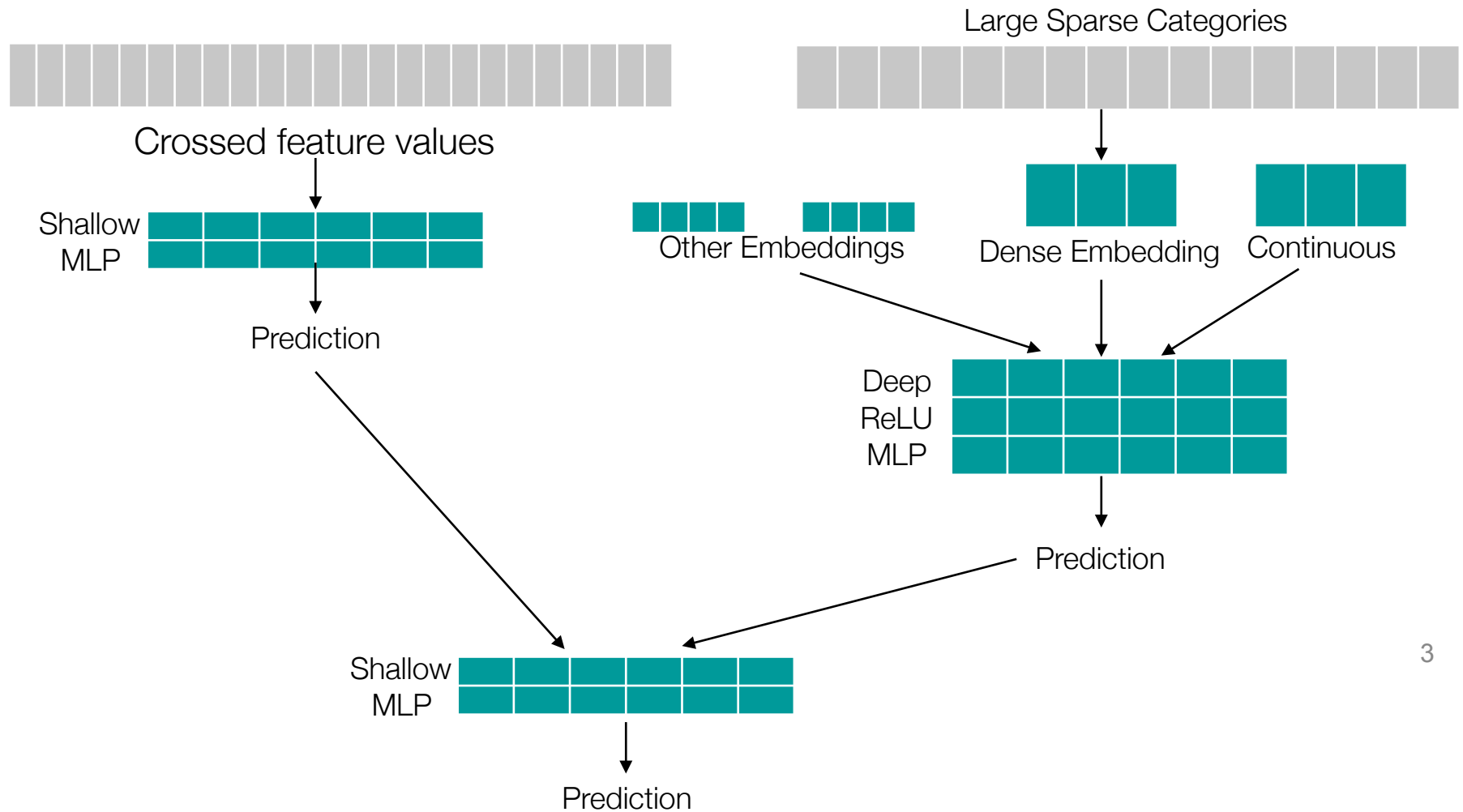# Lecture Notes for
Machine Learning in Python

Professor Eric Larson
Basic Convolutional Neural Networks

Logistics and Agenda

- Logistics
  - Wide/Deep due this week
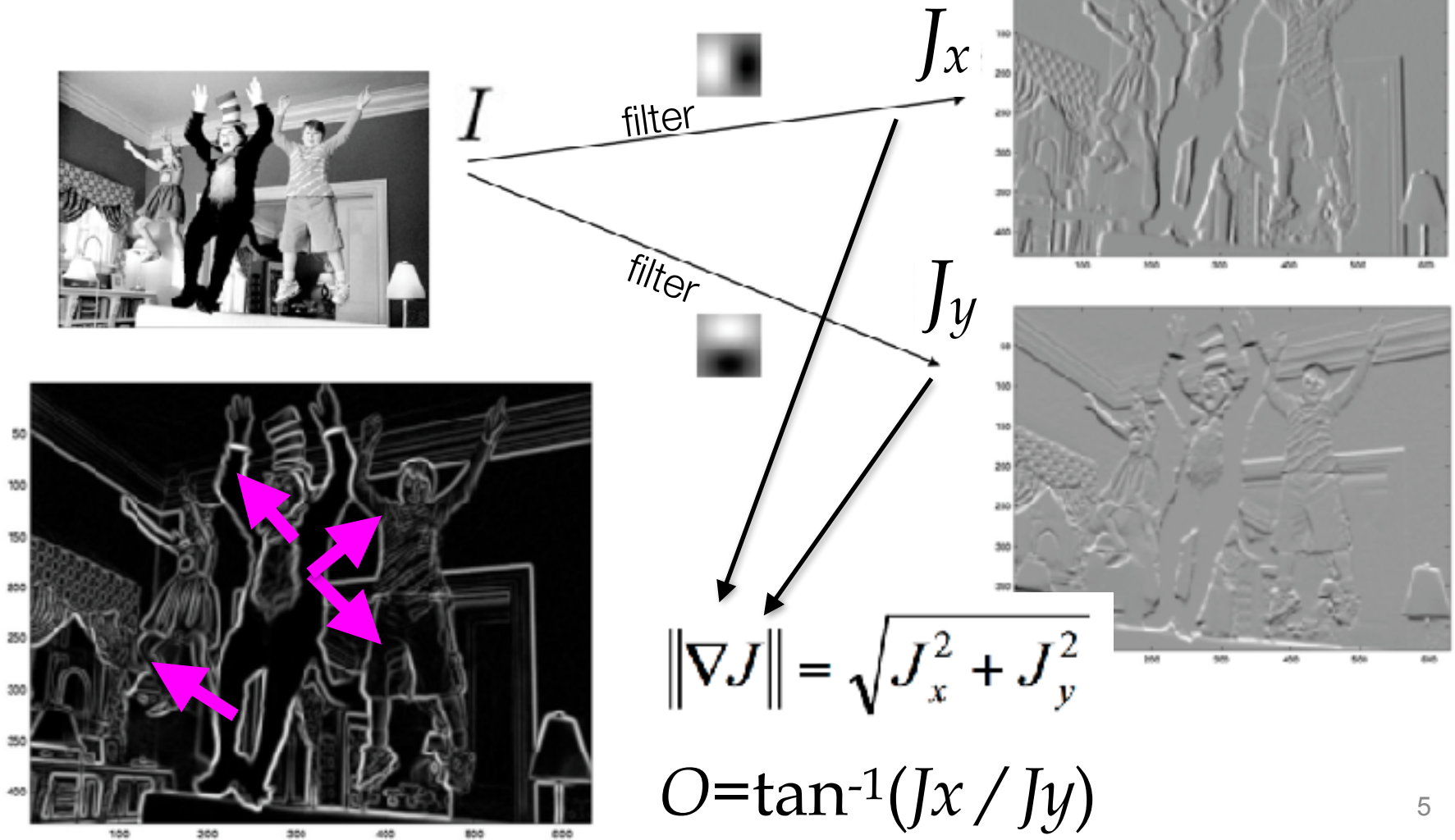- Agenda
  - Basic CNN architectures and Demo

Last Time:

Large Sparse Categories
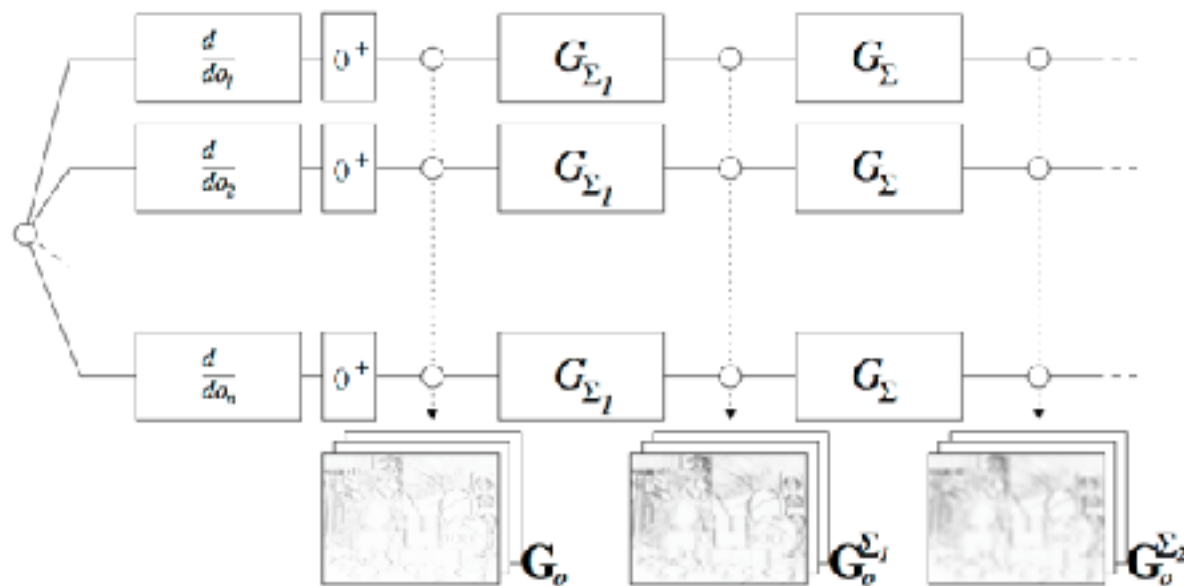
Crossed feature values

Shallow
MLP

Prediction

Other Embeddings

Dense Embedding

Continuous

Deep
ReLU
MLP

Prediction

Shallow
MLP

Prediction

Convolutional Neural Networks

What we did before

- the gradient (2D derivative)

I

filter

filter

J_x

J_y

$$\|\nabla J\| = \sqrt{J_x^2 + J_y^2}$$

$O = \tan^{-1}(Jx / Jy)$

images: Jianbo Shi, Upenn

What we did before

take normalized histogram at point u,v

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

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

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

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

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

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

CNN Overview

- First layer(s):
  - convolution with different filters
  - nonlinearity
  - pooling
  - Each pooling layer *can* make the input image "smaller"
    - more summative explanations
- Final layers are densely connected
  - typically multi-layer perceptrons

CNN Overview: Self Test

- First layer(s):
  - convolution with different filters
  - nonlinearity
  - pooling
  - Each pooling layer *can* make the input image "smaller"
    - more summative explanations
- 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
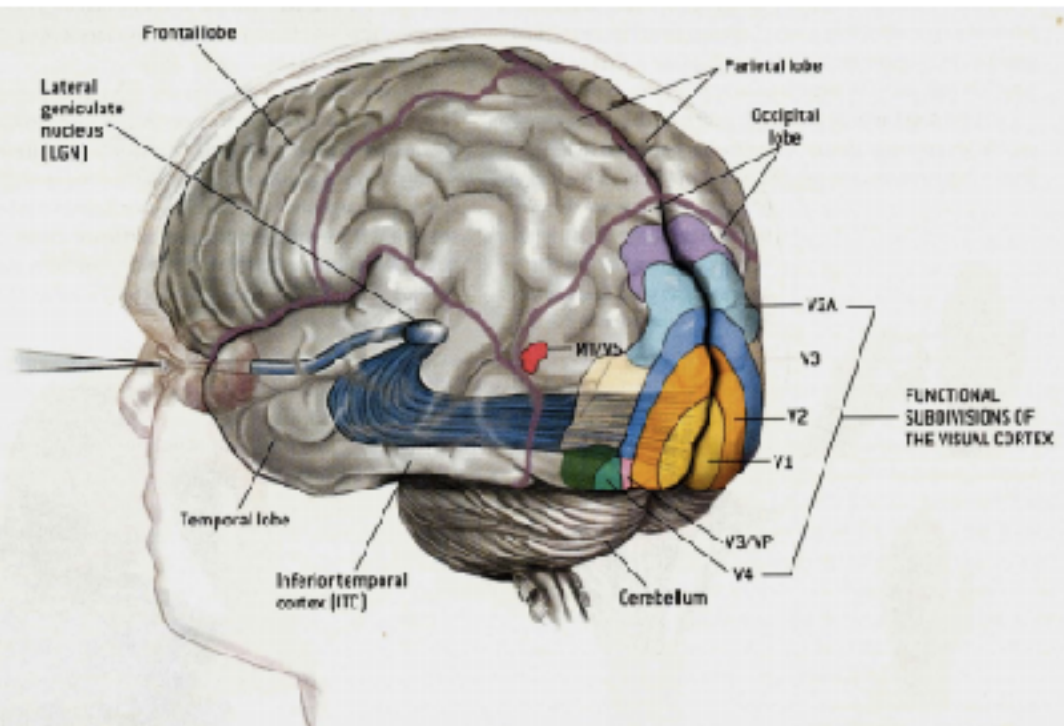 - (D) They are not a problem

CNN Filtering

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

Image responses for Gabor filter kernels
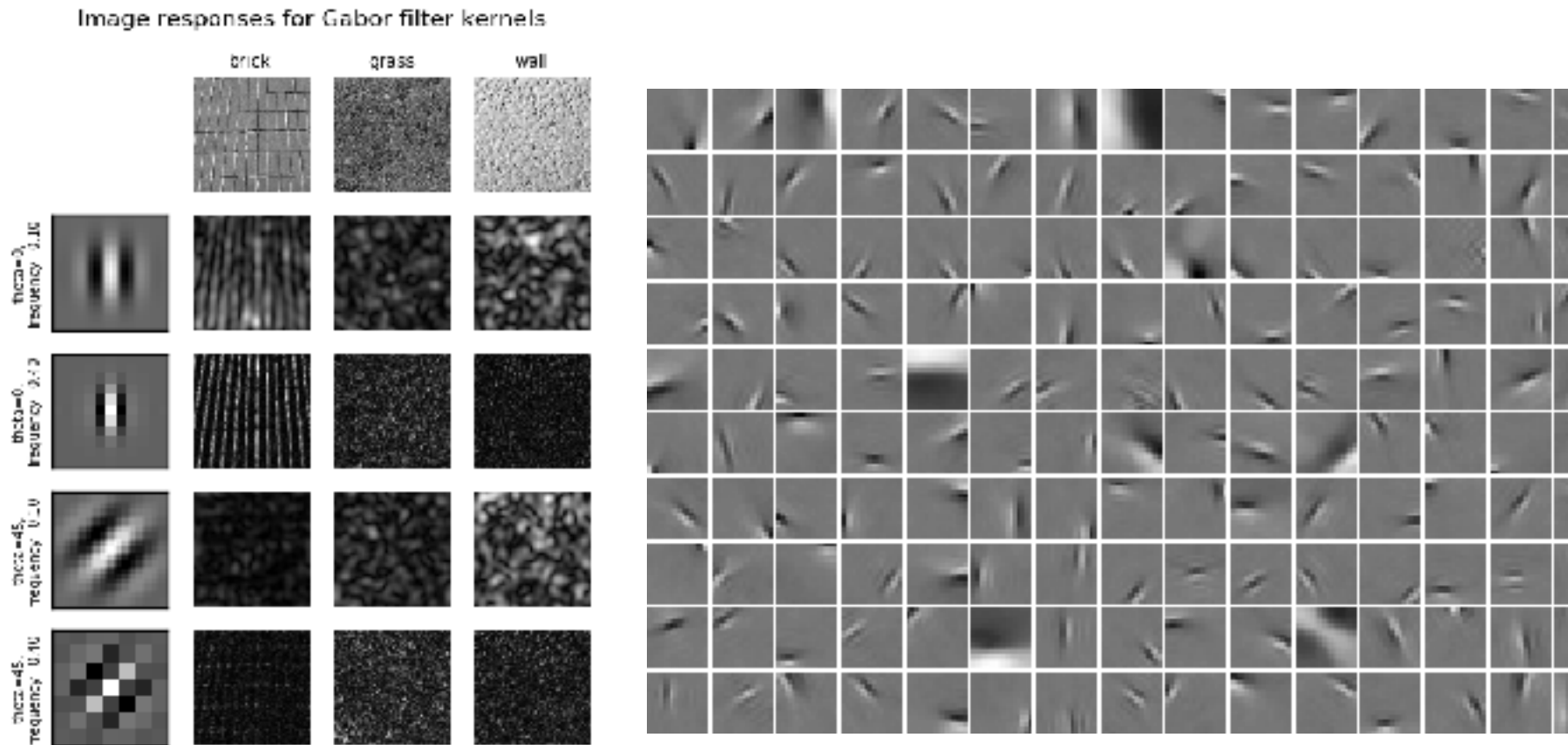
CNN Filtering

