

Lecture Notes for **Machine Learning in Python**

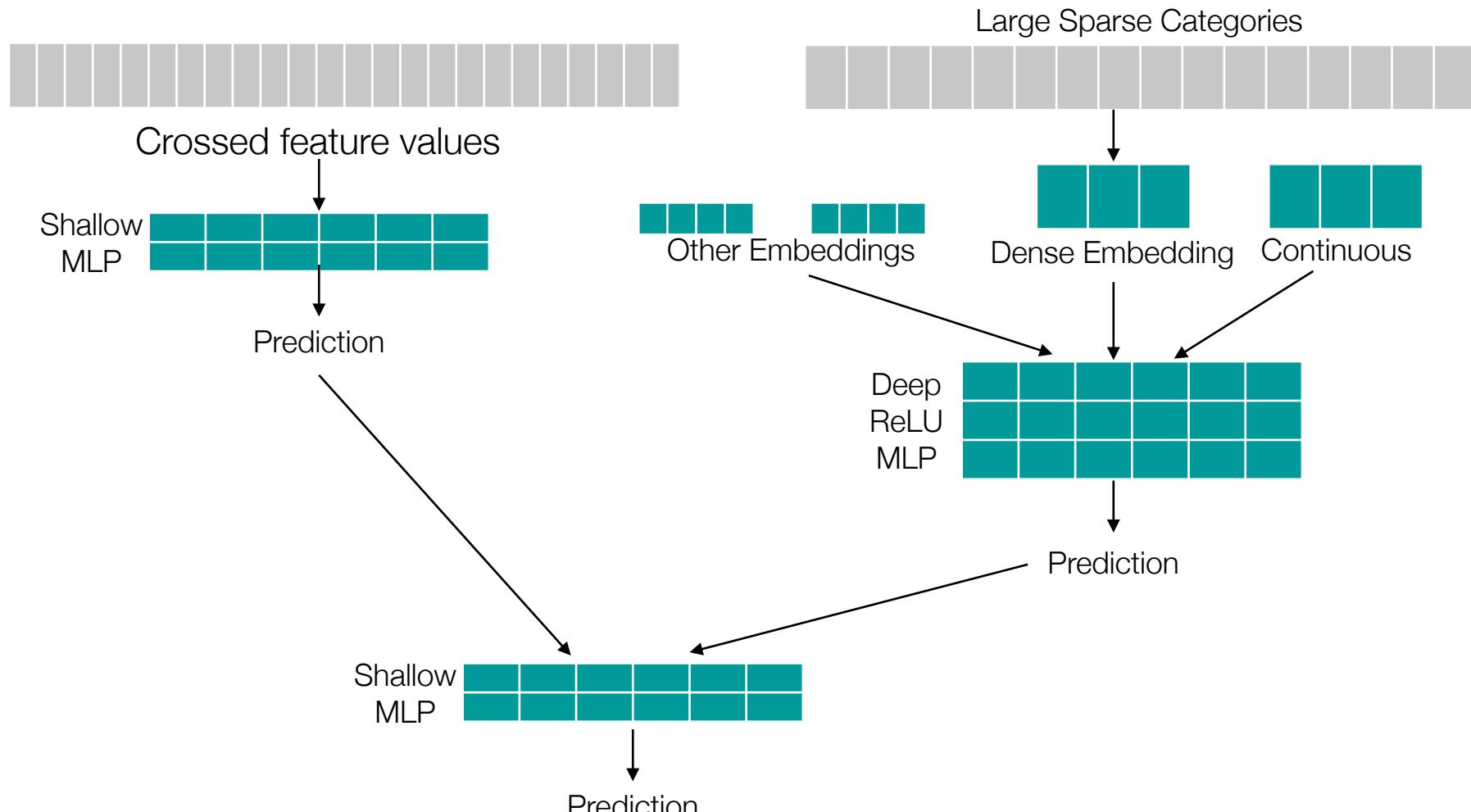
Professor Eric Larson
Basic Convolutional Neural Networks

Logistics and Agenda

- Logistics
 - Wide/Deep due date
- Agenda
 - Wide/Deep Town Hall
 - Basic CNN architectures and Demo

Last Time:

- Deep refers to increasingly smaller hidden layers
- Embed into sparse representations via ReLU



Town Hall, Wide and Deep



WHEN VISITING A NEW HOUSE, IT'S
GOOD TO CHECK WHETHER THEY HAVE
AN ALWAYS-ON DEVICE TRANSMITTING
YOUR CONVERSATIONS SOMEWHERE.

Convolutional Neural Networks

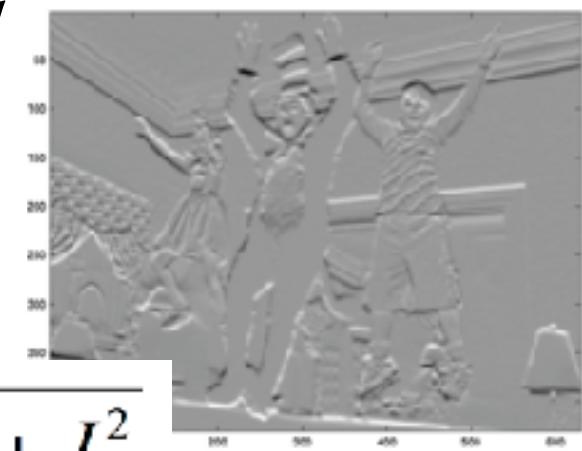
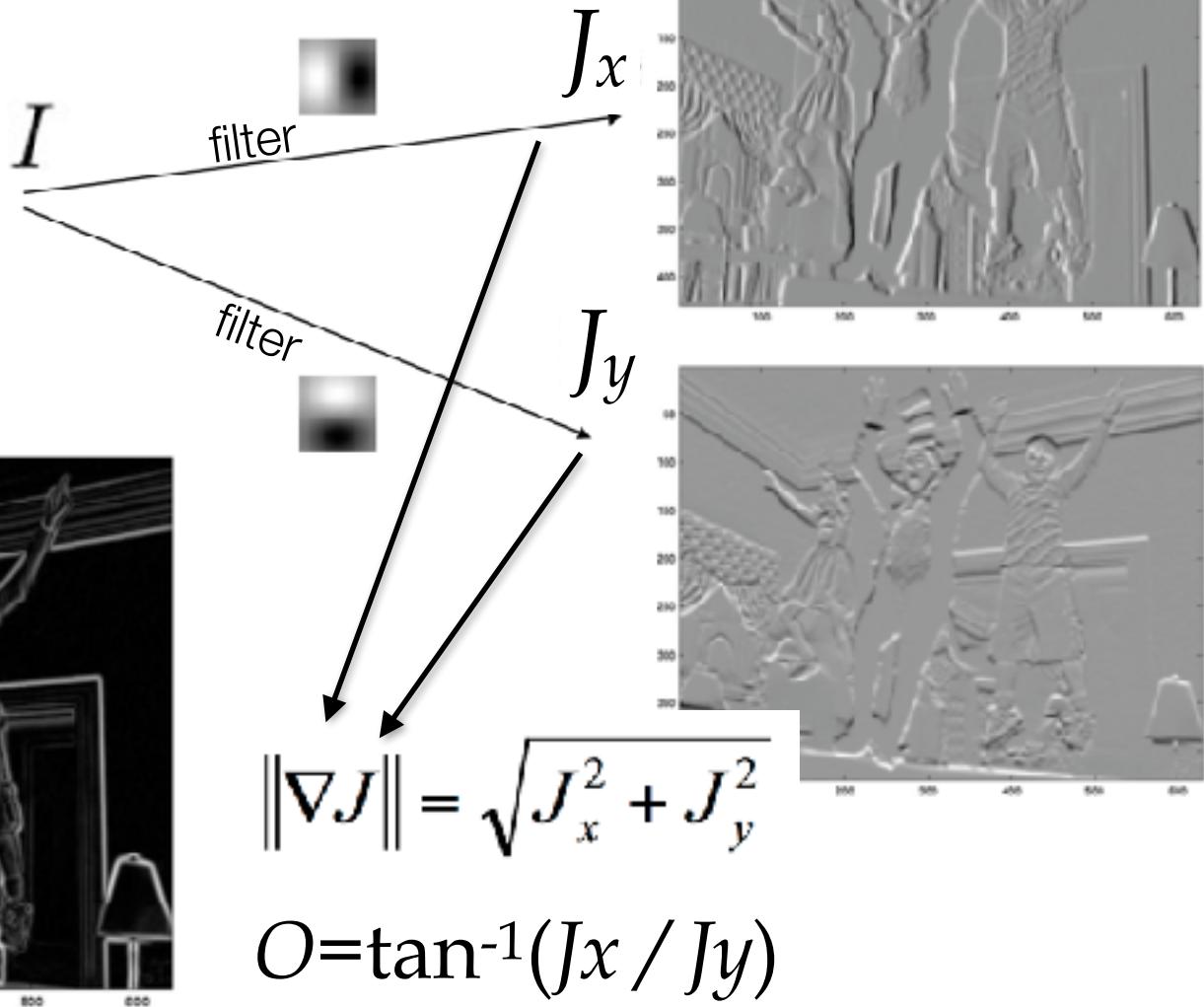


IN CS, IT CAN BE HARD TO EXPLAIN
THE DIFFERENCE BETWEEN THE EASY
AND THE VIRTUALLY IMPOSSIBLE.

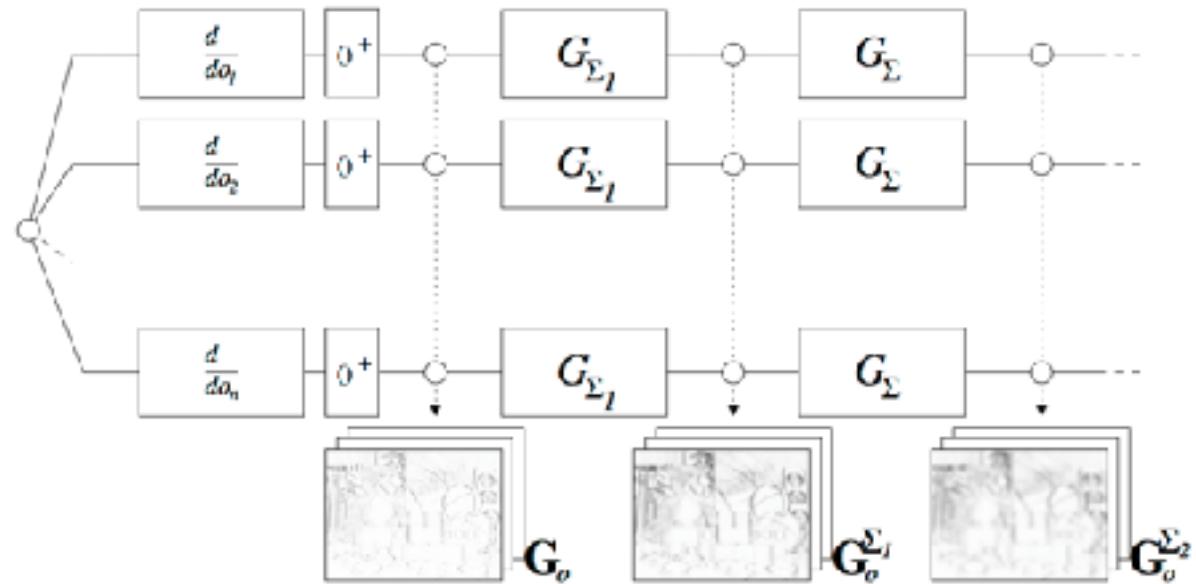
Reminder Convolution

What we did before

- the gradient (2D derivative)



What we did before



take normalized histogram at point u, v

$$\tilde{\mathbf{h}}_{\Sigma}(u, v) = \left\| [\mathbf{G}_1^{\Sigma}(u, v), \dots, \mathbf{G}_H^{\Sigma}(u, v)]^T \right\|$$

$$\mathcal{D}(u_0, v_0) =$$

$$[$$

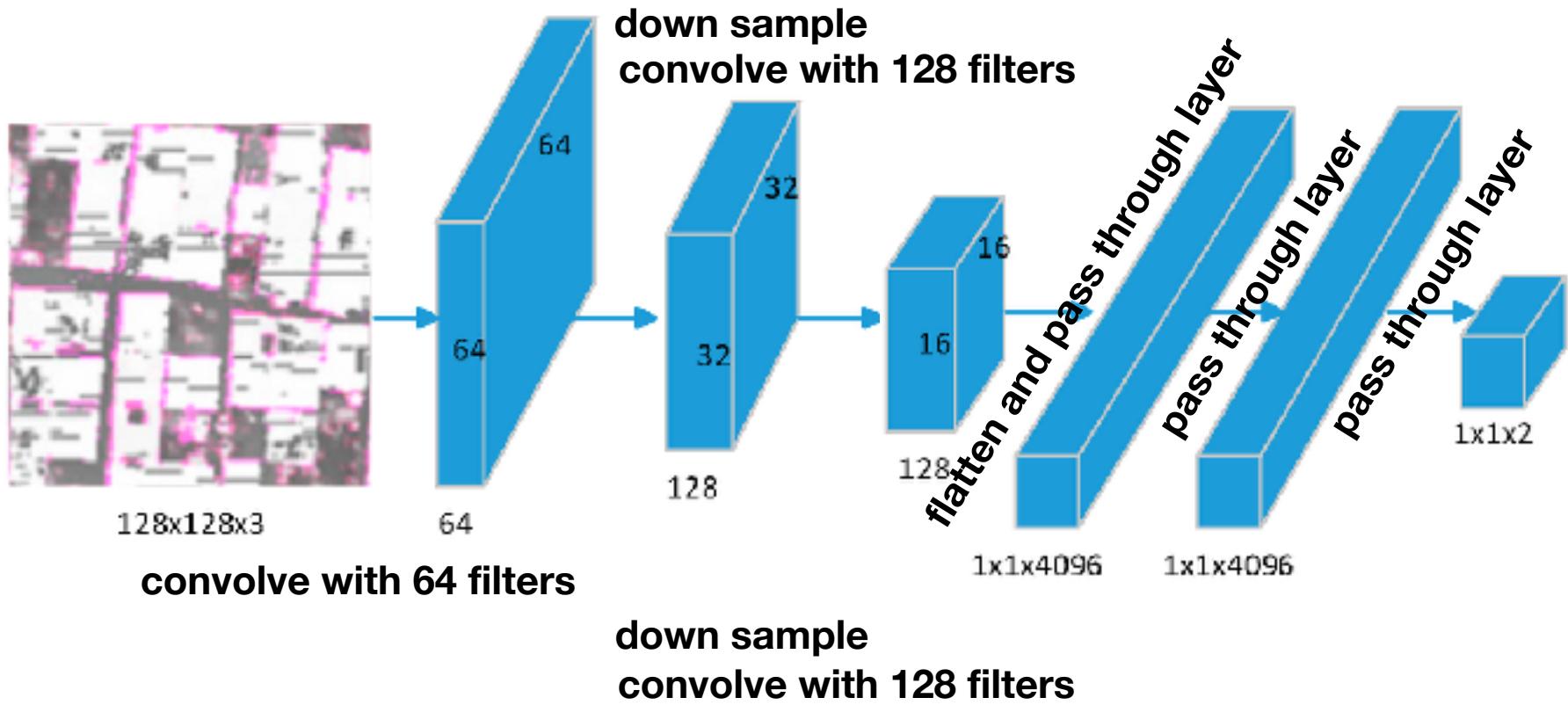
$$\tilde{\mathbf{h}}_{\Sigma_1}^{\top}(u_0, v_0),$$

$$\tilde{\mathbf{h}}_{\Sigma_1}^{\top}(\mathbf{l}_1(u_0, v_0, R_1)), \dots, \tilde{\mathbf{h}}_{\Sigma_1}^{\top}(\mathbf{l}_T(u_0, v_0, R_1)),$$

$$\tilde{\mathbf{h}}_{\Sigma_2}^{\top}(\mathbf{l}_1(u_0, v_0, R_2)), \dots, \tilde{\mathbf{h}}_{\Sigma_2}^{\top}(\mathbf{l}_T(u_0, v_0, R_2)),$$

