Neuron interpretation

CSEP 590B: Explainable AI

Hugh Chen, Ian Covert & Su-In Lee
University of Washington

Neuron interpretation

- Previous approaches provide a concise summary of model dependencies
  - E.g., feature/concept importance for single prediction

- **Neuron interpretation** is a more fine-grained approach
- Aims to understand model's internal features
  - Understand individual neurons, filters, layers
  - Often produces visualizations (most useful for image models)

Today

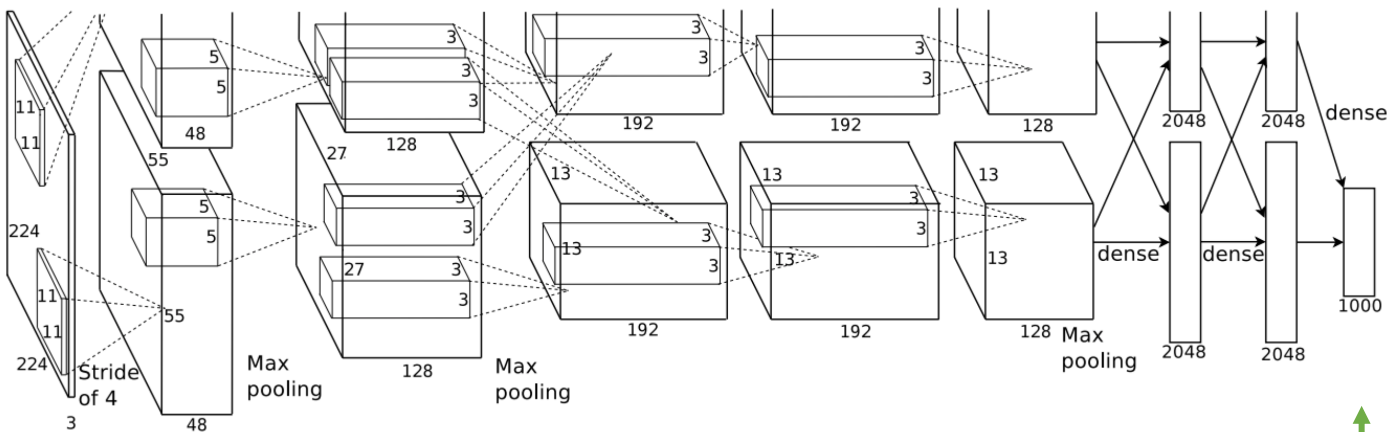
- Section 1
  - Concept-based explanations
- Section 2
  - Visualizing convolutional features
  - Feature visualization
  - Neuron Shapley

Background: ImageNet and AlexNet

- ImageNet = database of labeled images
  - Introduced by Fei-Fei Li's lab in 2009, turned into ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2010
  - Deng et al. "ImageNet: A large-scale hierarchical image database" (2009)
- In 2012, a CNN now called AlexNet won ILSVRC by a large margin
  - Top-5 test error rate of 15.3%, compared to 26.2% by the second-best entry
  - Krizhevsky et al., "ImageNet classification with deep convolutional neural networks" (2012)

AlexNet architecture

Internal activations



Image

Image dimensions:
h = 224, w = 224, c = 3

Classes

Krizhevsky et al., "ImageNet classification with deep convolutional neural networks" (2012)

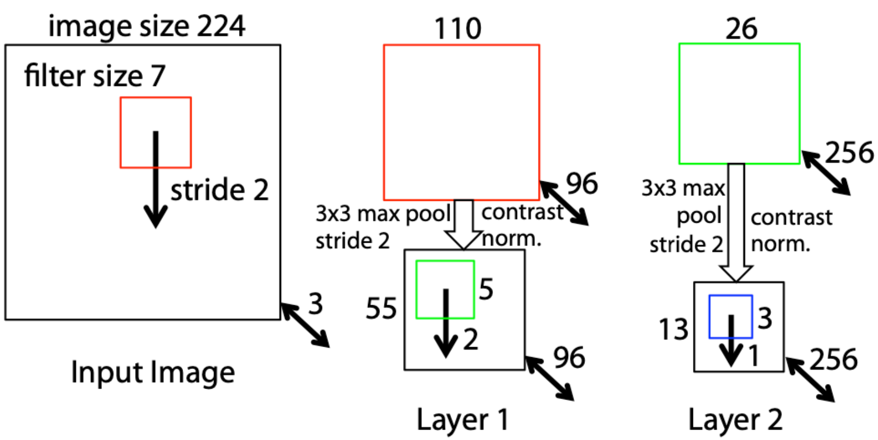
Understanding AlexNet

- After AlexNet, the ML community wanted to understand why/how CNNs worked so well
- To do so, Zeiler & Fergus (future ILSVRC winners) developed a procedure to interpret activations within CNN layers
  - Generated visualizations for individual samples
  - Collectively, these helped understand a model's convolutional filters

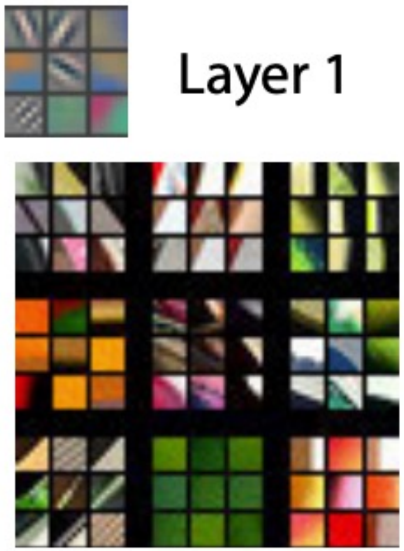
Zeiler & Fergus, "Visualizing and understanding convolutional networks" (2014)

Visualizing layer 1

Can directly examine learned filters



7x7x3 filters (9/96 total filters)



Patches that maximize each filter

Zeiler & Fergus, "Visualizing and understanding convolutional networks" (2014)