- 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

CNN Pooling

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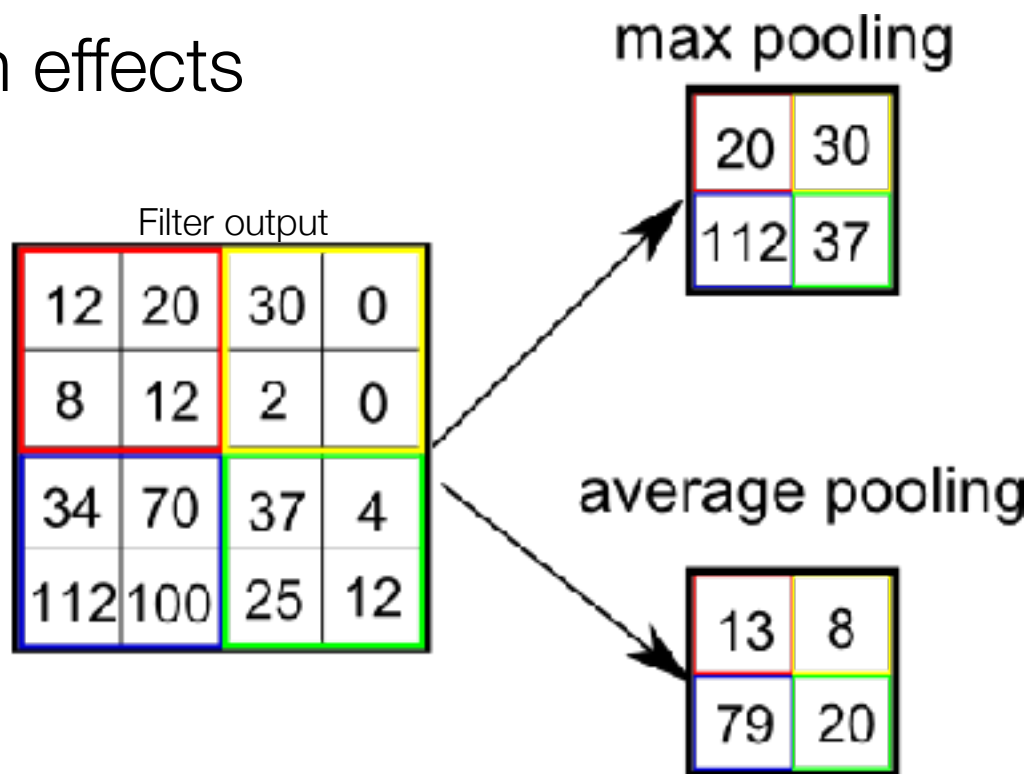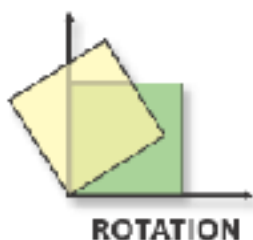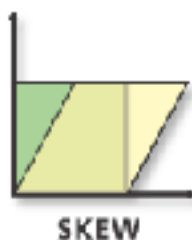
Filter output

max pooling

average pooling

DIFFERENTIAL SCALING

SKEW

ROTATION

TRANSLATION

Lecture Notes for
Machine Learning in Python

Professor Eric Larson
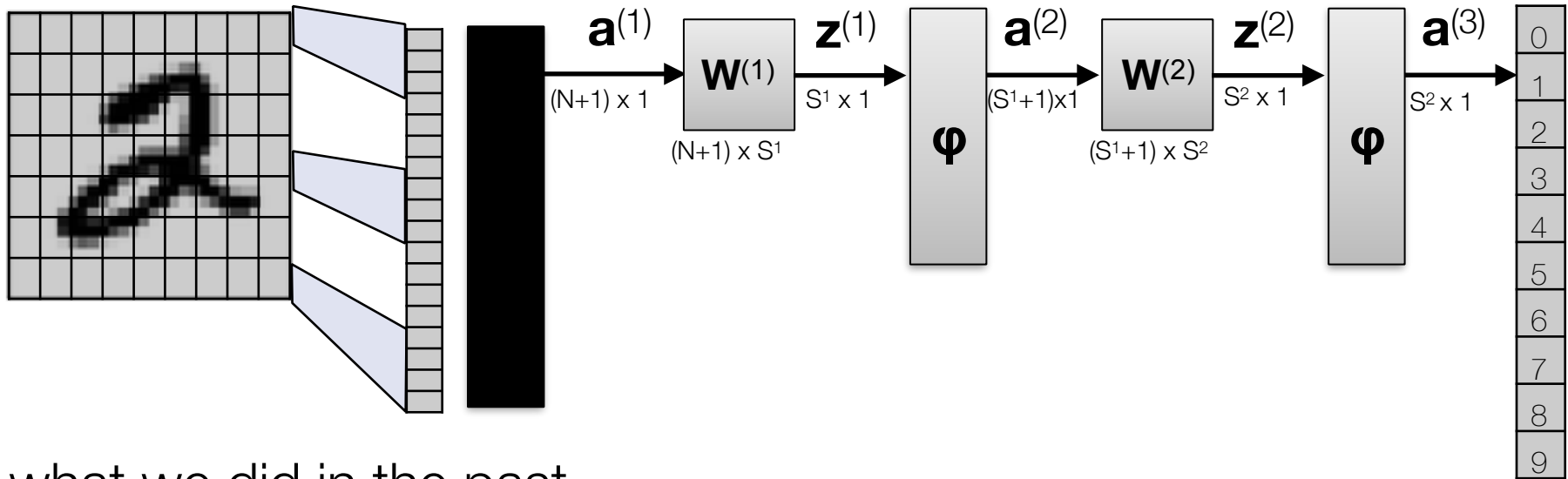## Basic Convolutional Neural Networks

Logistics and Agenda

- Logistics
  - Welcome back!
    CNNs due one week from tomorrow
- Agenda
  - Basic CNN architectures
  - Next Time: Finish CNN architecture