Tola et al. "Daisy: An efficient dense descriptor applied to wide-baseline stereo." Pattern Analysis and Machine Intelligence, IEEE Transactions

Anatomy of a convolution

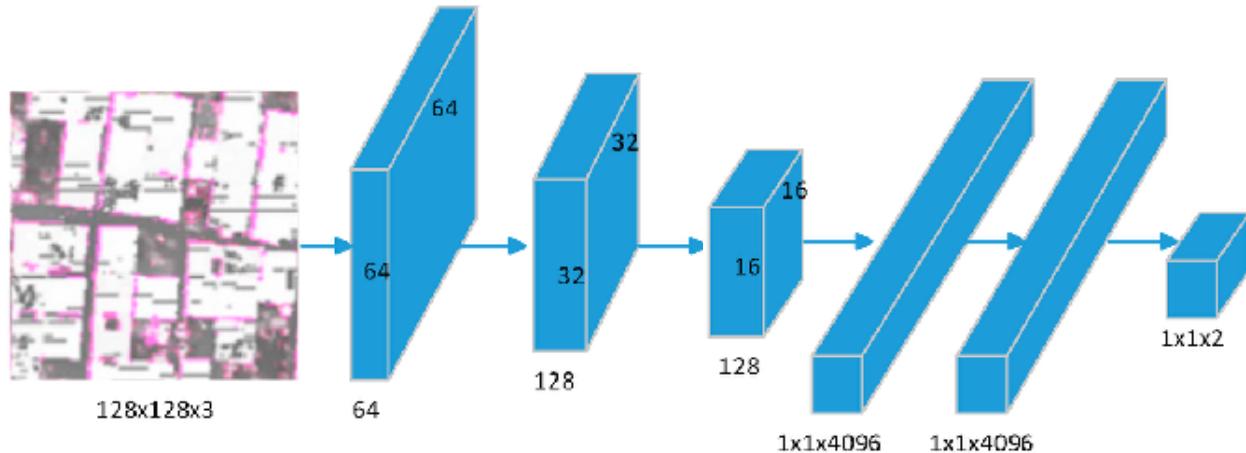


Blue Tensors: Outputs of Each Layer

Learned Params: Weights in Each Filter and Fully Connected Layer

CNN Overview

- First layer(s):
 - convolution
 - nonlinearity
 - pooling
 - Each pooling layer can make the input image “smaller”
 - allows for “Information Distillation”
 - less dependence on exact pixels
- Final layers are densely connected
 - typically multi-layer perceptrons

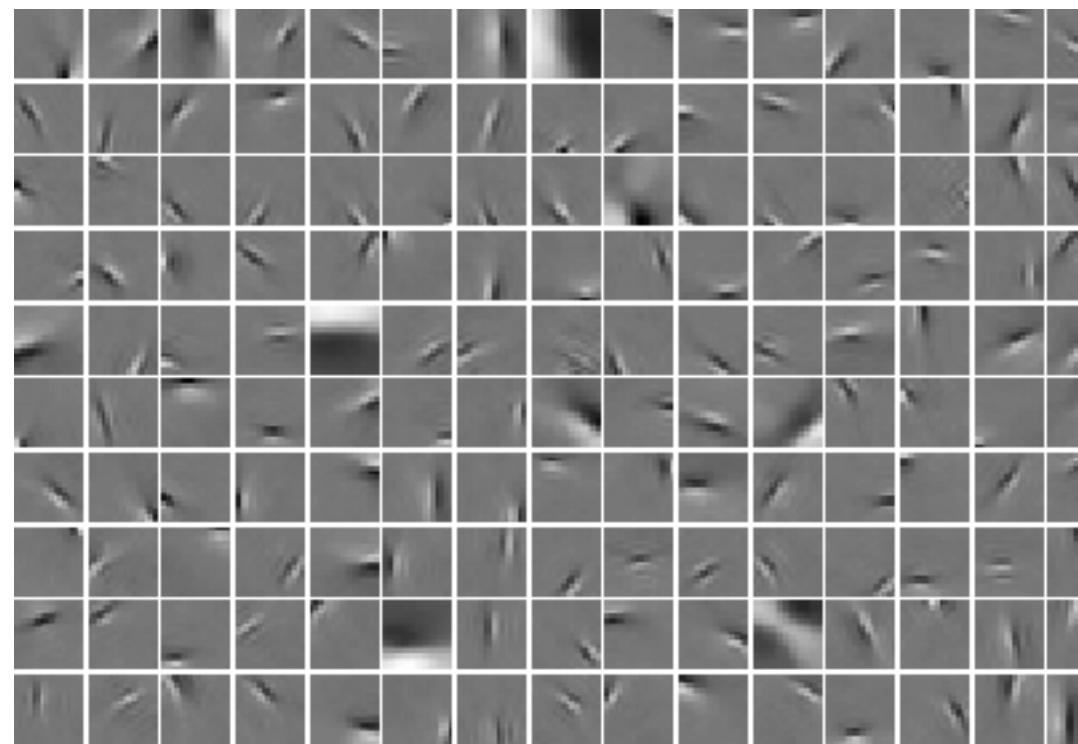
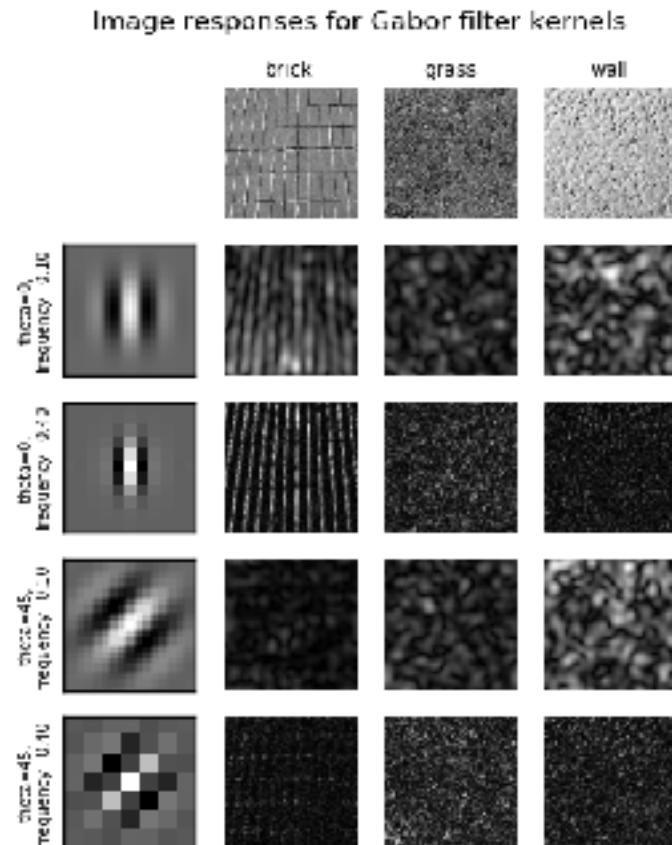


CNN Overview: Self Test

- First layer(s):
 - convolution
 - nonlinearity
 - pooling
 - Each pooling layer *can* make the input image “smaller”
 - allows for “Information Distillation”
 - less dependence on exact pixels
- Final layers are densely connected
 - typically multi-layer perceptrons
- Where are unstable gradients **most** problematic?
 - (A) During Convolution Layer(s) updates
 - (B) During Fully Connected Layer(s) updates
 - (C) Both A and B
 - (D) They are not a problem

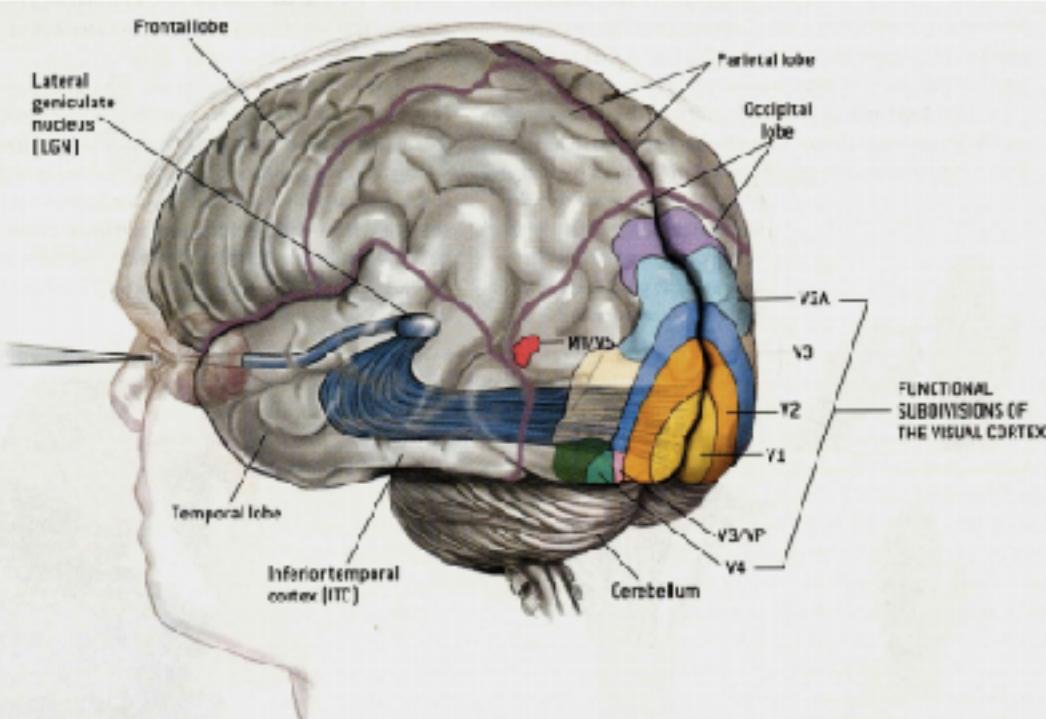
CNN Filtering

- Why perform lots of filtering?
 - recall gabor filtering?



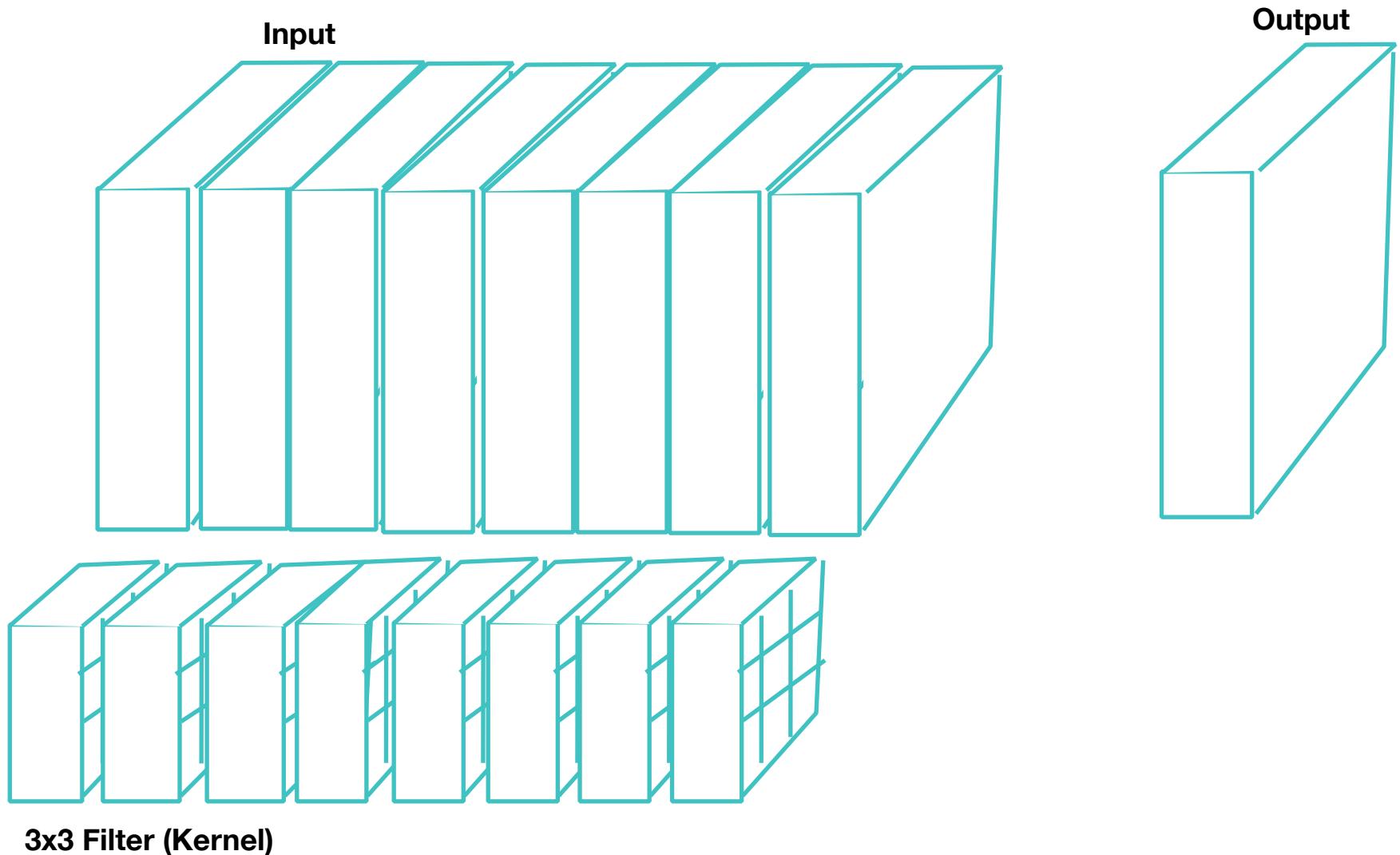
CNN Filtering

- Why perform lots of filtering?
 - recall gabor filtering?