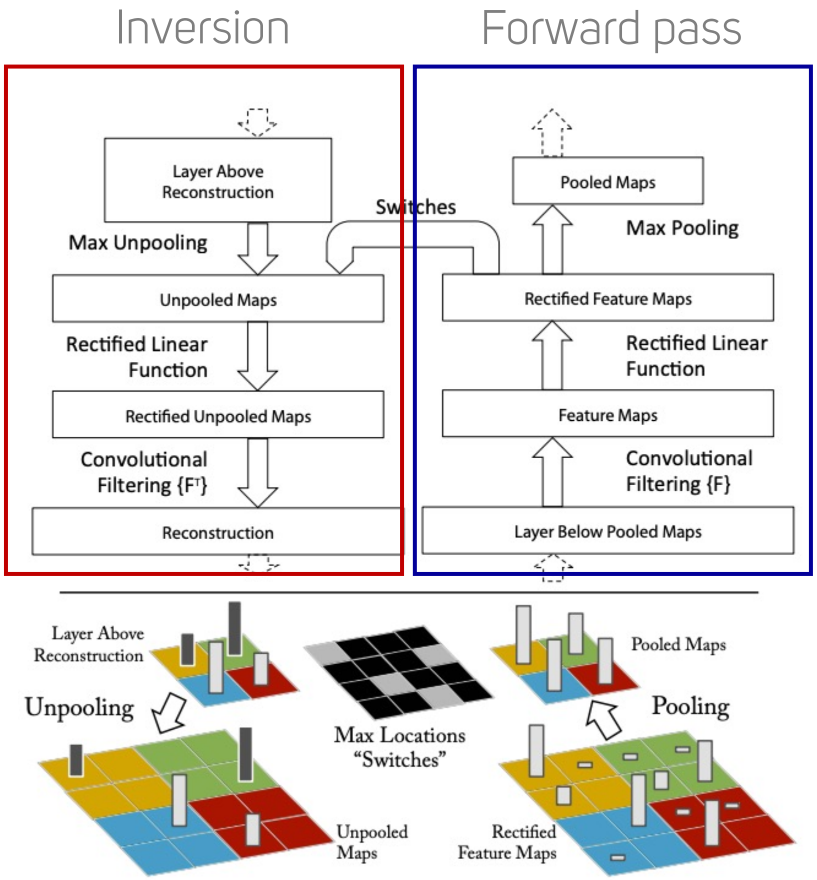
Visualizing later layers

- The previous case is special because we can directly visualize first-layer filters
- Idea for subsequent layers:
  - Propagate an image through the network to a certain layer
  - Set all activations to zero except for one
  - “Invert” each operation in the network to return to input pixels
    - Problem: operations are not truly invertible

Zeiler & Fergus, "Visualizing and understanding convolutional networks” (2014)

How to invert operations?

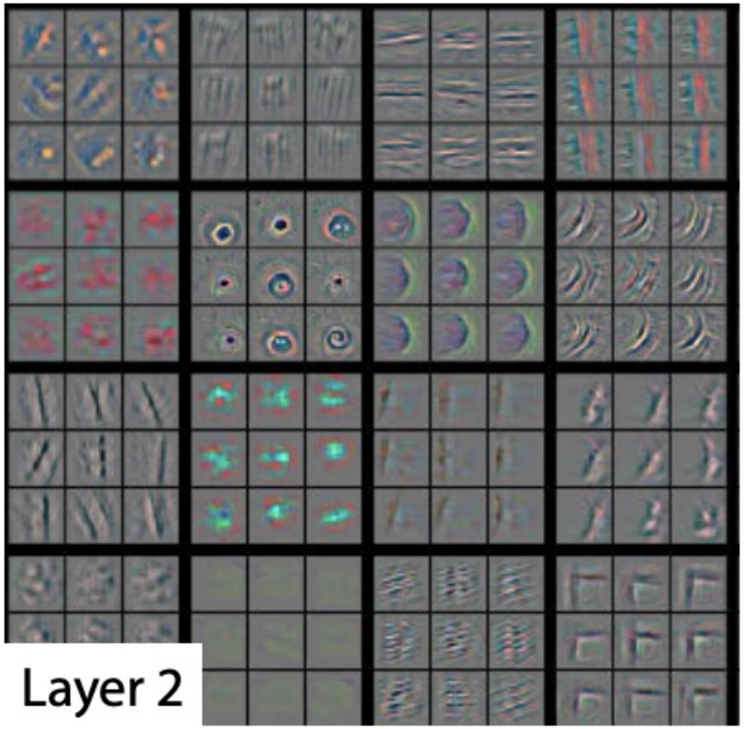
- **Conv. filters:** transposed versions of the same filters (“deconvolution”)
- **Unpooling:** max pooling is non-invertible, but we can approximately invert by keeping track of the max switches
- **ReLU:** pass reconstructed signal through ReLU



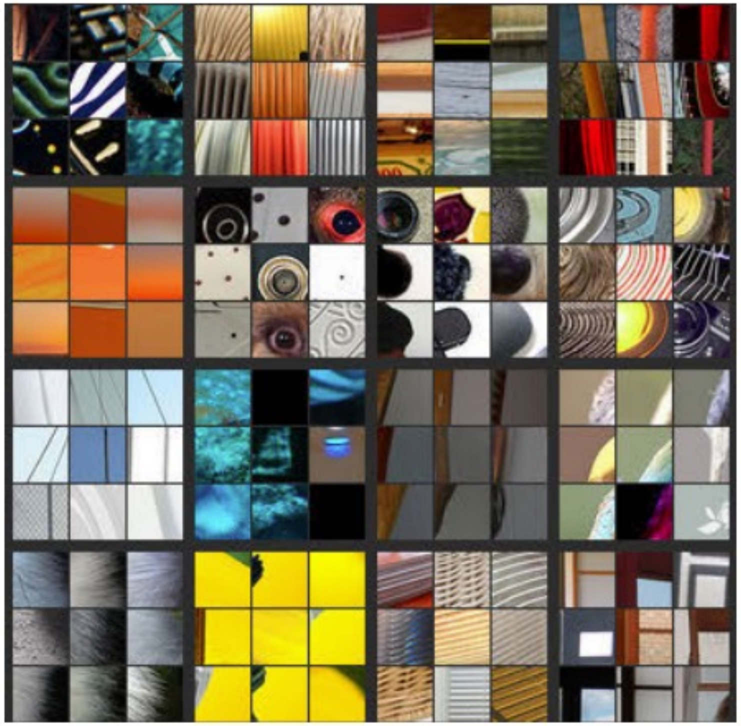
Visualizing layer 2

Each 3x3 group is a single neuron (9 patches with largest activation value)