Convolution Review (and clarification)
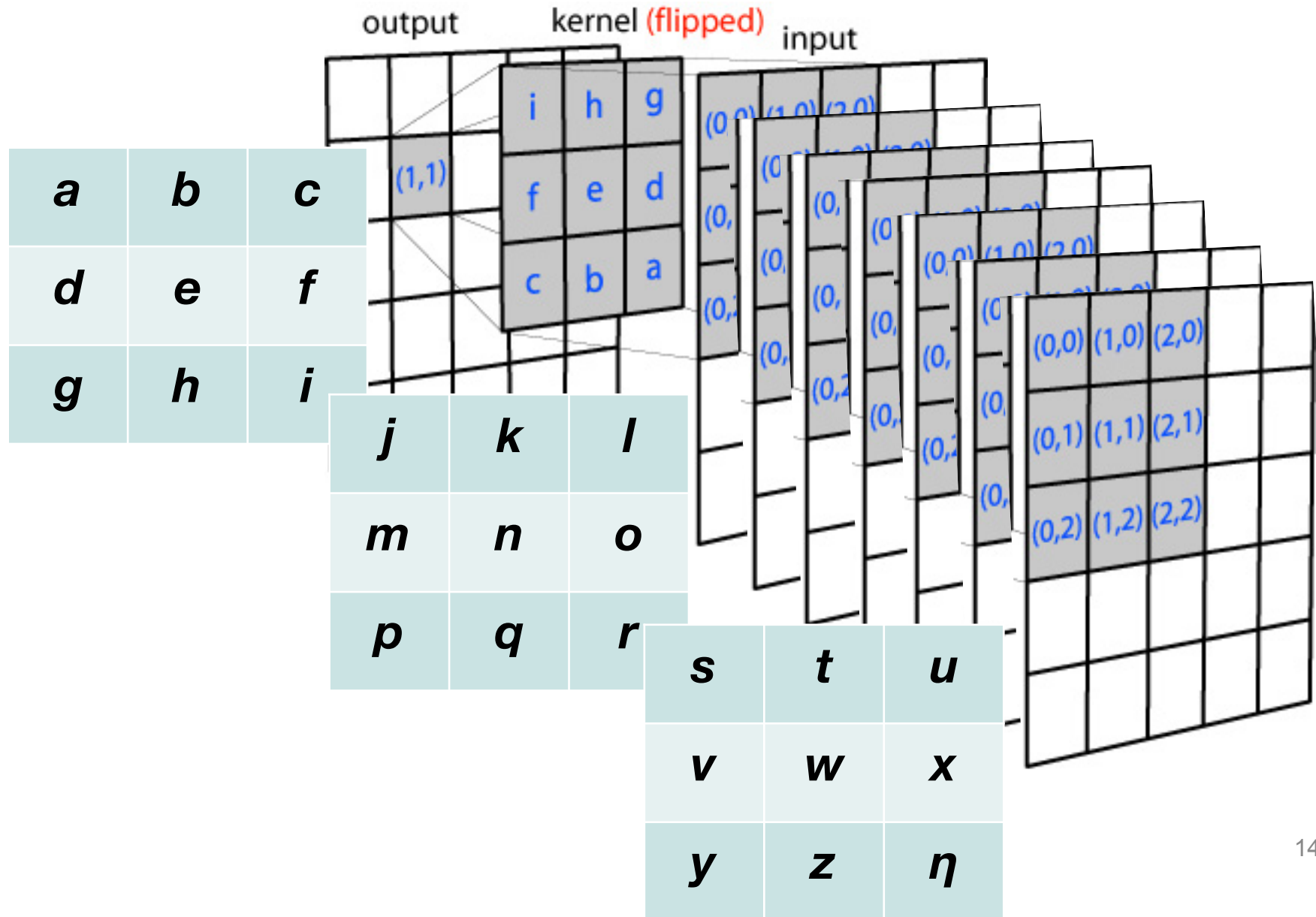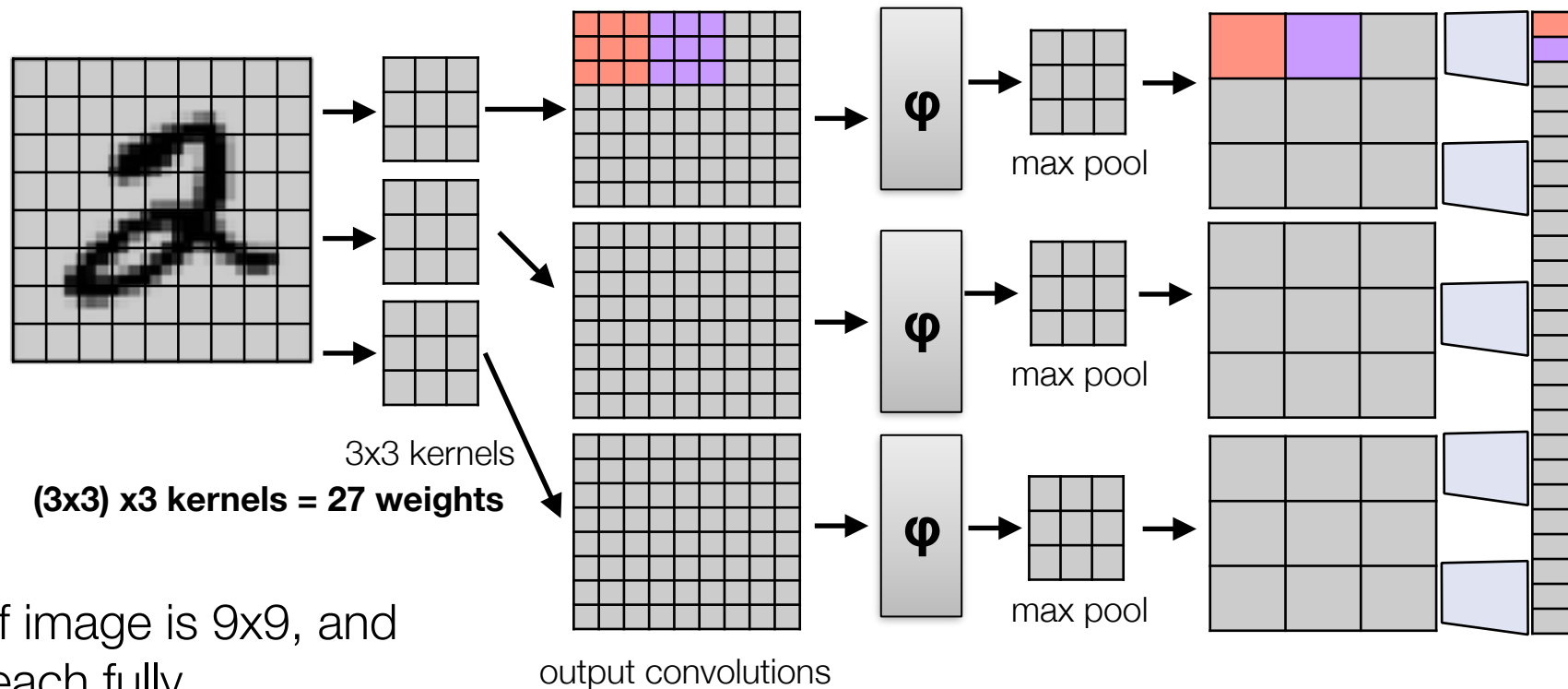
From Fully Connected to CNN

what we did in the past

If image is 9x9, and each fully connected layer is 20 hidden neurons wide, how many parameters are in this NN (ignore bias)?
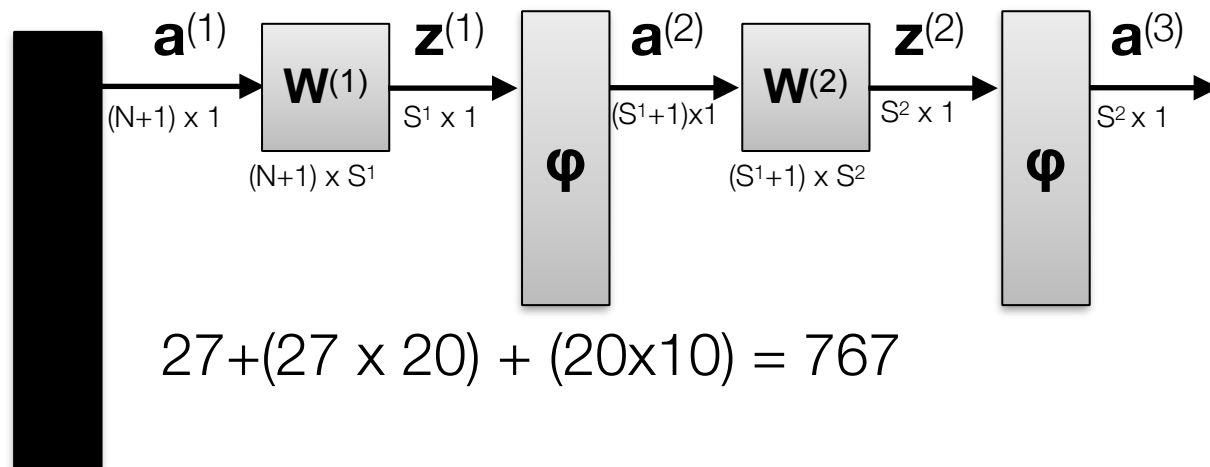
$(K^2 \times 20) + (20 \times 10) = 200 + 20\ K^2$