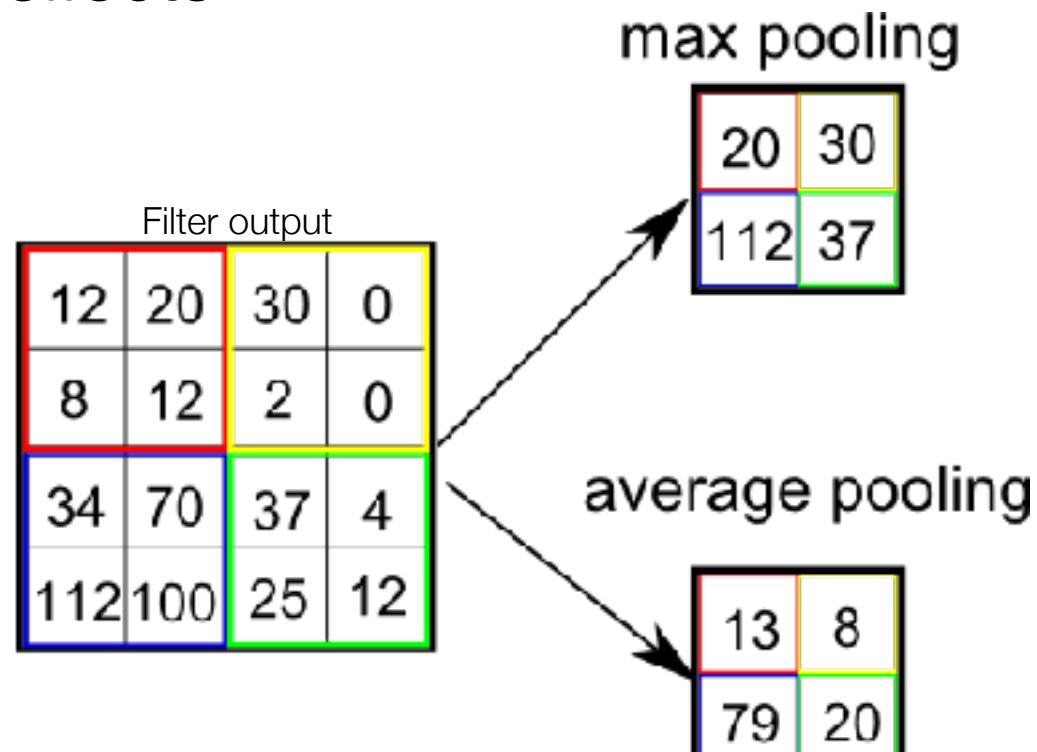
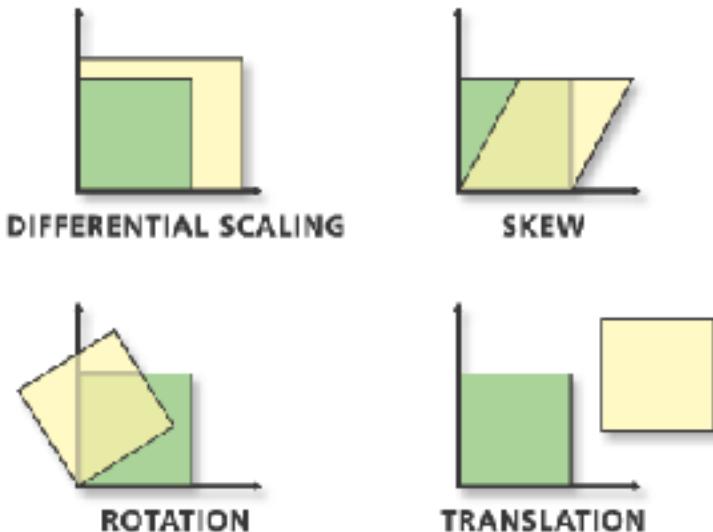
V1	Motion
V2	Stereo
V3	Color
V3a	Texture segregation
V3b	Segmentation, grouping
V4	Recognition
V7	Face recognition
MT	Attention
MST	Working memory/mental imagery

Convolution in a CNN

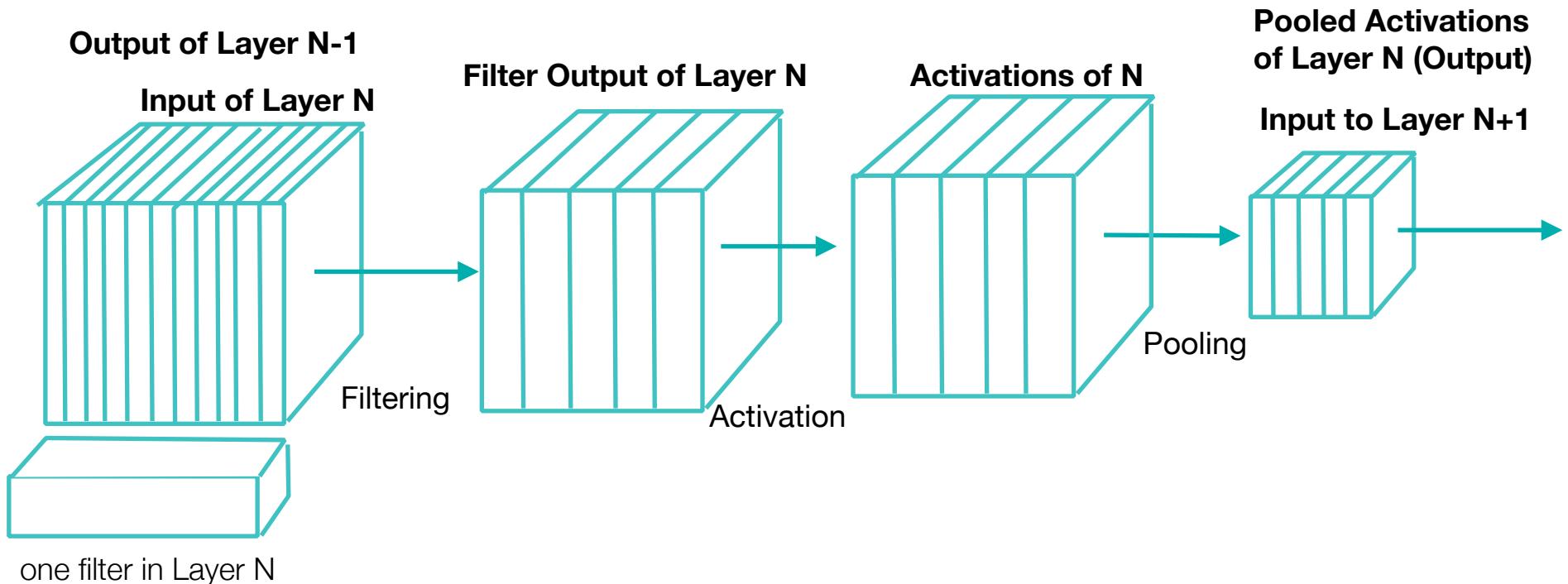


CNN Pooling

- Why perform pooling?
- Why max pooling?
 - reduce translation effects
 - param reduction

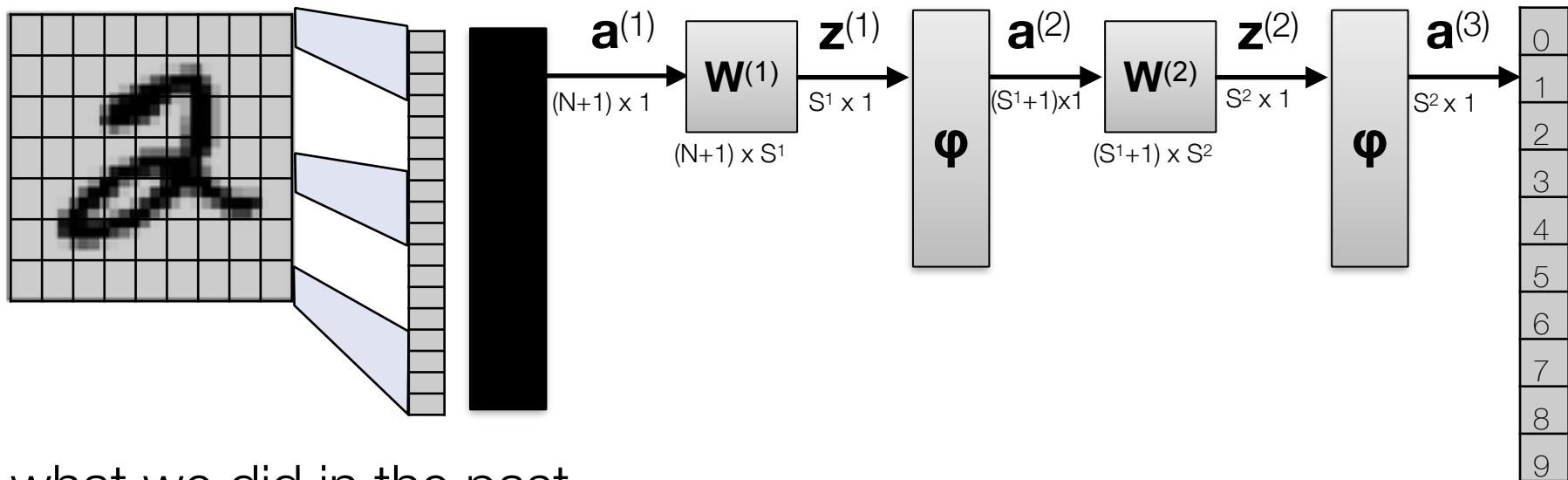


CNNs: Putting it together



Structure of Each Tensor: Channels x Rows x Columns

Simple Example: From Fully Connected to CNN



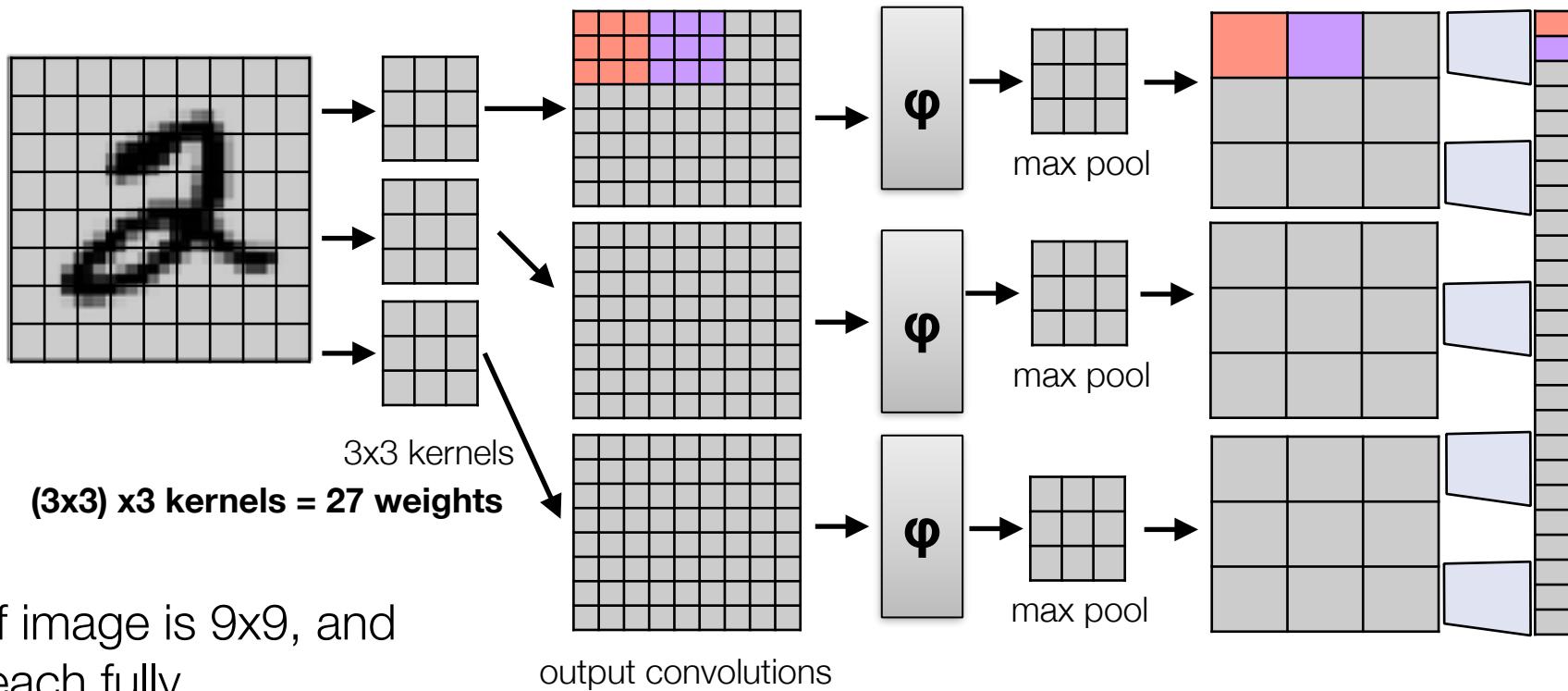
what we did in the past

If image is 9×9 , and each fully connected layer is 20 hidden neurons wide, how many parameters are in this NN (ignore bias)?

$$(K^2 \times 20) + (20 \times 10) = 200 + 20 K^2$$

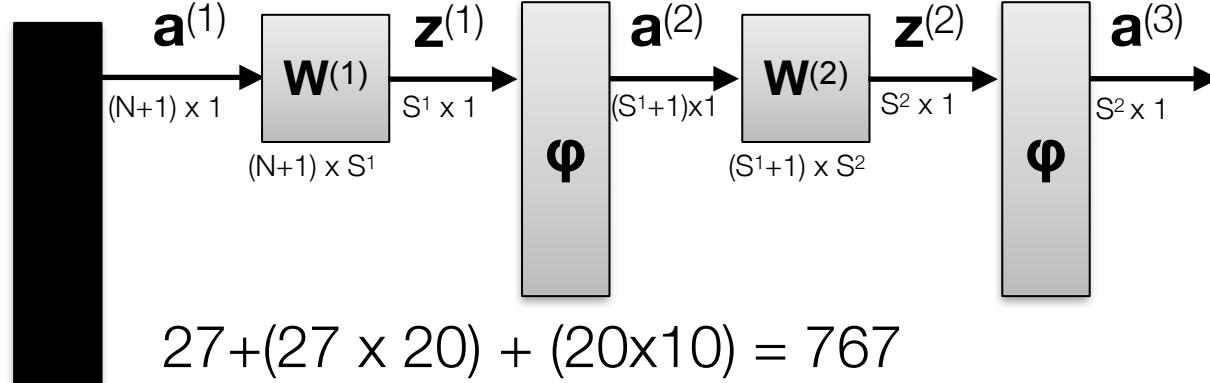
$$\text{for } 9 \times 9 = 200 + 20 \times 9^2 = 1,820 \text{ parameters}$$