Patch size corresponds to neuron's receptive field

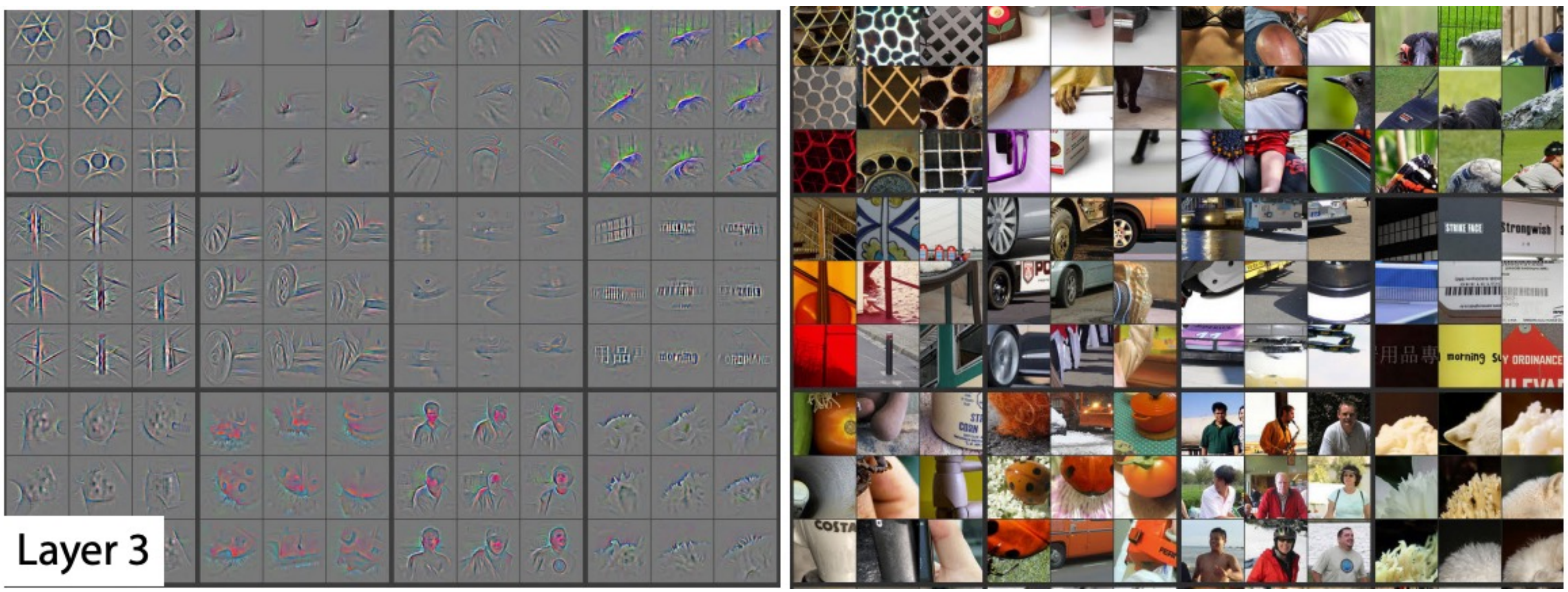


Activation-based reconstructions



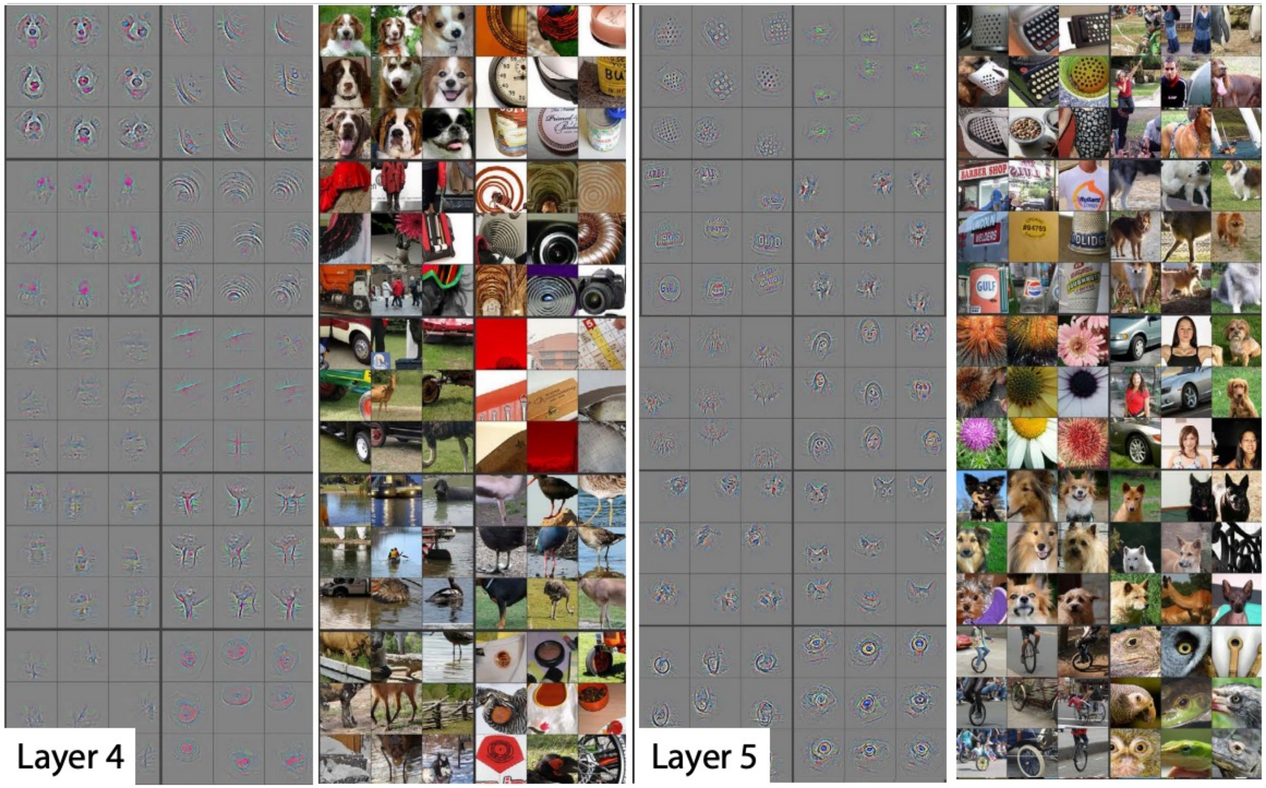
Corresponding patches

Visualizing layer 3



Zeiler & Fergus, "Visualizing and understanding convolutional networks" (2014)

Visualizing layers 4–5