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

From Fully Connected to CNN

3x3 kernels

(3x3) x3 kernels = 27 weights
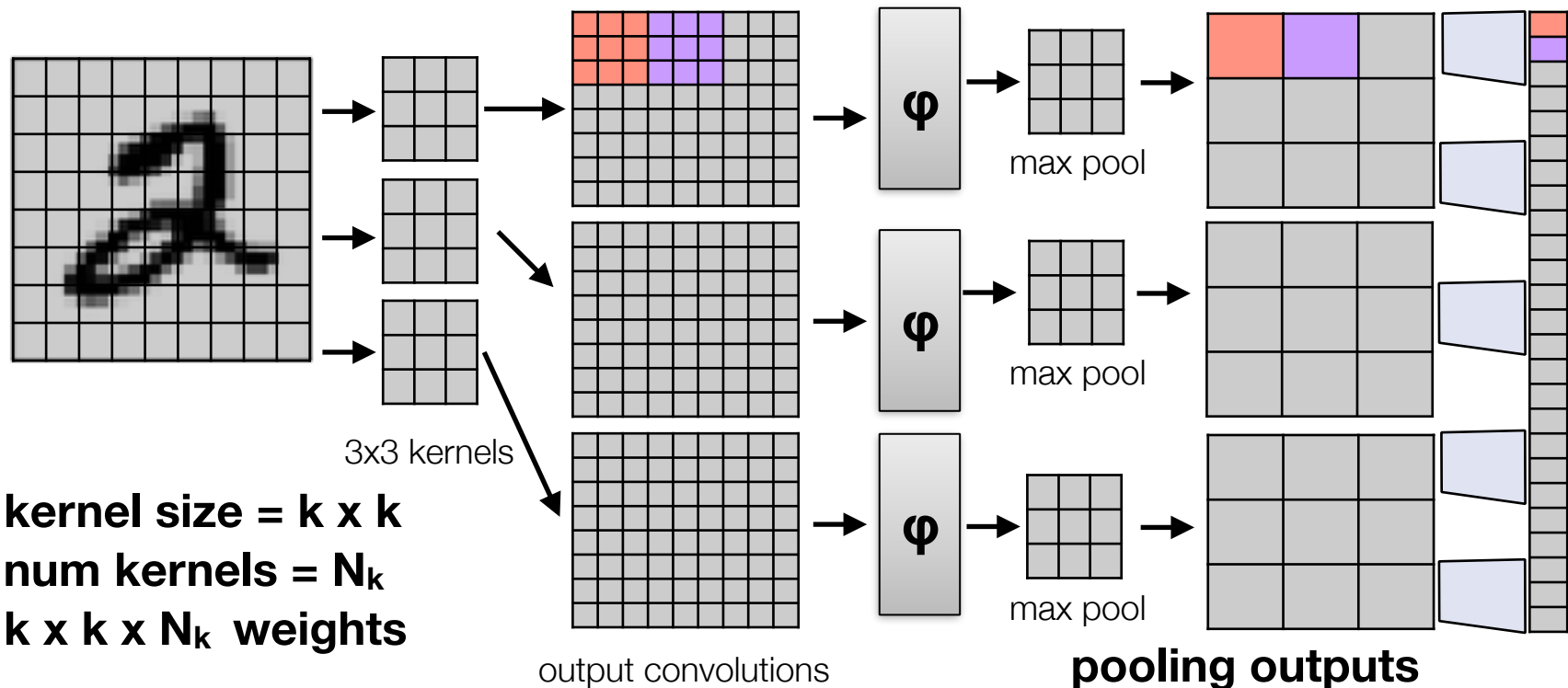
max pool

max pool

max pool

output convolutions

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

3x3x3 = 27 weights

$\mathbf{a}^{(1)}$ $(N{+}1) \times 1$ $\mathbf{W}^{(1)}$ $(N{+}1) \times S^1$ $\mathbf{z}^{(1)}$ $S^1 \times 1$ φ $\mathbf{a}^{(2)}$ $(S^1{+}1) \times 1$ $\mathbf{W}^{(2)}$ $(S^1{+}1) \times S^2$ $\mathbf{z}^{(2)}$ $S^2 \times 1$ φ $\mathbf{a}^{(3)}$ $S^2 \times 1$

$$27 + (27 \times 20) + (20 \times 10) = 767$$

0
1
2
3
4
5
6
7
8
9

From Fully Connected to CNN

3x3 kernels

kernel size = k x k
num kernels = N_k
k x k x N_k weights

output convolutions

max pool

max pool

max pool

pooling outputs
if stride = k
(K/k) x (K/k) x N_k

total inputs
N_k x (k^2 +K^2/k^2)