Simple Example: From Fully Connected to CNN



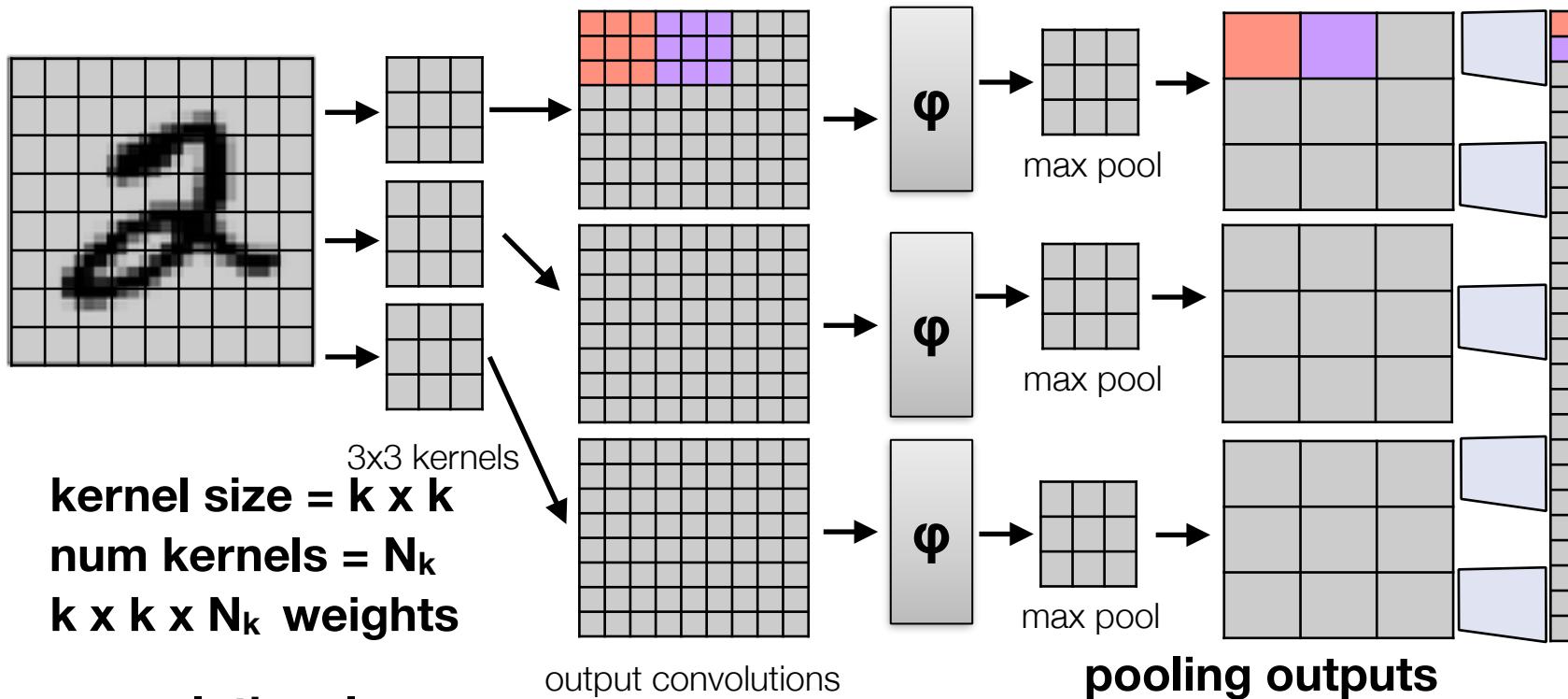
If image is 9x9, and each fully connected layer is 20 hidden neurons wide, how many parameters are in this NN (ignore bias)?

$$3 \times 3 \times 3 = 27 \text{ weights}$$



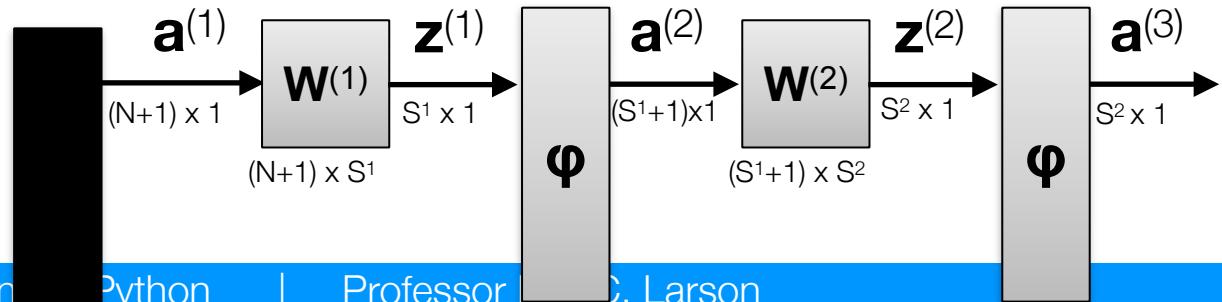
0
1
2
3
4
5
6
7
8
9
18

Simple Example: From Fully Connected to CNN

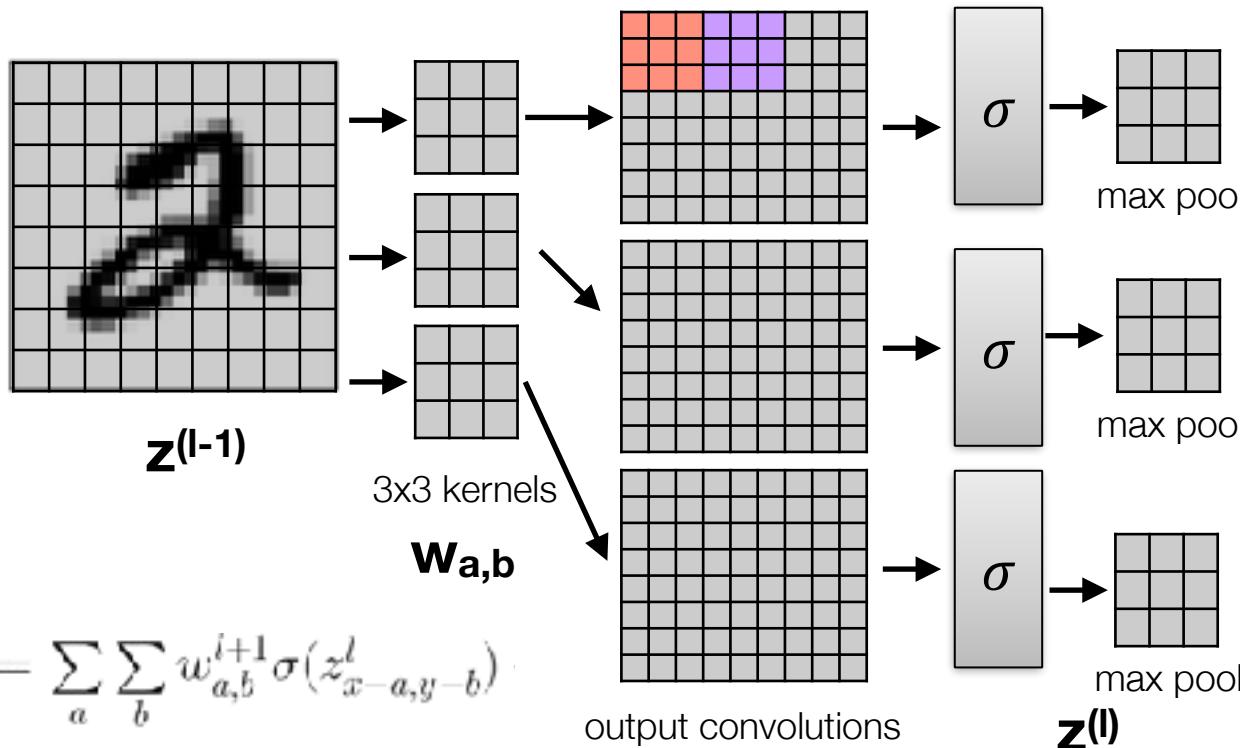


Input to MLP
 $N_k \times (K^2/k^2)$

num filters
image dimension



CNN gradient



Derivative of max pool is easy:

for each input z_i

$$f'(z_i) = \begin{cases} 1 & \text{if } z_i \text{ is max} \\ 0 & \text{else} \end{cases}$$

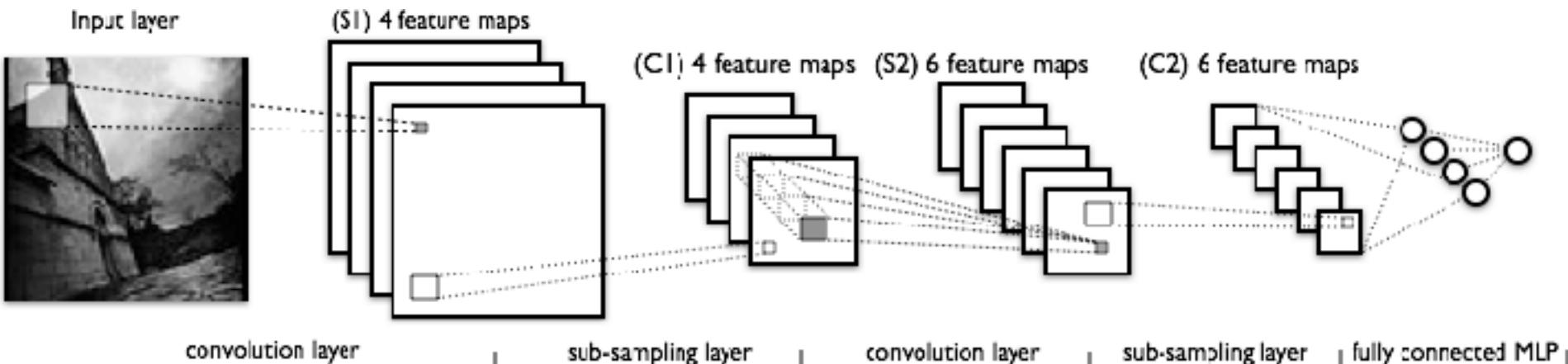
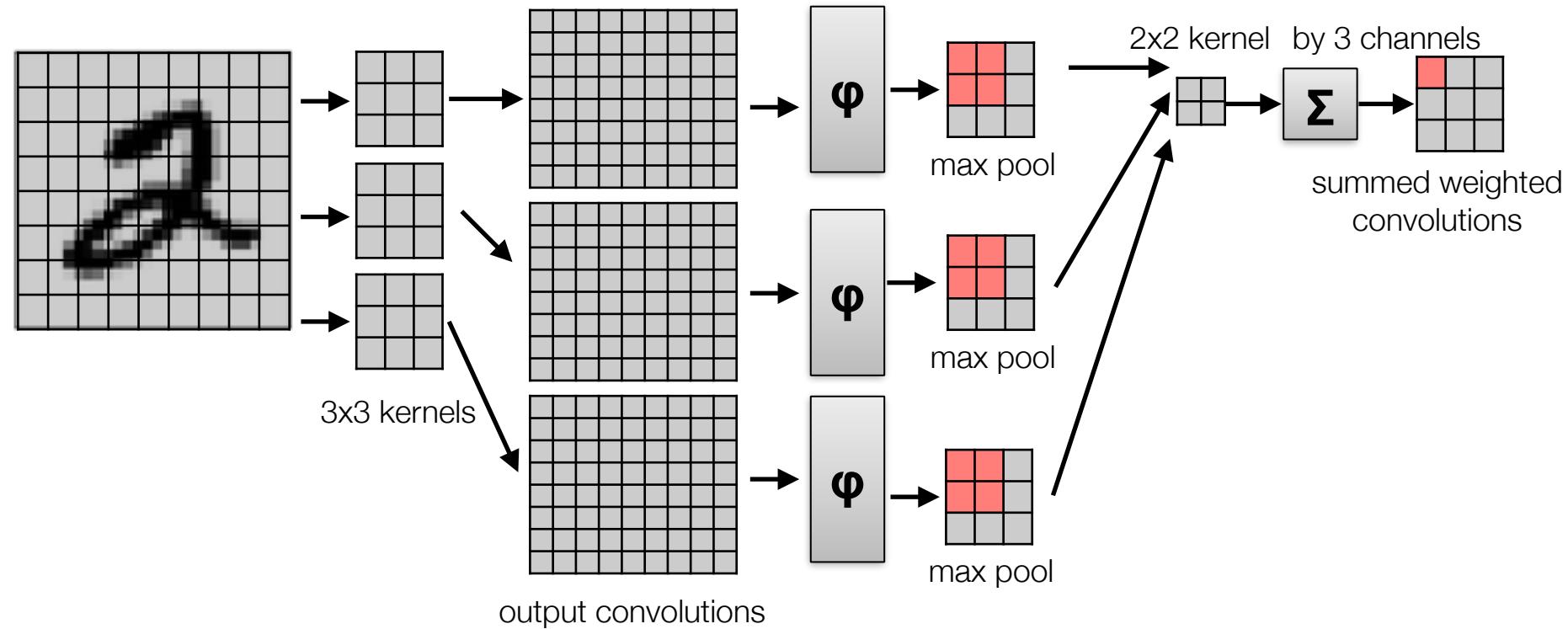
Derivative of convolution is more involved:

$$\frac{\partial C}{\partial w_{a,b}^l} = \sum_x \sum_y \frac{\partial C}{\partial z_{x,y}^l} \frac{\partial z_{x,y}^l}{\partial w_{a,b}^l} = \sum_x \sum_y \delta_{x,y}^l \frac{\partial (\sum_{a'} \sum_{b'} w_{a',b'}^l \sigma(z_{x-a',y-b'}^l) + b_{x,y}^l)}{\partial w_{a,b}^l} =$$
$$\sum_x \sum_y \delta_{x,y}^l \sigma(z_{x-a,y-b}^{l-1}) = \delta_{a,b}^l * \sigma(z_{-a,-b}^{l-1}) = \delta_{a,b}^l * \sigma(ROT180(z_{a,b}^{l-1}))$$