Zeiler & Fergus, "Visualizing and understanding convolutional networks" (2014)

Visualizing layer 5

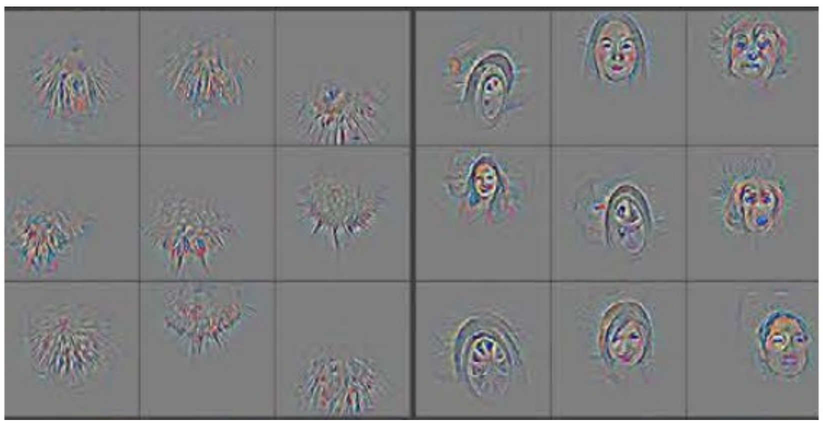
Text?

Dog torsos?

Zeiler & Fergus, "Visualizing and understanding convolutional networks" (2014)

Visualizing layer 5

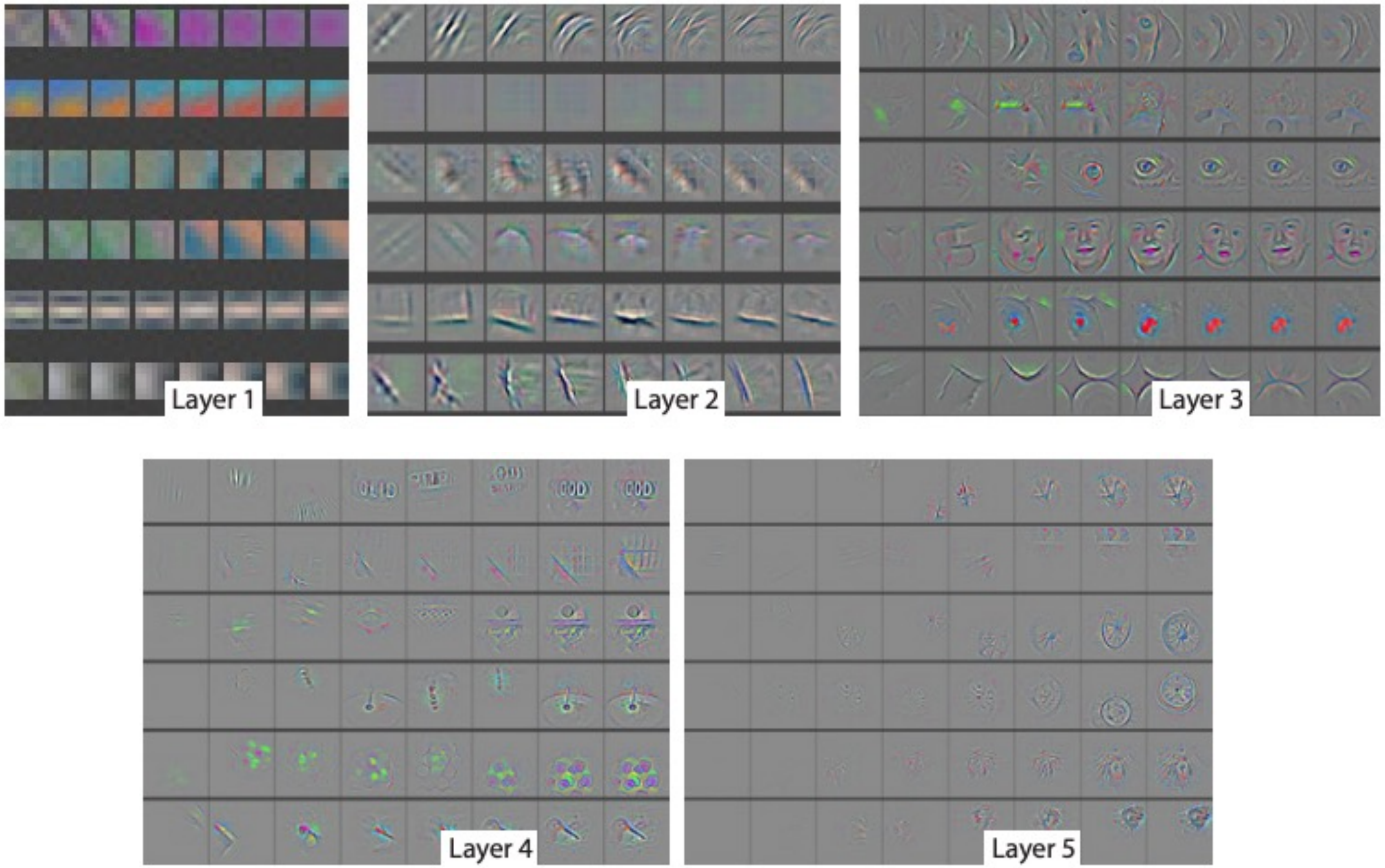
Spiky things? Faces or wheels?