num filters

filter dimension

image dimension

$\mathbf{a}^{(1)}$
$(N+1) \times 1$

$\mathbf{W}^{(1)}$
$(N+1) \times S^1$

$\mathbf{z}^{(1)}$
$S^1 \times 1$

$\boldsymbol{\varphi}$

$\mathbf{a}^{(2)}$
$(S^1+1) \times 1$

$\mathbf{W}^{(2)}$
$(S^1+1) \times S^2$

$\mathbf{z}^{(2)}$
$S^2 \times 1$

$\boldsymbol{\varphi}$

$\mathbf{a}^{(3)}$
$S^2 \times 1$

CNN gradient

3x3 kernels

W$_{a,b}$

output convolutions   **C**
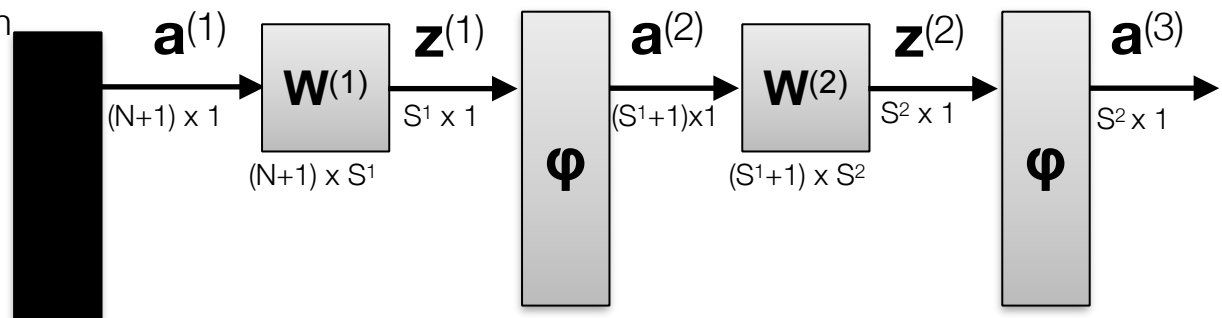
max pool
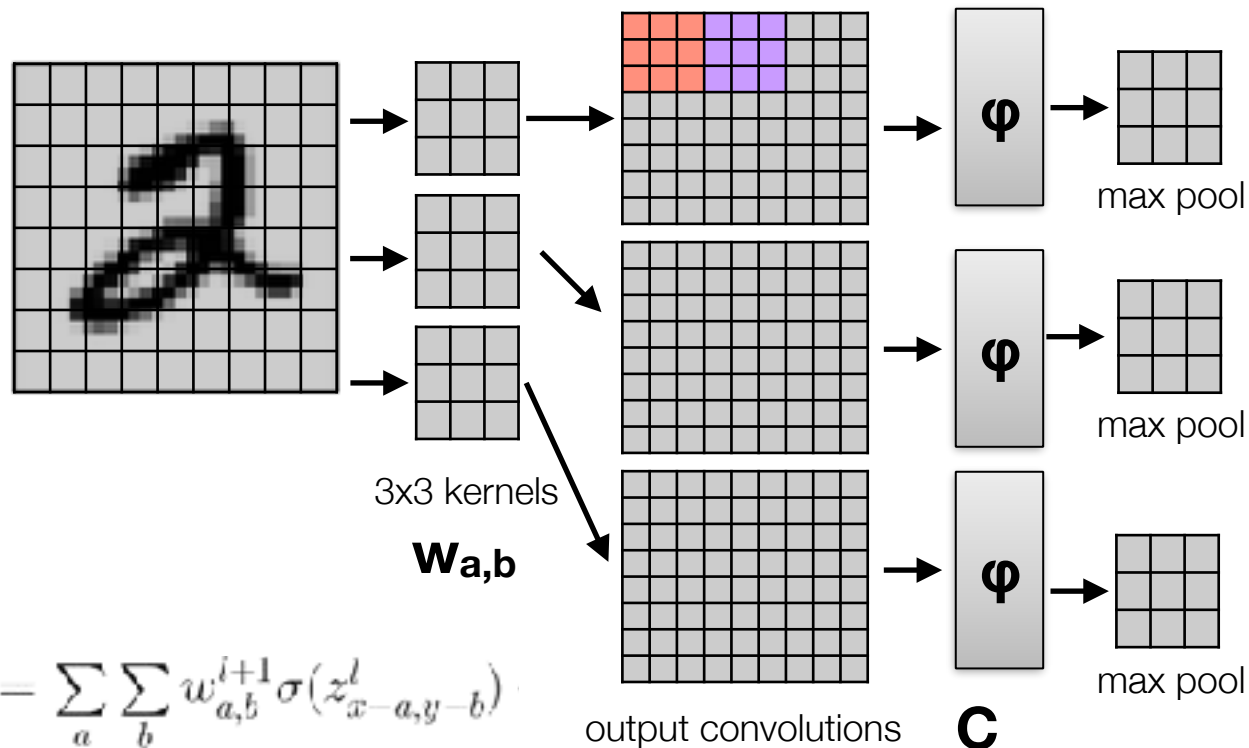
max pool

max pool

Derivative of max pool is easy:

for each input x_i

$$f'(x_i) = 1 \text{ if } x_i \text{ is max}$$
$$0 \text{ else}$$

$$= \sum_a \sum_b w_{a,b}^{l+1} \sigma(z_{x-a,y-b}^l)$$
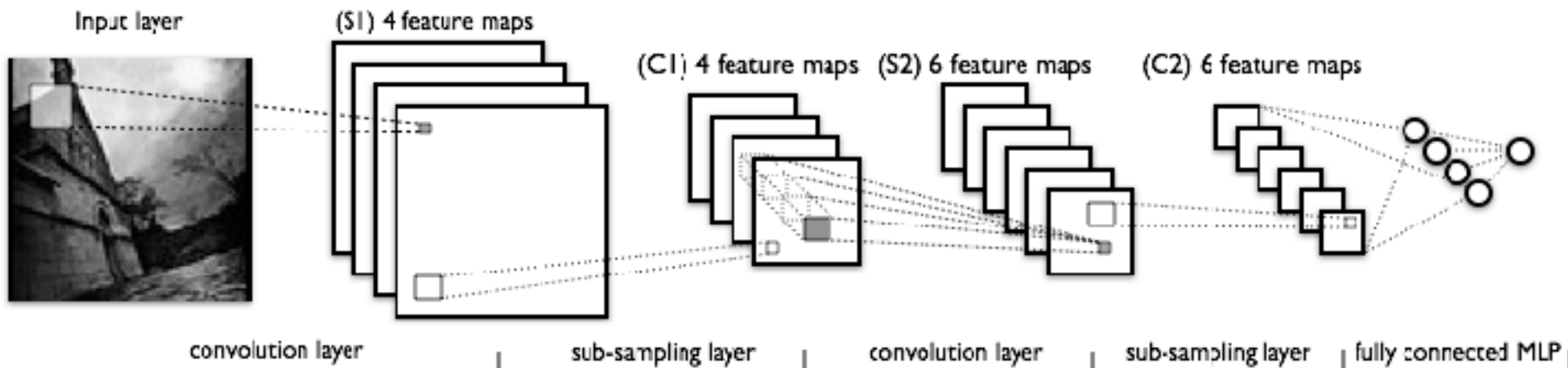
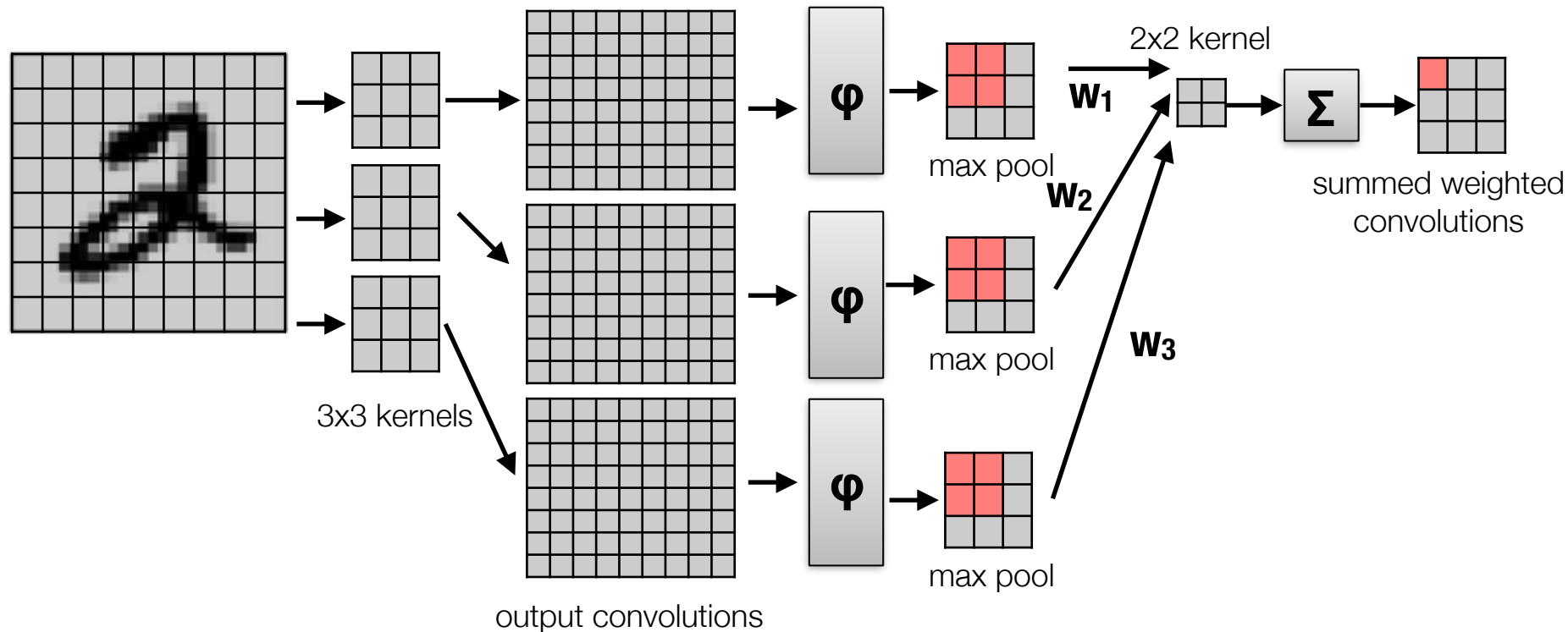Derivative of convolution is more involved:

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

$$\sum_x \sum_y \delta_{x,y}^l \sigma(z_{x-a,y-b}^{l-1}) = \delta_{a,b}^l * \sigma(z_{-a,-b}^{l-1}) = \delta_{a,b}^l * \sigma(ROT180(z_{a,b}^{l-1}))$$
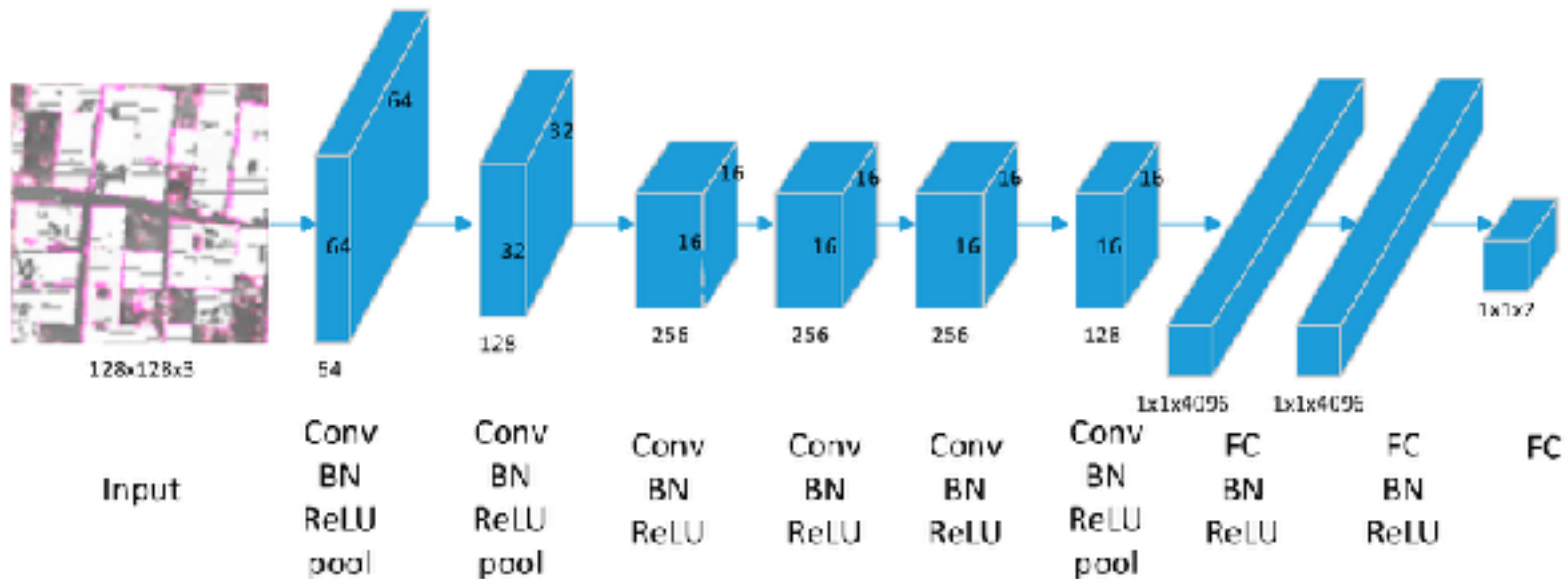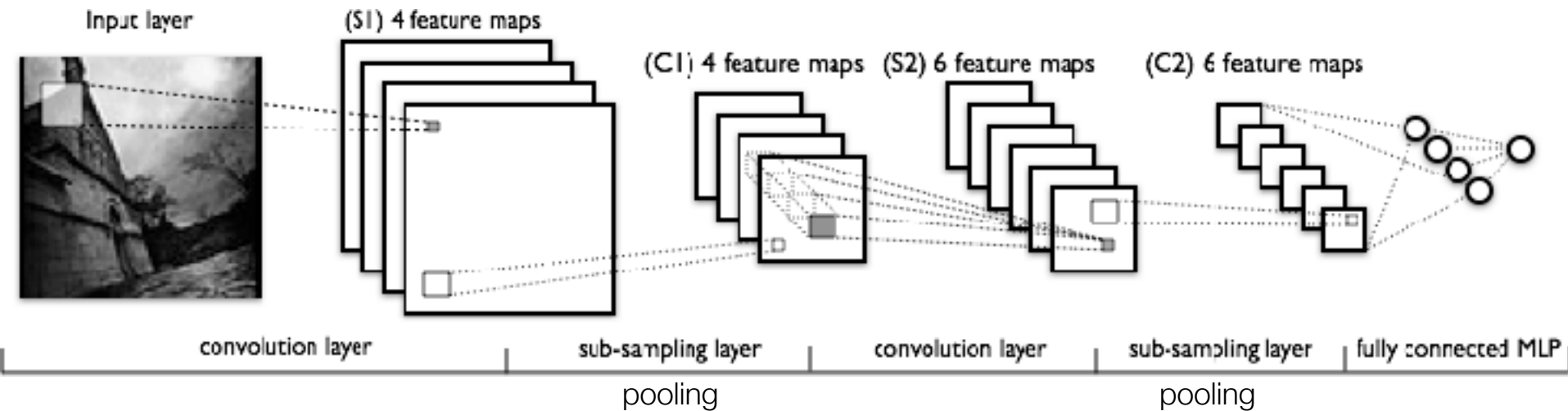
CNN gradient

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

CNN adding more convolutional layers

3x3 kernels

output convolutions

φ

max pool

2x2 kernel

W₁

W₂

W₃

Σ

summed weighted convolutions

Input layer

(S1) 4 feature maps

(C1) 4 feature maps

(S2) 6 feature maps

(C2) 6 feature maps

convolution layer

sub-sampling layer

convolution layer

sub-sampling layer

fully connected MLP

Some Example CNN Architectures

Input layer | (S1) 4 feature maps | (C1) 4 feature maps | (S2) 6 feature maps | (C2) 6 feature maps

convolution layer | sub-sampling layer | convolution layer | sub-sampling layer | fully connected MLP

pooling   pooling

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   UNIVERSITY of WYOMING   NASA Jet Propulsion Laboratory California Institute of Technology

