

CPSC 340: Machine Learning and Data Mining

More CNNs

Admin

- Assignment 5
 - due tonight
- Assignment 6:
 - Posted yesterday, due next Friday
- Assignment weights: 3% for hw0, 4% each for hw1-5, 2% for hw6
- Final exam:
 - April 25 (8:30am, ESB 1013)
 - Covers Assignments 1-6.
 - Past exams posted on GitHub.
 - Closed-book, cheat sheet: 1-page double-sided (same as midterm).

Today's class is bonus material

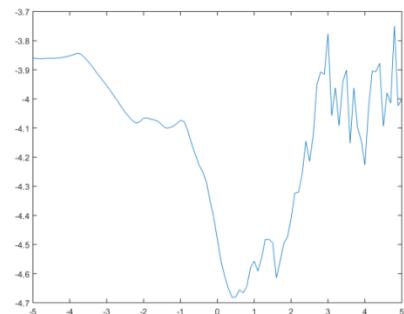
- Nothing in today's class will be in hw6 or on the final exam.
- We'll just discuss interesting & fun applications of CNNs.
- CNNs more generally are not on hw6 so there won't be a big question about them on the exam, maybe just some small multiple-choice stuff like the midterm.

Last Time: Convolutions

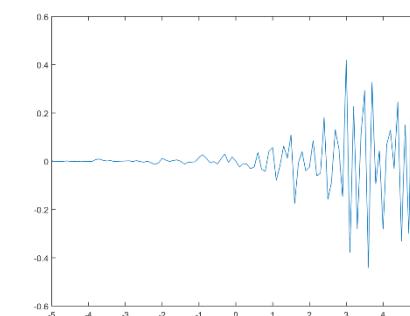
- **1D convolution:**

- Takes **signal 'x'** and **filter 'w'** to produces vector 'z':

$$x * w = z$$



$$\begin{bmatrix} -1 & 2 & -1 \end{bmatrix}$$



- Can be written as a **matrix multiplication**:

$$W_x = \begin{bmatrix} 1 & -2 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & & & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & -2 & 1 \end{bmatrix} = z$$

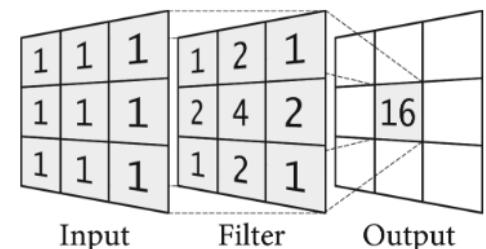
Last Time: Convolutions

- **2D convolution:**

- Signal ‘x’, filter ‘w’, and output ‘z’ are now all **images/matrices**:

$$x * w = z$$


$$\begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

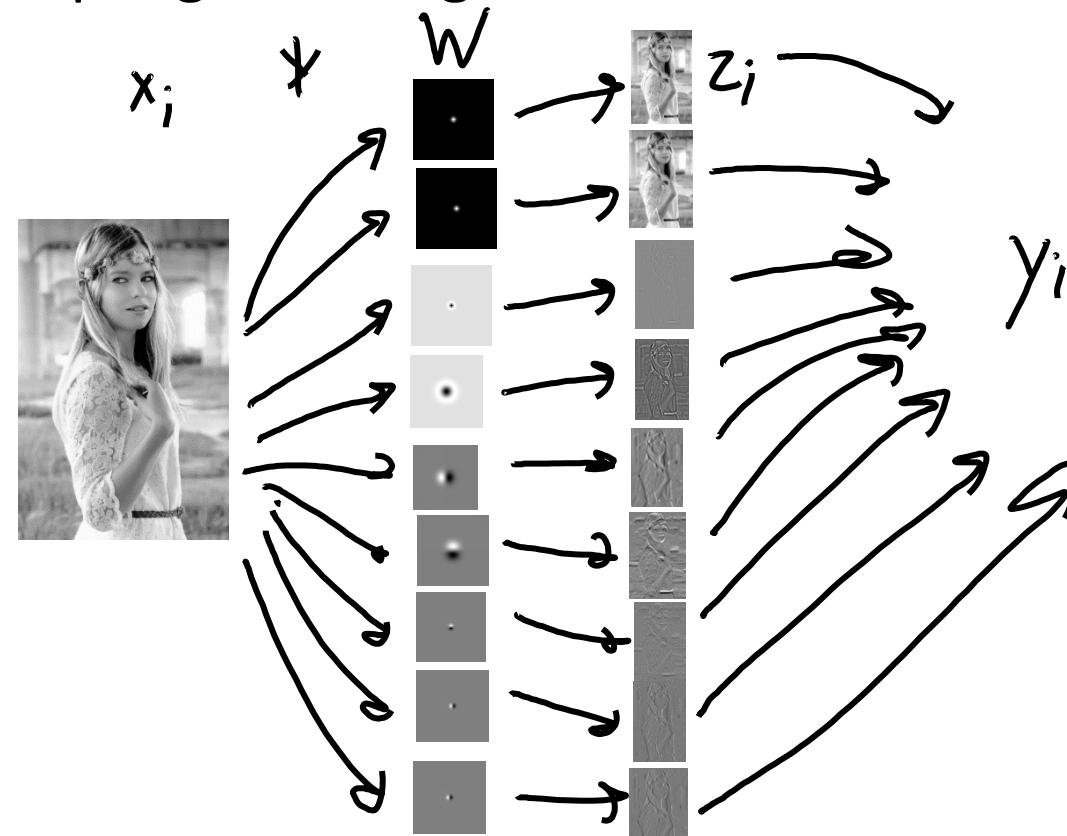
- Vectorized ‘z’ can be written as a **matrix multiplication** with vectorized ‘x’:

$$W = \begin{bmatrix} -2 & -1 & 0 & 0 & 0 & \dots & 0 & -1 & 0 & 1 & 0 & 0 & \dots & 0 & 0 & 1 & 2 & 0 & 0 & \dots & 0 & 0 \\ 0 & -2 & -1 & 0 & 0 & \dots & 0 & 0 & -1 & 0 & 1 & 0 & 0 & \dots & 0 & 1 & 0 & 0 & 1 & 2 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & -6 & -5 & 0 & 0 & 0 & 0 & 2 & -1 & -1 & 0 & -6 & 0 & 0 & 0 & -1 & -1 & 0 & 1 & 0 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

5

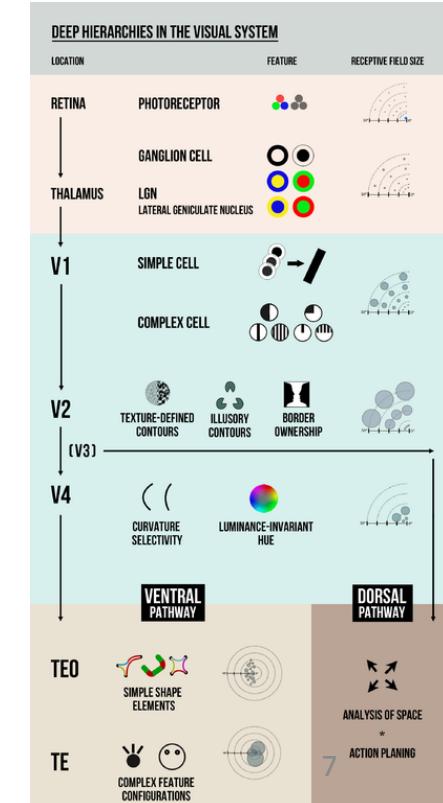
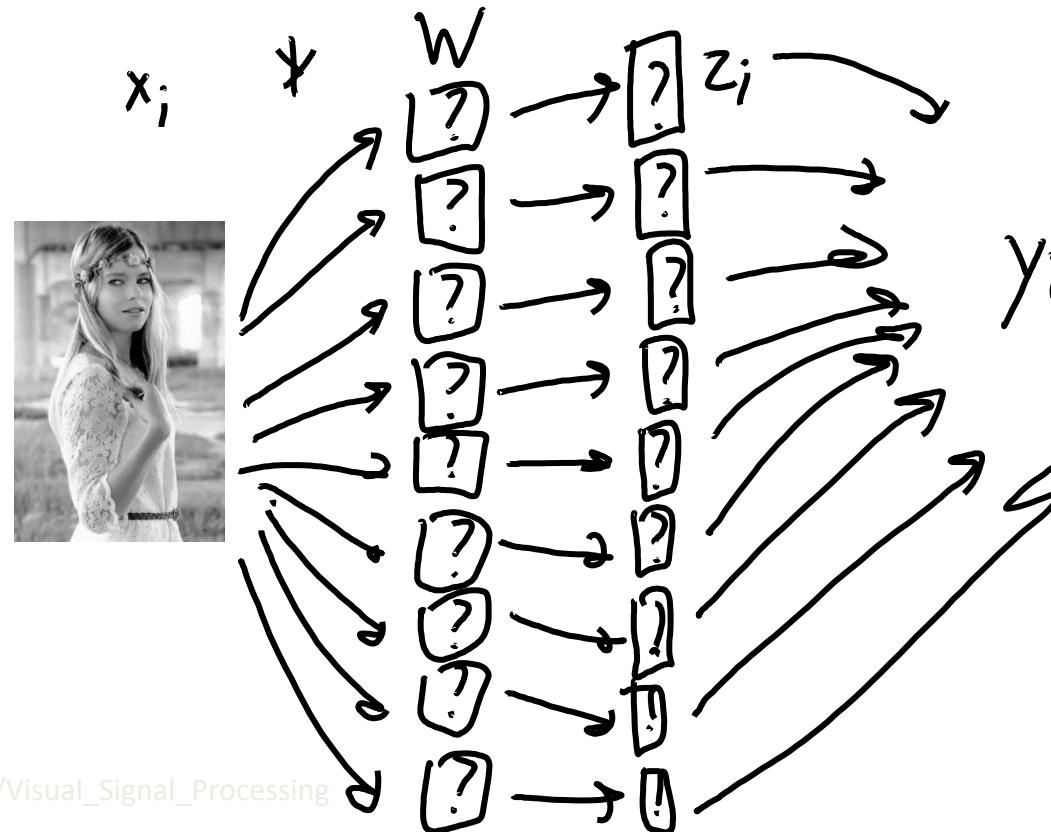
Last Time: Convolutional Neural Networks

- Classic approach uses **fixed convolutions** as features:
 - Usually have **different types/variances/orientations**.
 - Can do subsampling or taking **maxes** across locations/orientations/scales.



