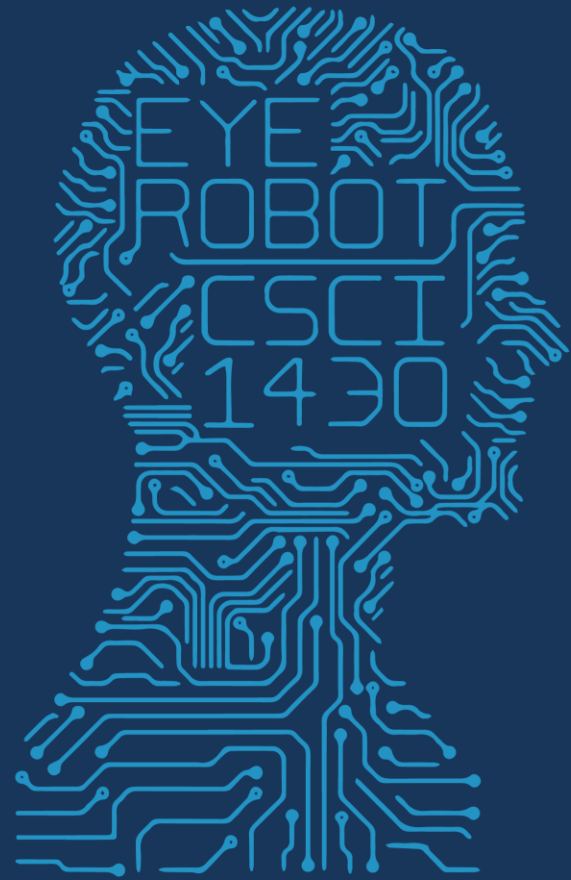




1950

FUTURE VISION



13 MARCH 2019

COMPUTER VISION



Simultaneous contrast

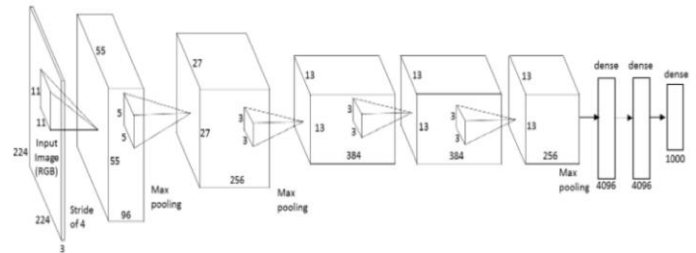






Training Neural Networks

- Build network architecture and define loss function
- Pick hyperparameters – learning rate, batch size
- Initialize weights + bias in each layer randomly

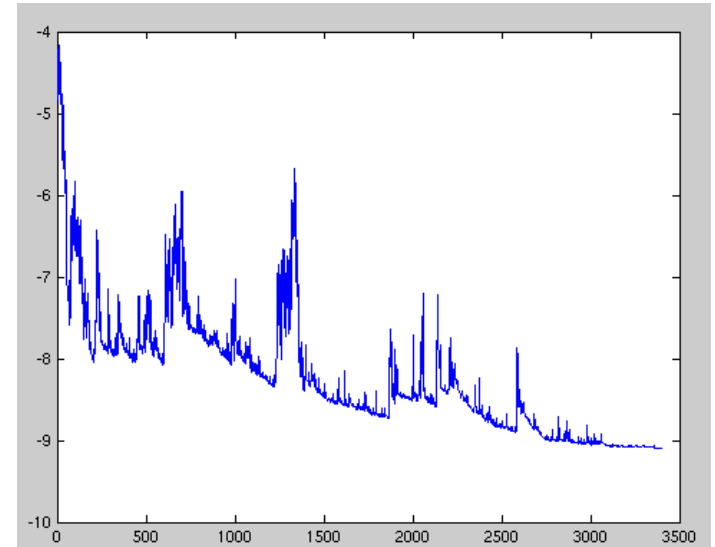


- While loss still decreasing
 - Shuffle training data
 - For each data point $i=1\dots n$ (maybe as mini-batch)
 - *Gradient descent*
 - Check validation set loss
- } “Epoch”

Stochastic Gradient Descent

Try to speed up processing with random training subsets

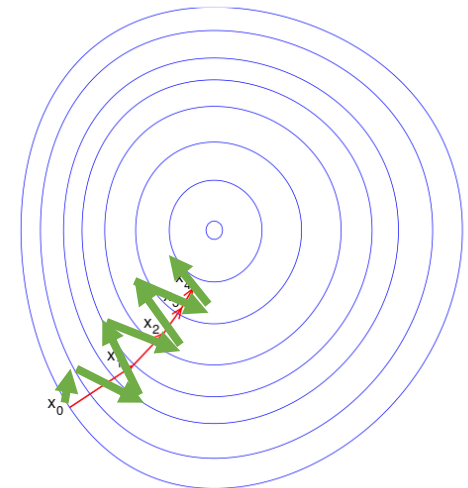
Loss will not always decrease (locally) as training data point is random, but converges over time.



Momentum

Gradient descent step size is weighted combination over time to dampen ping pong.

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \gamma \left(\alpha \left[\frac{\partial L}{\partial \boldsymbol{\theta}} \right]_{t-1} + \left[\frac{\partial L}{\partial \boldsymbol{\theta}} \right]_t \right)$$

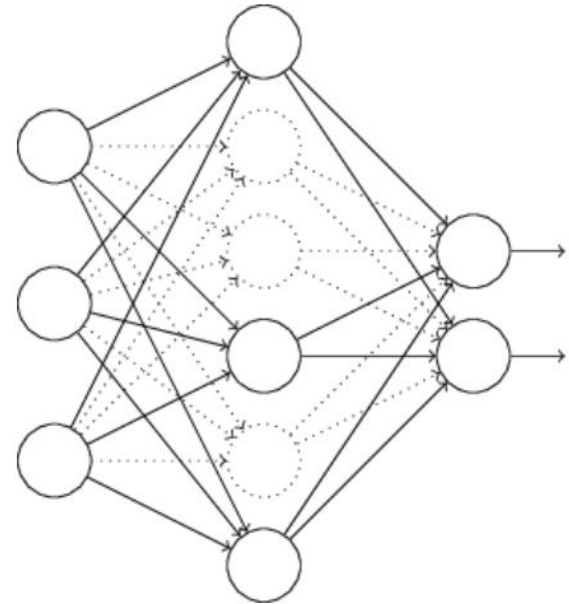


Regularization

- Penalize weights for simpler solution
 - Occam's razor

$$C = C_0 + \lambda \sum_w w^2,$$

- Dropout half of neurons for each minibatch
 - Forces robustness



But James...

...I thought we were going to treat machine learning like a black box? I like black boxes.

Deep learning is:

- a black box
- *also a black art.*
- Grad student
gradient descent : (



Why in Project 4 Q5 do we initialize the weights to zero?

Good question - a mistake on our part.

Setting zero weights makes all neurons equivalent as there is no difference in the gradient computed across neurons. Called “symmetric updates”.

Setting zero bias is OK; often we standardize the data beforehand (subtract mean, divide by std. dev) so a zero bias is a good initialization.

*Why in Project 4 Q5 do we initialize the weights to zero?
So why did it still train?*

$Wx = 0$ for the layer output

-> produces uniform softmax output (each class equally probable; i.e., on MNIST, 0.1)

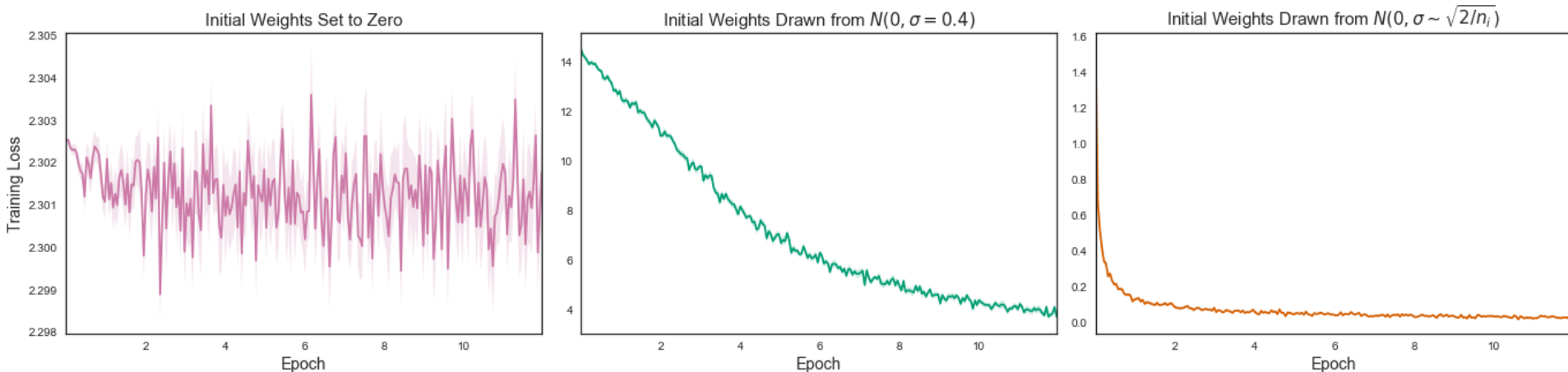
Gradient update rule and supervision label y_j still provides the right signal

p_j vs $1 - p_j$

...but all neurons not of class j receive same gradient update.

Why in Project 4 Q5 do we initialize the weights to zero?

In general, initialization is very important.



Good strategy: He et al. 2015:

For ReLU, draw random weights from Gaussian distribution with variance = $2 / \#$ inputs to layer

What is activation function for?

To allow multiple layers; to avoid resulting composition of linear functions collapsing to a single layer.

Difference between CNN and convolution in feature extraction?

No difference! Same operation [correlation/convolution]

Why do we shave off pixels?

We only use the valid region of convolution; typically no padding. Some recent works special case these edge convolutions.

Why multidimensional kernels?

We wish to convolve over the outputs of many other learned kernels -> 'integrating' information via weighted sum of previous layer outputs.

How to know which kernels to use in 2^{nd} + convolution layers?

Gradient descent + back propagation learns them.

How to set weights on fully connected layers?

Gradient descent + back propagation learns them.

What even is back propagation again?

Project 4 Q5 has some good references.

How do we decide the best parameters for network architecture?

For less complicated situations, we can use 'guess & check' [trial and error]. Is there any method?

'Grid search' -> trial and error

'Bayesian optimization' -> meta-learning; optimize the hyperparameters

General strategy:

Bottleneck -> extract information (kernel) and learn to squeeze representation into a smaller number of parameters.

I've heard about many more terms of jargon!

Skip connections

Residual connections

Batch normalization

...we'll get to these in a little while.

How can free will coexist with divine preordination?

Ah yes.

Related works:

- The Ontological Argument
- The Problem of Evil
- Ship of Theseus / Sorites Paradox
- What is Art
- $0.999\dot{9} = 1$
- Chinese Room AI

Short cuts to AI

With billions of images on the web, it's often possible to find a close nearest neighbor.

We can shortcut hard problems by “looking up” the answer, stealing the labels from our nearest neighbor.



So what is intelligence?

Weak AI:

The simulation of a 'mind' is a model for the 'mind'.

Strong AI:

The simulation of a 'mind' is a 'mind'.

Chinese Room experiment, John Searle (1980)

If a machine can convincingly simulate an intelligent conversation, does it understand?



Chinese Room experiment, John Searle (1980)

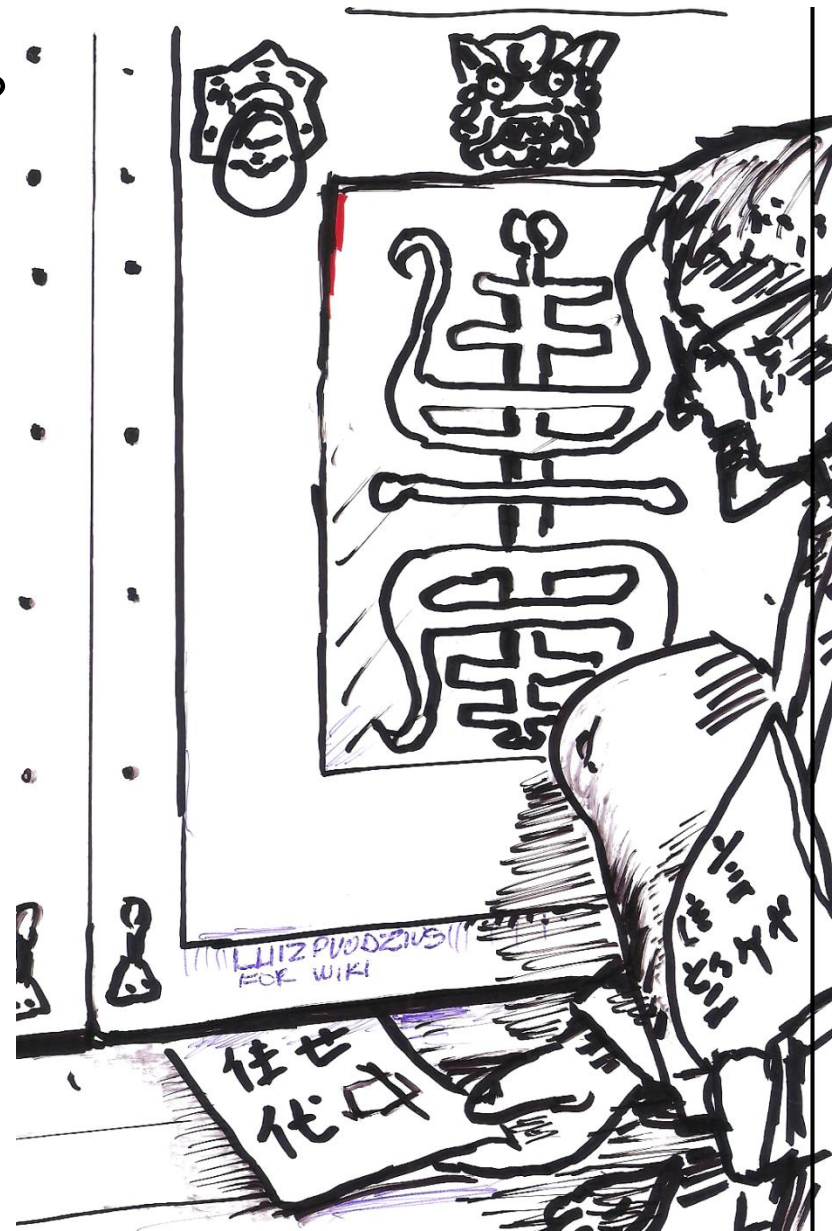
If a machine can convincingly simulate an intelligent conversation, does it understand?