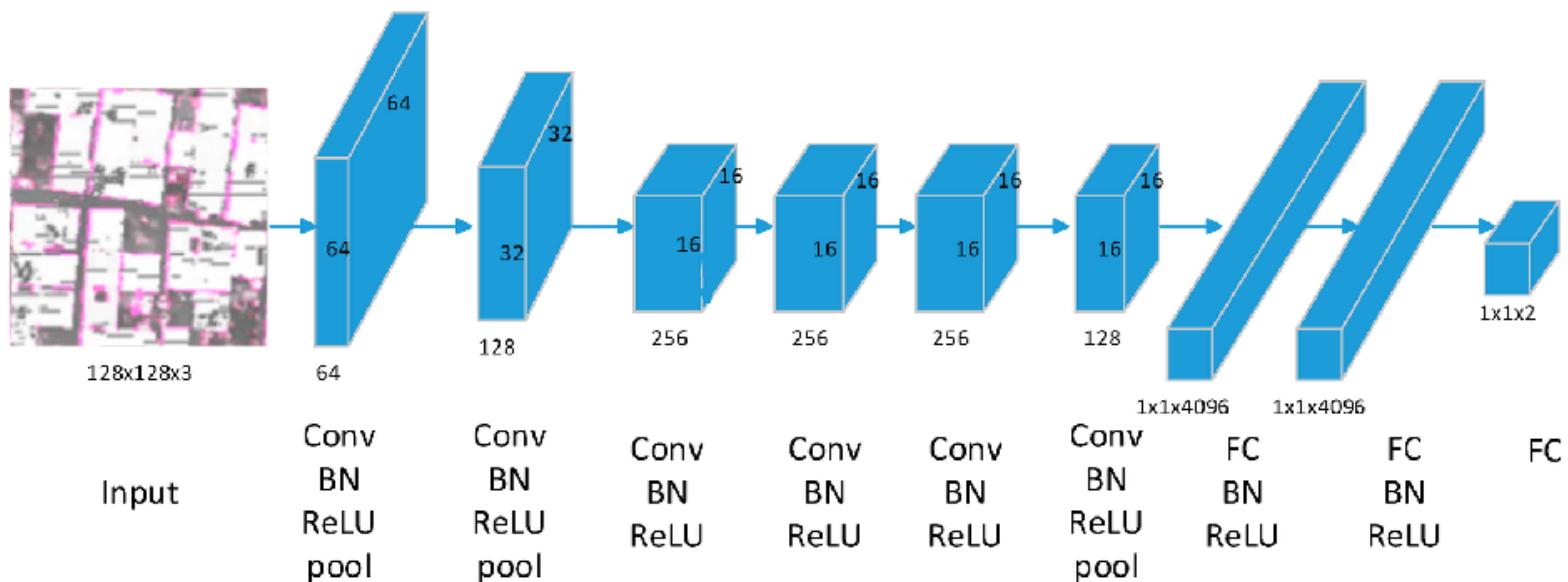
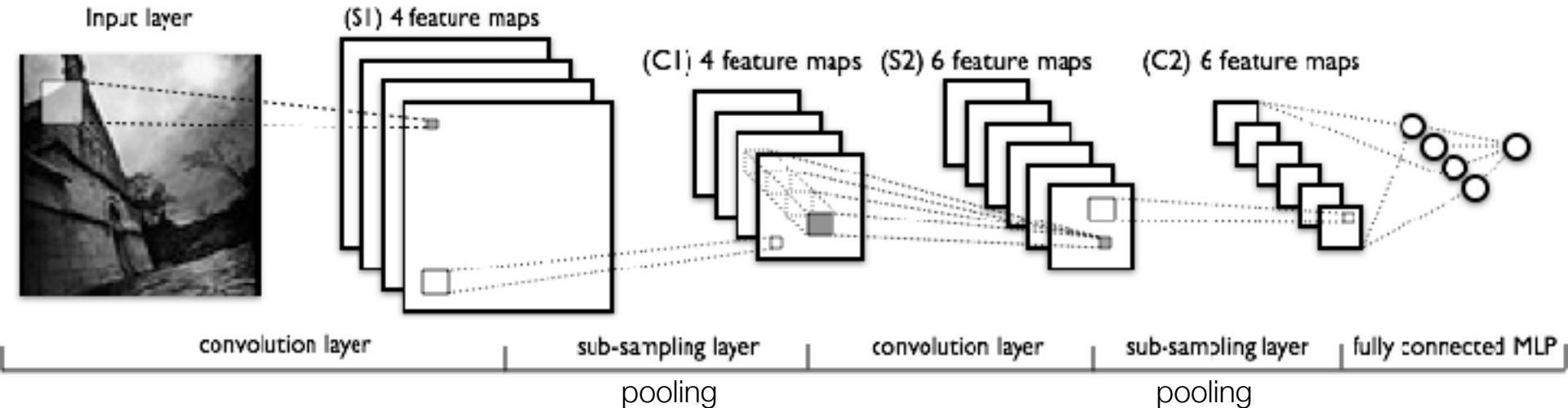
CNN gradient

- But we really want to understand the process!
- These are great guides:
 - [https://grzegorzgwardys.wordpress.com/
2016/04/22/8/](https://grzegorzgwardys.wordpress.com/2016/04/22/8/)
 - [http://andrew.gibiansky.com/blog/machine-
learning/convolutional-neural-networks/](http://andrew.gibiansky.com/blog/machine-learning/convolutional-neural-networks/)

CNN adding more convolutional layers



Some Example CNN Architectures



CNN: What does it all mean?

Deep Visualization Toolbox

yosinski.com/deepvis

#deepvis



Jason Yosinski



Jeff Clune



Anh Nguyen



Thomas Fuchs



Hod Lipson



Cornell University



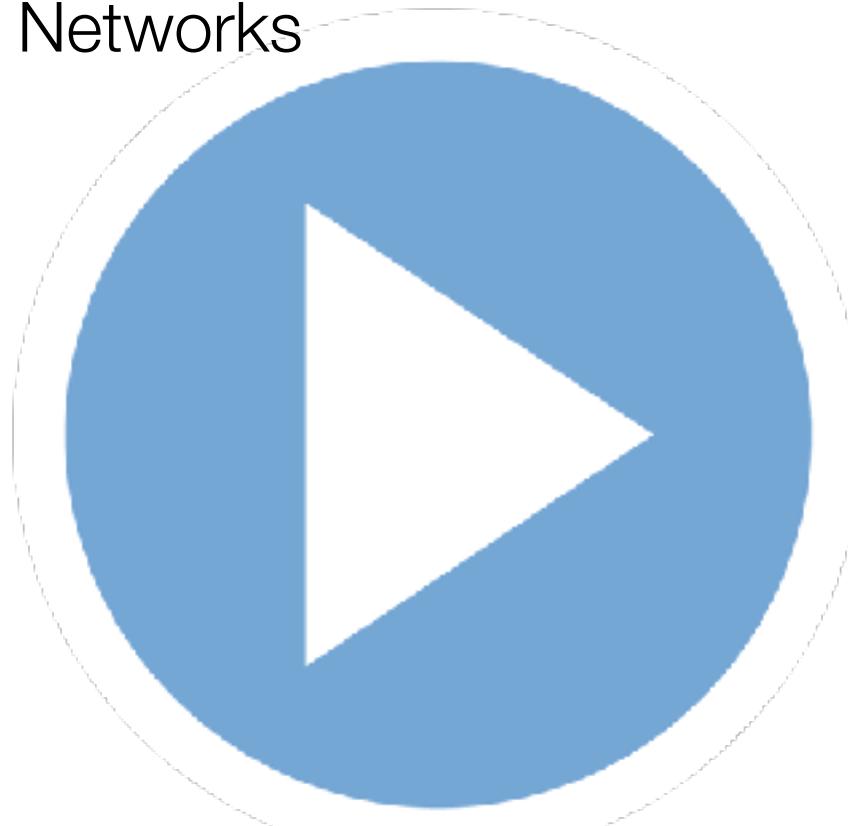
Jet Propulsion Laboratory
California Institute of Technology

<https://github.com/yosinski/deep-visualization-toolbox>

TensorFlow and Basic CNNs

Demo

Convolutional Neural Networks
in TensorFlow
with Keras



11. Convolutional Neural Networks.ipynb

Next Lecture

- More CNN architectures and CNN history

Lecture Notes for **Machine Learning in Python**

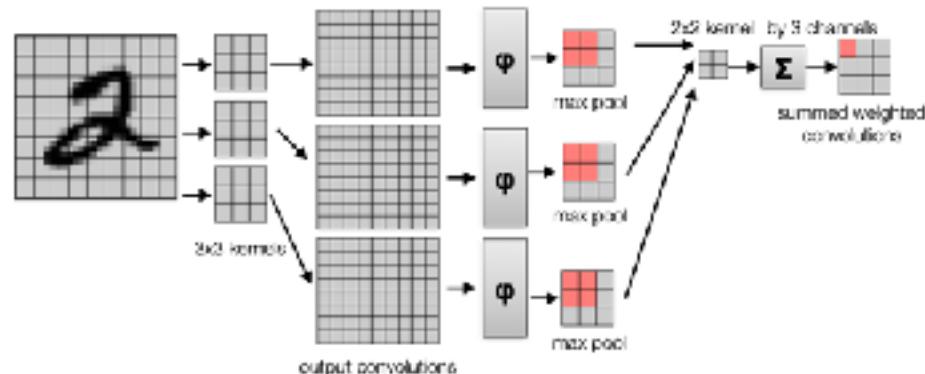
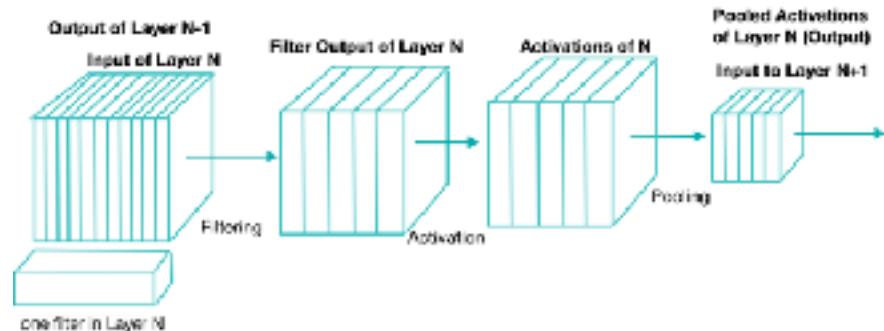
Professor Eric Larson

An Ongoing History of Convolutional Networks

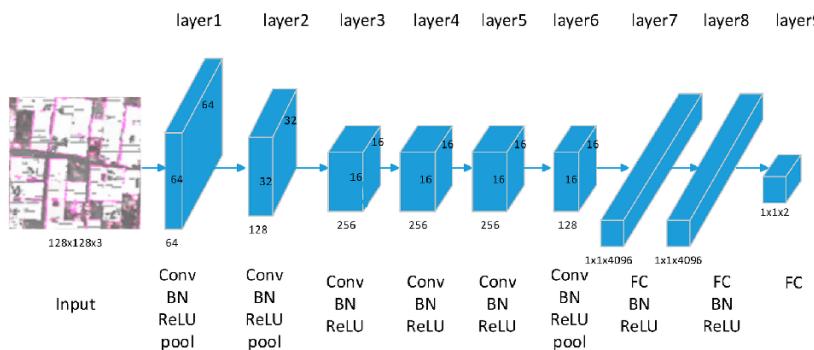
Class logistics and Agenda

- Wide/Deep Lab due soon!
- Agenda:
 - Finish CNN Demo
 - History of CNNs
 - with Modern CNN Architectures
- Next Time:
 - More Advanced CNN Demo

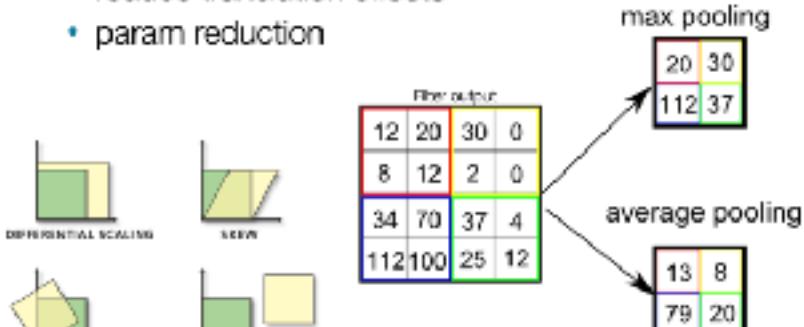
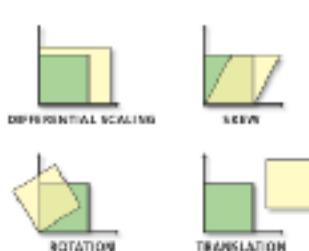
Last Time:



Structure of Each Tensor: Channels x Rows x Columns



- Why perform pooling?
- Why max pooling?
 - reduce translation effects
 - param reduction



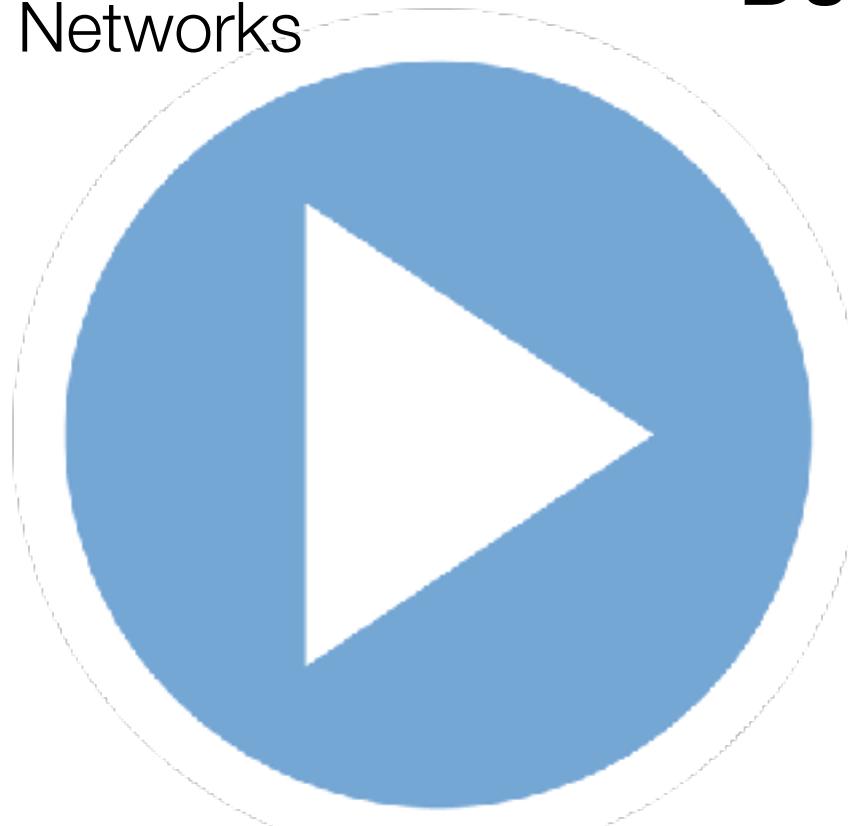
TensorFlow and Basic CNNs

**Last
Time!**

Convolutional Neural Networks
in TensorFlow
with Keras

with Sequential API!