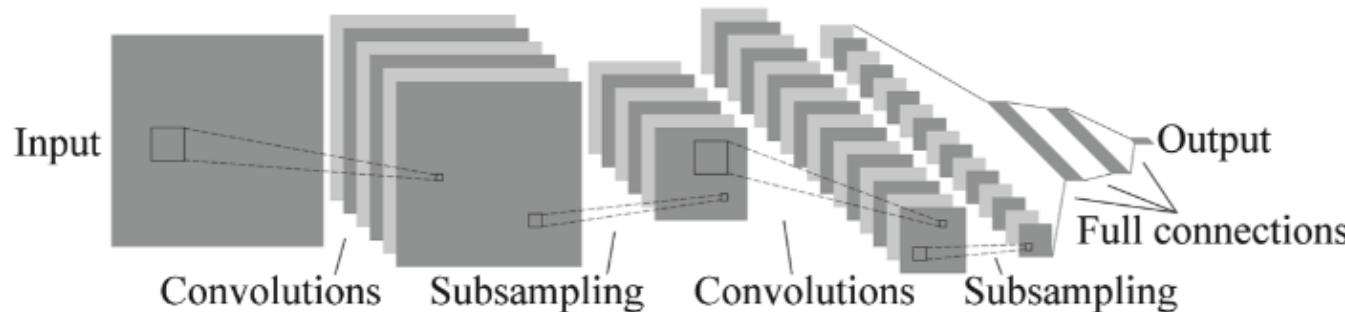
Last Time: Convolutional Neural Networks

- Convolutional neural networks learn the features:
 - Learning ‘W’ and ‘w’ automatically chooses types/variances/orientations.
 - Can do multiple layers of convolution to get deep hierarchical features.



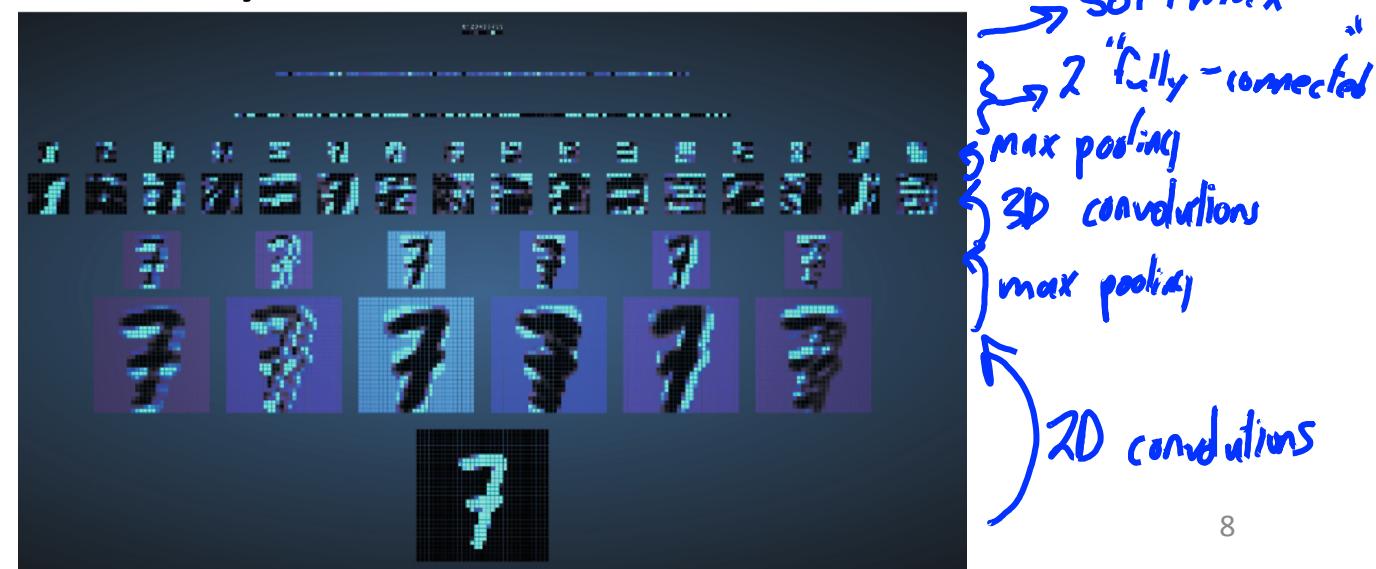
Last Time: Convolutional Neural Networks

- Classic convolutional neural network (LeNet):



- Visualizing the “activations” of the layers:

- <http://scs.ryerson.ca/~aharley/vis/conv>



AlexNet Convolutional Neural Network

- ImageNet 2012 won by **AlexNet**:
 - 15.4% error vs. 26.2% for closest competitor.
 - 5 convolutional layers.
 - 3 fully-connected layers.
 - SG with momentum.
 - ReLU non-linear functions.
 - Data translation/reflection/cropping.
 - L2-regularization + Dropout.
 - 5-6 days on two GPUs.

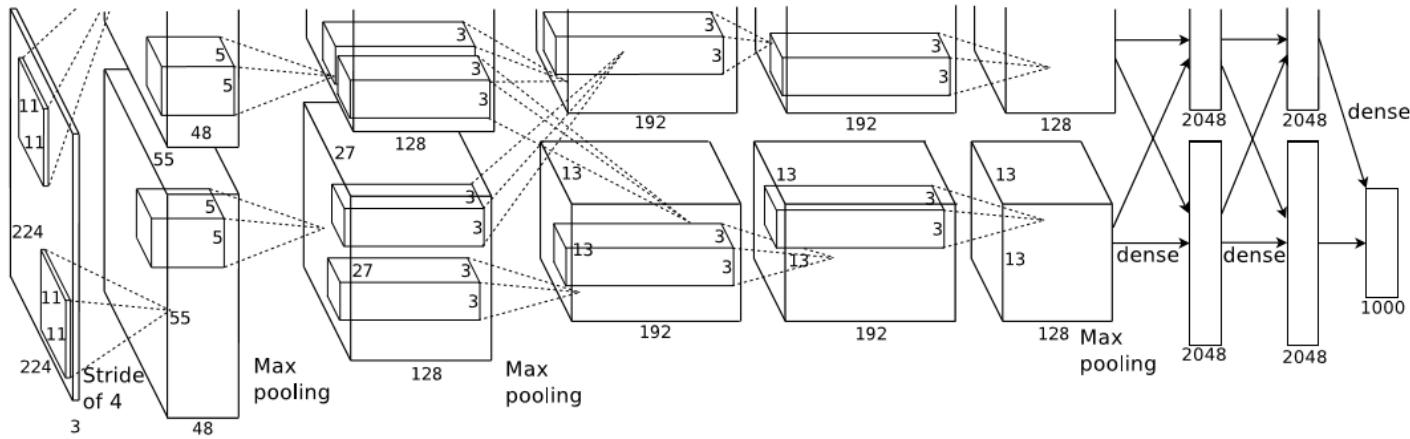


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

AlexNet Convolutional Neural Network

- ImageNet 2012 won by AlexNet:
 - 15.4% error vs. 26.2% for closest competitor.

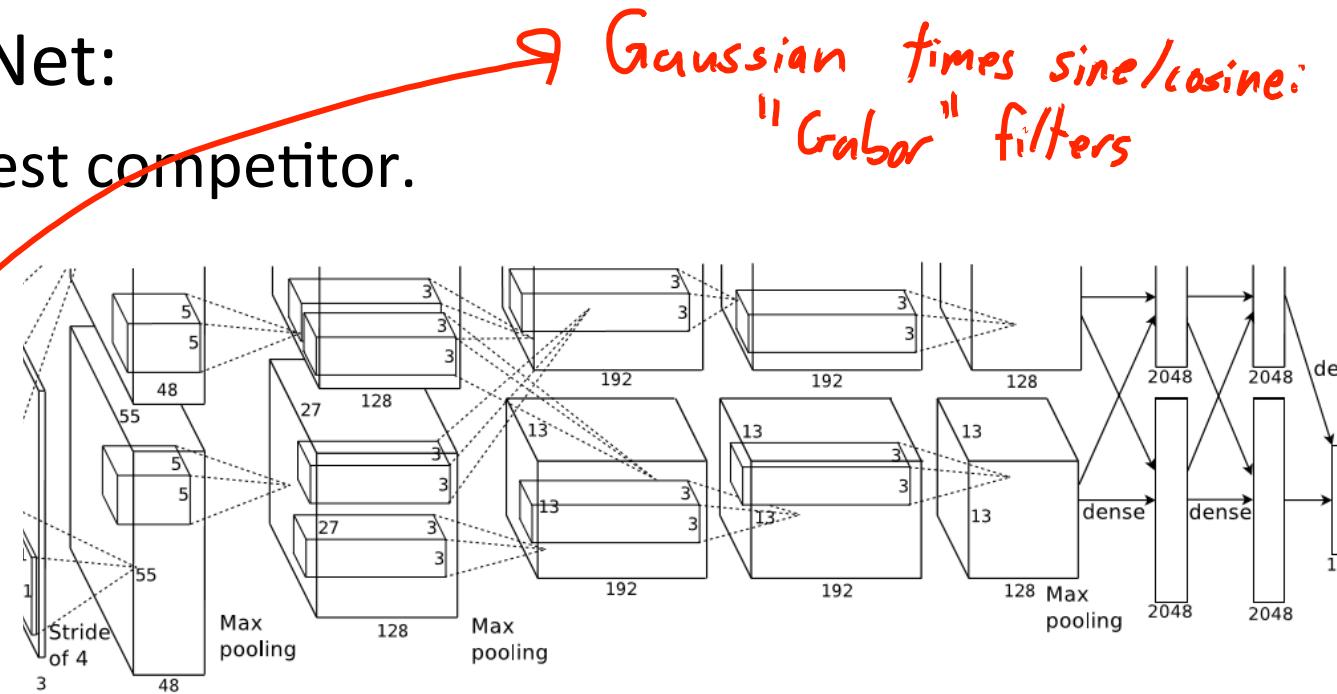
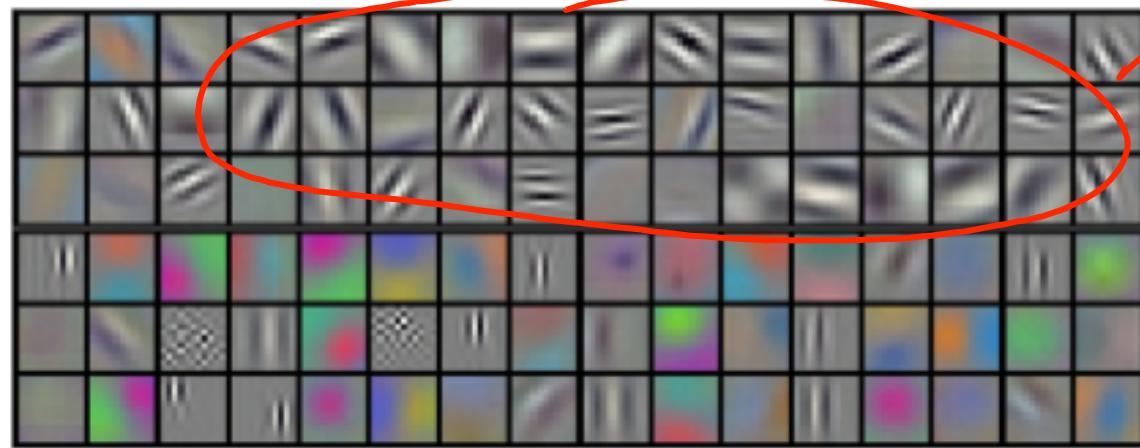


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The

ure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–6–4096–1000.