Searle imagines himself in a room, acting as a computer by manually executing a program that convincingly simulates the behavior of a native Chinese speaker.

Most of the discussion consists of attempts to refute it.

"The overwhelming majority," notes *BBS* editor Stevan Harnad, "still think that the Chinese Room Argument is dead wrong."

The sheer volume of the literature that has grown up around it inspired Pat Hayes to quip that the field of cognitive science ought to be redefined as "the ongoing research program of showing Searle's Chinese Room Argument to be false."





Yann LeCun

October 23 at 9:58pm · 🌐

Questions from the piece:

Q1. Does the Chinese Room argument prove the impossibility of machine consciousness?

A1: Hell no. ... [See More](#)



Can Machines Become Moral?

The question is heard more and more often, both from those who think that machines cannot become moral, and who think that to believe otherwise is a dangerous illusion, and from those who think that machines must become moral,...

BIGQUESTIONSONLINE.COM | BY DON HOWARD

   You and 156 others

30 Comments 20 Shares

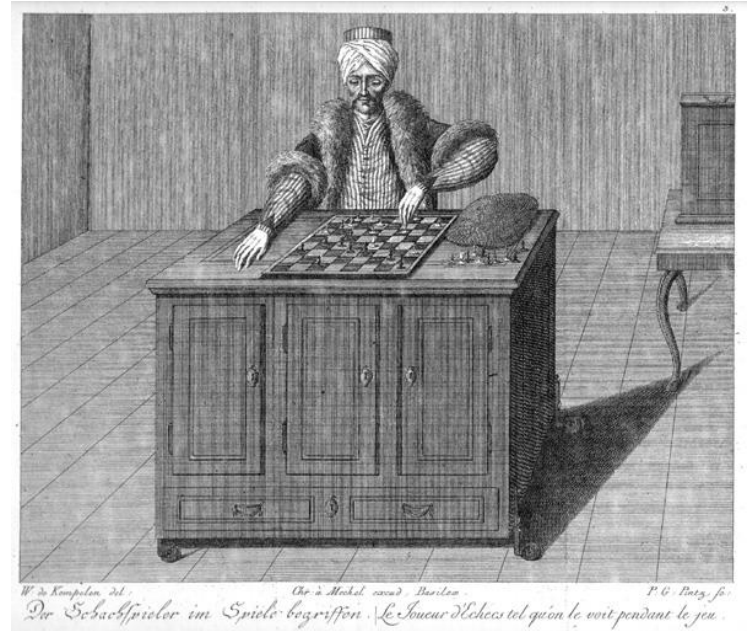
 Like

 Comment

 Share

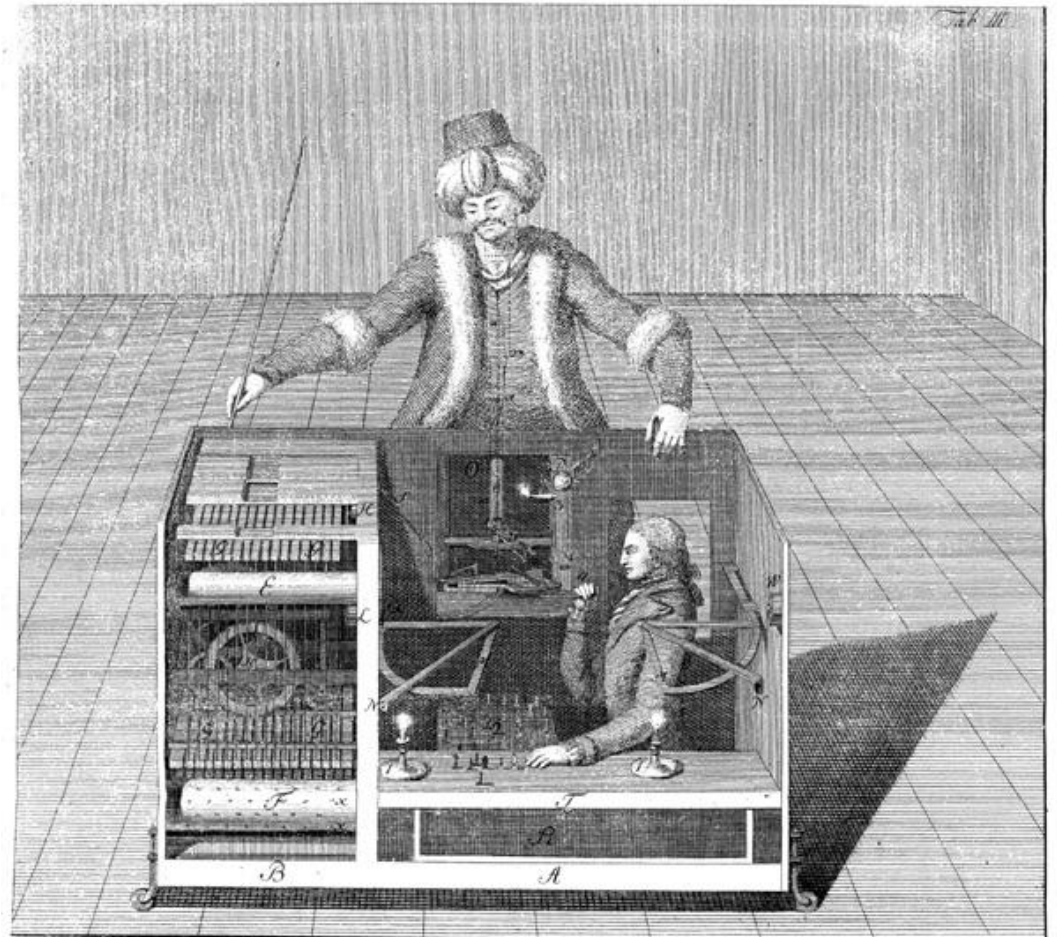
Mechanical Turk

- von Kempelen, 1770.
- Robotic chess player.
- Clockwork routines.
- Magnetic induction (not vision)
- Toured the world; played Napoleon Bonaparte and Benjamin Franklin.



Mechanical Turk

- It was all a ruse!
- Ho ho ho.



"Can machines fly?"

Yes; aeroplanes exist.

"Can machines fly like a bird?"

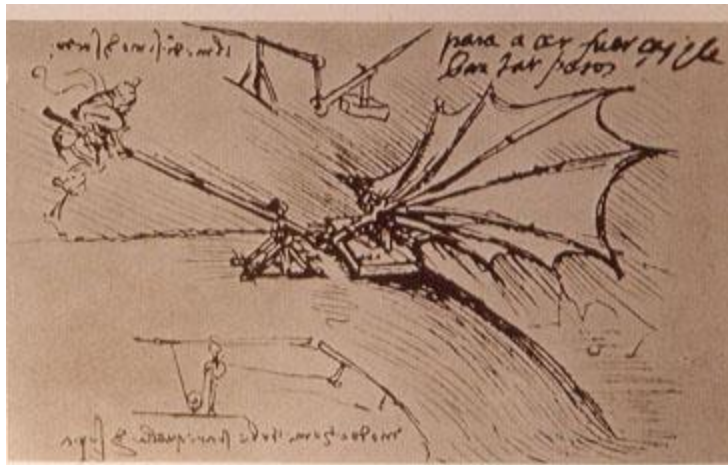
No, because aeroplanes don't flap.

"Can machines perceive?"

"Can machines understand?"

Are these question like the first, or like the second?

Ornithopters



Festo SmartBird [2011]

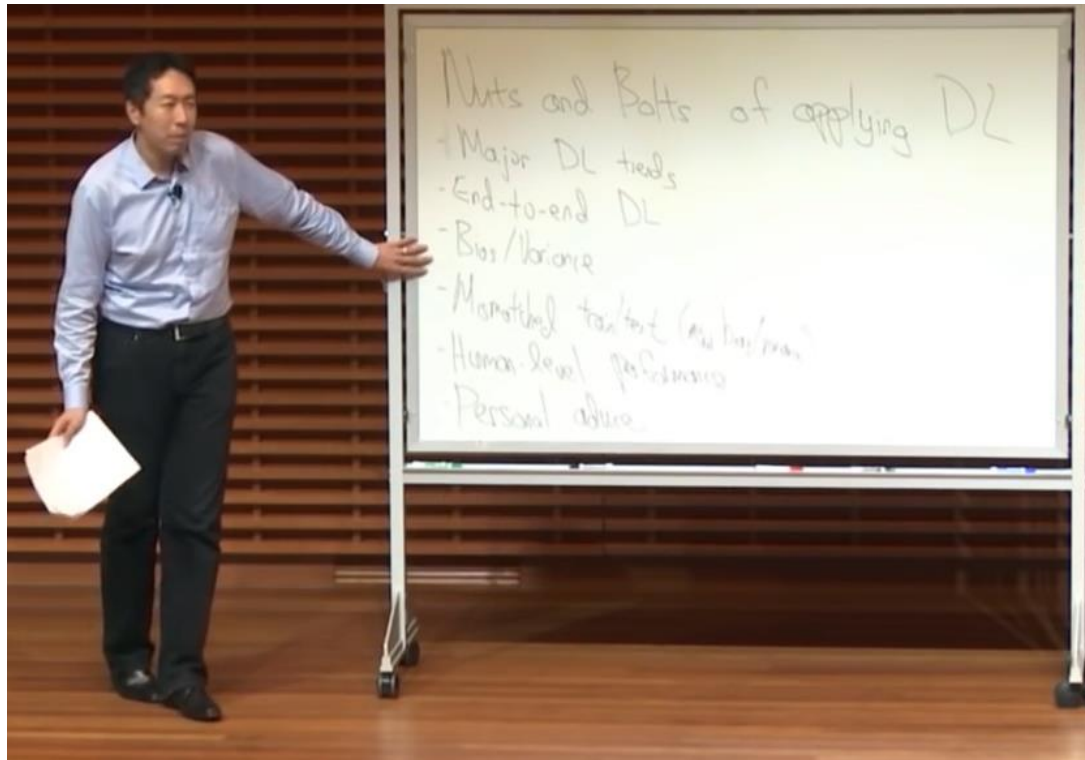


When something is not working...

...how do I know what to do next?