If needed:
**Finish
Demo**



11. Convolutional Neural Networks.ipynb

History of Convolutional Neural Networks



Thanks to machine-learning algorithms,
the robot apccalypse was short-lived.

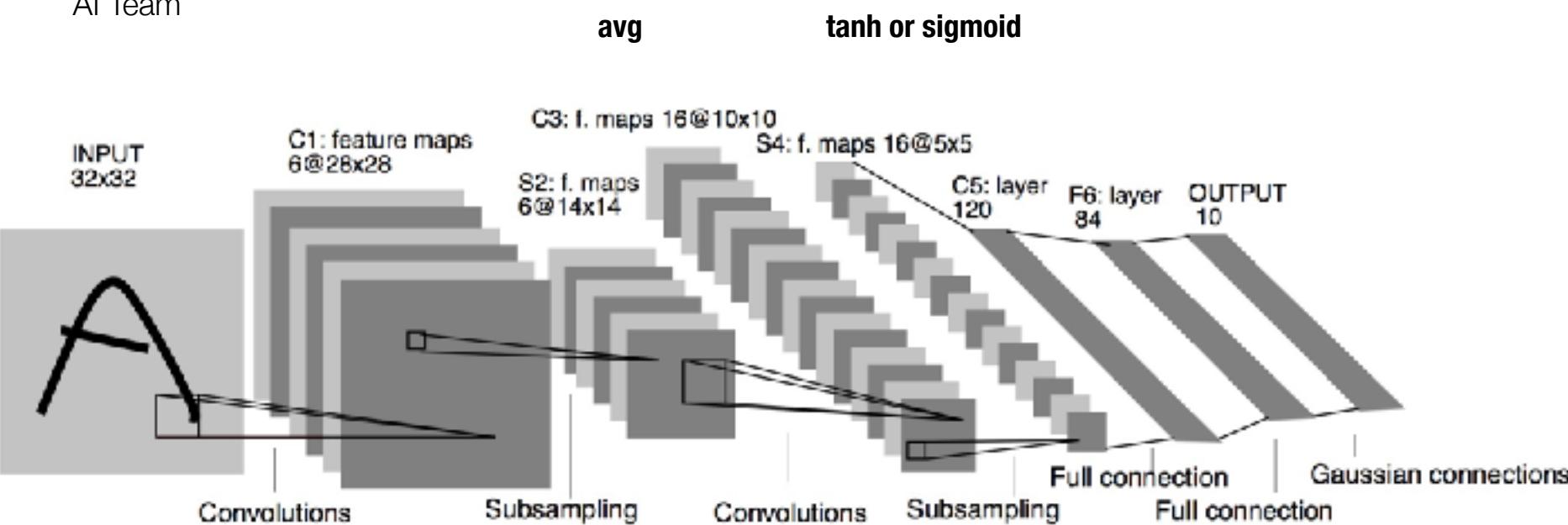
Types of CNN, 1988-1998



Yann
LeCun

Heads Facebook
AI Team

- **LeNet-1 (1988)**
 - ~2600 params, not many layers
- **LeNet-5 (1998)**
 - 7 layers, gets excellent MNIST performance
- Major contribution, general structure:
 - conv=>pool=>non-linearity=> ...=>MLP



CNN History

- List of major breakthroughs from 1998 through 2010 in convolutional networks:



- 2010



Types of CNN, 2010



Dan
Ciresan

AI Researcher
IDSIA, Switzerland

- **Ciresan Net**
- Publishes code for running CNN via GPU
 - Subsequently wins 5 international competitions
 - from stop signs => cancer detection
- Major contribution: NVIDIA parallelized training algorithms

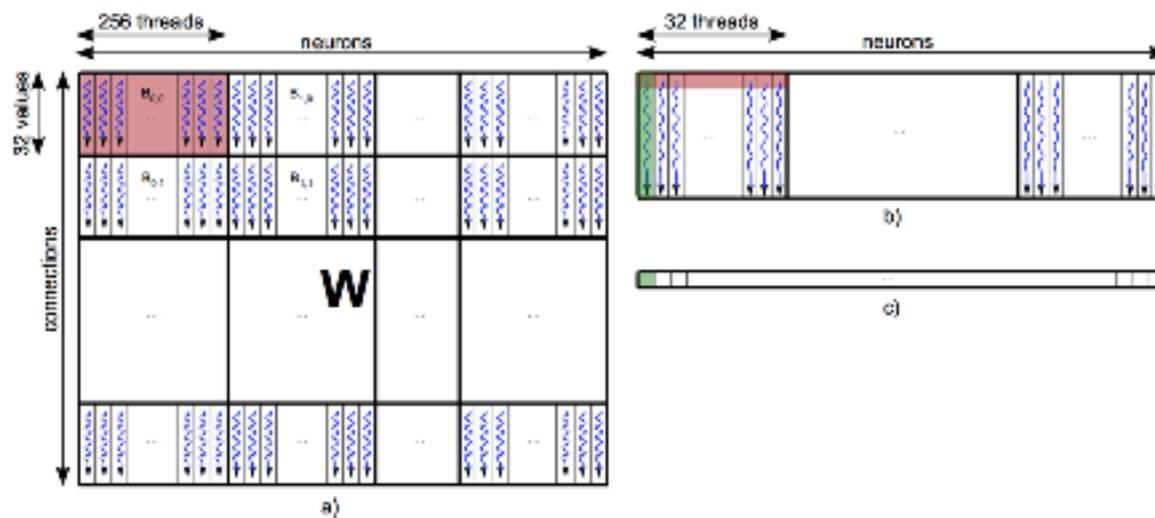
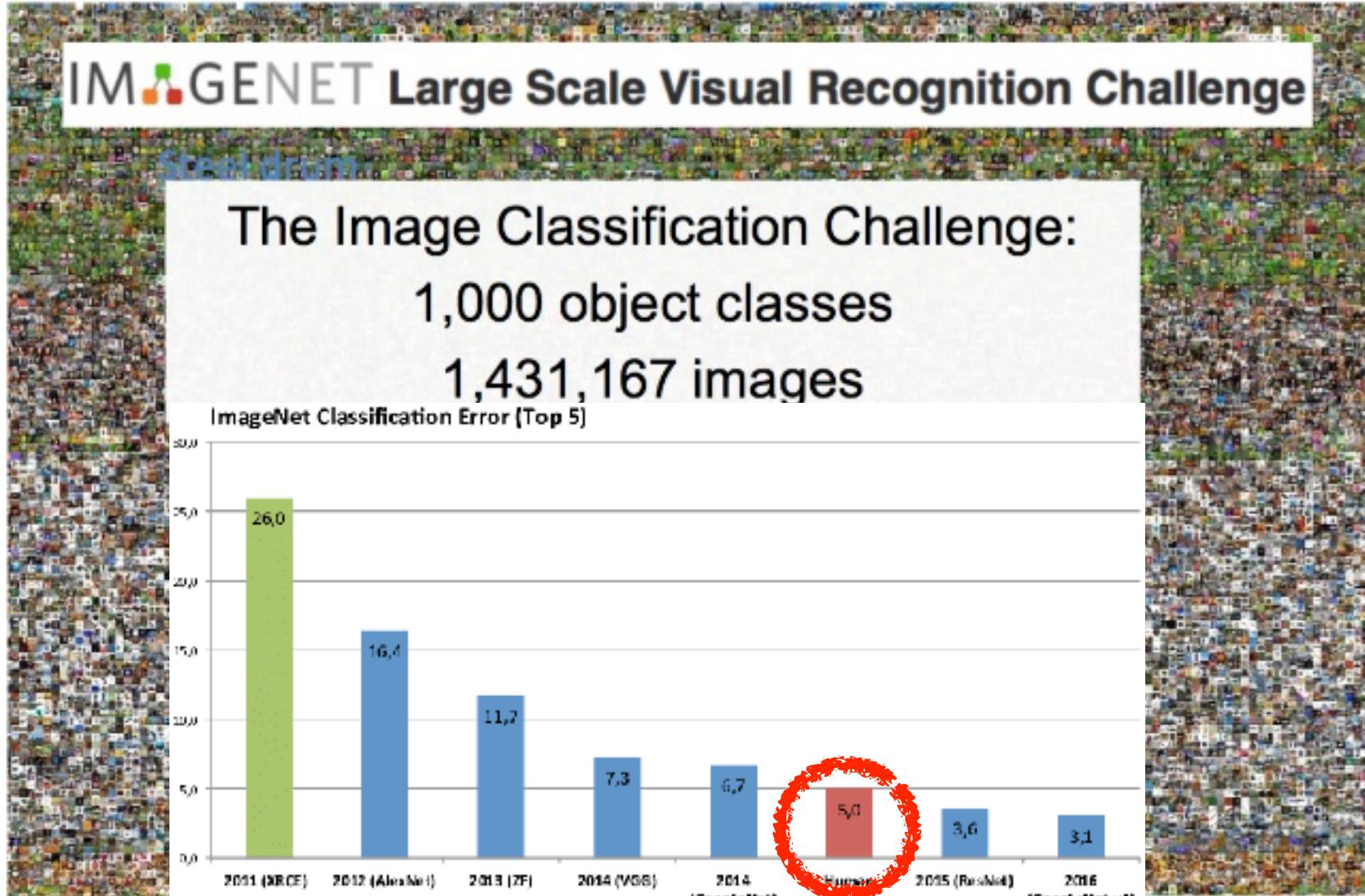


Figure 2: Forward propagation: a) mapping of kernel 1 grid onto the padded weight matrix; b) mapping the kernel 2 grid onto the partial dot products matrix; c) output of forward propagation.

<https://arxiv.org/pdf/1003.0358.pdf>

ImageNet Competition (2010-2016)



https://www.researchgate.net/figure/Winner-results-of-the-ImageNet-large-scale-visual-recognition-challenge-LSVRC-of-the_fig7_324476862

<https://www.slideshare.net/nmhkahn/case-study-of-convolutional-neural-network-61556303>

Types of CNN, 2012

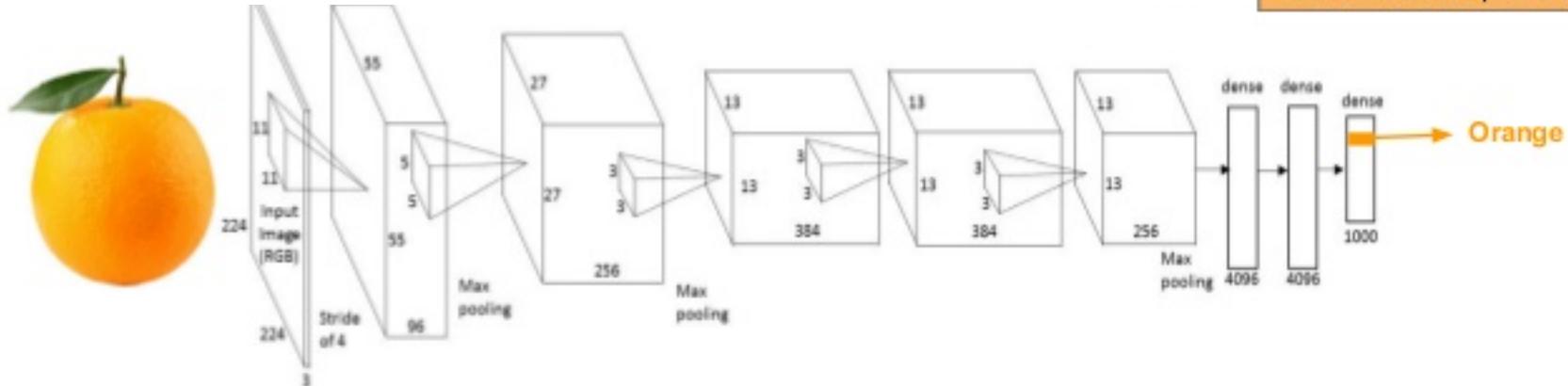
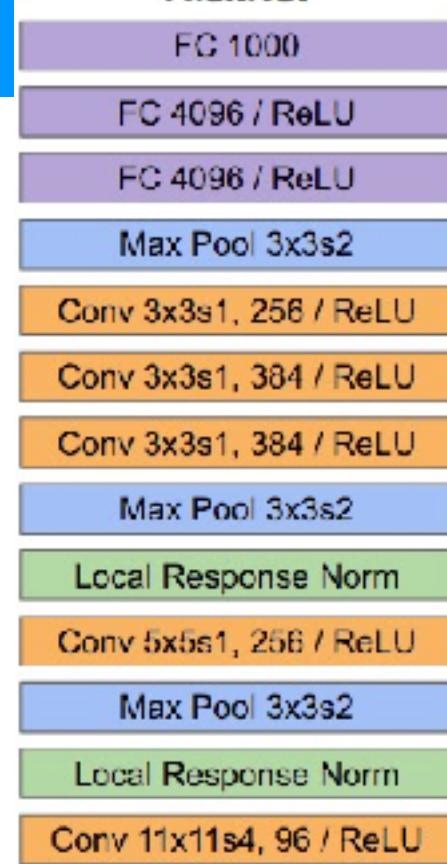
AlexNet



Alex
Krizhevsky

Google

- **AlexNet**, Hinton is mentor
 - wins ImageNet competition
- Major contributions:
 - dropout for regularization
 - systematic use of ReLU
 - data expansion
 - ***overlapping max pool***



Warning



Types of CNN, 2013



Karen Simonyan
and
Andrew Zisserman



- Oxford **VGG Net** (Visual Geometry Group)
- Major contributions:
 - small cascaded kernels
 - way more layers (19 versus ~7)
 - “emulates” biology “better”
 - trained on NVIDIA GPUs for 2-3 weeks

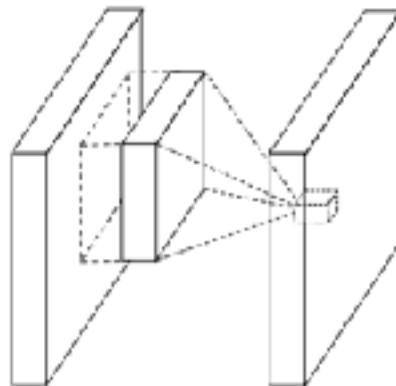
<http://www.robots.ox.ac.uk/~vgg/practicals/cnn/>

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64	conv3-64	conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128	conv3-128	conv3-128
maxpool					
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
maxpool					
conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: Number of parameters (in millions).