ZFNet Convolutional Neural Network

- ImageNet 2013 won by variation of AlexNet called ZF Net:
 - 11.2% error (now using 11x11 instead of 7x7).
 - Introduced **deconvolutional networks** to visualize what CNNs learn.

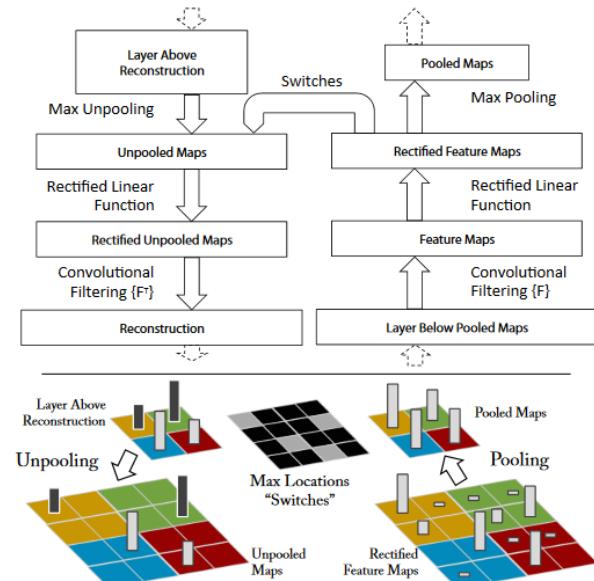
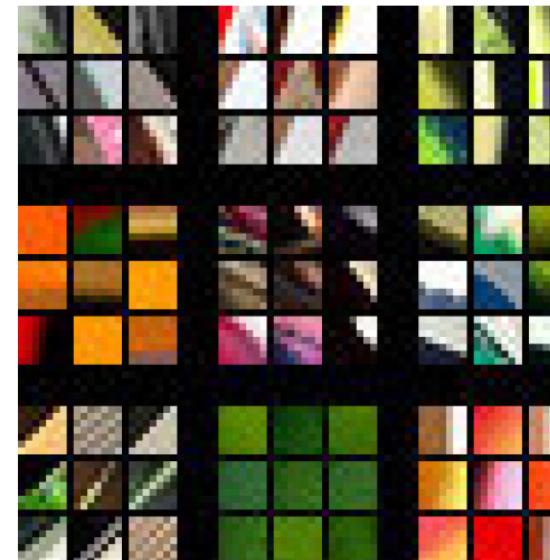


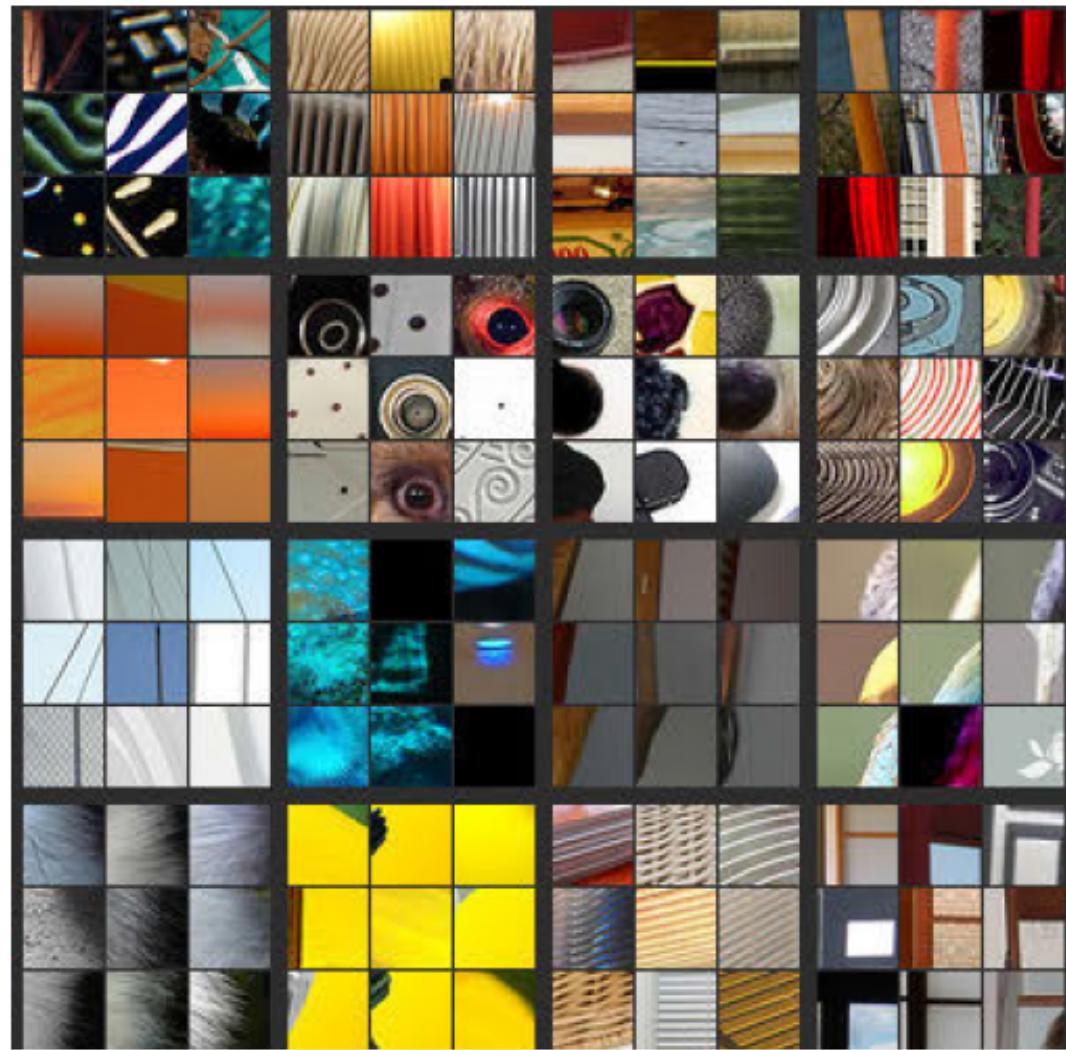
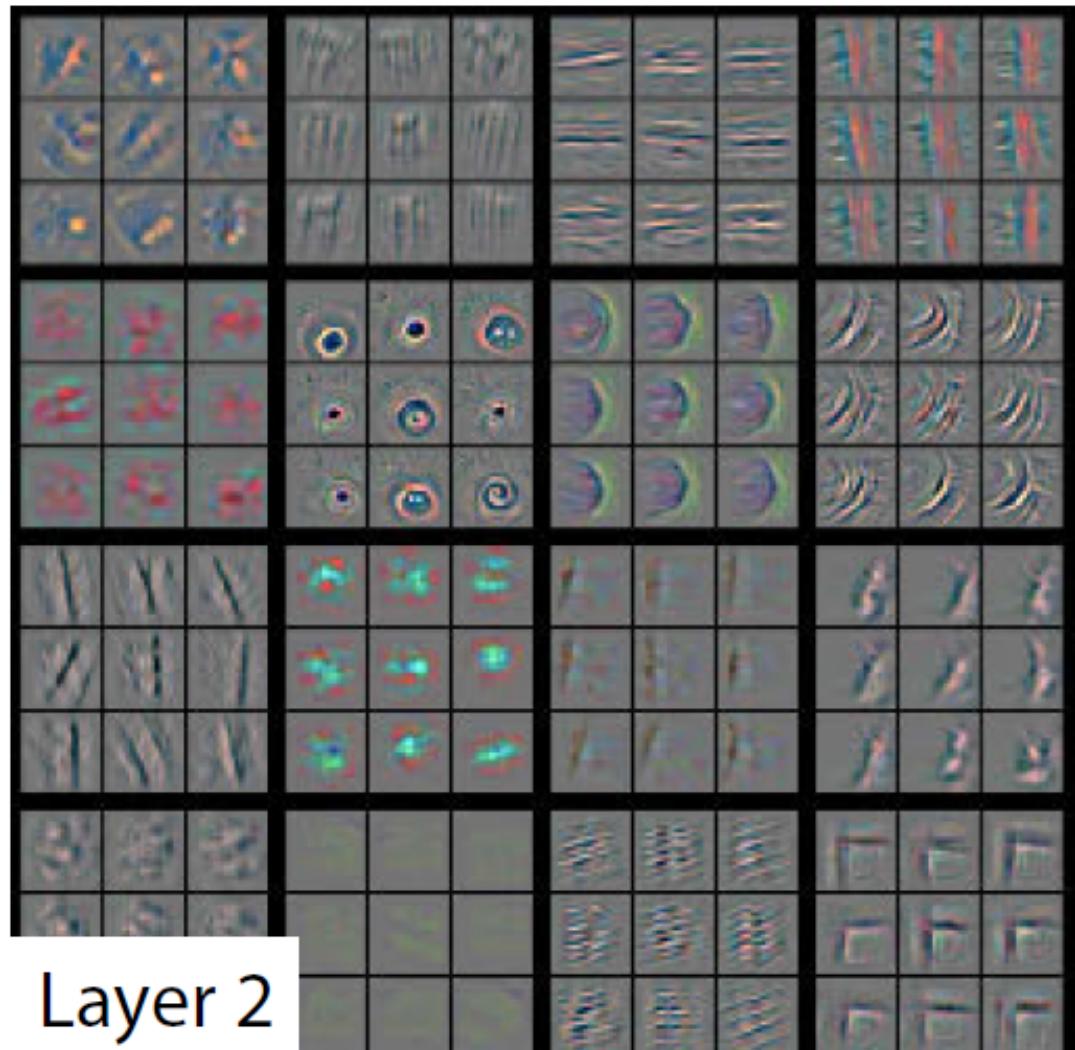
Figure 1. Top: A deconvnet layer (left) attached to a convnet layer (right). The deconvnet will reconstruct an approximate version of the convnet features from the layer beneath. Bottom: An illustration of the unpooling operation in the deconvnet, using *switches* which record the location of the local max in each pooling region (colored zones) during pooling in the convnet.



