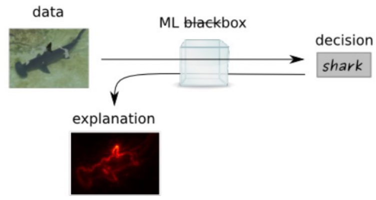
Which **types** of introspection techniques can you distinguish?

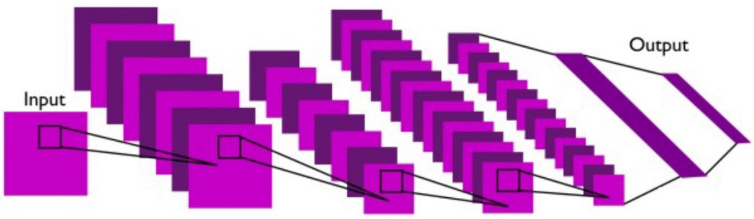
- Find at least one categorization scheme! (more are possible)
- Give as many **examples** as you can per category!

Types of Introspection

global	local (depending on input)
feature visualization e.g. DeepDream	**saliency maps** e.g. layer-wise relevance propagation (LRP)



Feature Visualization

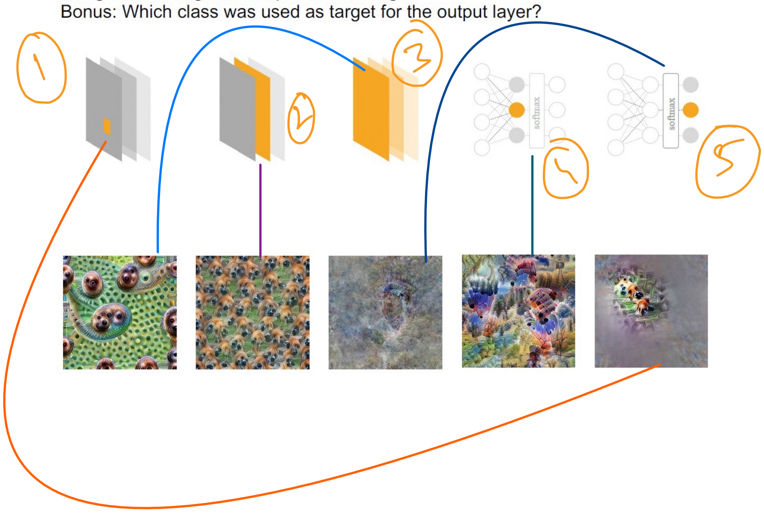


feature visualization by optimization
(find the input that optimizes a particular part of the network)

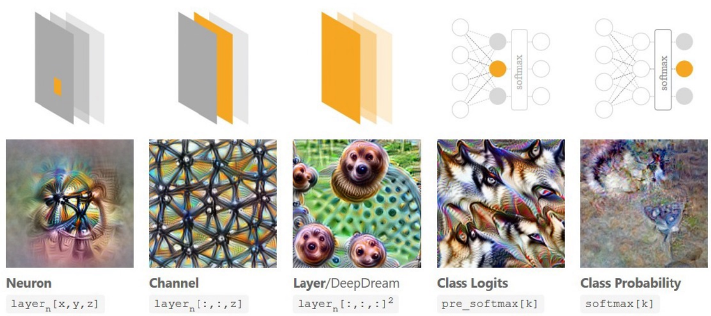
Task 3

Feature Visualization

Assign each image to it's optimization target!
Bonus: Which class was used as target for the output layer?