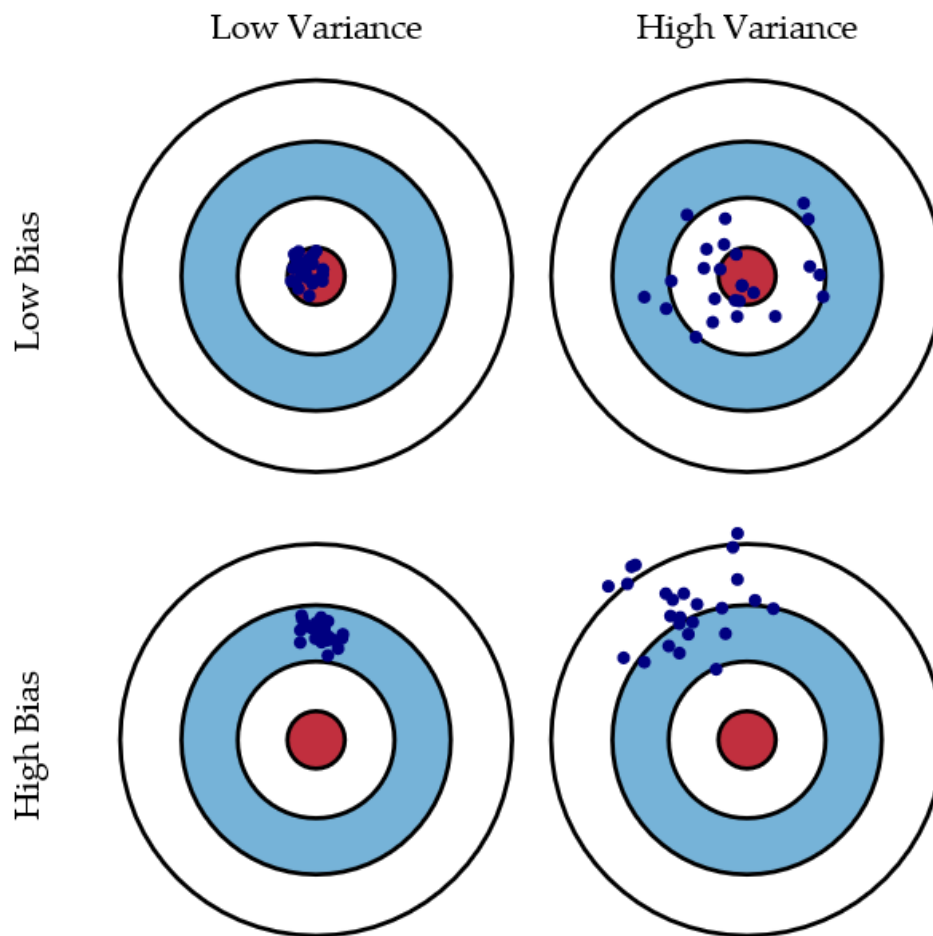
The Nuts and Bolts of Building Applications using Deep Learning

- Andrew Ng - NIPS 2016
- <https://youtu.be/F1ka6a13S9I>

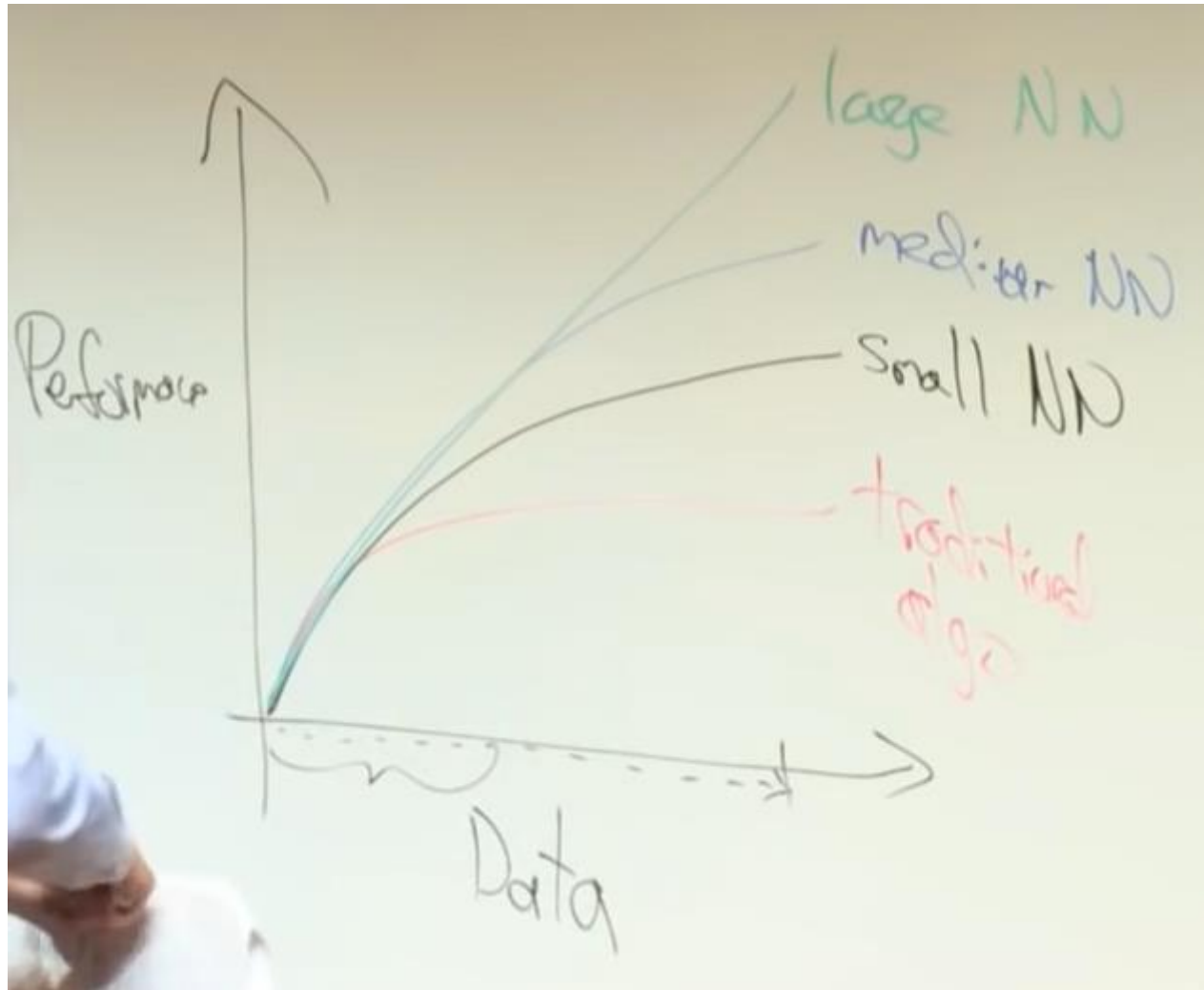


Bias/variance trade-off

"It takes surprisingly long time to grok bias and variance deeply, but people that understand bias and variance deeply are often able to drive very rapid progress." --Andrew Ng



Bias = accuracy
Variance = precision



Go collect a dataset

- Most important thing:
 - Training data must represent target application!

- Take all your data
 - 60% training
 - 40% testing
 - 20% testing
 - 20% validation (or 'development')

Properties

- Human level error = 1%
- Training set error = 10%
- Validation error = 10.2%
- Test error = 10.4%

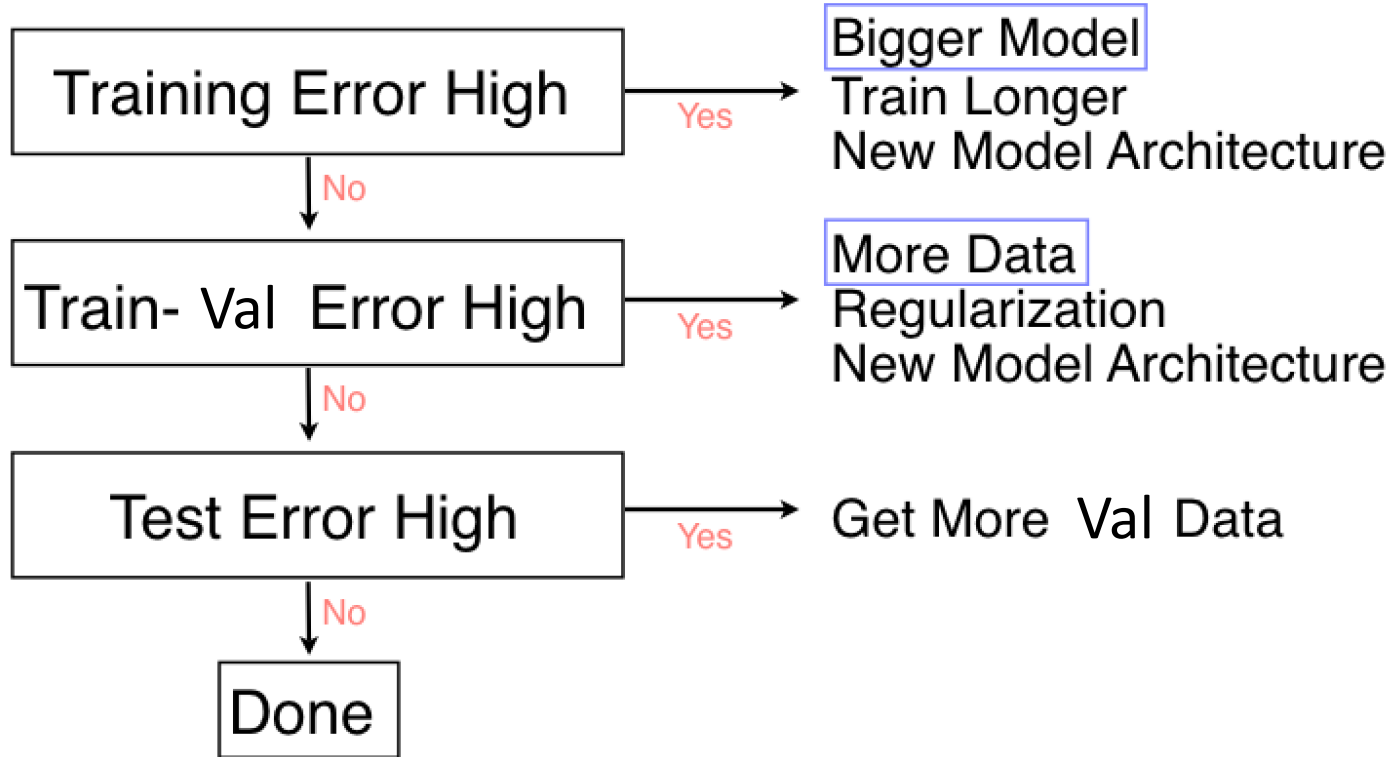


“Bias”

“Variance”

Overfitting to
validation

The Nuts and Bolts of Building Applications Using Deep Learning



My Neural Network isn't working! What should I do?

Created on Aug. 19, 2017, 5:56 p.m.

So you're developing the next great breakthrough in deep learning but you've hit an unfortunate setback: your neural network isn't working and you have no idea what to do. You go to your boss/supervisor but they don't know either - they are just as new to all of this as you - so what now?

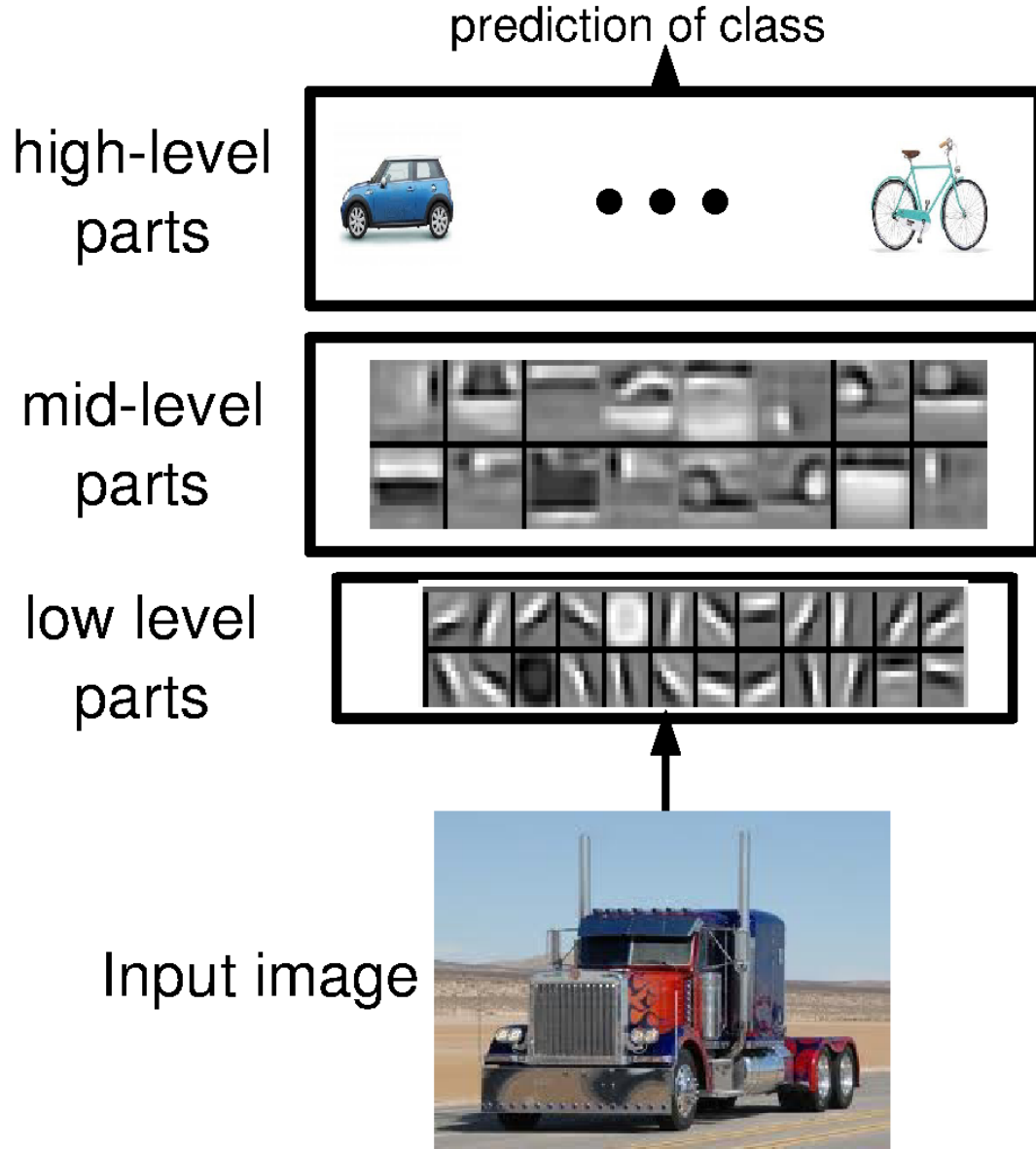
Well luckily for you I'm here with a list of all the things you've probably done wrong and compiled from my own experiences implementing neural networks and supervising other students with their projects:

1. [You Forgot to Normalize Your Data](#)
2. [You Forgot to Check your Results](#)
3. [You Forgot to Preprocess Your Data](#)
4. [You Forgot to use any Regularization](#)
5. [You Used a too Large Batch Size](#)
6. [You Used an Incorrect Learning Rate](#)
7. [You Used the Wrong Activation Function on the Final Layer](#)
8. [Your Network contains Bad Gradients](#)
9. [You Initialized your Network Weights Incorrectly](#)
10. [You Used a Network that was too Deep](#)
11. [You Used the Wrong Number of Hidden Units](#)

Daniel Holden

<http://theorangeduck.com/page/neural-network-not-working>

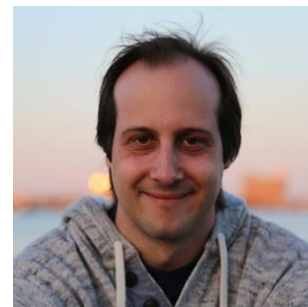
Interpretation



- distributed representations
- feature sharing
- compositionality

Object Detectors Emerge in Deep Scene CNNs

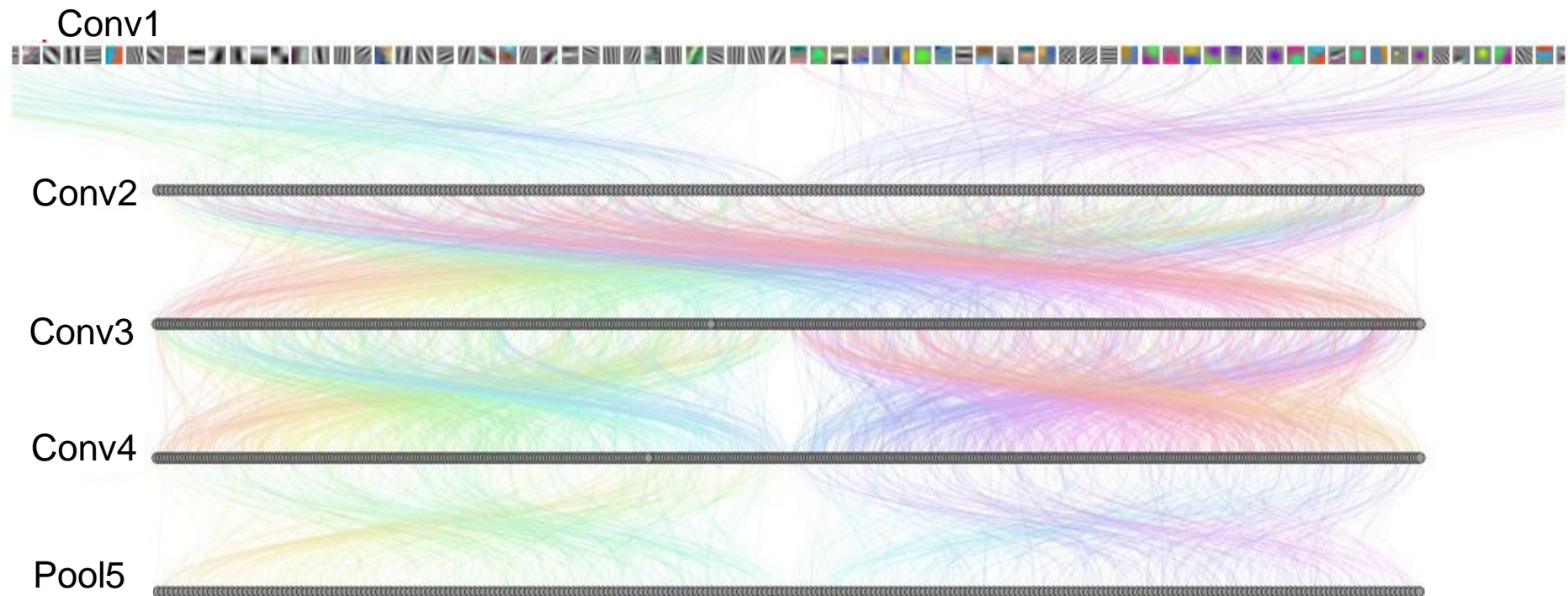
Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba



Massachusetts Institute of Technology

How Objects are Represented in CNN?

CNN uses **distributed code** to represent objects.



Estimating the Receptive Fields

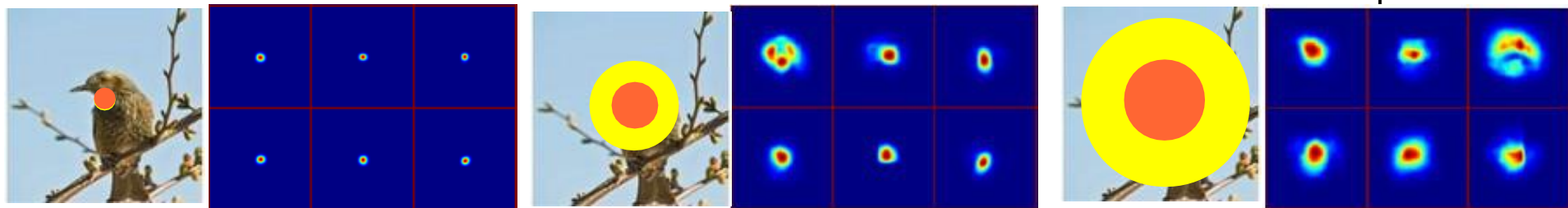
Estimated receptive fields

Actual size of RF is much smaller than the theoretic size

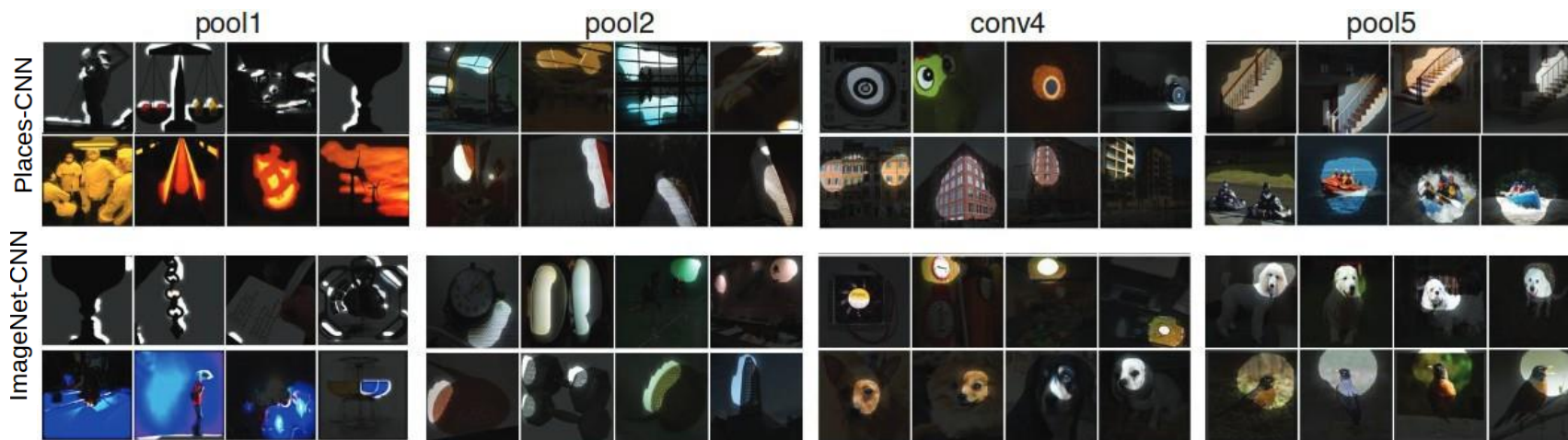
pool1

conv3

pool5



Segmentation using the RF of Units



More semantically meaningful

Annotating the Semantics of Units

Top ranked segmented images are cropped and sent to Amazon Turk for annotation.

Task 1

Word/Short description:

lower

Task 2

Mark (by clicking on them) the images which don't correspond to the short description you just wrote



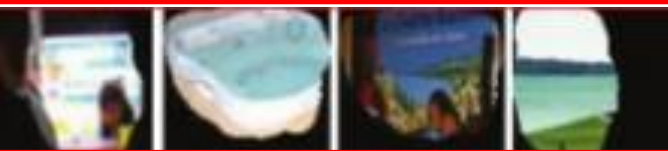
Task 3

Which category does your short description mostly belong to?

- Scene (kitchen, corridor, street, beach, ...)
- Region or surface (road, grass, wall, floor, sky, ...)
- Object (bed, car, building, tree, ...)
- Object part (leg, head, wheel, roof, ...)
- Texture or material (striped, rugged, wooden, plastic, ...)
- Simple elements or colors (vertical line, curved line, color blue, ...)

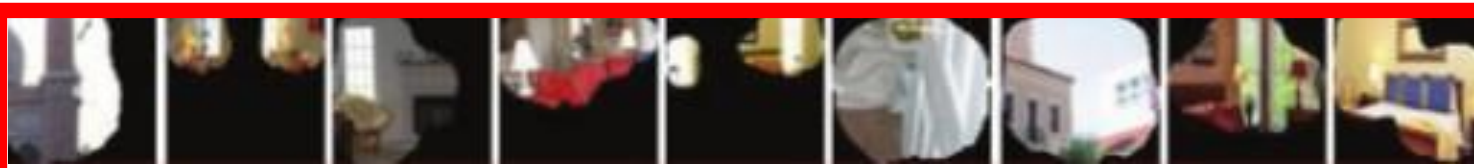
Annotating the Semantics of Units

Pool5, unit 76; Label: ocean; Type: scene; Precision: 93%



Annotating the Semantics of Units

Pool5, unit 13; Label: Lamps; Type: object; Precision: 84%



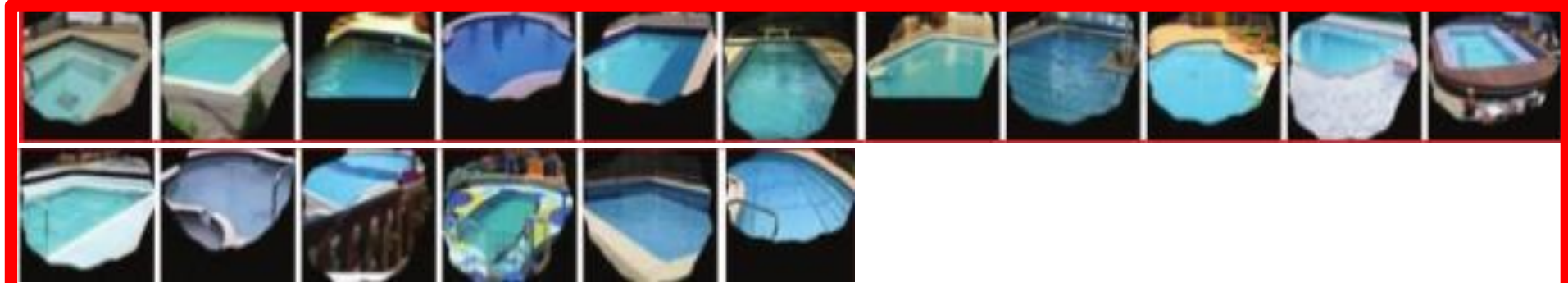
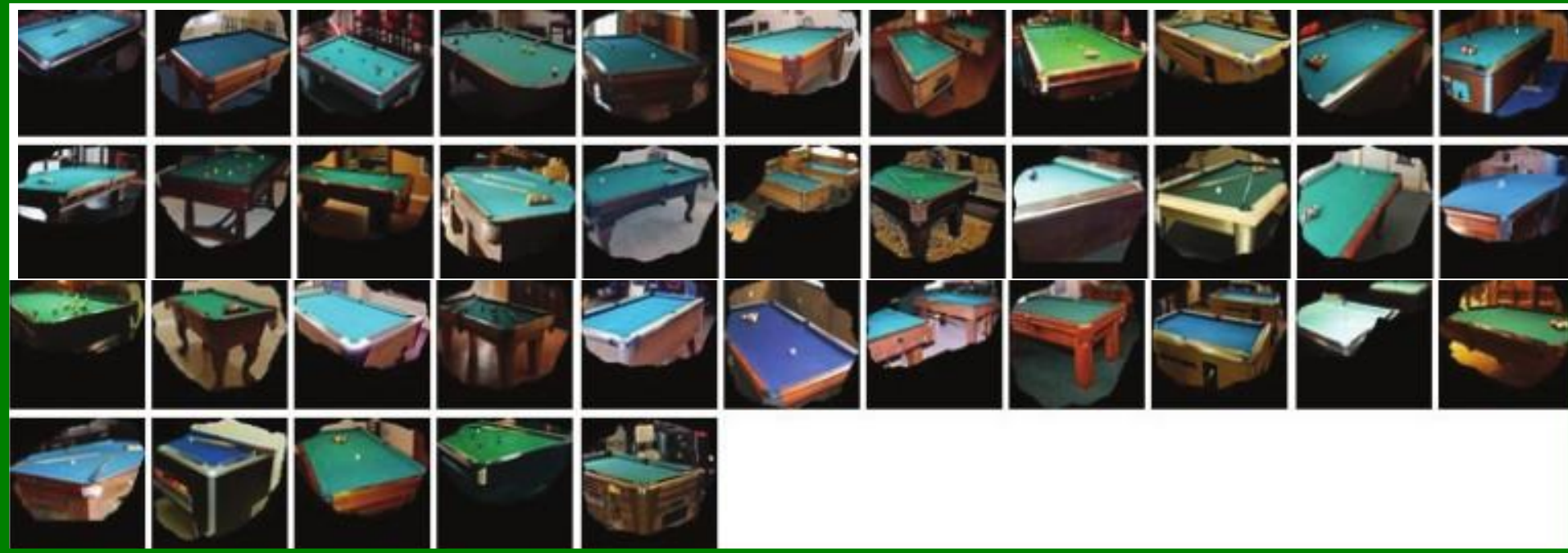
Annotating the Semantics of Units