Layer 1



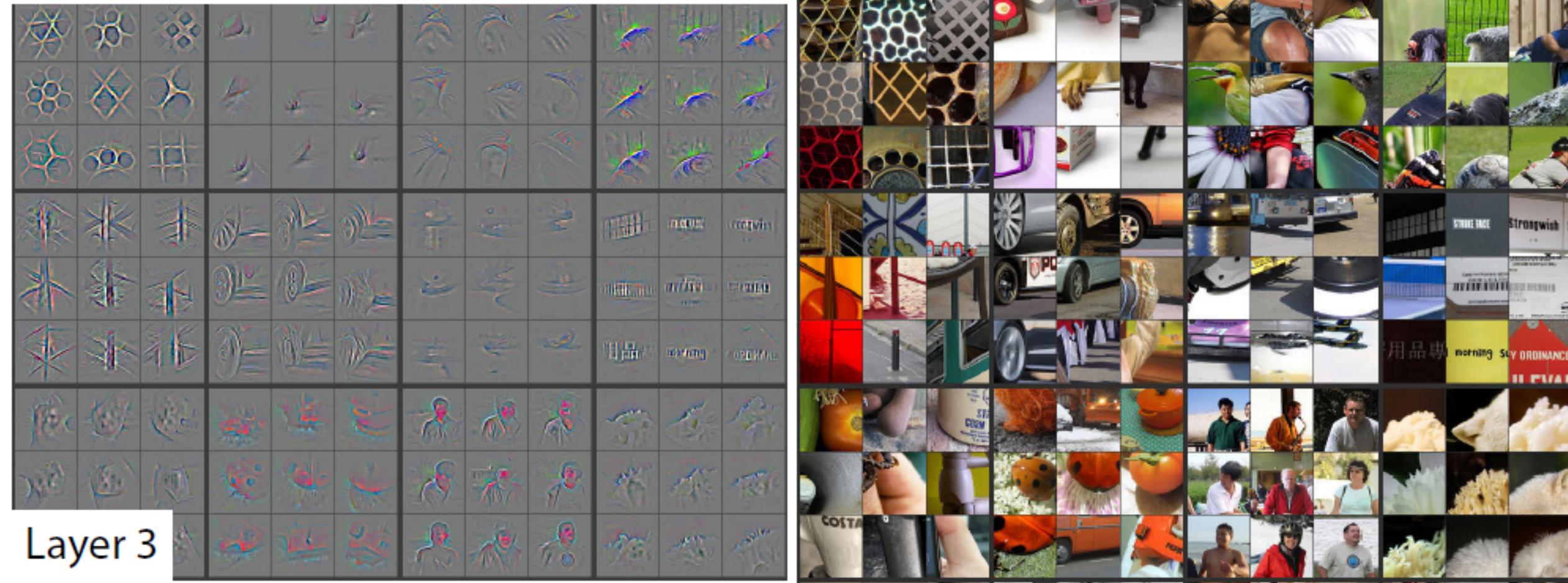
ZFNet Convolutional Neural Network



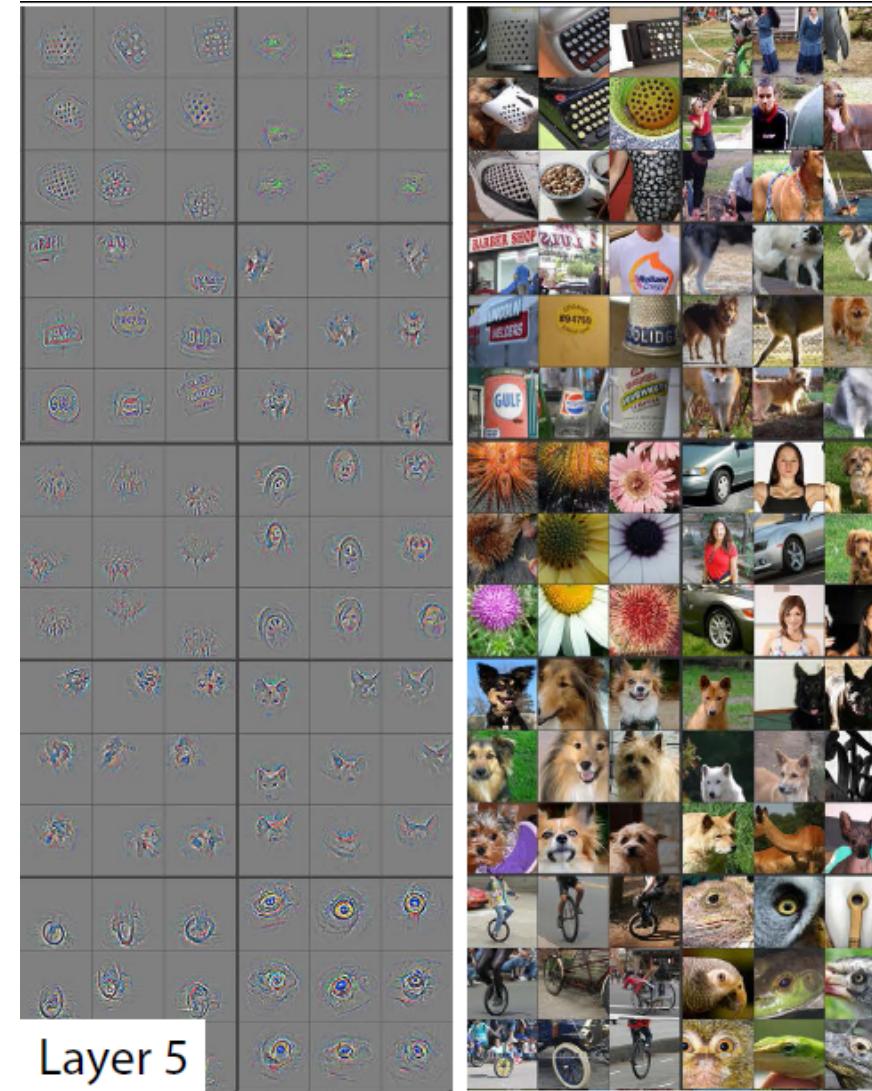
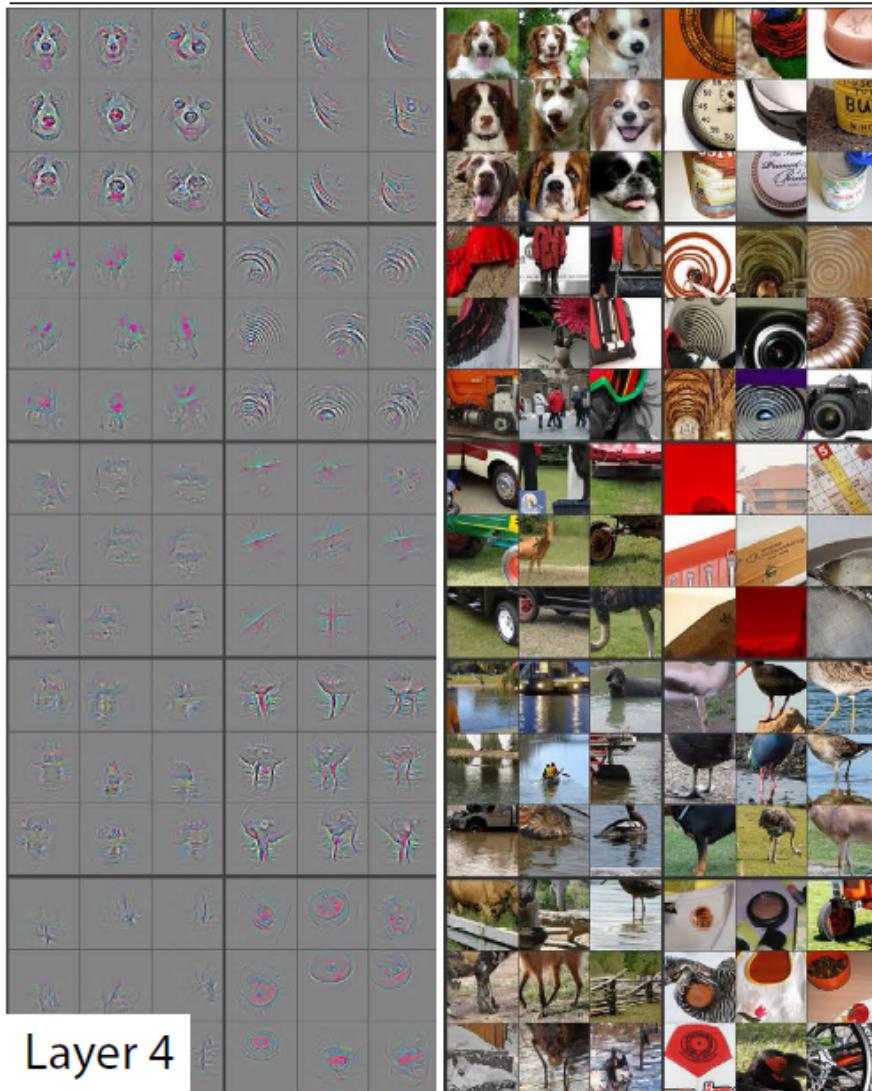
12