Zeiler & Fergus, "Visualizing and understanding convolutional networks" (2014)

Feature evolution over training

Epochs (1, 2, 5, 10, 20, 30, 40, 64)



Remarks

- **Pros:**
  - These visualizations are fast and straightforward
- **Cons:**
  - Inversions are approximate (potentially low-quality)
  - Does not generalize to arbitrary DNN operations
  - Visualizations can be hard to interpret

- One of the most cited papers on this topic, but not the first:
  - See Erhan et al, "Visualizing higher-layer features of a deep network" (2009)

Today

- Section 1
  - Concept-based explanations
- Section 2
  - Visualizing convolutional features
  - Feature visualization
  - Neuron Shapley

Feature visualization

- Olah et al. is an interactive article about feature visualization techniques
  - Examples on GoogLeNet, trained on the ImageNet dataset
- This work visualizes learned features by **activation maximization**
  - Solves a per-neuron optimization problem
  - Identifies prototypical examples that activate each neuron within the model

Olah et al., "Feature visualization" (2017)

Activation maximization

- Neural networks are usually differentiable with respect to their inputs
- If model parameters θ are fixed and $h_{ij}(\theta, x)$ is activation of node i from layer j, then we want:

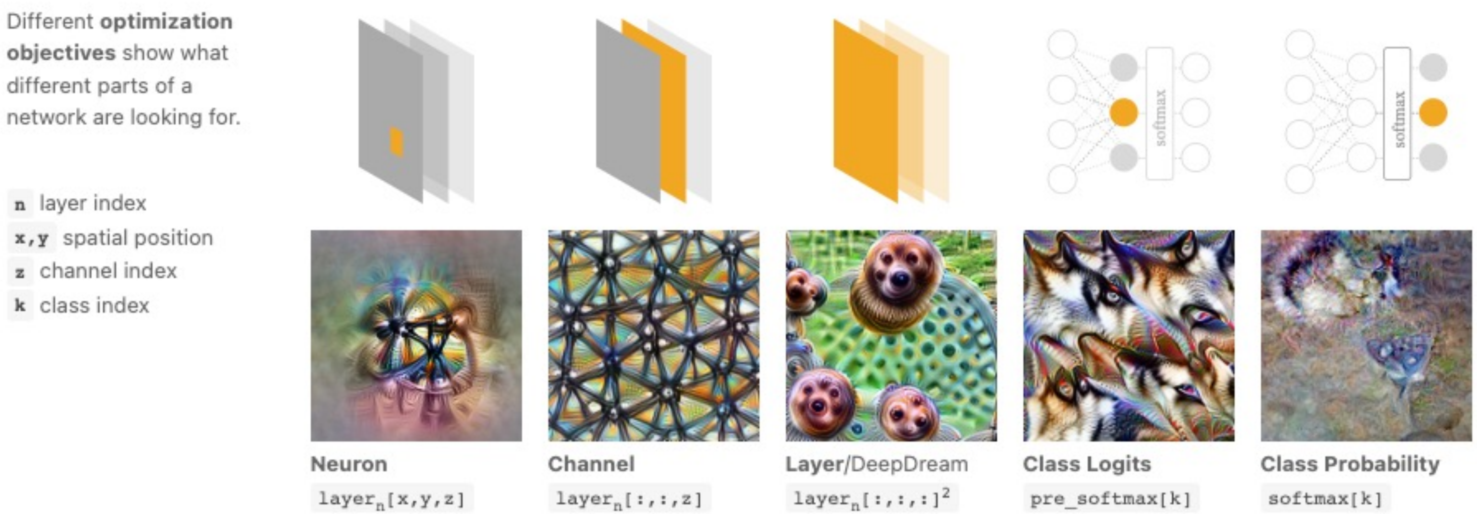
$$x^* = \mathrm{argmax}_x \, h_{ij}(\theta, x)$$

- A difficult optimization problem, but we can find a local optimum

Erhan et al., "Visualizing higher-layer features of a deep network" (2009)

Optimization objectives

- We can use arbitrary optimization objectives:



Olah et al., "Feature visualization" (2017)

Optimization approach

- How to solve the optimization problem?

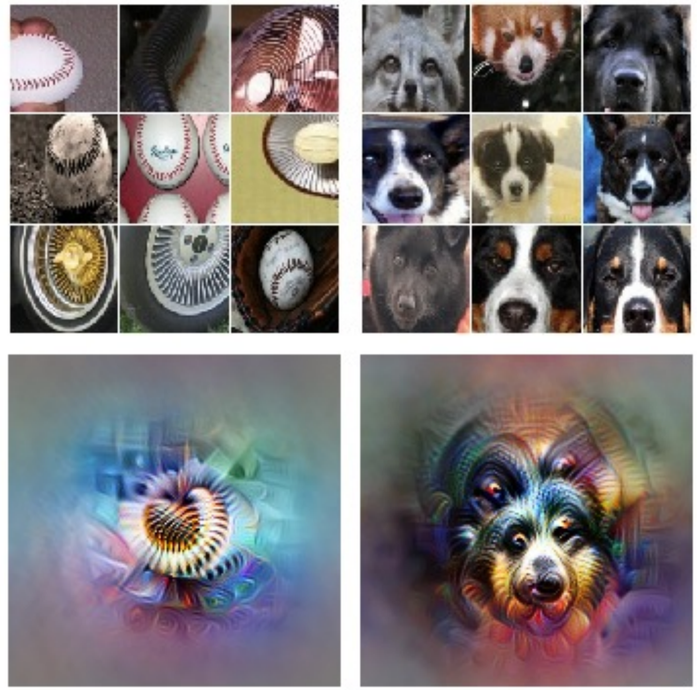
$$x^* = \mathrm{argmax}_x \, h_{ij}(\theta, x)$$