Feature Visualization



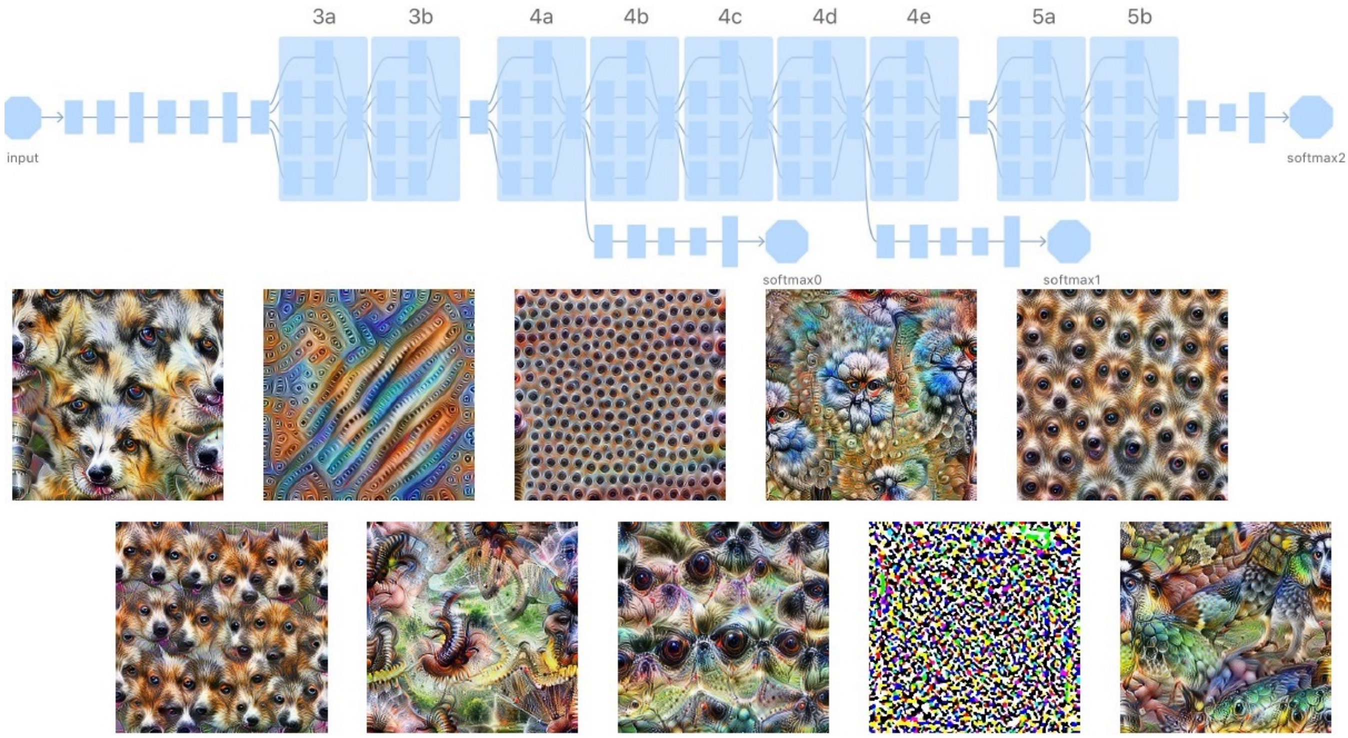
https://distill.pub/2017/feature-visualization/

① → in the beginning the receptive field is small
② → we are looking at one Filter convolving, so we can see so many things repeated.
③ → we are trying to see an abstract of the
entire channels depth (for e.g filters = 64. Then we can see some repeated info but not everything.
④ It doesn't consider an other neurons before softmax.
⑤ It considers other Neurons after softmax.