→ Deconvolution network! giving patch that leads to largest response

ZFNet Convolutional Neural Network

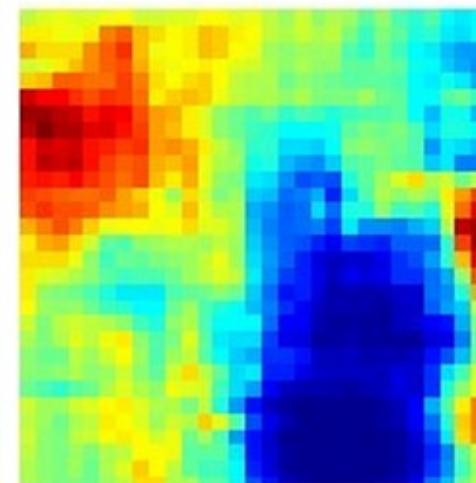
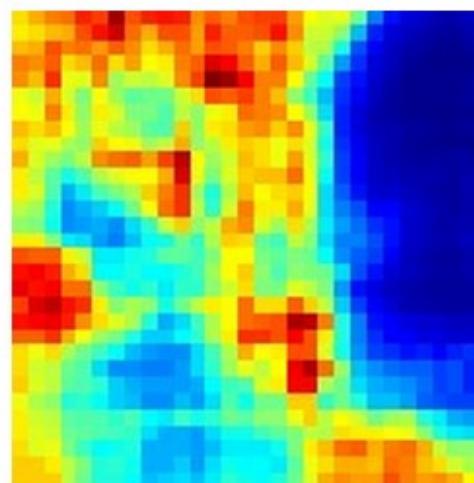
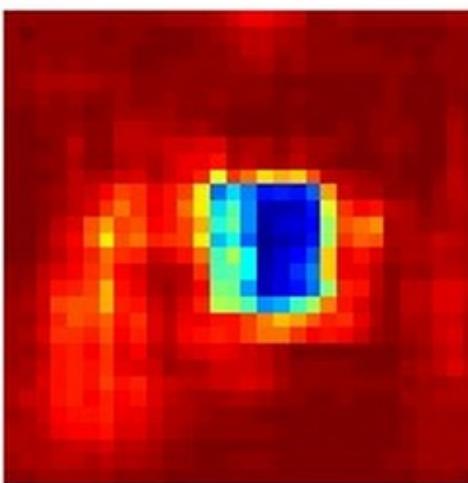


ZFNet Convolutional Neural Network



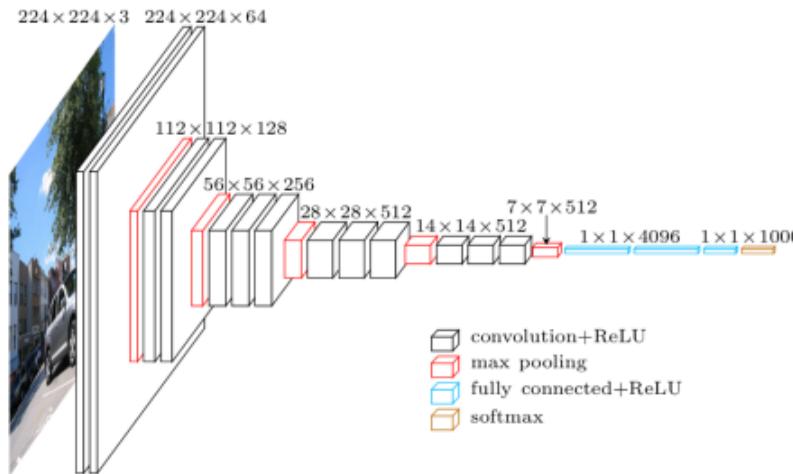
ZFNet Convolutional Neural Network

- Looked at how prediction changes if we hide part of the image:



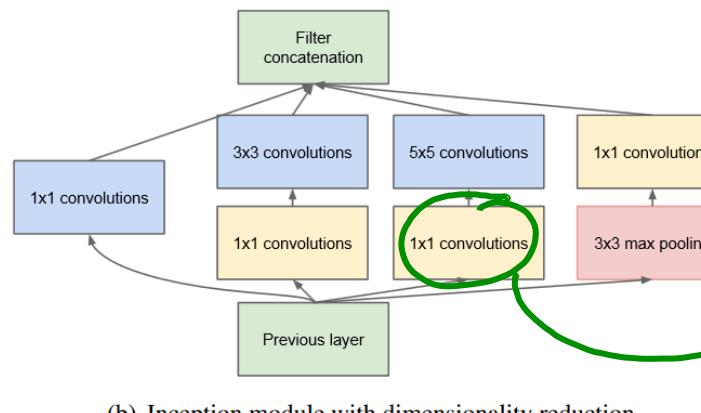
VGG Convolutional Neural Network

- Image 2014 “Localization” Task won by a **19-layer VGG** network:
 - 7.3% error for classification (2nd place).
 - Uses **3x3 convolution layers** with stride of 1:
 - 3x3 followed by 3x3 simulates a 5x5, and another 3x3 simulates a 7x7, and so on.
 - Speeds things up and reduces number of parameters.
 - Increases number of non-linear ReLU operations.
 - “Deep and simple”: variants of VGG are the most popular CNNs.



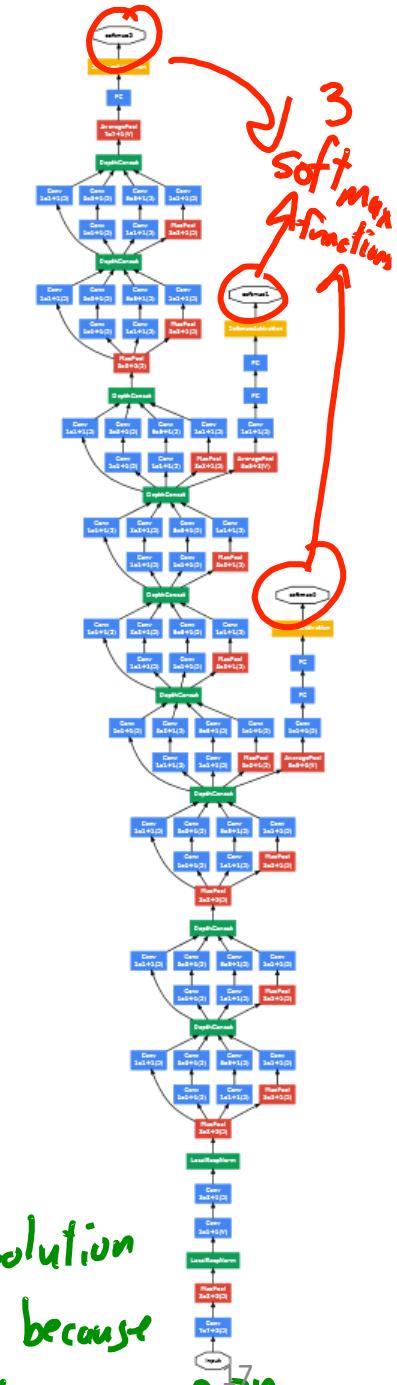
GoogLeNet

- Image 2014 classification task won by GoogLeNet:
 - 6.7% errors.
 - 22 layers
 - No fully connected layers.
 - During training, try to predict label at multiple locations.
 - During testing, just take the deepest predictions.
 - “Inception” modules: combine convolutions of different sizes.



(b) Inception module with dimensionality reduction

'1 | x |' convolution makes sense because they are first 2 dimensions of 3D conv.



ResNet

- Image 2015 won by Resnet (all 5 tasks):
 - 3.6% error (below estimate 5% human error).
 - 152 layers (2-3 weeks on 8 GPUs to train).
 - “Residual learning” allows better performance with deep networks:
 - Include input to layer in addition to non-linear transform.