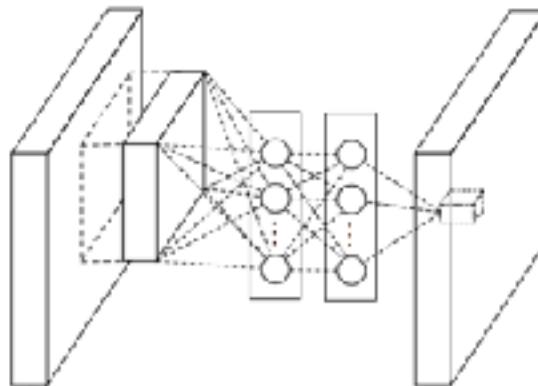
Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

- Network in Network **NiN**
 - or MLPConv

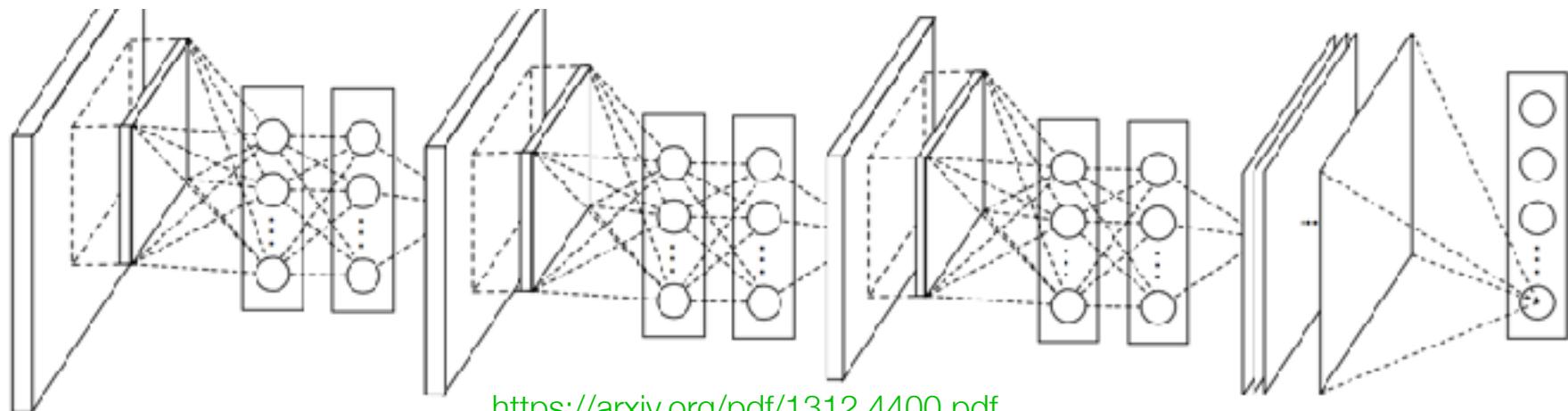


(a) Linear convolution layer

Min Lin^{1,2}, Qiang Chen², Shuicheng Yan²
¹Graduate School for Integrative Sciences and Engineering
²Department of Electronic & Computer Engineering
National University of Singapore, Singapore
`{linmin,chenqiang,elec yans}@nus.edu.sg`



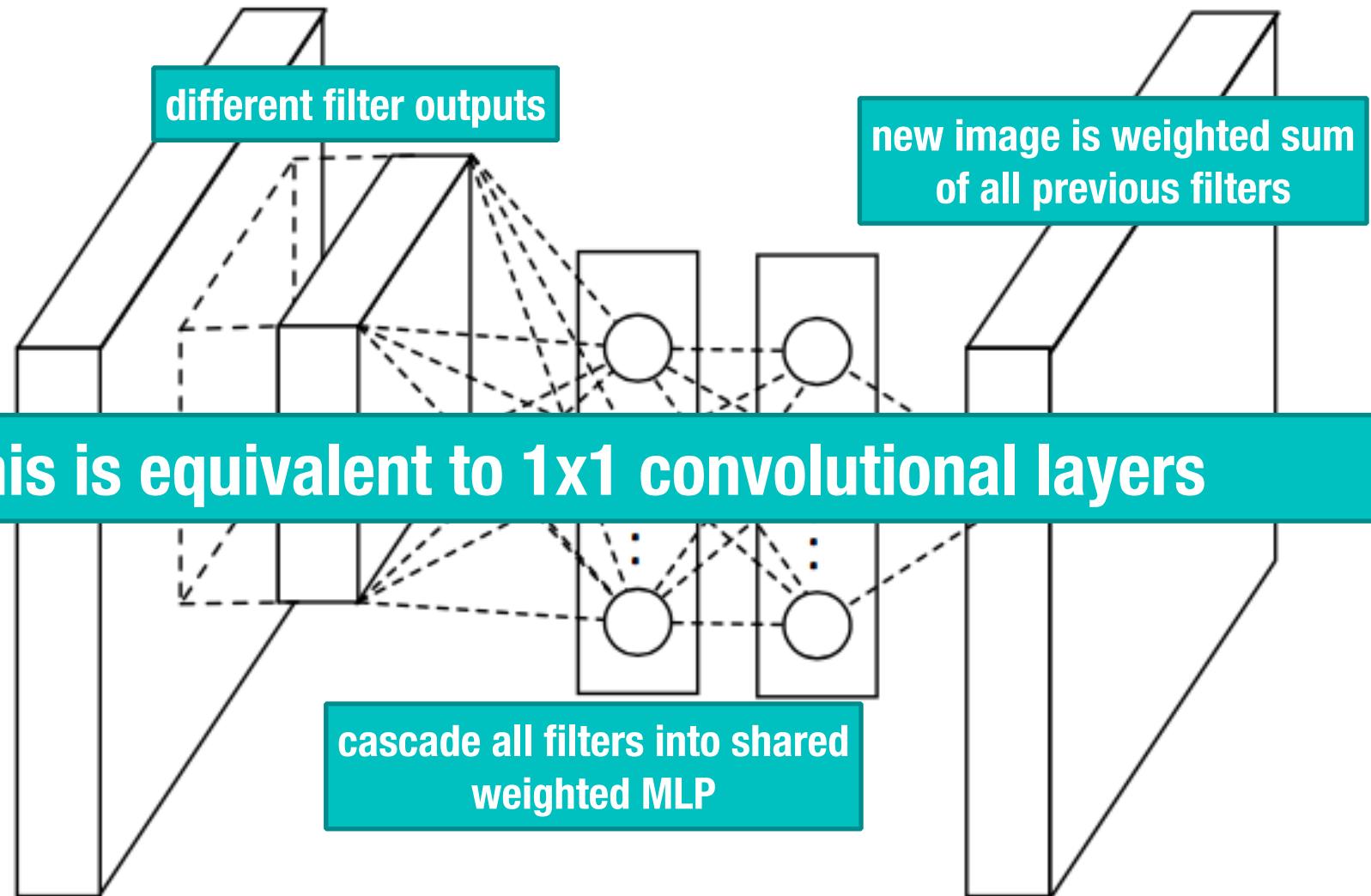
(b) Mlpconv layer



<https://arxiv.org/pdf/1312.4400.pdf>

Types of CNN, 2014

- Network in Network



Types of CNN, 2014

Research at Google

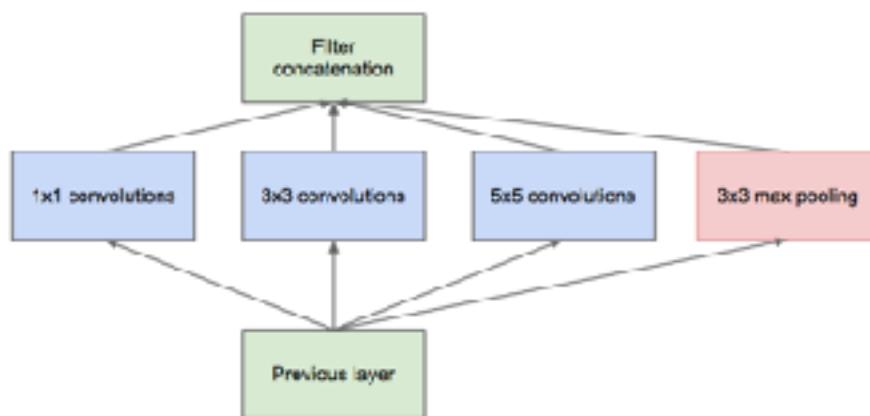
Home Publications People Teams Outreach

Christian Szegedy

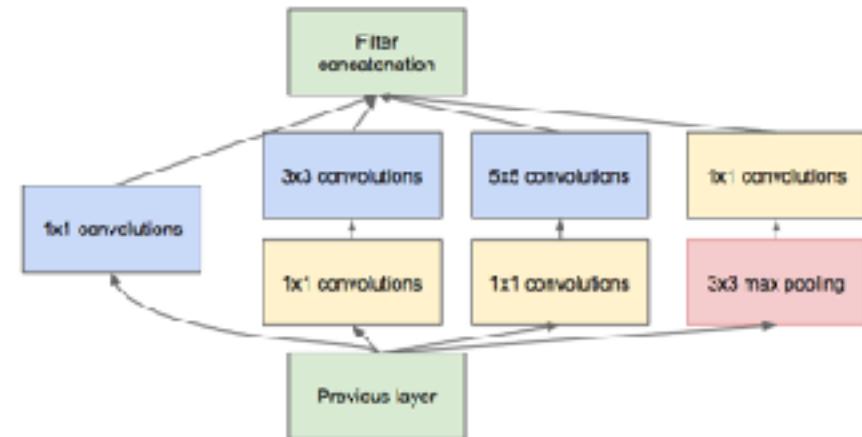


Research Area(s)
Machine Intelligence
Machine Perception

- **GoogLeNet**
 - or **Inception V1**
- Major contribution:
 - bottleneck layering
 - parallel NiN



(a) Inception module, naïve version



(b) Inception module with dimension reductions

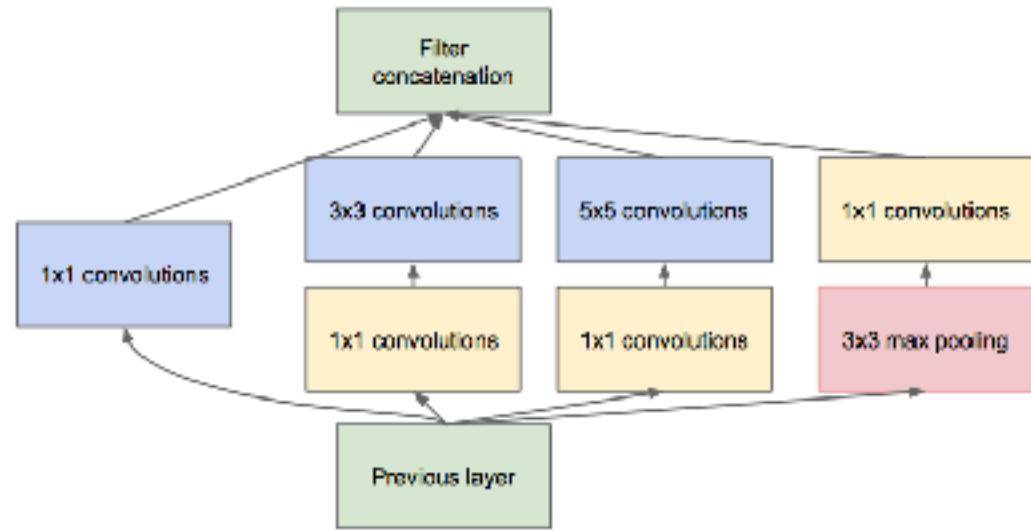
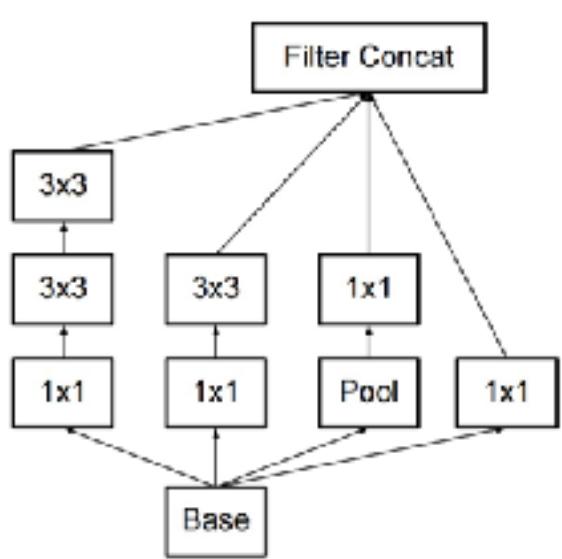
Figure 2: Inception module

<https://arxiv.org/pdf/1409.4842.pdf>

Types of CNN, 2015 February and December

A screenshot of Christian Szegedy's profile on Google Research. It shows his name, a photo, and research areas: Machine Intelligence and Machine Perception.