- Two main options:
  - Iterate over dataset examples
    - Solution is guaranteed to be a real example
  - Gradient descent
    - Update with $x^{(t+1)} = x^{(t)} - \alpha \nabla_x h_{ij}\left(\theta, x^{(t)}\right)$
    - Not guaranteed to be a real example

Optimization vs. dataset examples

Dataset examples
On-manifold examples

Grad. descent optimization
Off-manifold examples



Baseball—or stripes?
mixed4a, Unit 6

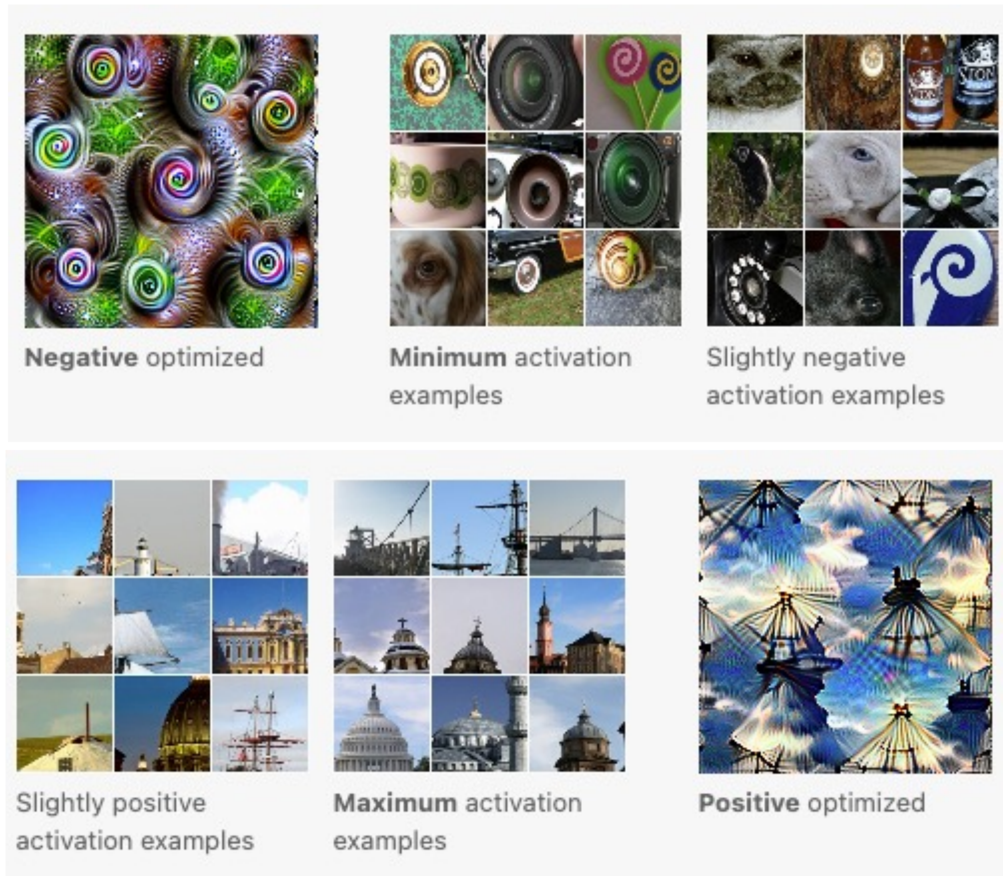
Animal faces—or snouts?
mixed4a, Unit 240

Optimization can potentially separate correlated attributes

Olah et al., "Feature visualization" (2017)

More examples

Investigating a single neuron
(layer mixed 4a, unit 492)



Olah et al., "Feature visualization" (2017)

Optimization challenges

- Naively optimizing neuron activation leads to poor solutions
  - Noisy, high-frequency, checkerboard patterns
  - Possibly due to strided convolutions and pooling operations
  - Semantically meaningless (like adversarial examples)

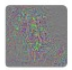
Olah et al., "Feature visualization" (2017)

Regularization approaches

- Frequency regularization:
  - Penalize high-frequency patterns (e.g., via total variation norm)
- Transformation robustness:
  - Find inputs that maximize activation even under small transformations (jitter, rotations, scaling)
- Learned priors:
  - Learn a model of the real data and use it to generate realistic samples (e.g., GAN)

Olah et al., "Feature visualization" (2017)

Regularization approaches

		Unregularized	Frequency Penalization	Transformation Robustness	Learned Prior	Dataset Examples
	**Erhan, *et al.*, 2009** [3] Introduced core idea. Minimal regularization.	■	□	□	□	□
	**Szegedy, *et al.*, 2013** [11] Adversarial examples. Visualizes with dataset examples.	■	□	□	□	■
	**Mahendran & Vedaldi, 2015** [7] Introduces total variation regularizer. Reconstructs input from representation.	□	■	□	□	□
	**Nguyen, *et al.*, 2015** [14] Explores counterexamples. Introduces image blurring.	□	■	□	□	□
	**Mordvintsev, *et al.*, 2015** [4] Introduced jitter & multi-scale. Explored GMM priors for classes.	□	□	■	■	□
	**Øygard, *et al.*, 2015** [15] Introduces gradient blurring. (Also uses jitter.)	□	■	■	□	□
	**Tyka, *et al.*, 2016** [16] Regularizes with bilateral filters. (Also uses jitter.)	□	■	■	□	□
	**Mordvintsev, *et al.*, 2016** [17] Normalizes gradient frequencies. (Also uses jitter.)	□	■	■	□	□
	**Nguyen, *et al.*, 2016** [18] Paramaterizes images with GAN generator.	□	□	□	■	□
	**Nguyen, *et al.*, 2016** [10] Uses denoising autoencoder prior to make a generative model.	□	□	□	■	□

Olah et al., "Feature visualization" (2017)

Regularization strength

- Strong regularization leads to more realistic examples, but it can introduce misleading correlations
  - Using dataset examples is very strong regularization
  - However, hard to tell if model relies on a baseball's shape, color, strings; or a dog's ears, nose, eyes
- Weak regularization avoids misleading correlations, but may lead to noisy images