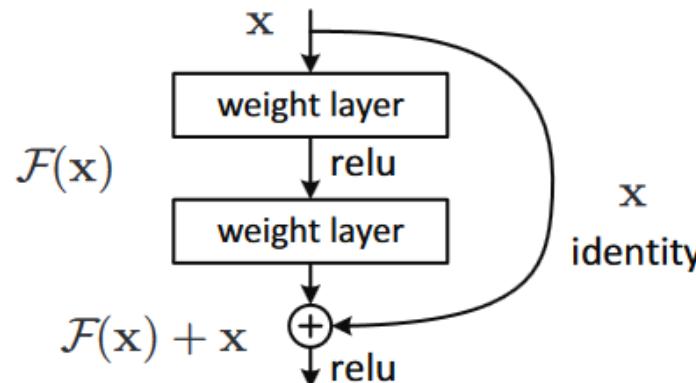
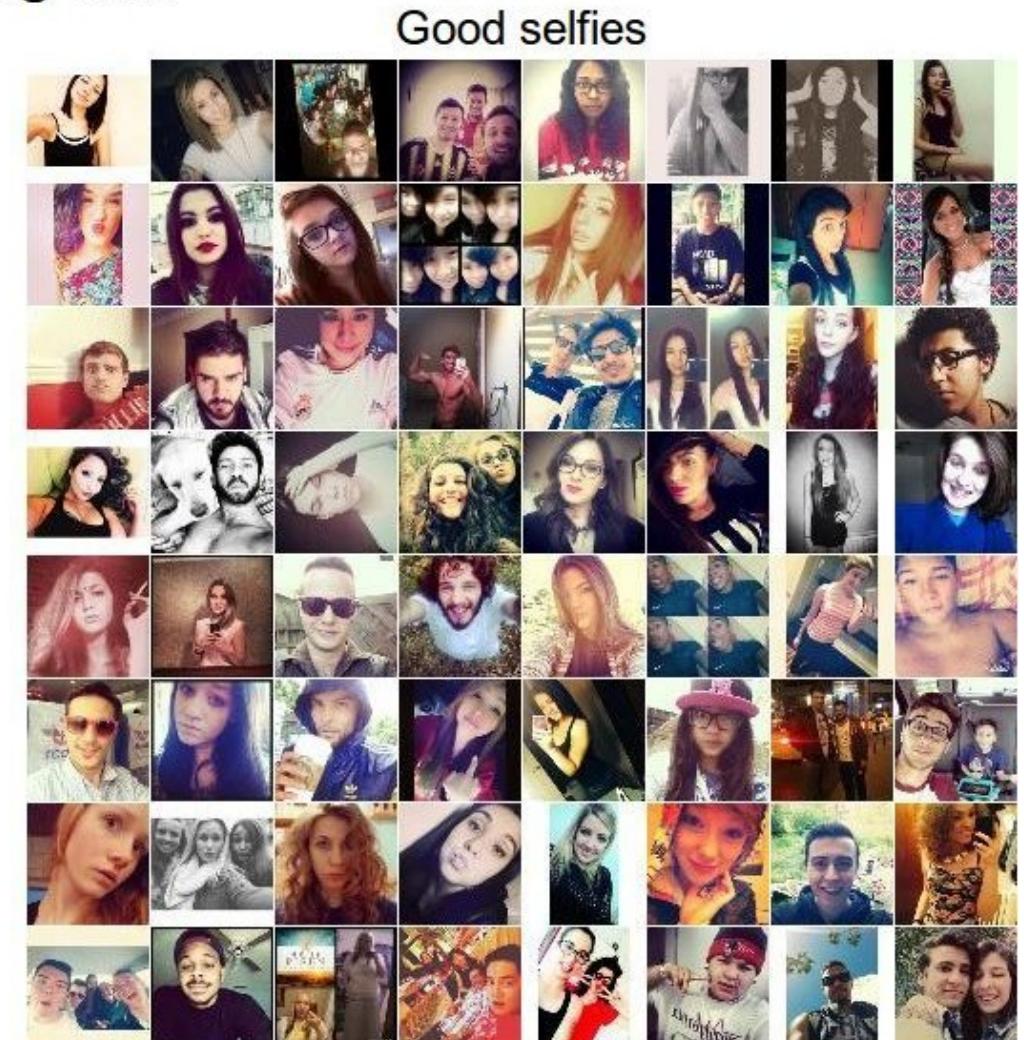
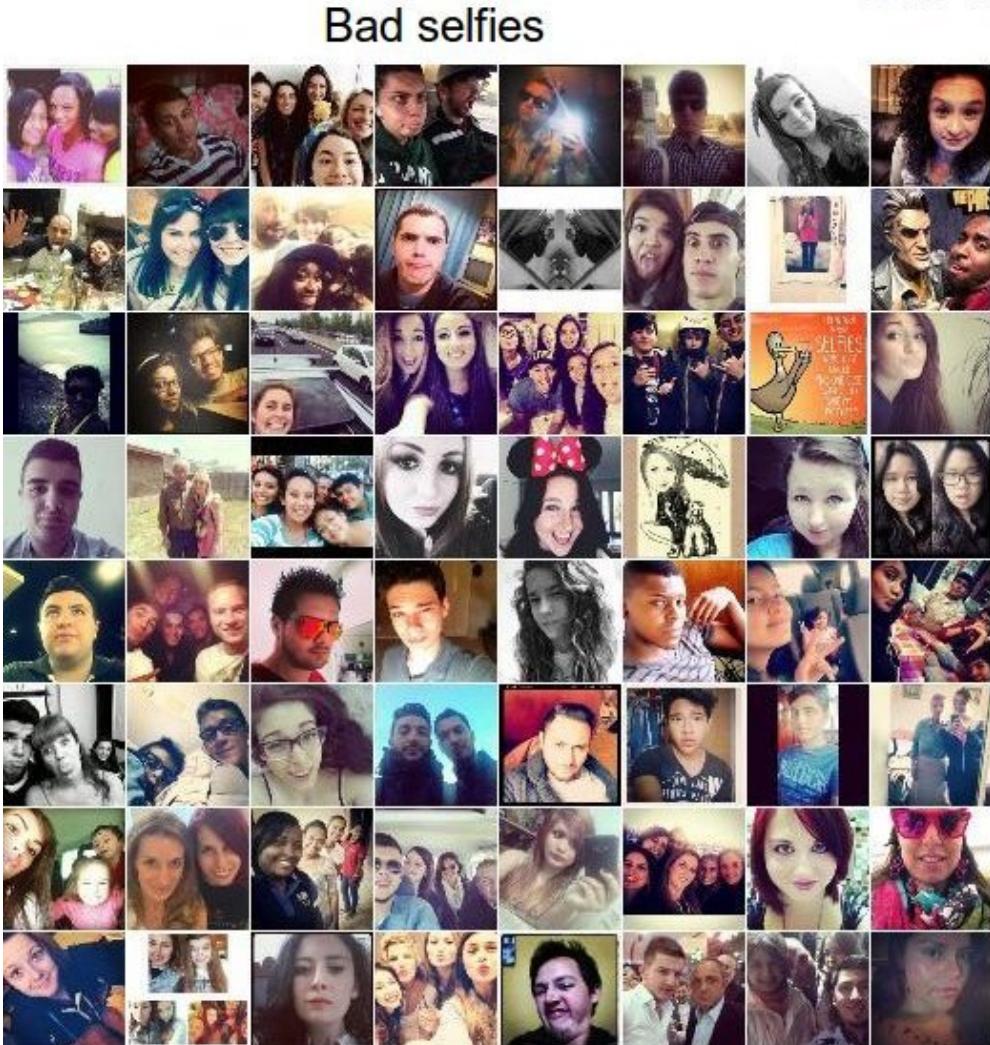


Figure 2. Residual learning: a building block.

- Network just focuses on “residual”: what is not captured in original signal.
- So if the learned weights are zero then the network is just the identity.

CNNs for Rating Selfies

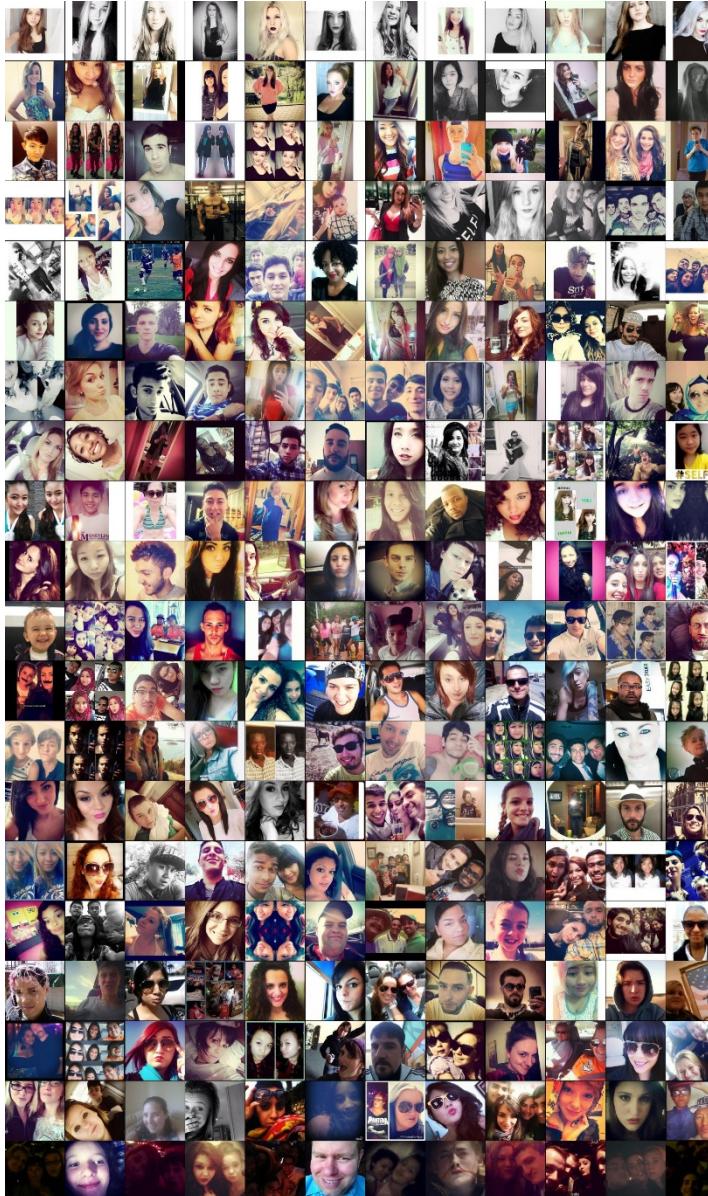
Our training data



CNNs for Rating Selfies

Do:

- Be female
- Have face be $\frac{1}{3}$ of image
- Cut off forehead
- Show long hair
- Oversaturate face
- Use filter
- Add border

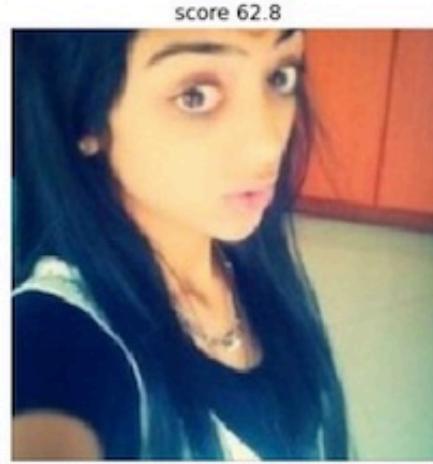
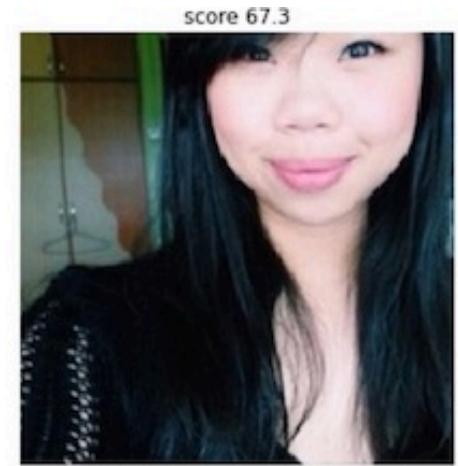
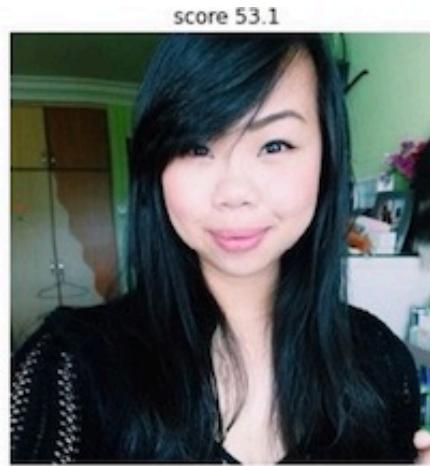
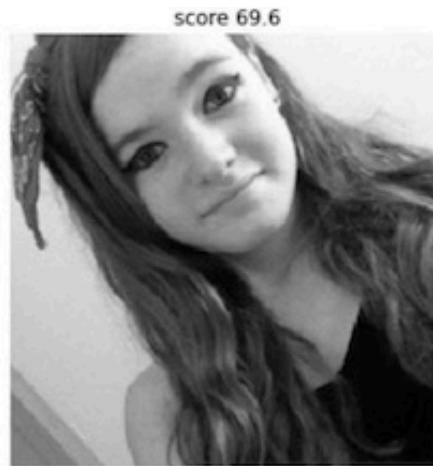
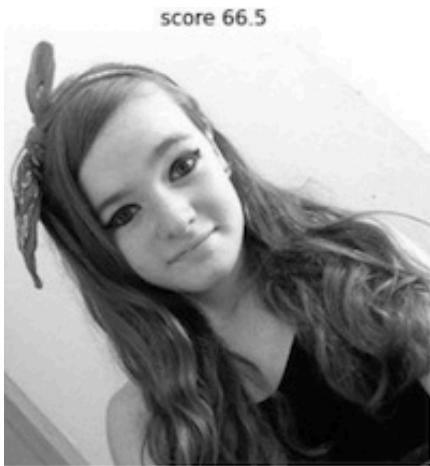