- **Inception V2**, Inception V1 with batch normalization
- **Inception V3:**
 - replace 5x5 with multiple 3x3



<https://arxiv.org/pdf/1512.00567.pdf>

42

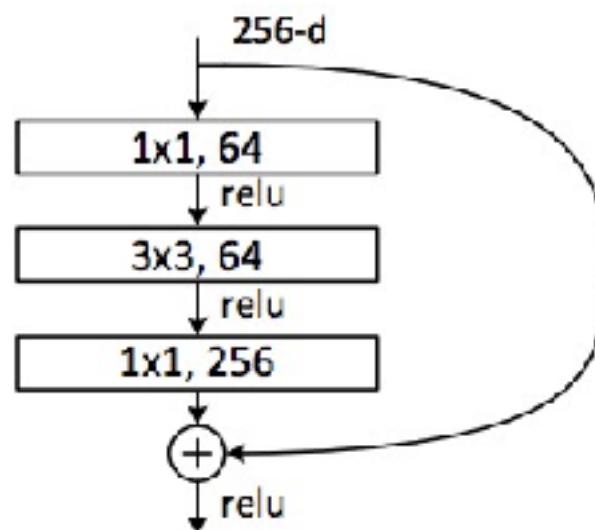
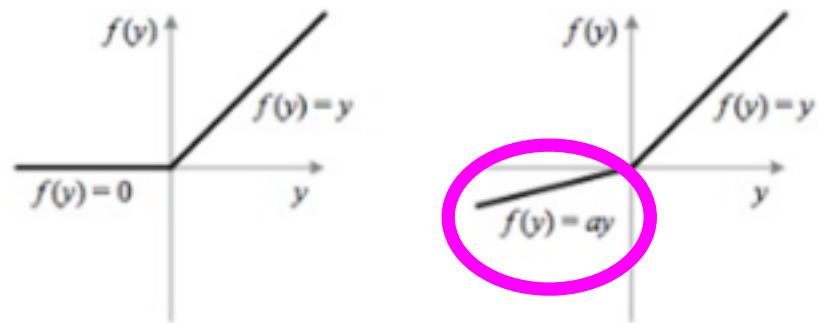
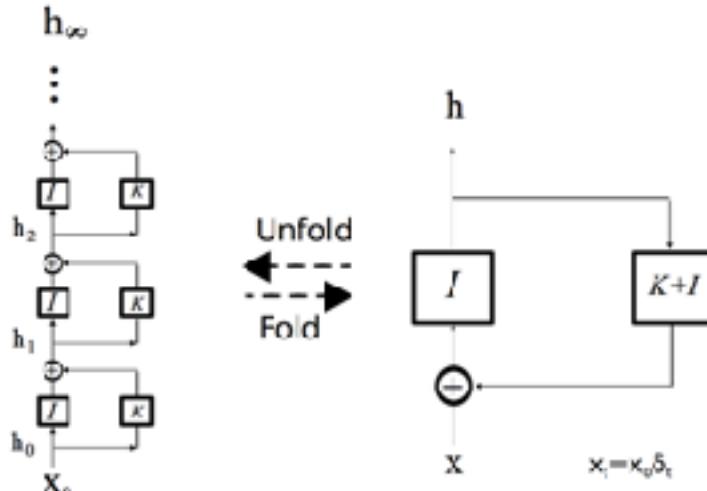
Types of CNN, 2015 December

Microsoft Research

- Major Contributions:
 - ensembles, not strictly sequential
 - “bio-plausible” with feedback

• ResNet

- PReLU: adaptive trained slope



- NiN: triple bypass layer
 - similar to bottleneck

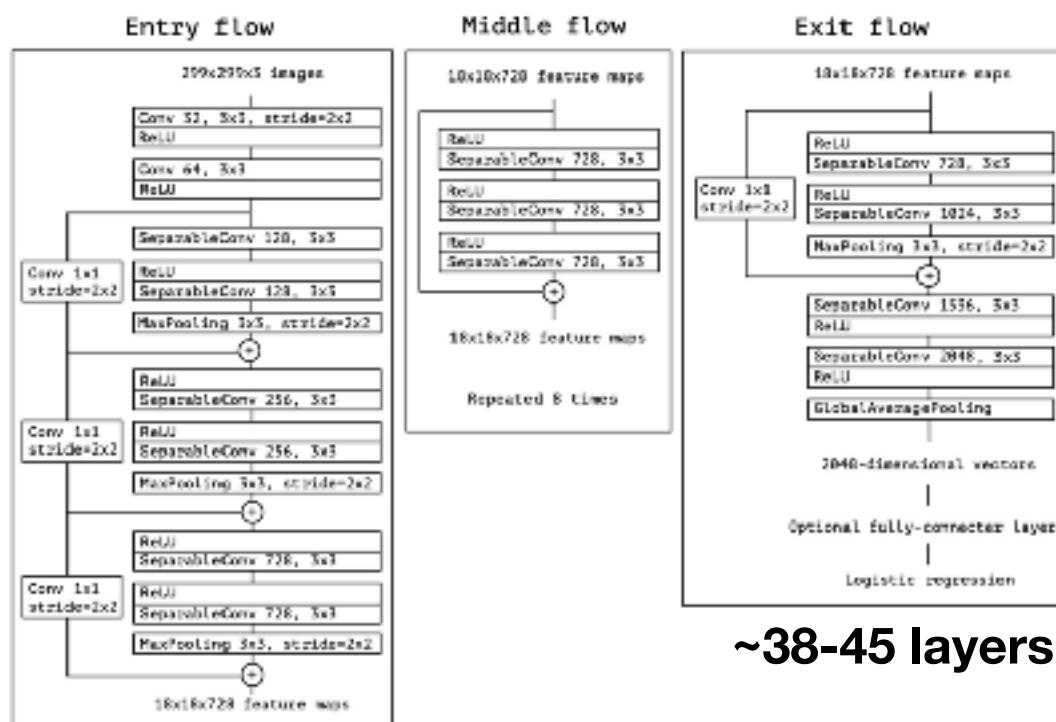
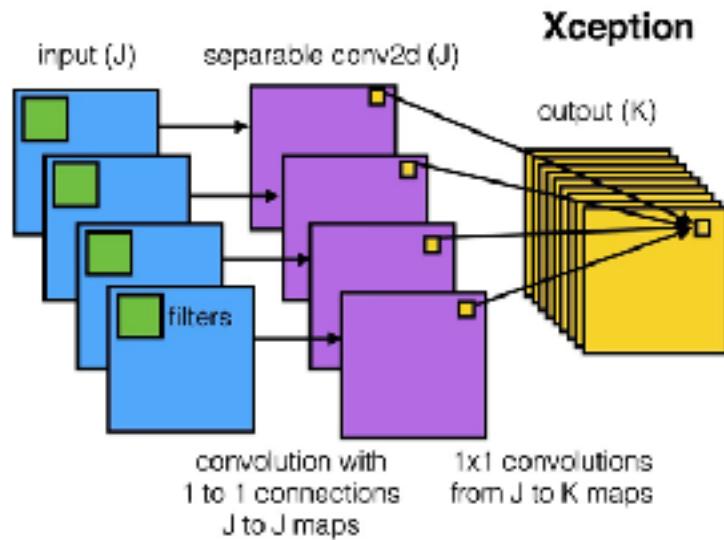
Types of CNN, 2017 April



Francois Chollet
Google

Xception

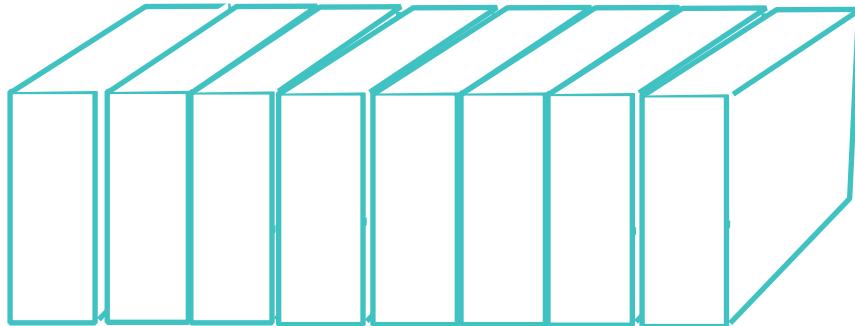
- Major Contributions:
 - ◆ combining branching / residual blocks
 - ◆ separable convolutions



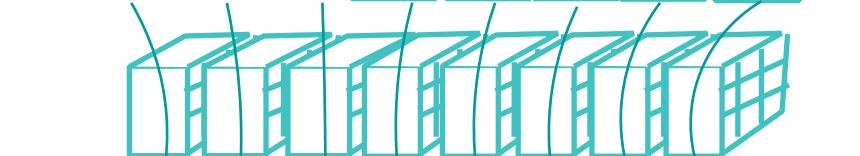
~38-45 layers

<https://arxiv.org/pdf/1610.02357.pdf>

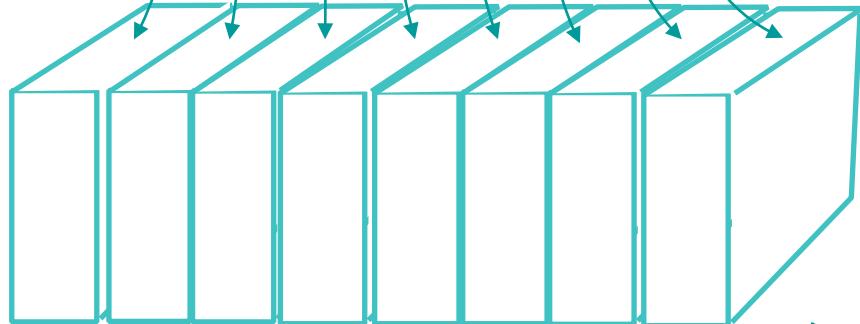
Separable Convolution Primer



Inputs, From Layer N-1
Num Channels = J

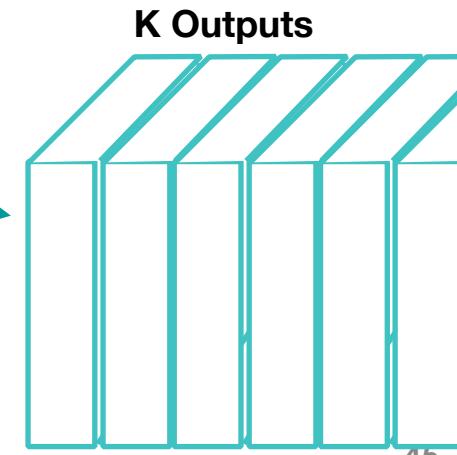


Filters, Layer N
Convolve Each Channel
Separately
Num filters = J



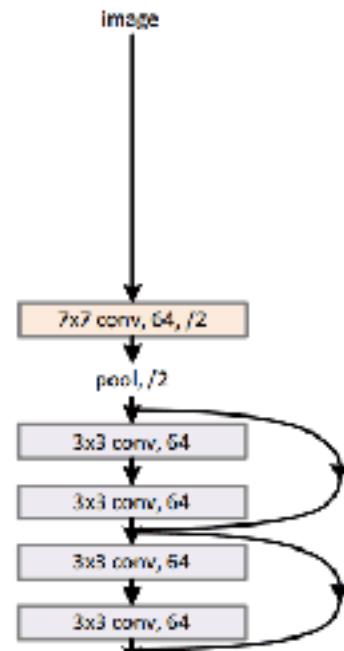
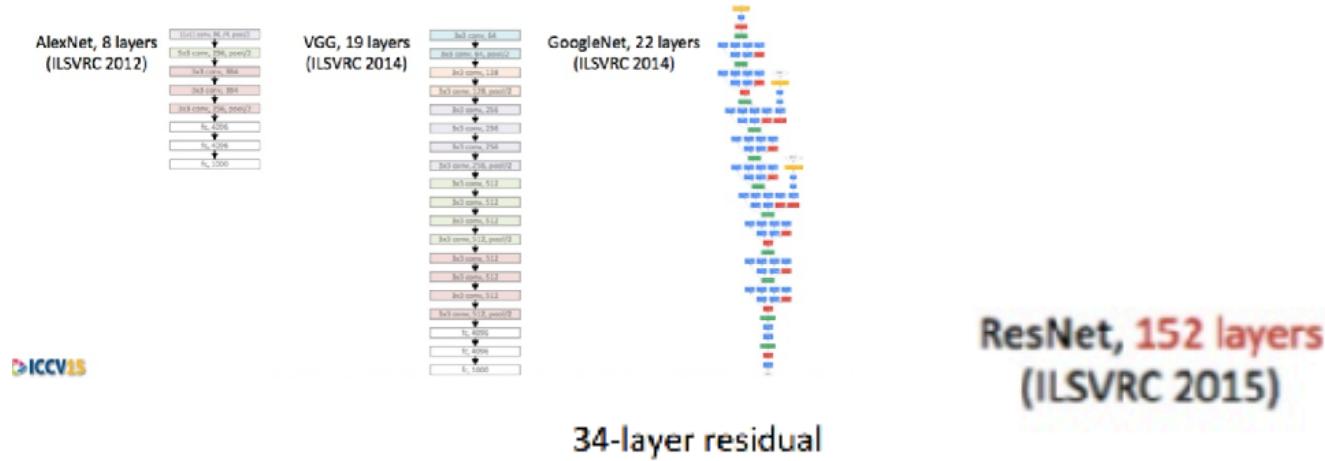
Concat Channel Outputs
Num Intermediate
Channels = J

Perform K 1x1
Traditional Convolutions
to get K Outputs



How big are these networks?

How big are these networks?



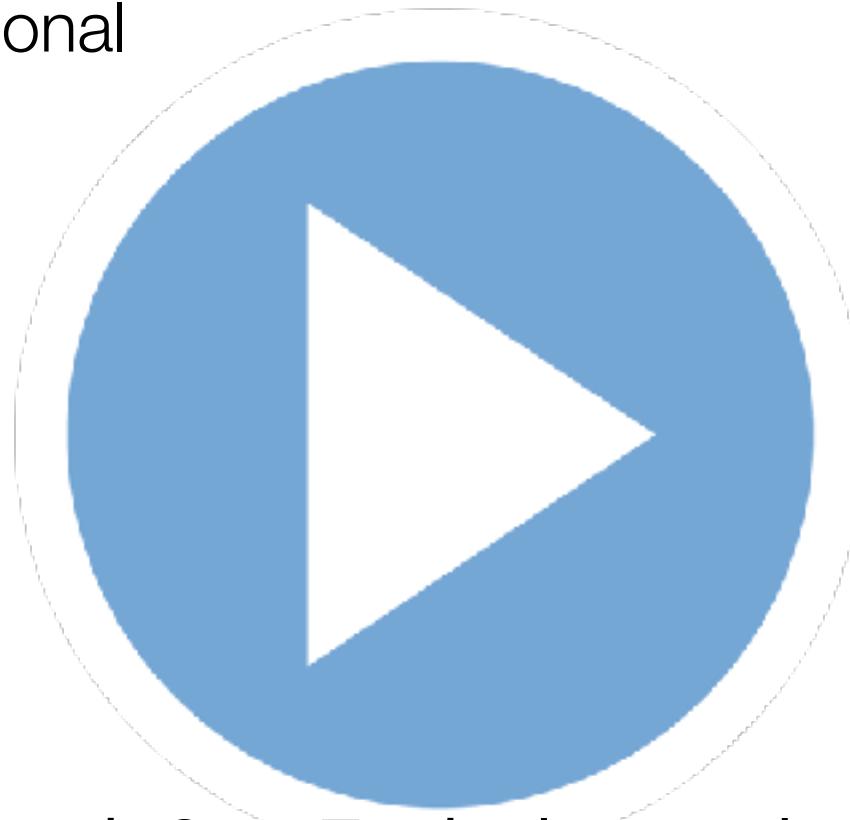
Self Test

- We have seen a lot of different networks.
- The most important concept to understand in using convolutional neural networks is:
 - A. Use proper initialization of layers
 - B. Have plenty of data or use expansion
 - C. Set aside time for training
 - D. Use batch normalization

Demo

Even more Convolutional
Neural Networks

...in TensorFlow
...with Keras



12. More Advanced CNN Techniques.ipynb

Next Time:

- CNN Lab Discussion (Town Hall)
- Intro to Recurrent Neural Network Architectures
 - RNNs, GRUs, LSTMs
 - Training for characters