Olah et al., "Feature visualization" (2017)

Remarks

- **Pros:**
  - Activation maximization is general, only requires that the model is differentiable w.r.t. its inputs
  - Cool visualizations
- **Cons:**
  - Can be difficult to interpret optimized images
  - Neurons may not correspond to simple, human-interpretable concepts
  - Hyperparameters and regularization can be heuristic
  - Large number of neurons to interpret

Today

- Section 1
  - Concept-based explanations
- Section 2
  - Visualizing convolutional features
  - Feature visualization
  - Neuron Shapley

Motivation

- Neuron interpretation is often *ad hoc*, because we don't know which ones to investigate
- Neuron Shapley quantifies neuron importance while accounting for interactions

Ghorbani & Zou, "Neuron Shapley: Discovering the responsible neurons" (2020)

Neuron Shapley

- **Idea:** instead of finding important features, find important neurons (here, filters)
- Remove filters to produce subsets of the network
  - Fix each removed filter's output to its mean (based on a set of validation images)
- Then, evaluate model behavior based on a given metric $V(\cdot)$ (e.g., accuracy, loss, etc.)
- Find important ones using the Shapley value

Ghorbani & Zou, "Neuron Shapley: Discovering the responsible neurons" (2020)

Computation

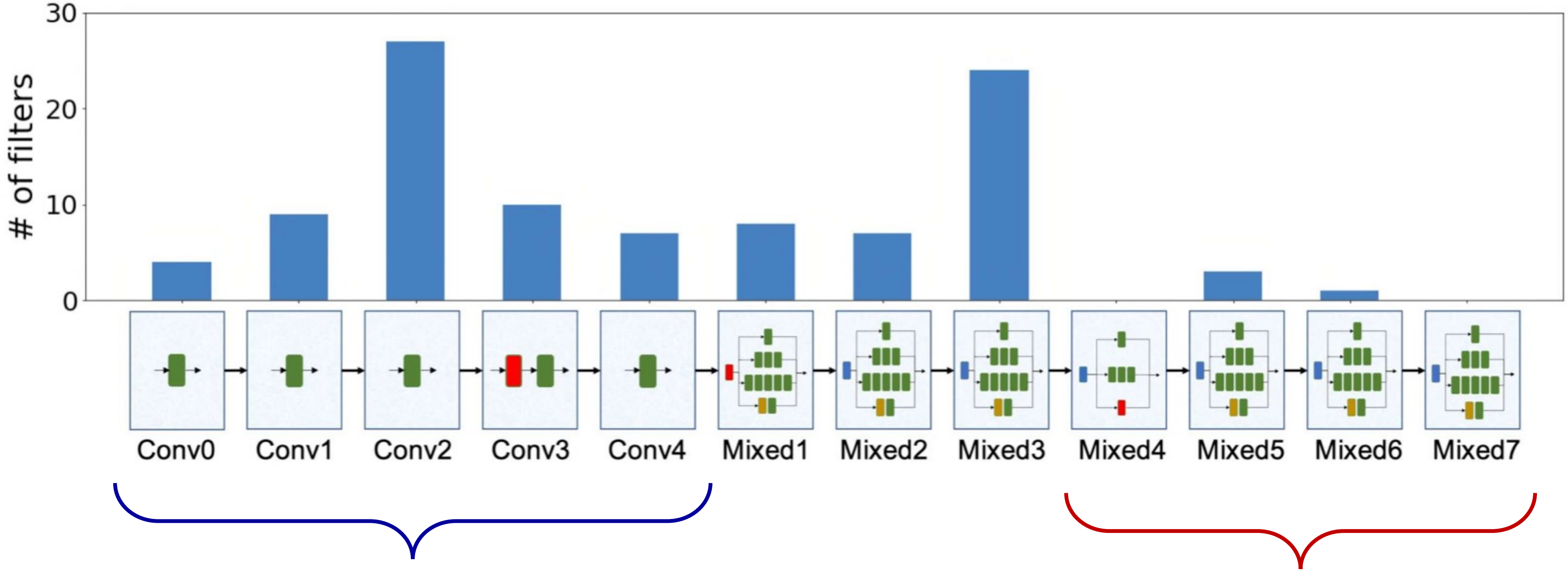
- Shapley values are difficult to calculate for games with many players
- Here, players are convolutional filters
  - >10k players, creates a challenging approximation problem
- The authors use a modified permutation sampling algorithm (recall HW1)

Ghorbani & Zou, "Neuron Shapley: Discovering the responsible neurons" (2020)

Quantitative evaluation

- Apply Neuron Shapley to Inception-v3 trained on ImageNet
  - Explain the 17K filters preceding the logits
  - Use the overall network accuracy as $V(\cdot)$
- Sparse explanations:
  - A small number of filters have largest importance
  - Removing top 10 filters dropped test accuracy from 74% to 38%
  - Removing top 20 dropped test accuracy to 8%