Pool5, unit 77; Label:legs; Type: object part; Precision: 96%



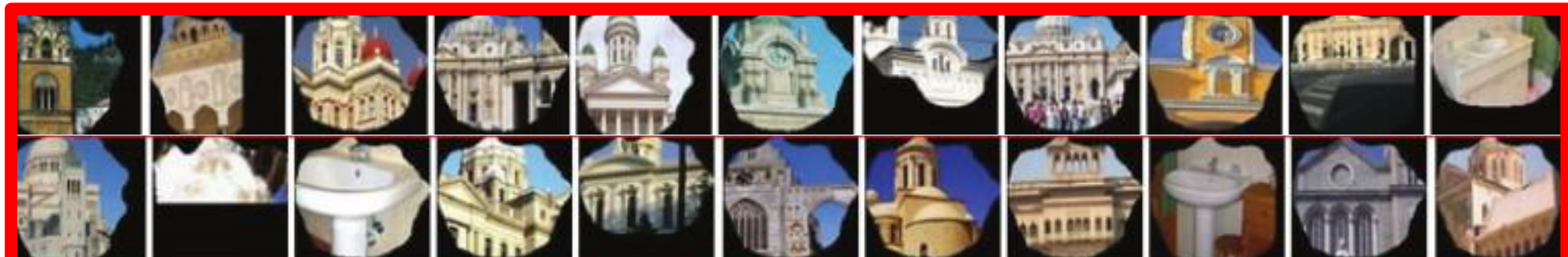
Annotating the Semantics of Units

Pool5, unit 112; Label: pool table; Type: object; Precision: 70%



Annotating the Semantics of Units

Pool5, unit 22; Label: dinner table; Type: scene; Precision: 60%



ImageNet vs. PlacesNet

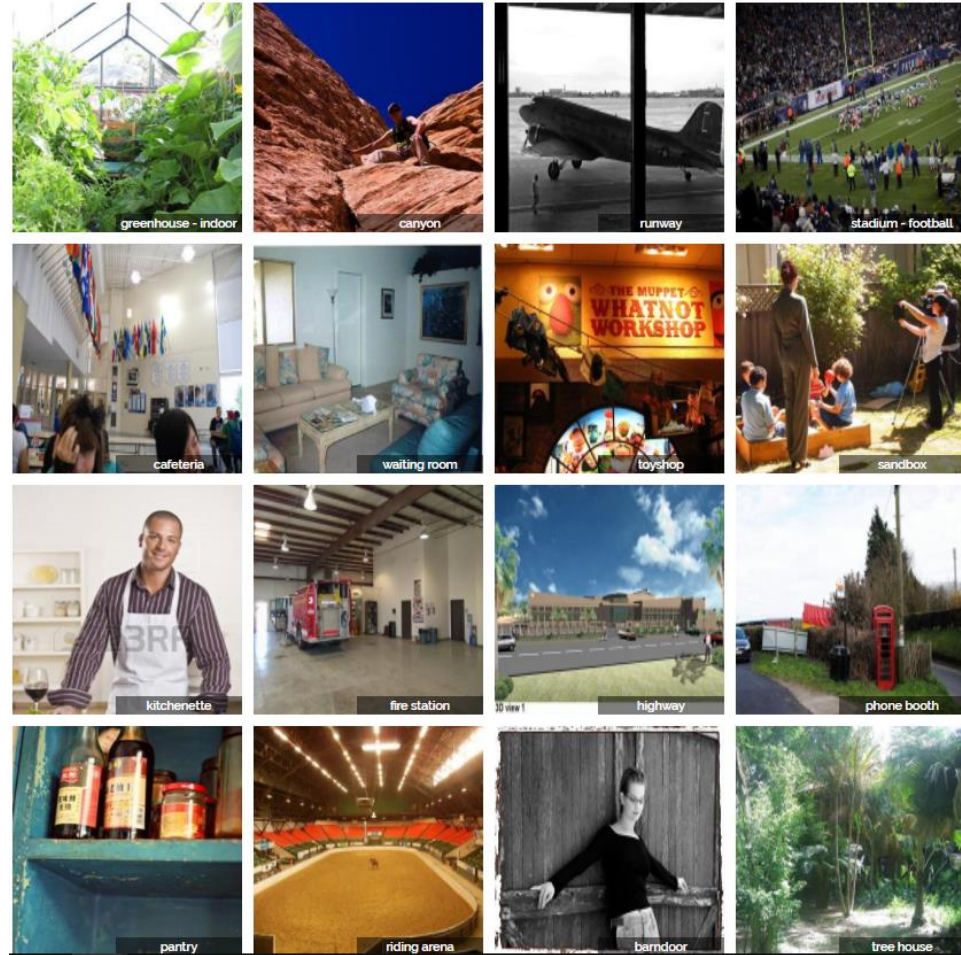
<http://places2.csail.mit.edu/demo.html>

ImageNet

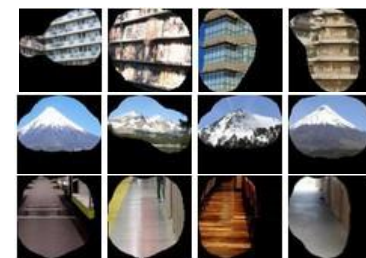
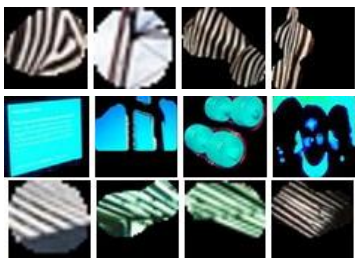
- ~1 mil object-level images over 1000 classes

PlacesNet

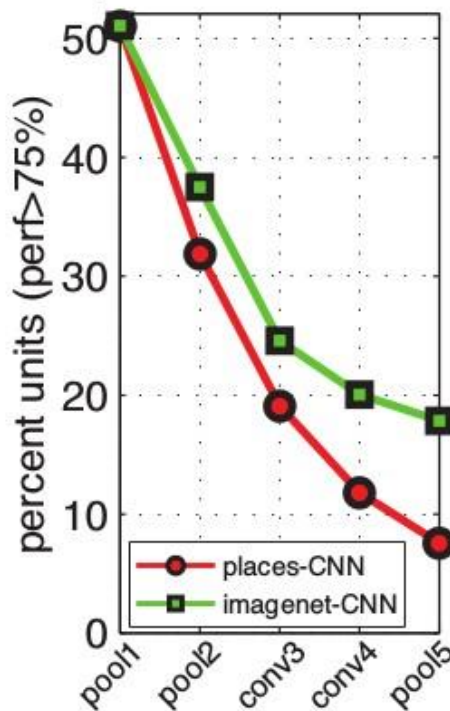
- ~1.8 million images from 365 scene categories (at most 5000 images per category).



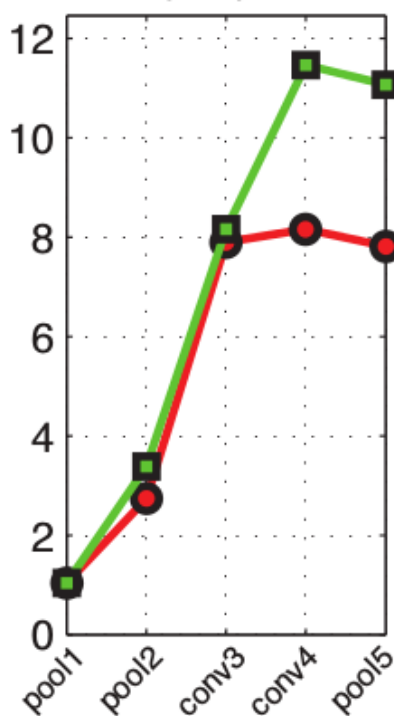
Distribution of Semantic Types at Each Layer



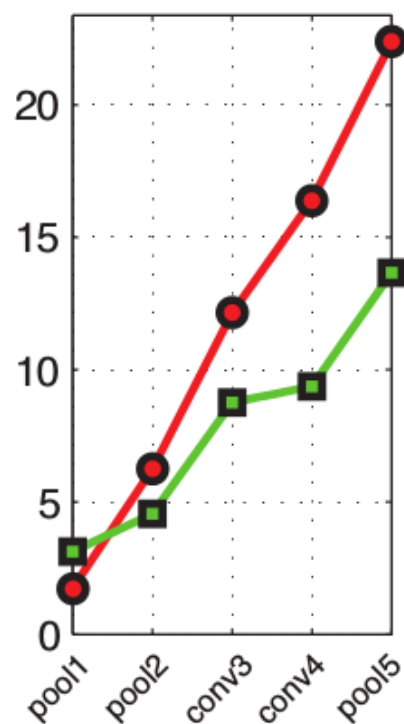
Simple elements & colors



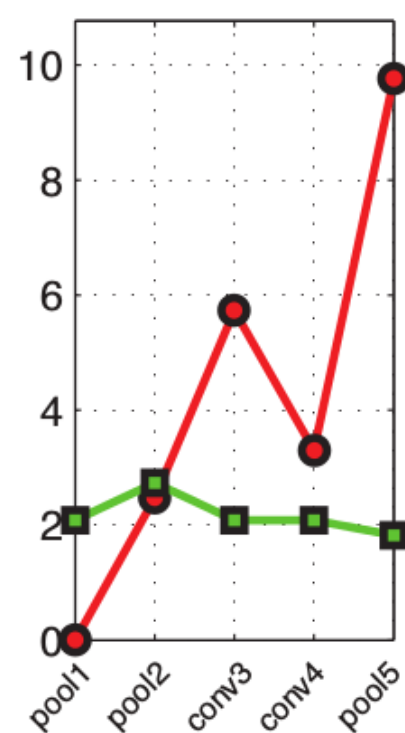
Object part



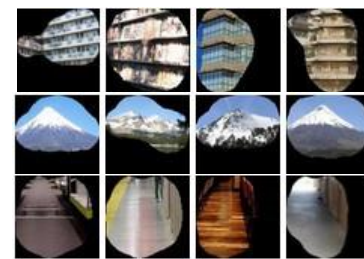
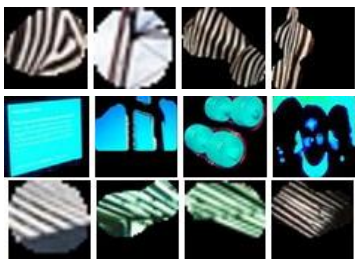
Object



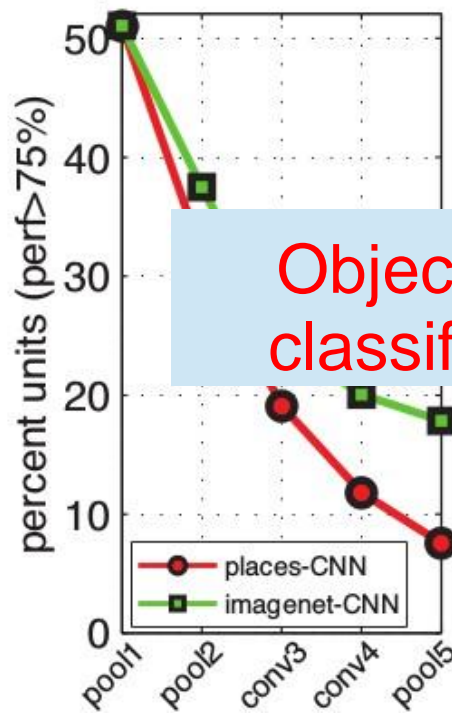
Scene



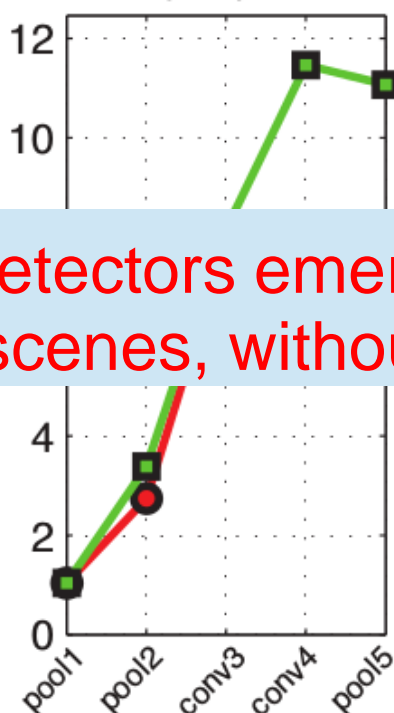
Distribution of Semantic Types at Each Layer