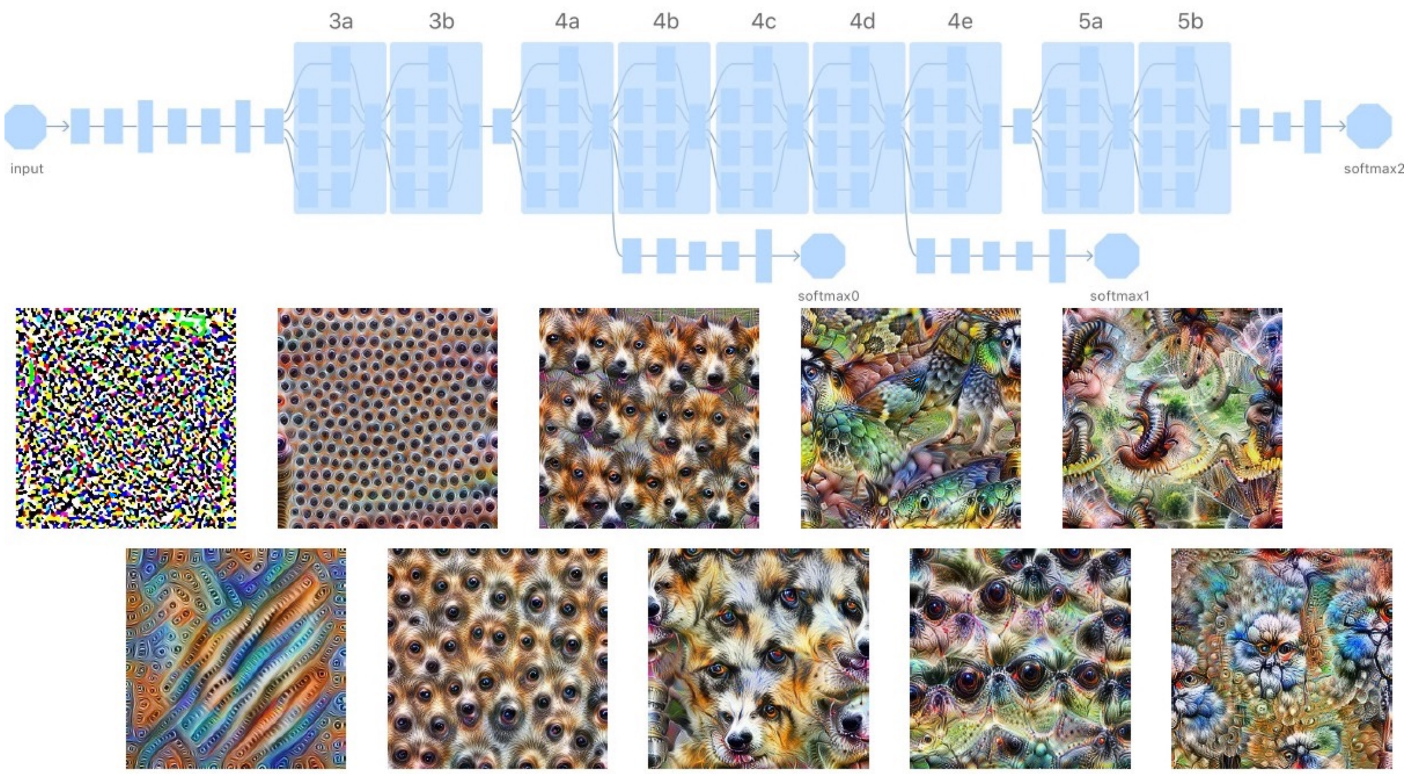
Task 4

Feature Visualization: DeepDream

Which layers were used as optimization target to generate these images?



Feature Visualization: DeepDream



Task 5

Feature Visualization

What's the main problem with the (vanilla) optimization approach?

vs

How do we solve this problem?

Feature Visualization

What's the main problem with the (vanilla) optimization approach?

unregularized optimization is unnatural

vs

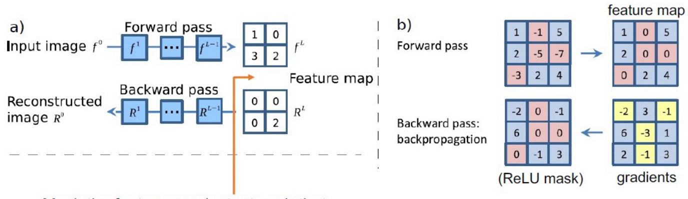
How do we solve this problem?

by regularizing the optimization

frequency penalization	transformation robustness	learned prior

gradient-based Saliency Maps

easiest method: using backpropagation values (i.e. the gradients) directly



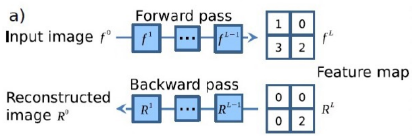
Mask the feature map/output such that all except the value of interest are zeroed out e.g. the output neuron of the predicted class

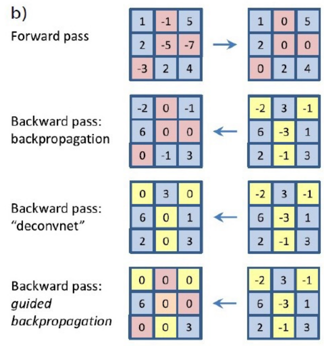
compute gradients for the masked feature map (or output)

Propagating back to the input gives a saliency map.
Each position tells how sensitive the value of interest is to changes in this position.
Hence the name **sensitivity analysis**.

[Springenberg et al. (2014). Striving for Simplicity: The All Convolutional Net]

gradient-based Saliency Maps





c)

activation: $f_i^{l+1} = relu(f_i^l) = \max(f_i^l, 0)$

backpropagation: $R_i^l = (f_i^l > 0) \cdot R_i^{l+1}$, where $R_i^{l+1} = \frac{\partial f^{out}}{\partial f_i^{l+1}}$

backward 'deconvnet': $R_i^l = (R_i^{l+1} > 0) \cdot R_i^{l+1}$

guided backpropagation: $R_i^l = (f_i^l > 0) \cdot (R_i^{l+1} > 0) \cdot R_i^{l+1}$

[Springenberg et al. (2014). Striving for Simplicity: The All Convolutional Net]

Deep Taylor Decomposition and LRP

What's the difference?