Distribution of important filters by layer

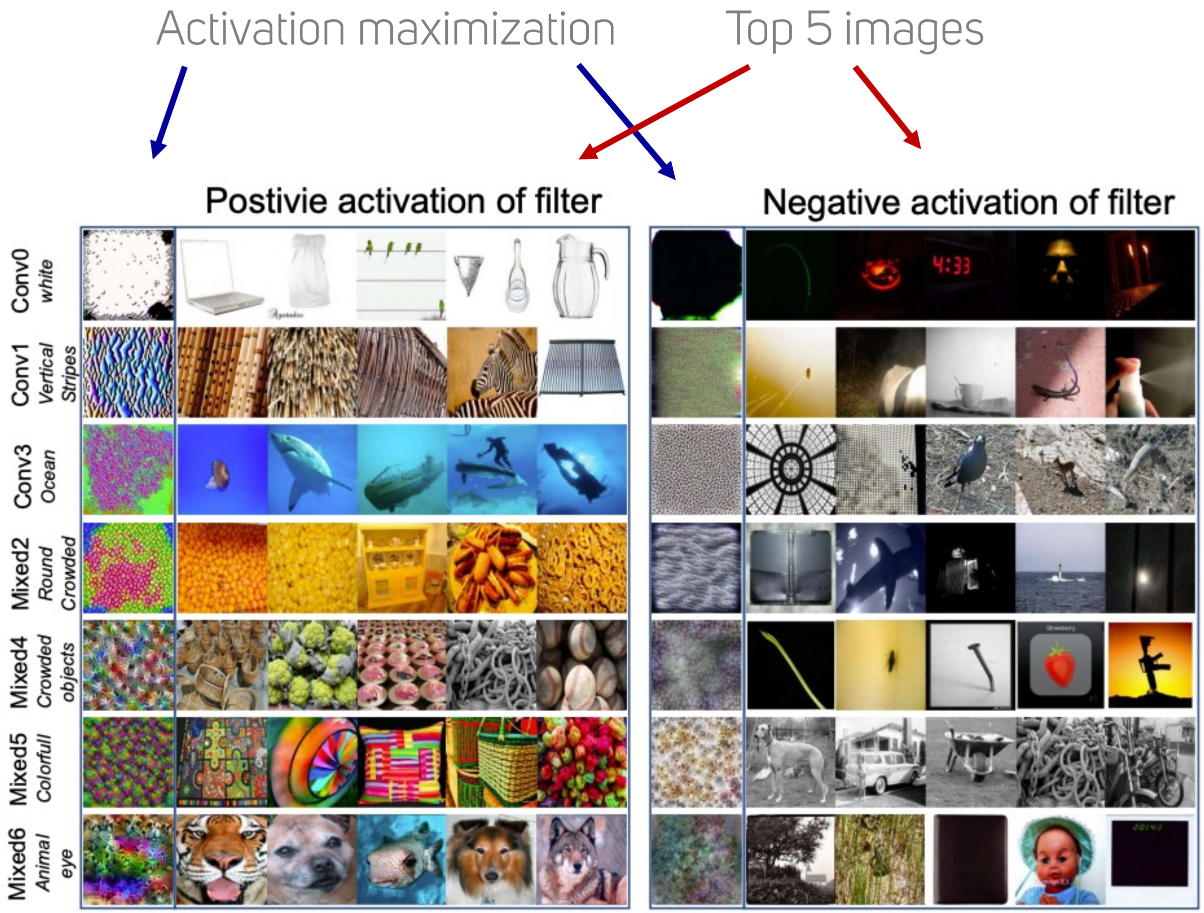


Early feature extraction layers have many important filters

Perhaps surprisingly, layers closest to classification do not

Visualizing important filters

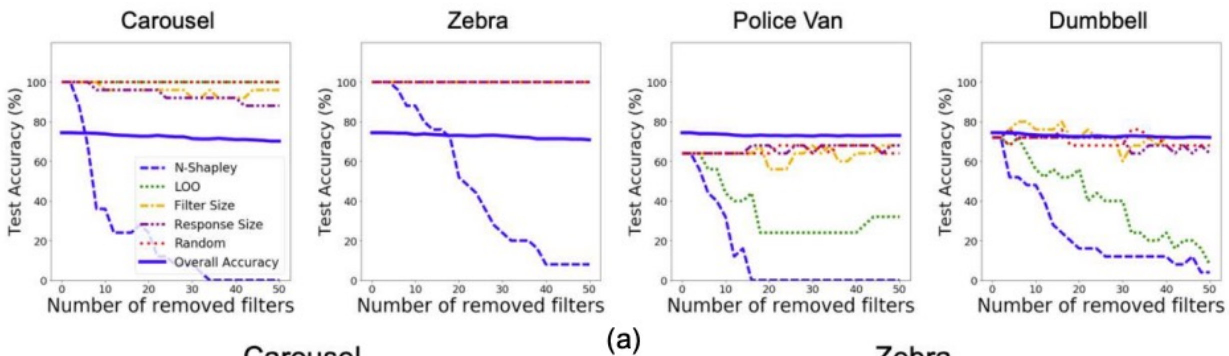
High Shapley value filters



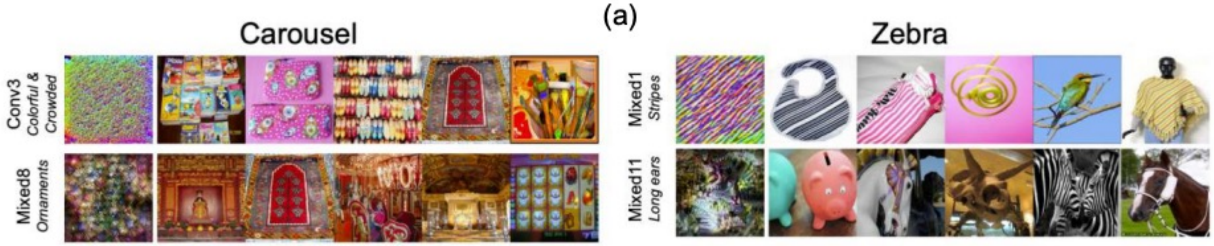
Class-specific experiments

- Here, $V(\cdot)$ is class-specific prediction accuracy

Ablation tests



Filter visualization



Other use cases

- Identifying filters to remove
- Filters that are responsible for biased prediction
  - Removing these filters increased gender classification accuracy for minorities
- Filters that are vulnerable to adversarial attacks
  - Removing filters that contribute to adversary's success lowers the attack success rate significantly

Ghorbani & Zou, "Neuron Shapley: Discovering the responsible neurons" (2020)

Remarks

- **Pros:**
  - Neuron Shapley identifies important neurons
  - Can identify neurons to visualize, remove
- **Cons:**
  - Extremely expensive (naively, exponential in number of neurons)
    - Authors improve computation using bandit algorithm
    - Find that 10k samples is sufficient (10k evaluations of V, where V itself requires many model evaluations)

Summary

- Multiple techniques for understanding individual neurons, based on…
  - Operation inversion
  - Visualizing highly activated examples
  - Activation maximization
- Quantifying neuron importance can help prioritize our analysis, and suggest model adjustments