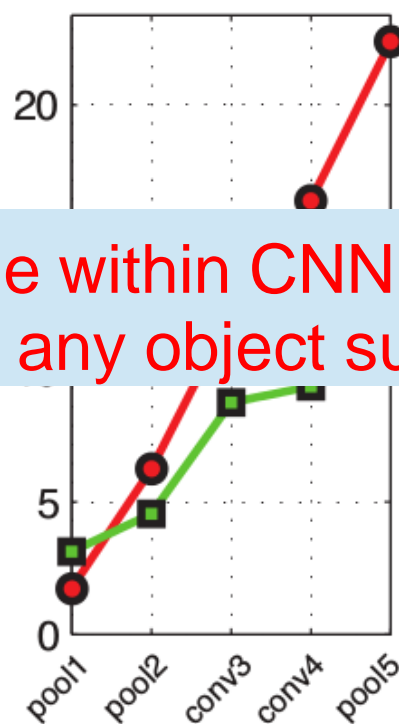
Simple elements & colors



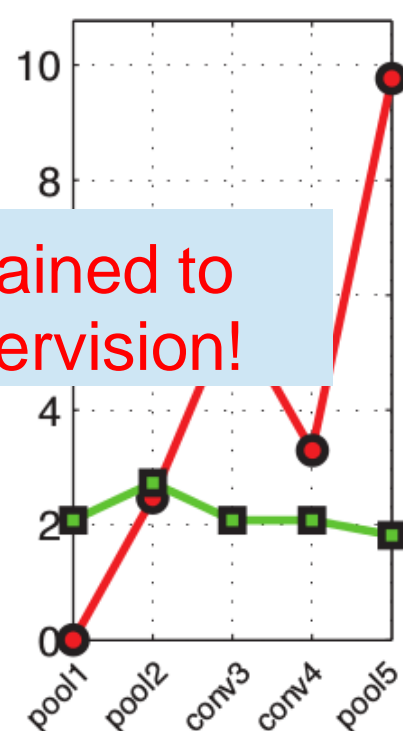
Object part



Object



Scene



Object detectors emerge within CNN trained to classify scenes, without any object supervision!



@teenybiscuit



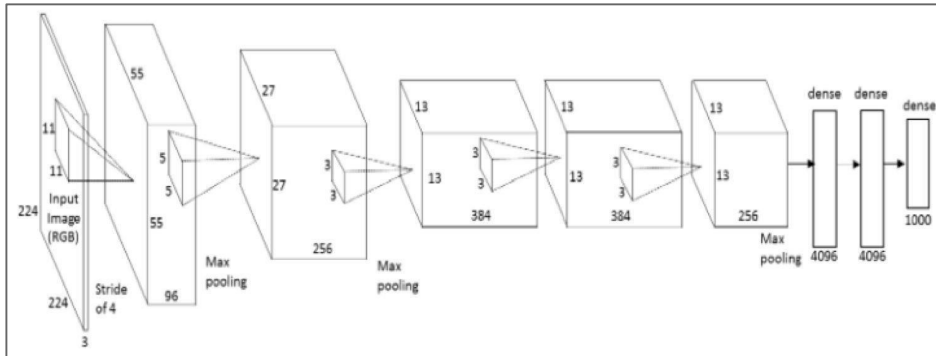


Interesting CNN properties

...or other ways to measure reception

<http://yosinski.com/deepvis>

What input to a neuron maximizes a class score?



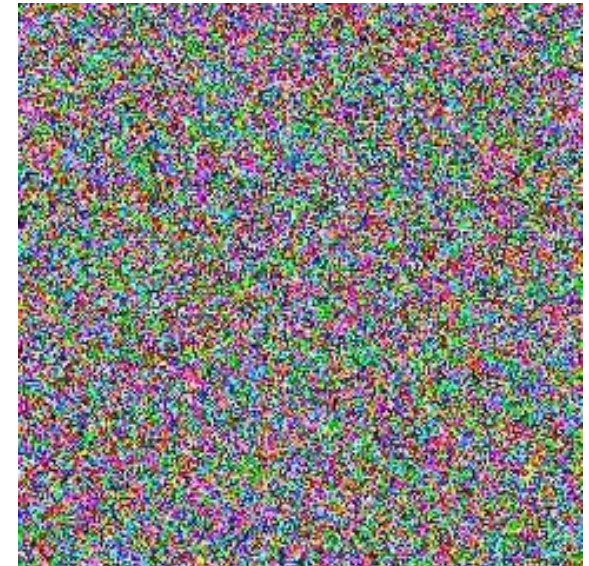
To visualize the function of a specific unit in a neural network, we synthesize an input to that unit which causes high activation.

Neuron of choice i

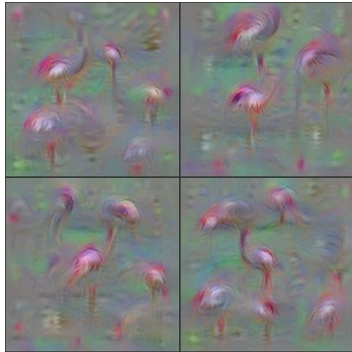
An image of random noise x .

Repeat:

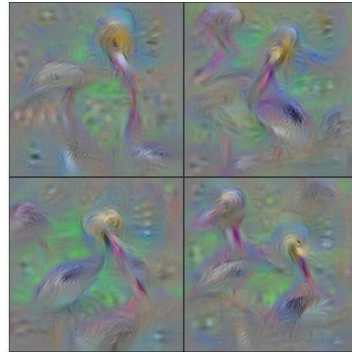
1. Forward propagate: compute activation $a_i(x)$
2. Back propagate: compute gradient at neuron $\partial a_i(x) / \partial x$
3. Add small amount of gradient back to noisy image.



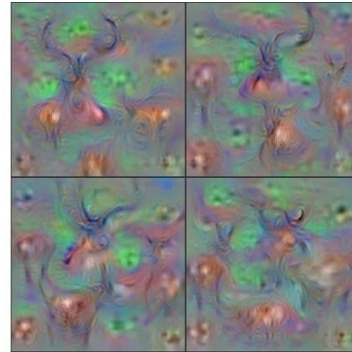
What image maximizes a class score?



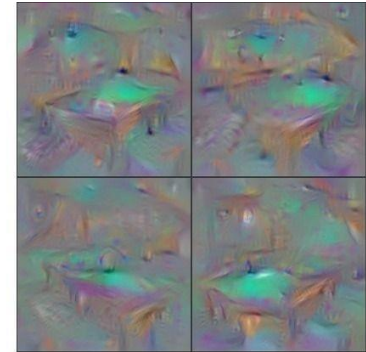
Flamingo



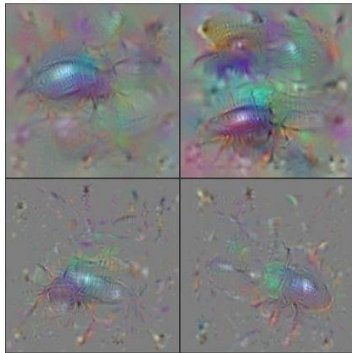
Pelican



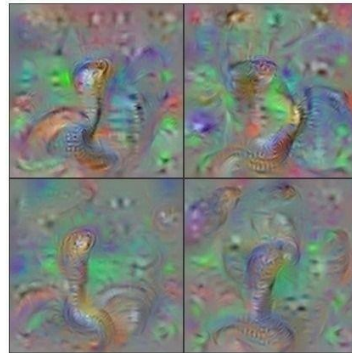
Hartebeest



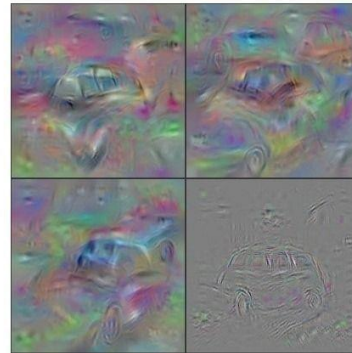
Billiard Table



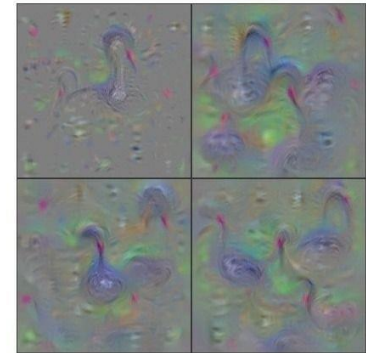
Ground Beetle



Indian Cobra



Station Wagon



Black Swan

[Understanding Neural Networks Through Deep Visualization, Yosinski et al. , 2015]

<http://yosinski.com/deepvis>

What image maximizes a class score?

Layer 8



Pirate Ship

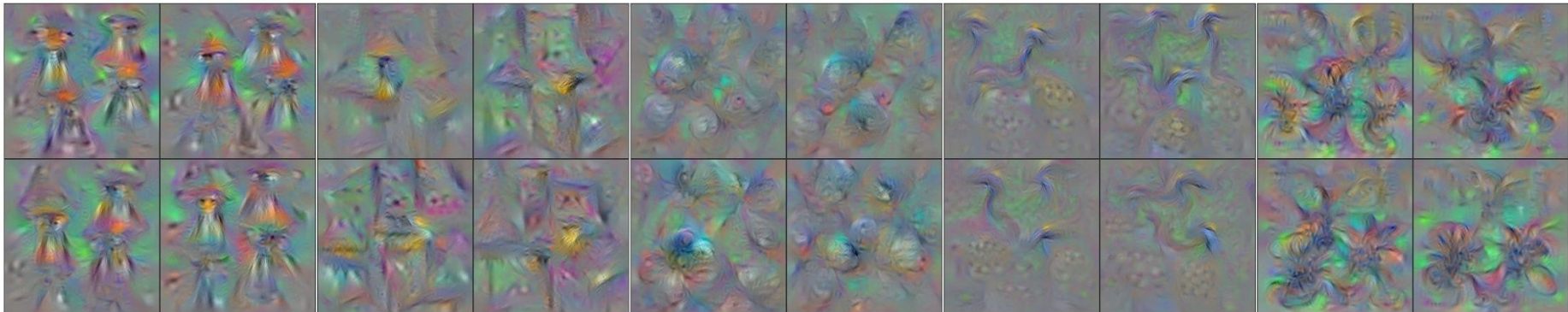
Rocking Chair

Teddy Bear

Windsor Tie

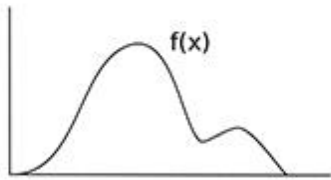
Pitcher

Layer 7

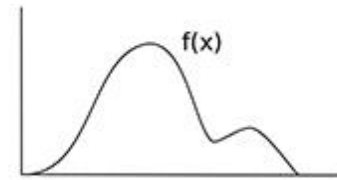


Wow!

They just 'fall out'!



Panda!



Gibbon!