deep Taylor decomposition

$$R_d^{(1)} = (x - x_0)_{(d)} \cdot \frac{\partial f}{\partial x_{(d)}}(x_0)$$

- root point x_0 must be determined
- computationally efficient (backprop)

layer-wise relevance propagation

$$R_{i \leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j} \cdot R_j^{(l+1)}$$

- no root point needed
- computationally expensive

GradCAM: Gradient-weighted Class Activation Mapping

(a) Original Image (c) Grad-CAM 'Cat'

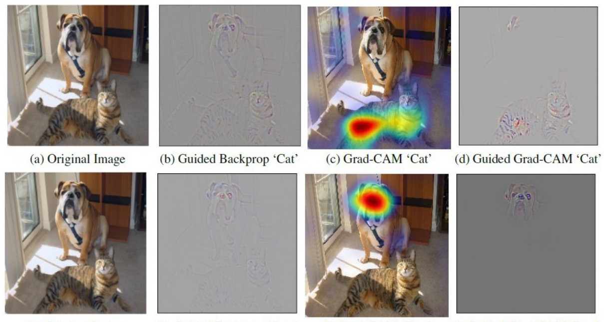
$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i \sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

gradient (sensitivity) of class c to changes in feature map A^k averaged over all positions

$$L_{\text{Grad-CAM}}^c = ReLU \underbrace{\left(\sum_k \alpha_k^c A^k\right)}_{\text{linear combination}}$$

combine all feature maps A^k in one layer as weighted sum using α_k^c as weight

GradCAM: Gradient-weighted Class Activation Mapping

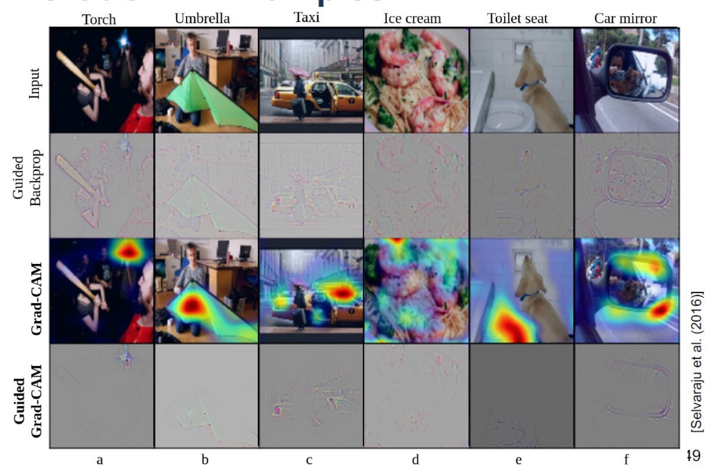


(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat' (d) Guided Grad-CAM 'Cat'

(g) Original Image (h) Guided Backprop 'Dog' (i) Grad-CAM 'Dog' (j) Guided Grad-CAM 'Dog'

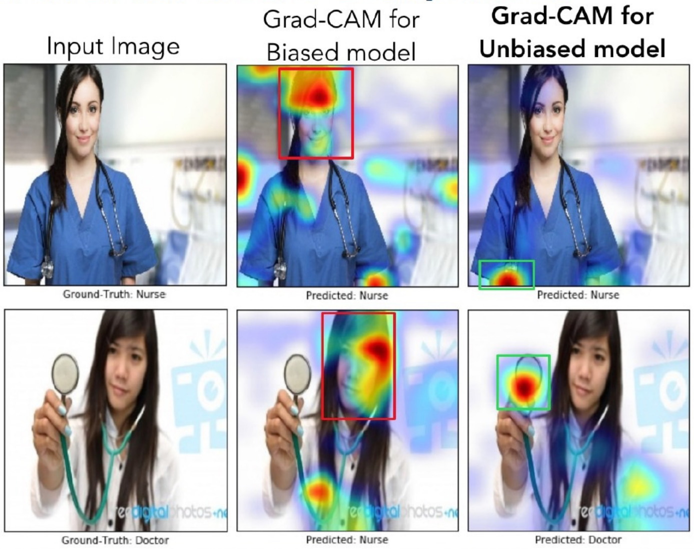
[Selvaraju et al. (2016). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization.]

GradCAM: Examples



[Selvaraju et al. (2016)]

GradCAM: Model Comparison



Problems

- these methods
  - sometimes require particular architectures (e.g. only 2D-convolution with max-pooling)
  - mostly use ReLUs and a positive input space (which pixels positively influence an output class)
  - are mostly evaluated only for images (visually interpretable)
- not well applicable for
  - other activation functions (allowing negative activation)
  - real-valued input space (negative values)
  - visually hardly interpretable data (e.g. waveforms)