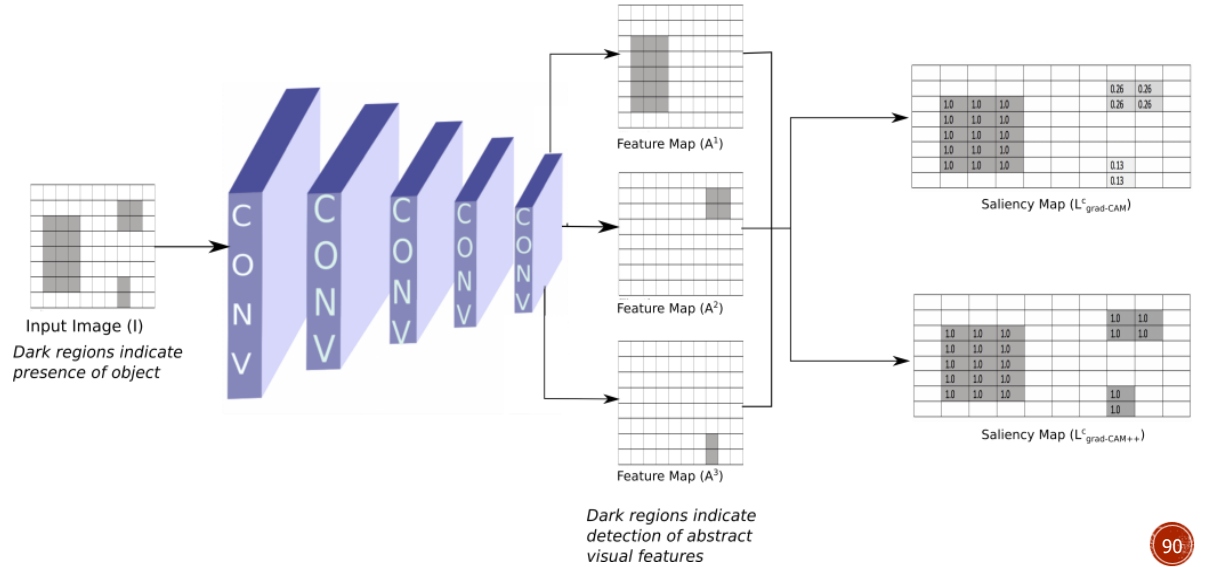
$$\alpha_k^c = \frac{1}{Z}\sum_i\sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

- This can suppress activation maps with comparatively lesser spatial footprint
- Since instances of objects in an image tend to have different shapes and orientations, some of them can fade away
- This can be corrected by using weighted average of pixel-wise gradients

$$w_k^c = \sum_i\sum_j \alpha_{ij}^{kc}.relu(\frac{\partial Y^c}{\partial A_{ij}^k})$$

- where relu to focus on positive gradients and α is the pixel-wise weight.

Grad-CAM++: Intuition



Grad-CAM++: Methodology

- For a particular class c and activation map k, the pixel-wise weight α^{kc} at pixel position (i, j) can be calculated as:

$$\alpha_{ij}^{kc} = \frac{\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2}}{2\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} + \sum_a \sum_b A_{ab}^k \{\frac{\partial^3 Y^c}{(\partial A_{ij}^k)^3}\}}$$

NOTE: Both a,b and i,j are iterators on the same activation map. They are only used to avoid confusion.

Section 3.1 to 3.4, Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks

Grad-CAM++: Methodology

- For a particular class c and activation map k, the pixel-wise weight α^{kc} at pixel position (i, j) can be calculated as:

$$\alpha_{ij}^{kc} = \frac{\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2}}{2\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} + \sum_a \sum_b A_{ab}^k \{\frac{\partial^3 Y^c}{(\partial A_{ij}^k)^3}\}}$$

  NOTE: Both a,b and i,j are iterators on the same activation map. They are only used to avoid confusion.

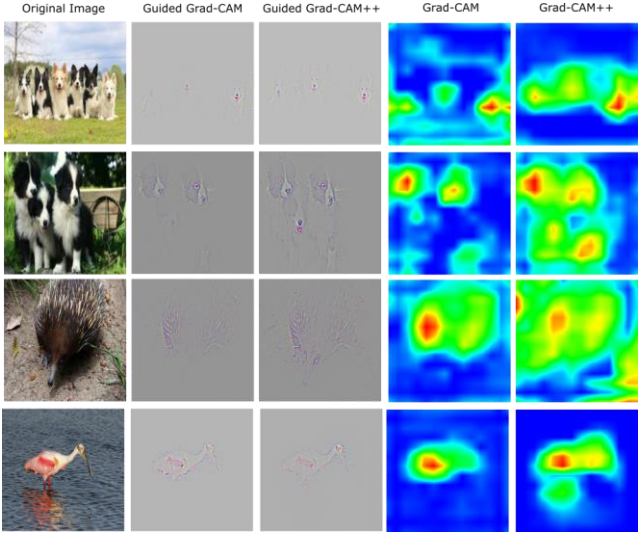
- Final localization map $L_{Grad-CAM++}$ (similar to that of GradCAM):