Gibbon class
gradient



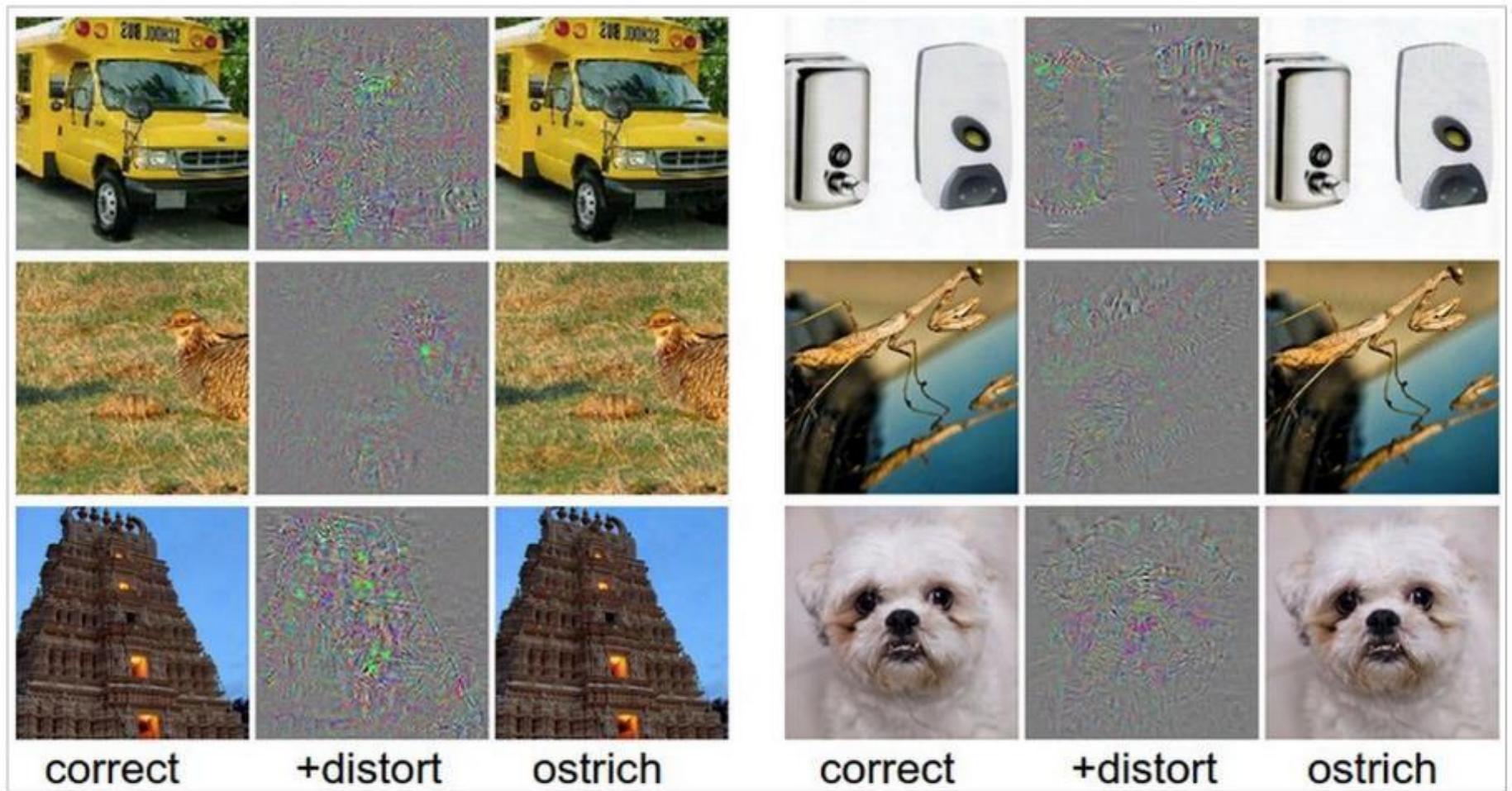
Panda



Adversarial example



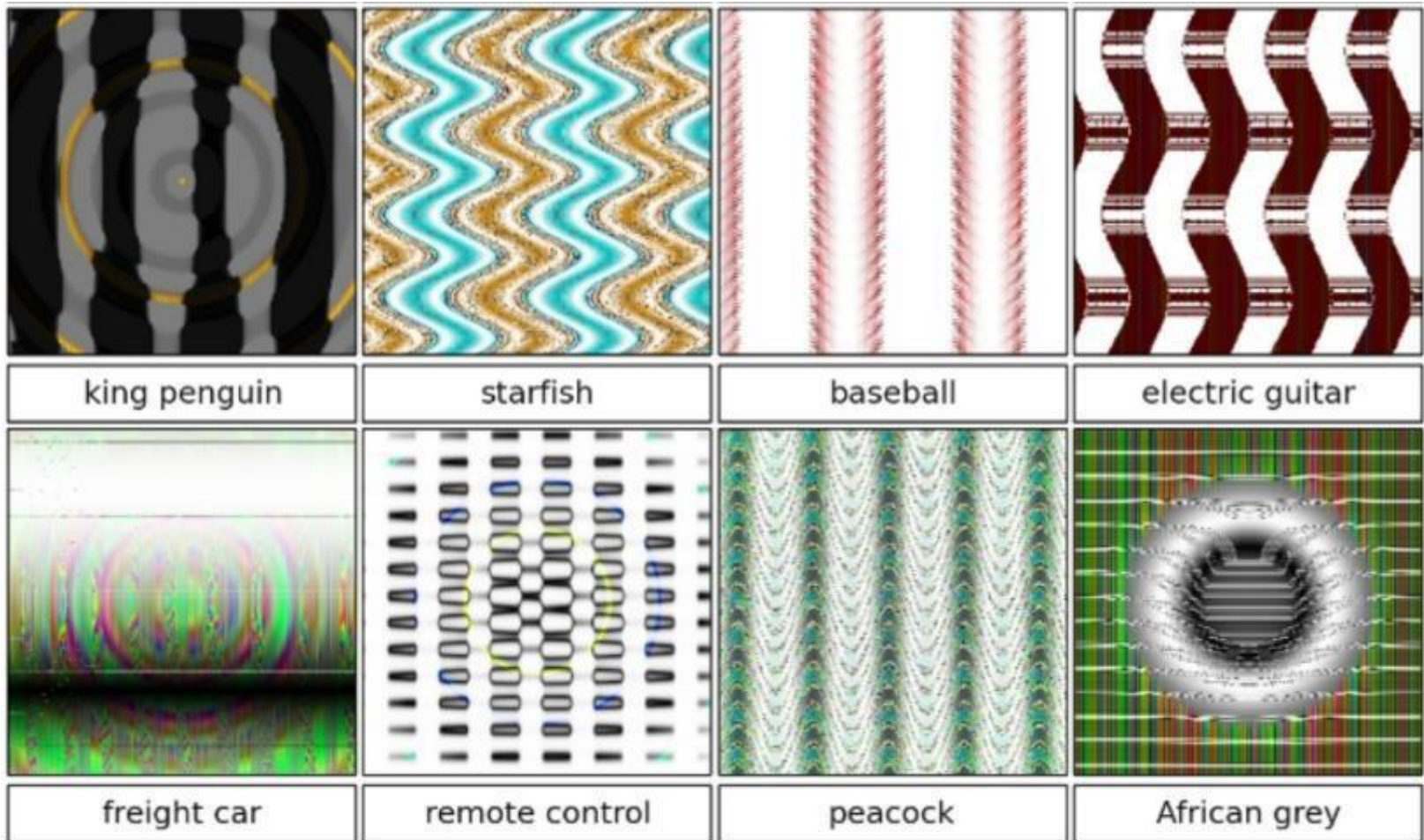
Breaking CNNs



Take a correctly classified image (left image in both columns), and add a tiny distortion (middle) to fool the ConvNet with the resulting image (right).

Intriguing properties of neural networks [[Szegedy ICLR 2014](#)]

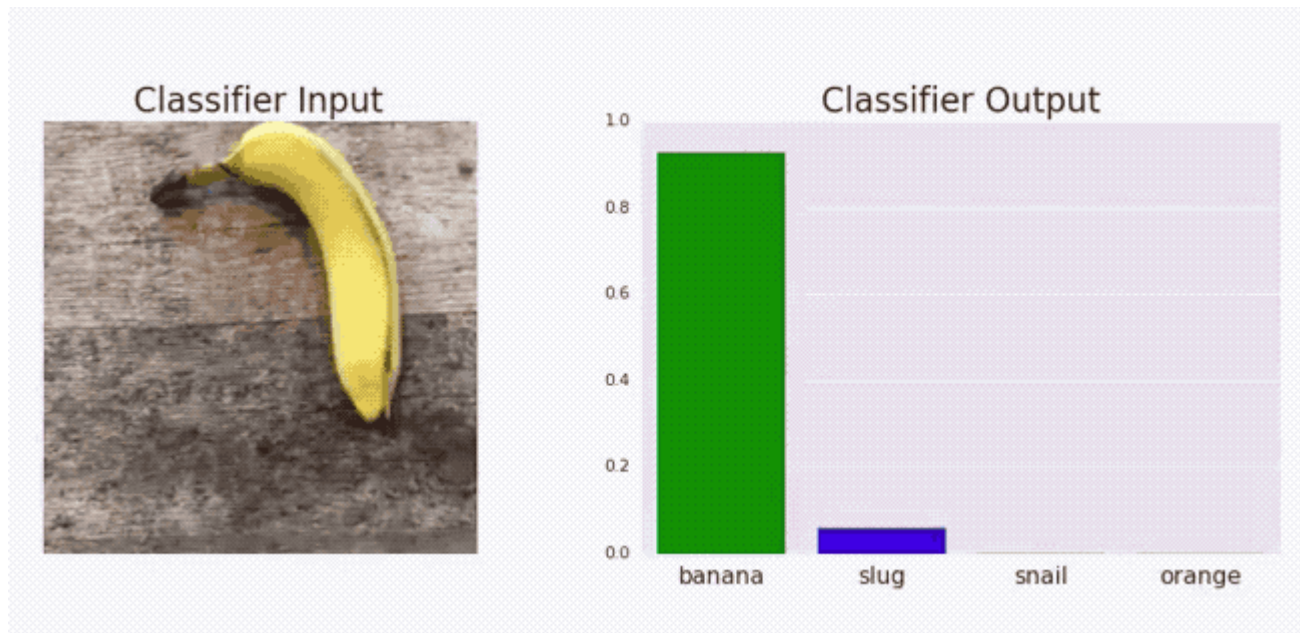
Breaking CNNs



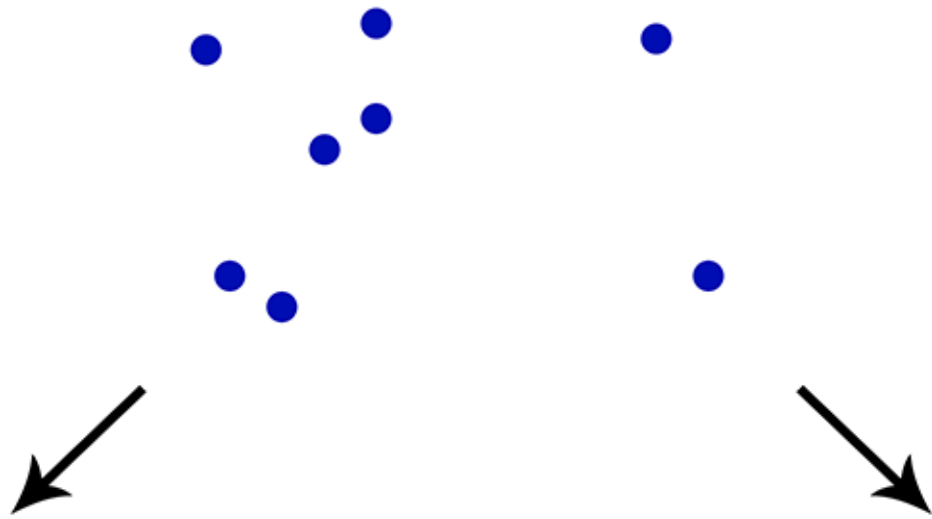
Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images [[Nguyen et al. CVPR 2015](#)]

Adversarial Patches

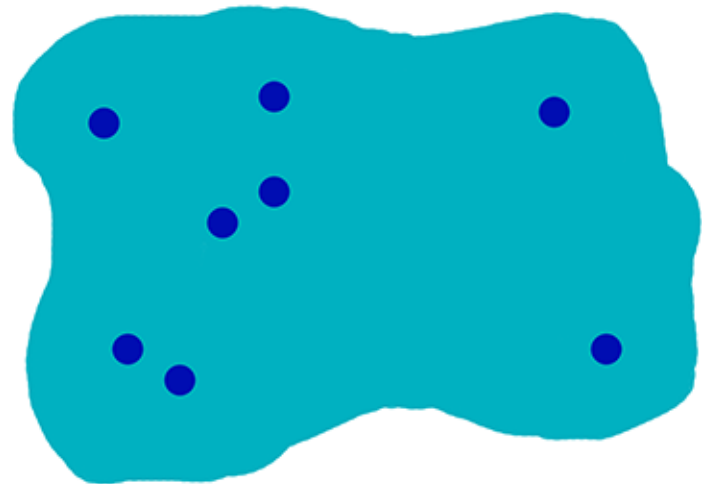
- <https://www.theverge.com/2018/1/3/16844842/ai-computer-vision-trick-adversarial-patches-google>
- <https://arxiv.org/pdf/1712.09665.pdf>



A same set of data points or Experience



Local generalization:
Generalization power of
pattern recognition



Extreme generalization:
Generalization power
achieved via
abstraction and reasoning



The boy is holding a baseball bat.