Don't:

- Use low lighting
- Make head too big
- Take group shots

CNNs for Rating Selfies

Finding best
image crop:



Beyond Classification

- “Fully convolutional” neural networks allow “dense” prediction:

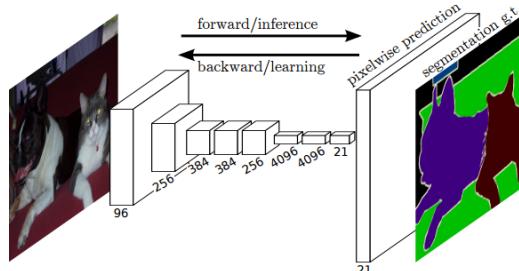


Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

- Best methods combine these with graphical models or LSTMs (CPSC 540).
- Image segmentation:

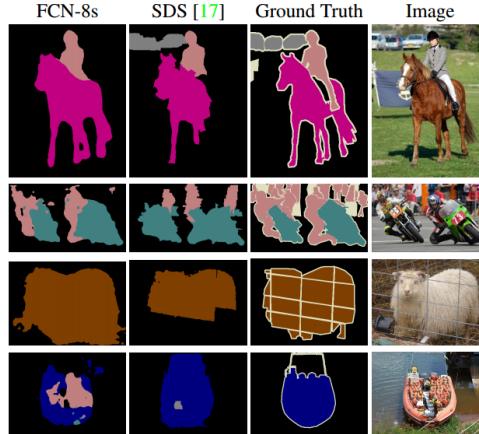


Figure 6. Fully convolutional segmentation nets produce state-of-the-art performance on PASCAL. The left column shows the output of our highest performing net, FCN-8s. The second shows the segmentations produced by the previous state-of-the-art system by Hariharan *et al.* [17]. Notice the fine structures recovered (first

Beyond Classification

- “Fully convolutional” neural networks allow “dense” prediction:

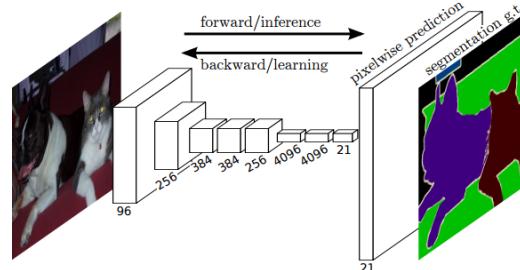
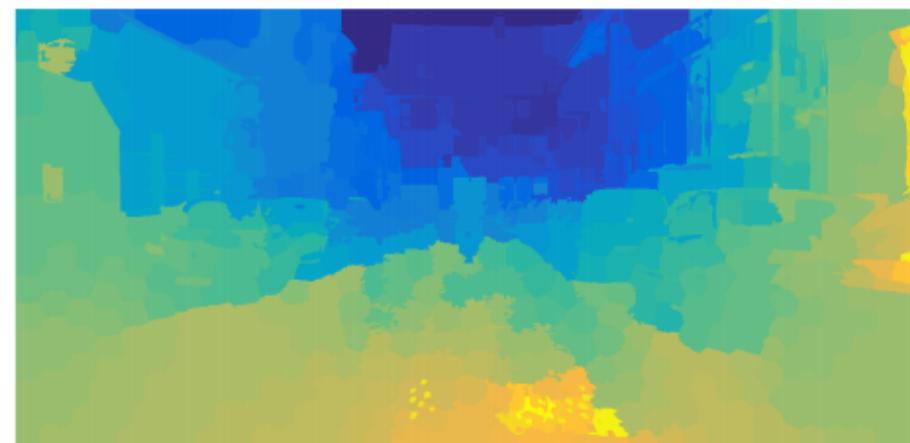


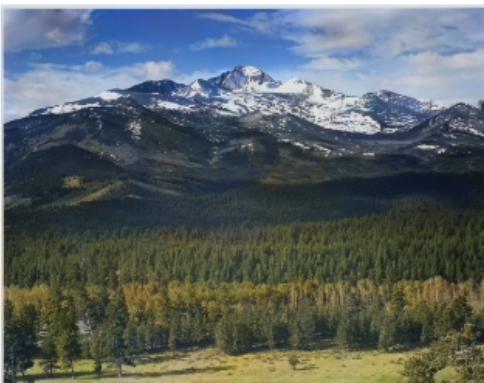
Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

- Best methods combine these with graphical models or LSTMs (CPSC 540).
- Depth Estimation:



Beyond Classification

- Image colorization:



Colorado National Park, 1941

Textile Mill, June 1937

Berry Field, June 1909

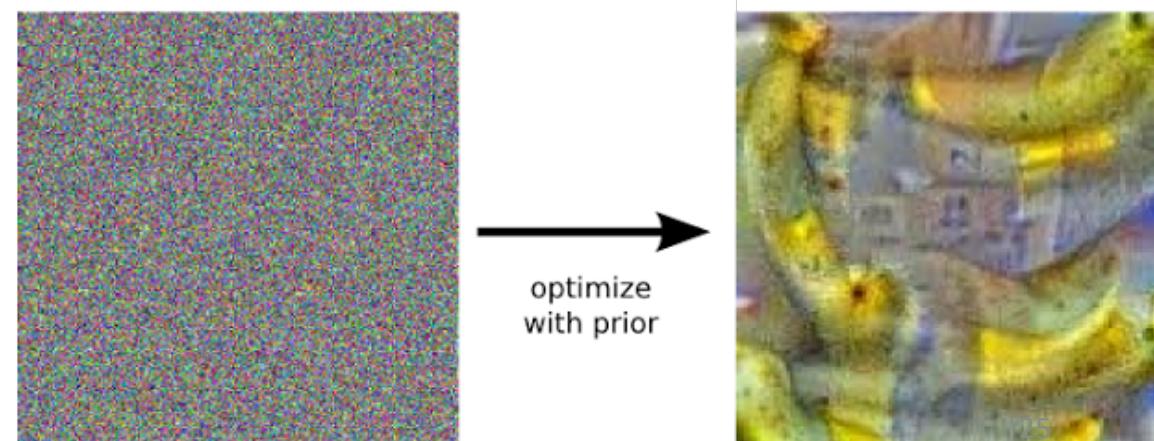
Hamilton, 1936

- [Image Gallery](#), [Video](#)

Inceptionism

- A crazy idea:
 - Instead of weights, use backpropagation to take gradient with respect to x_i .
- Inceptionism with trained network:
 - Fix the label y_i (e.g., “banana”).
 - Start with random noise image x_i .
 - Use gradient descent on image x_i .
 - Add a spatial regularizer on x_i :
 - Encourages neighbouring x_{ij} to be similar

“Show what you think a banana looks like.”



Inceptionism

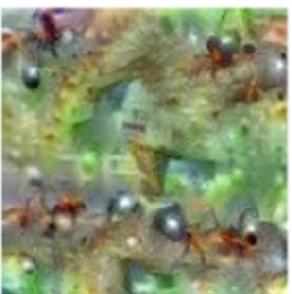
- Inceptionism for different class labels:



Hartebeest



Measuring Cup



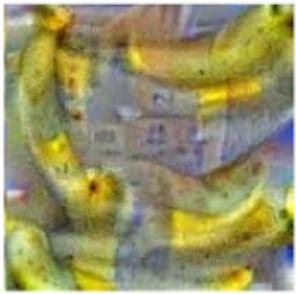
Ant



Starfish



Anemone Fish



Banana



Parachute



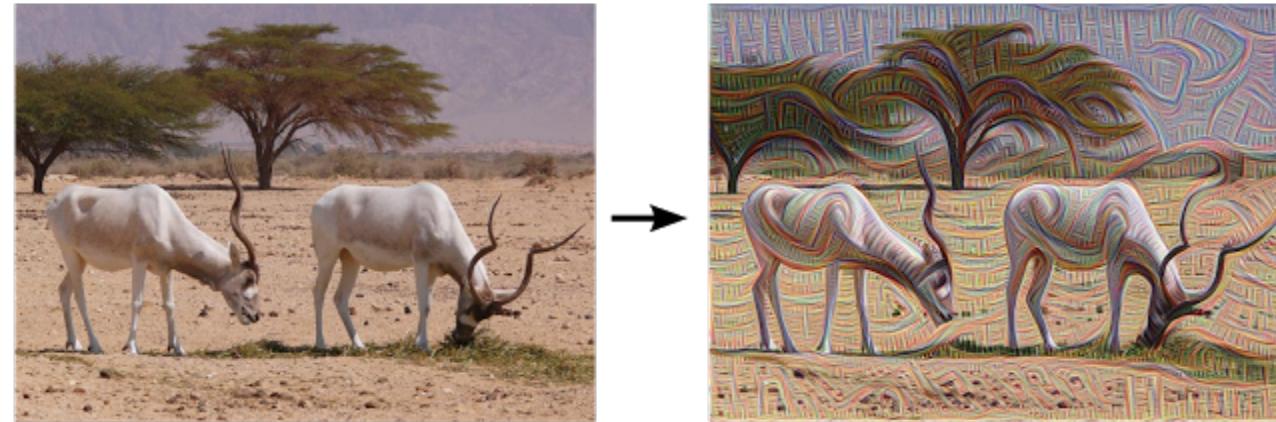
Screw

Dumbbell



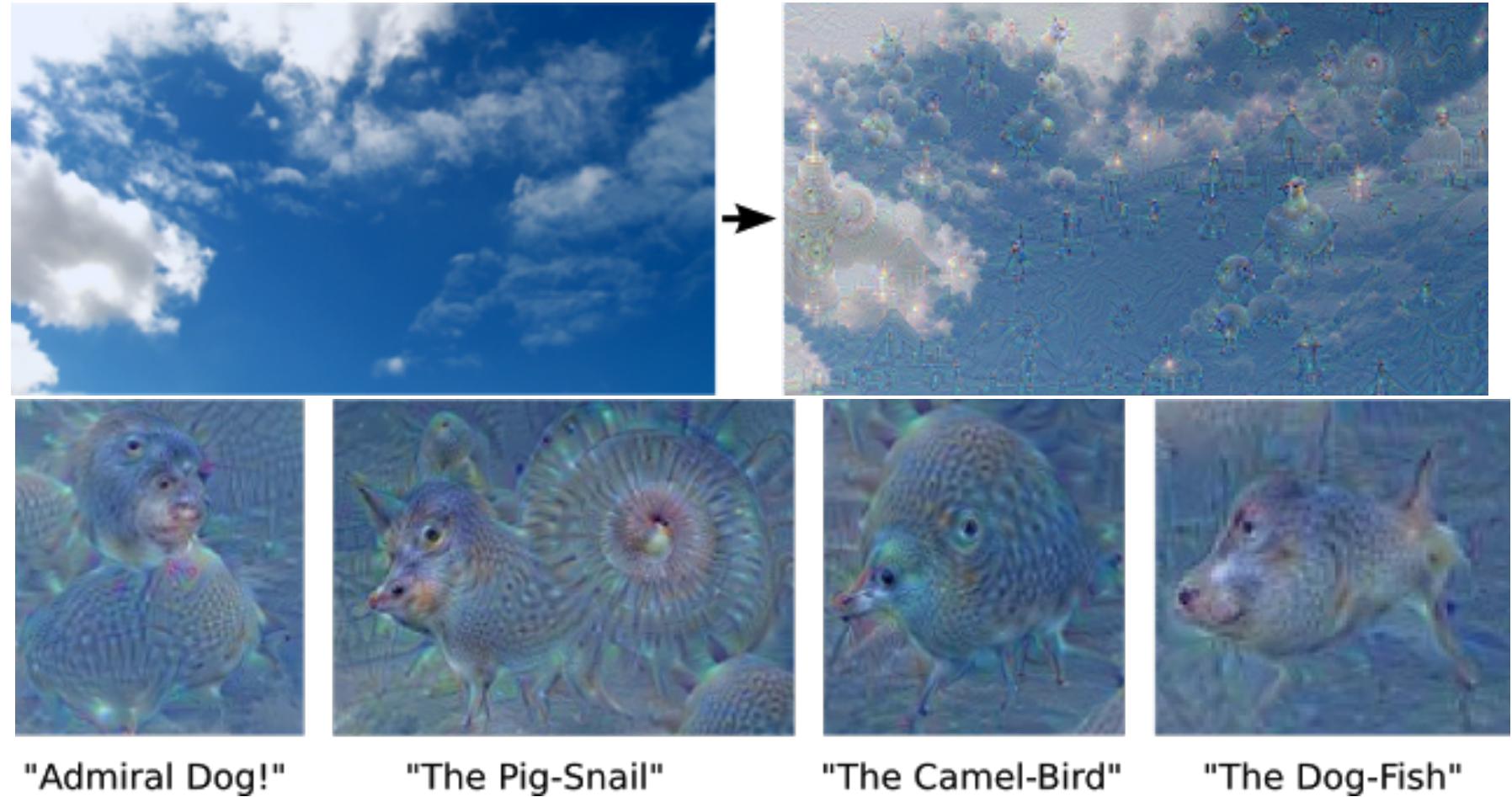
Inceptionism

- Inceptionism where we try to match $z_i^{(m)}$ values instead of y_i .
 - Shallow ‘m’:



Inceptionism

- Inceptionism where we try to match $z_i^{(m)}$ values instead of y_i .
 - Deepest ‘m’:



Inceptionism

- Inceptionism where we try to match $z_i^{(m)}$ values instead of y_i .
 - “Deep dream” starts from random noise:



- Inceptionism gallery
- [Deep Dream video](#)

Artistic Style Transfer

- Artistic style transfer:
 - Given a content image ‘C’ and a style image ‘S’.
 - Make a image that has content of ‘C’ and style of ‘S’.

Content:



Style:



Artistic Style Transfer

- Artistic style transfer:
 - Given a content image ‘C’ and a style image ‘S’.
 - Make a image that has content of ‘C’ and style of ‘S’.
- CNN-based approach applies gradient descent with 2 terms:
 - Loss function: match deep latent representation of content image ‘C’:
 - Difference between $z_i^{(m)}$ for deepest ‘m’ between x_i and ‘C’.
 - Regularizer: match all latent representation covariances of style image ‘S’.
 - Difference between covariance of $z_i^{(m)}$ for all ‘m’ between x_i and ‘C’.

Artistic Style Transfer

A



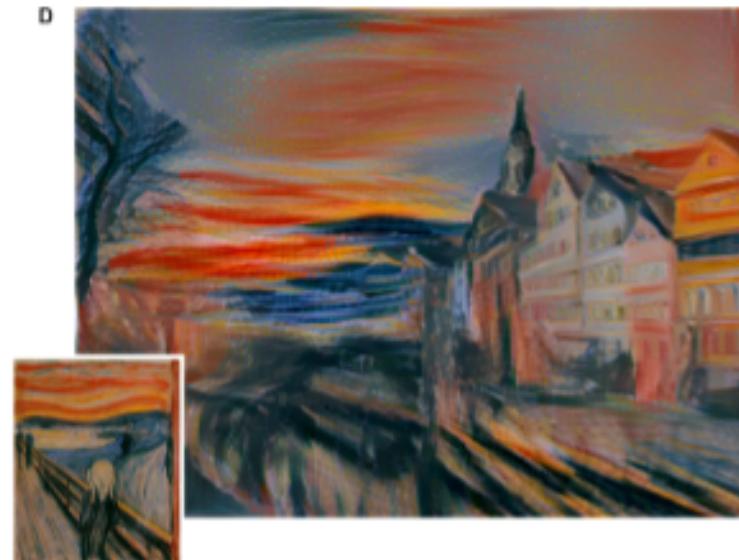
B



C



D



[Image Gallery](#)

Examples



Figure: **Left:** My friend Grant, **Right:** Grant as a pizza

Artistic Style Transfer

- Recent methods combine CNNs with graphical models (CPSC 540):



Input A



Input B



Content A + Style B



Content B + Style A

Artistic Style Transfer

- Recent methods combine CNNs with graphical models (CPSC 540):



Input style



Input content



Ours



Artistic Style Transfer for Video

- Combining style transfer with optical flow:
 - <https://www.youtube.com/watch?v=Khuj4ASldmU>
- Videos from Ricky's paper:



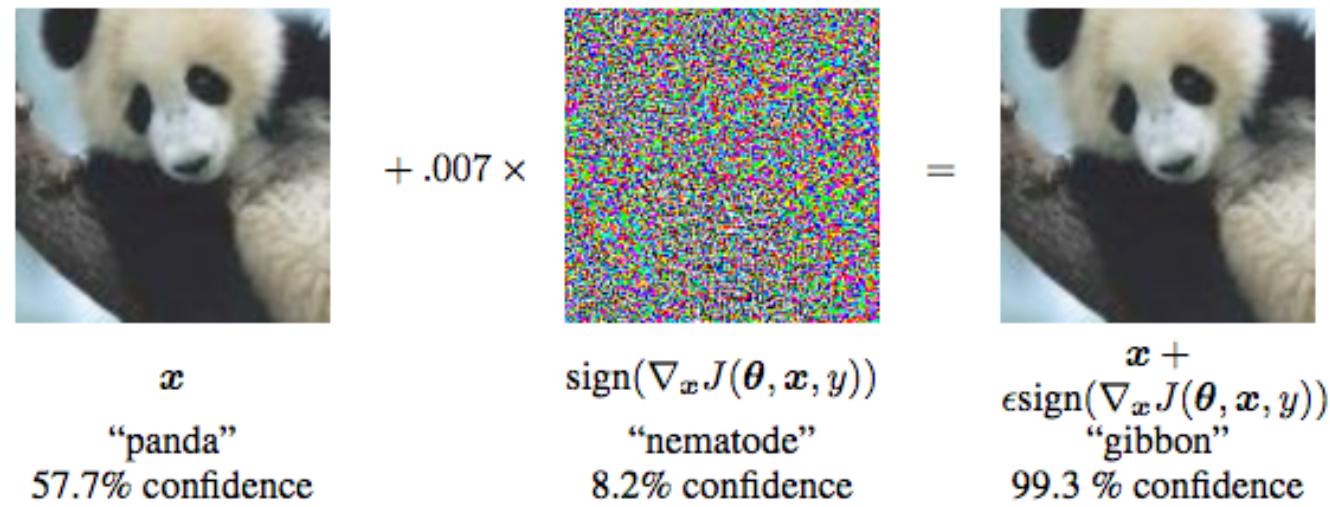
Mission Accomplished?

- For speech recognition and object detection:
 - No other methods have ever given the current level of performance.
 - But, we also don't know how to scale up other universal approximators.
 - There is likely some overfitting to these particular tasks.
- Despite high-level of abstraction, deep CNNs are easily fooled:
 - But progress on fixing ‘blind spots’.
- Do we really need 1000 training images for every object?



Fooling CNNs: Adversarial Examples

- Recent work: imperceptible noise that changes the predicted label
- Procedure: optimize the label with respect to the noise added



- Works disturbingly well at tricking a trained network
- This raises serious AI safety questions
 - Can a malicious person repaint a stop sign and fool self-driving cars?