$$L_{ij}^c = relu(\sum_k w_k^c . A_{ij}^k)$$

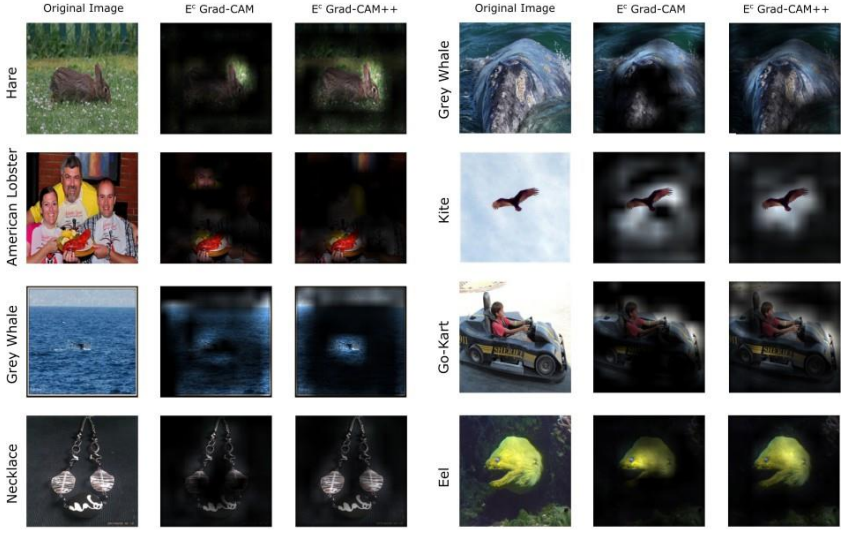
where $$w_k^c = \sum_i \sum_j \alpha_{ij}^{kc} . relu(\frac{\partial Y^c}{\partial A_{ij}^k})$$

Section 3.1 to 3.4, Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks

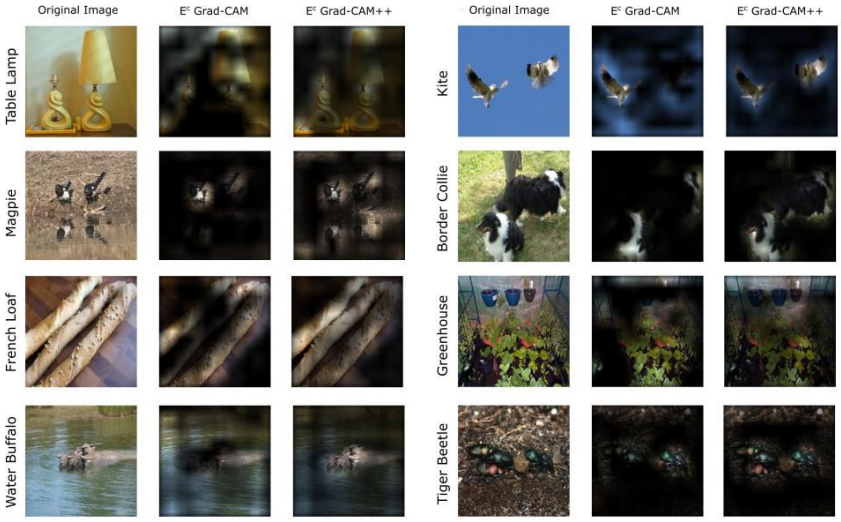
Grad-CAM++: Example



Grad-CAM++: Examples for Class Localization



Grad-CAM++: Examples for Multiple Occurrences



Readings

- Lecture Notes of CS231n, Stanford
- Deep Visualization Toolkit demo video and webpage by J. Yosinski et al.
- To see high-resolution t-SNE visualizations, visit here To know more about t-SNE, visit here
- Simonyan, Vedaldi, and Zisserman, Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, ICLR Workshop 2014
- Zeiler and Fergus, Visualizing and Understanding Convolutional Networks, ECCV 2014
- Springenberg et al, Striving for Simplicity: The All Convolutional Net, ICLR Workshop 2015
- Zhou et al, Learning Deep Features for Discriminative Localization, CVPR 2016
- Selvaraju et al, Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, ICCV 2017
  Chattopadhyay et al, Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks, WACV 2018

References

- Laurens van der Maaten and Geoffrey E. Hinton. “Visualizing Data using t-SNE”. In: 2008.
- Ross Girshick et al. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Nov. 2013).
- Matthew D. Zeiler and Rob Fergus. “Visualizing and Understanding Convolutional Networks”. In: *ECCV*. 2014.
- Jost Tobias Springenberg et al. “Striving for Simplicity: The All Convolutional Net”. In: *CoRR* abs/1412.6806 (2015).
- Min Lin, Qiang Chen, and Shuicheng Yan. “Network in network”. In: *arXiv preprint arXiv:1312.4400*
- (2013).
- Bolei Zhou et al. “Object detectors emerge in deep scene cnns”. In: *arXiv preprint arXiv:1412.6856*
- (2014).
- Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.
- Bolei Zhou et al. “Learning deep features for discriminative localization”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2921–2929.
- Ramprasaath R Selvaraju et al. “Grad-cam: Visual explanations from deep networks via gradient-based localization”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 618–626.
- Aditya Chattopadhay et al. “Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks”. In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2018.

Thank you

QUESTIONS?

Image Source: http://pluspng.com/best-of-luck-png-5072.html