For next time: The success of obfuscating gradients

- <https://github.com/anishathalye/obfuscated-gradients>

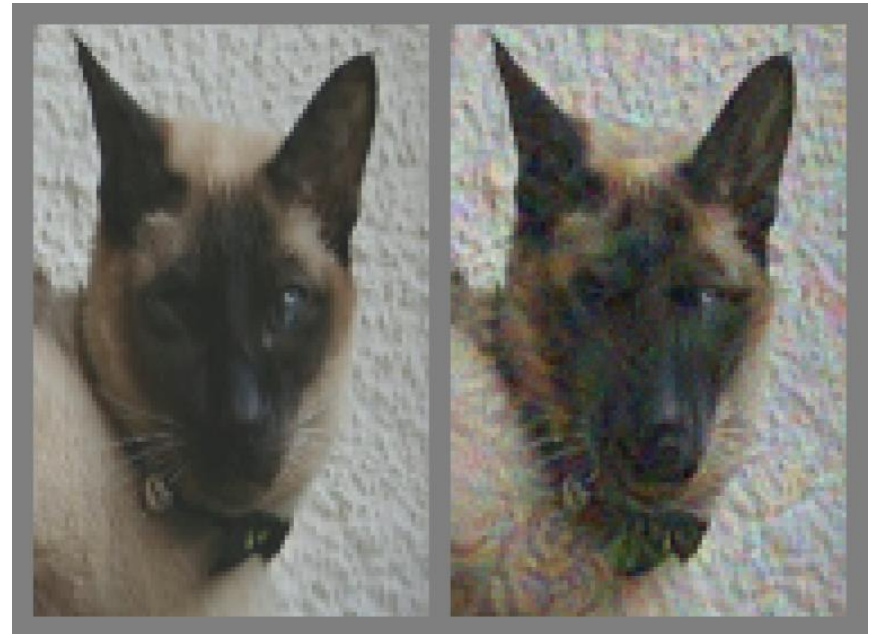
In our recent paper, we evaluate the robustness of eight papers accepted to ICLR 2018 as non-certified white-box-secure defenses to adversarial examples. We find that seven of the eight defenses provide a limited increase in robustness and can be broken by improved attack techniques we develop.

The only defense we observe that significantly increases robustness to adversarial examples within the threat model proposed is “Towards Deep Learning Models Resistant to Adversarial Attacks” (Madry et al. 2018), and we were unable to defeat this defense without stepping outside the threat model. Even then, this technique has been shown to be difficult to scale to ImageNet-scale (Kurakin et al. 2016). The remainder of the papers rely either inadvertently or intentionally on what we call *obfuscated gradients*. Standard attacks apply gradient descent to maximize the loss of the network on a given image to generate an adversarial example on a neural network. Such optimization methods require a useful gradient signal to succeed. When a defense obfuscates gradients, it breaks this gradient signal and causes optimization based methods to fail.

For next time: human + CV adversarial examples

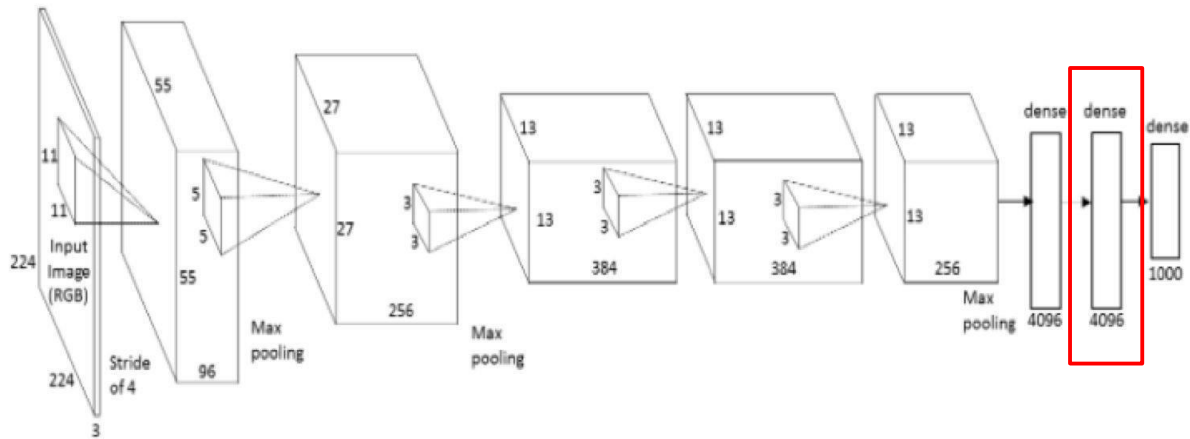
- <https://arxiv.org/abs/1802.08195>

JT note: does this actually work?



Reconstructing images

Question: Given a CNN **code**, is it possible to reconstruct the original image?



Reconstructing images

Find an image such that:

- Its code is similar to a given code
- It “looks natural”
 - Neighboring pixels should look similar

$$\text{Image } \mathbf{x}^* = \underset{\mathbf{x} \in \mathbb{R}^{H \times W \times C}}{\operatorname{argmin}} \ell(\Phi(\mathbf{x}), \Phi_0) + \lambda \mathcal{R}(\mathbf{x})$$

$$\ell(\Phi(\mathbf{x}), \Phi_0) = \|\Phi(\mathbf{x}) - \Phi_0\|^2$$

Reconstructing images

original image



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

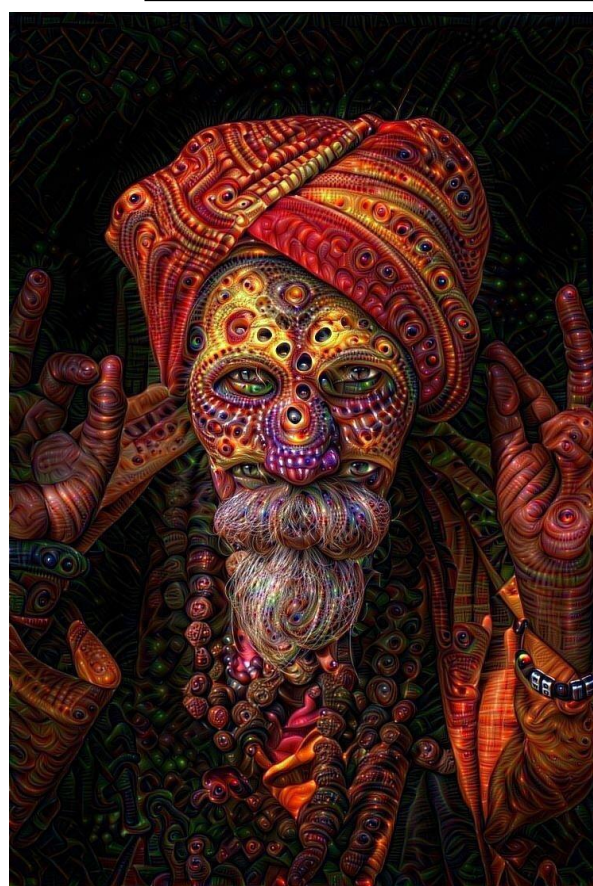
*Understanding Deep Image Representations by Inverting Them
[Mahendran and Vedaldi, 2014]*

Reconstructing images

Reconstructions from the representation after last last pooling layer
(immediately before the first Fully Connected layer)



DeepDream



DeepDream <https://github.com/google/deepdream>

DeepDream



DeepDream modifies the image in a way that “boosts” all activations, at any layer

This creates a feedback loop: e.g., any slightly detected dog face will be made more and more dog-like over time.



"Admiral Dog!"



"The Pig-Snail"



"The Camel-Bird"



"The Dog-Fish"

DeepDream

Deep Dream Grocery Trip

<https://www.youtube.com/watch?v=DgPaCWJL7XI>

Deep Dreaming Fear & Loathing in Las Vegas: the Great San Francisco AcidWave

<https://www.youtube.com/watch?v=oyxSerkkP4o>

Style transfer

Neural Style

[A Neural Algorithm of Artistic Style by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge, 2015]

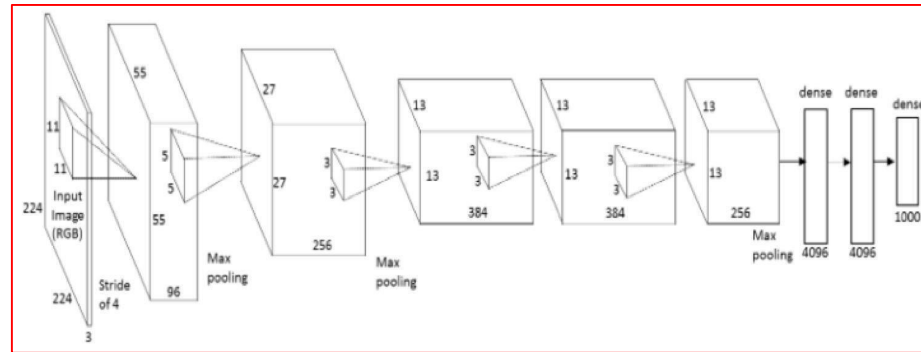
good implementation by Justin Johnson in Torch:

<https://github.com/jcjohnson/neural-style>



Neural Style

Step 1: Extract **content targets** (ConvNet activations of all layers for the given content image)

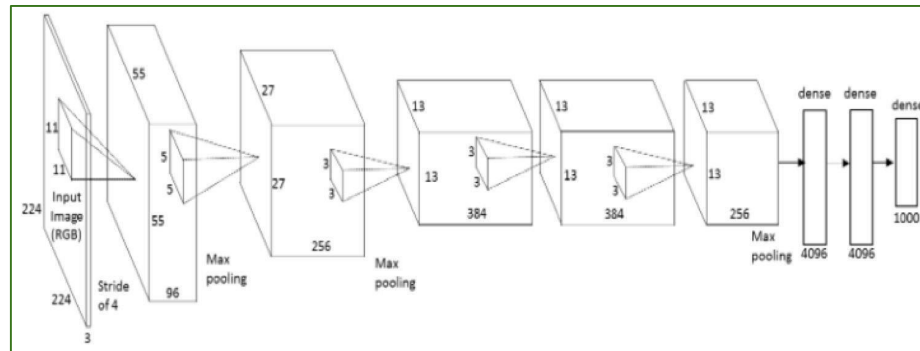


content activations

e.g.
at CONV5_1 layer we would have a [14x14x512] array of target activations

Neural Style

Step 2: Extract **style targets** (Gram matrices of ConvNet activations of all layers for the given style image)



style gram matrices

e.g.
at CONV1 layer (with [224x224x64] activations) would give a [64x64] Gram matrix of all pairwise activation covariances (summed across spatial locations)

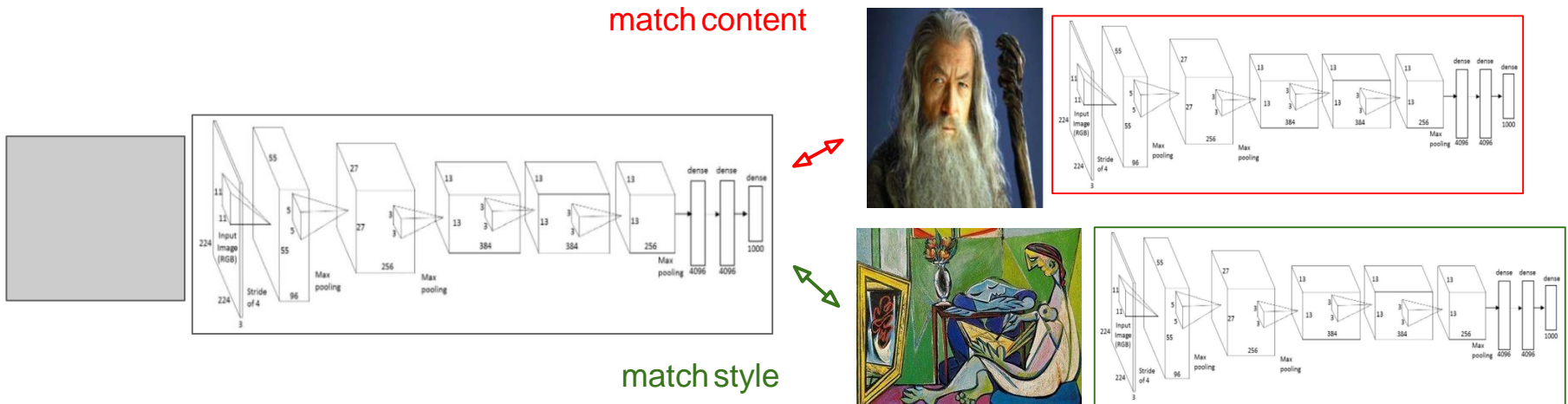
$$G = V^T V$$

Neural Style

Step 3: Optimize over image to have:

- The **content** of the content image (activations match content)
- The **style** of the style image (Gram matrices of activations match style)

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$



Neural Style



make your own easily on deepart.io

Dataset Distillation

- <https://arxiv.org/pdf/1811.10959.pdf>

DATASET DISTILLATION

Tongzhou Wang
Facebook AI Research, MIT CSAIL

Jun-Yan Zhu
MIT CSAIL

Antonio Torralba
MIT CSAIL

Alexei A. Efros
UC Berkeley

ABSTRACT

Model distillation aims to distill the knowledge of a complex model into a simpler one. In this paper, we consider an alternative formulation called *dataset distillation*: we keep the model fixed and instead attempt to distill the knowledge from a large training dataset into a small one. The idea is to *synthesize* a small number of data points that do not need to come from the correct data distribution, but will, when given to the learning algorithm as training data, approximate the model trained on the original data. For example, we show that it is possible to compress 60,000 MNIST training images into just 10 synthetic *distilled images* (one per class) and achieve close to original performance with only a few gradient descent steps, given a fixed network initialization. We evaluate our method in various initialization settings and with different learning objectives. Experiments on multiple datasets show the advantage of our approach compared to alternative methods.