Demo

TensorFlow and Basic CNNs

Convolutional Neural Networks
in TensorFlow
with Keras

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 this week
- But we will start RNN next time
- Agenda:
    - History of CNNs
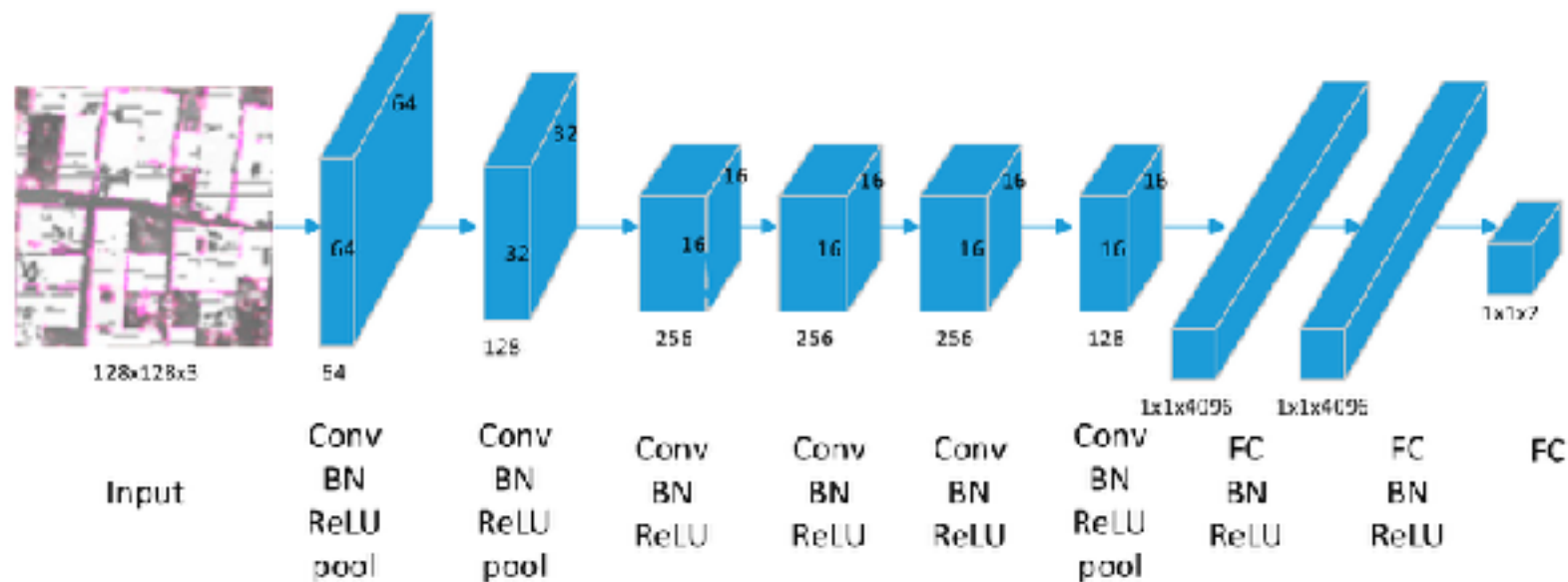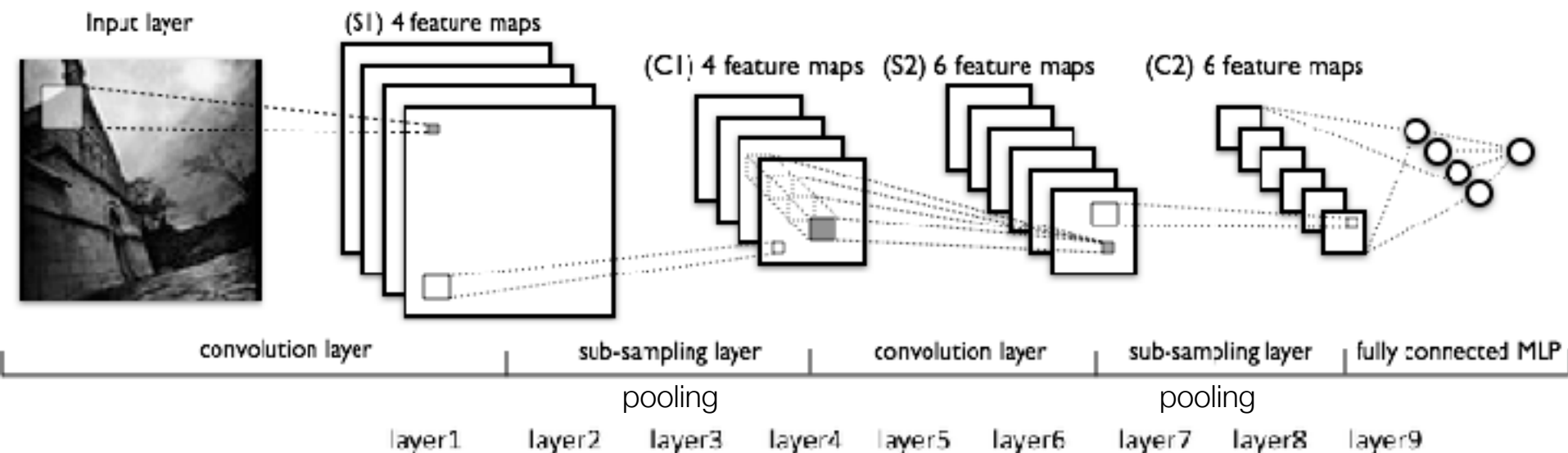   - with Modern CNN Architectures

TensorFlow and Basic CNNs

Finish Demo

Convolutional Neural Networks
in TensorFlow
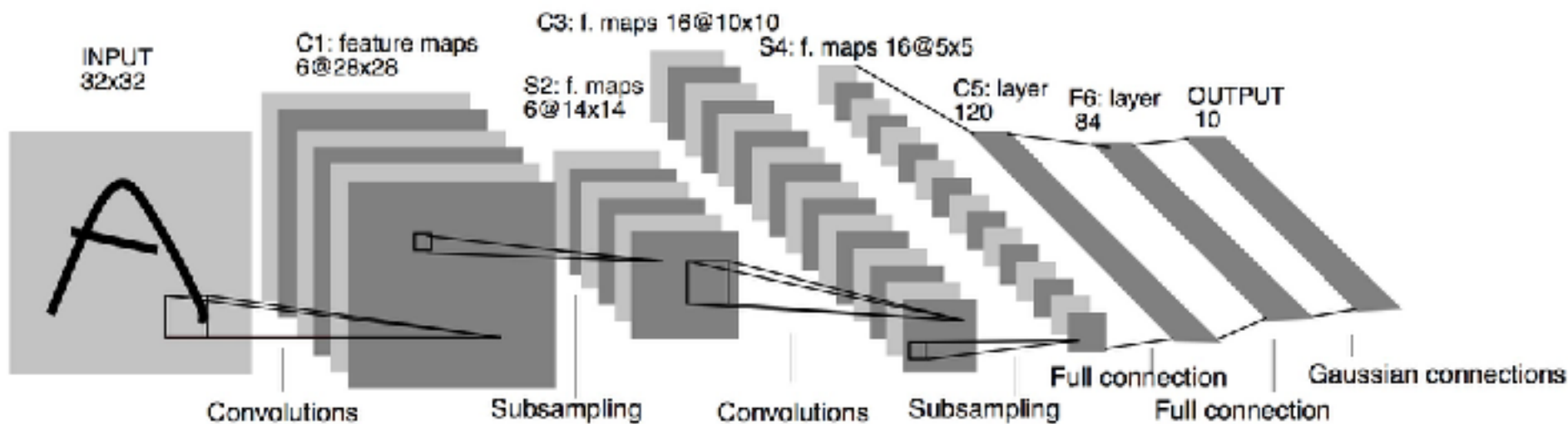with Keras
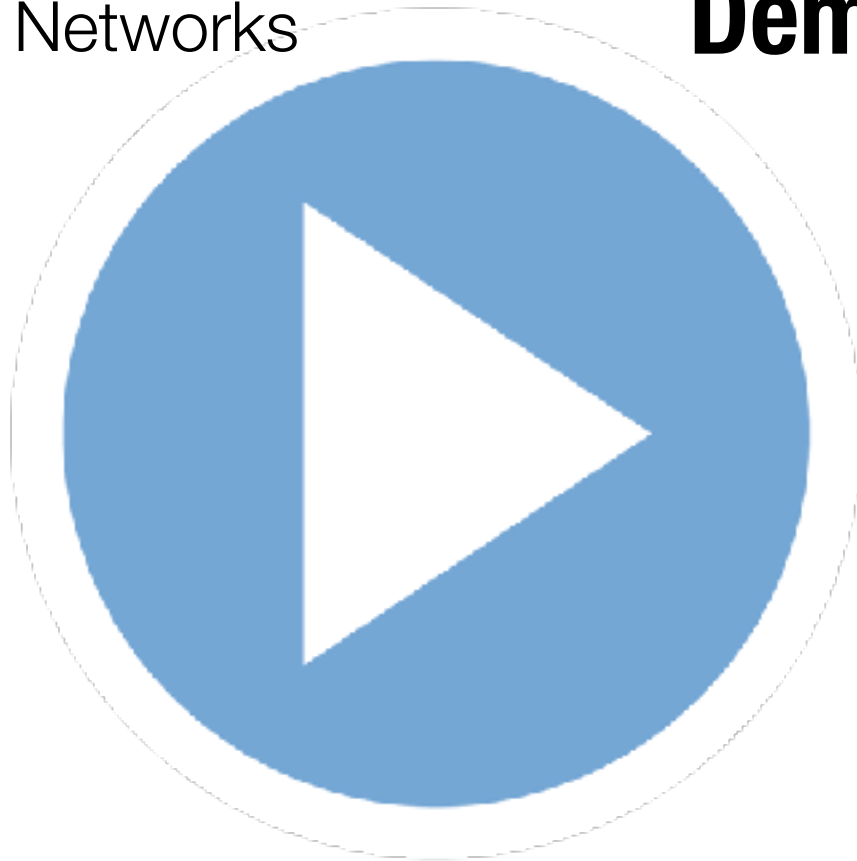
Last Time:

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
    avg   **tanh or sigmoid**

https://culurciello.github.io/tech/2016/06/04/nets.html

CNN History

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

¯_(ツ)_/¯

- 2010

Types of CNN, 2010

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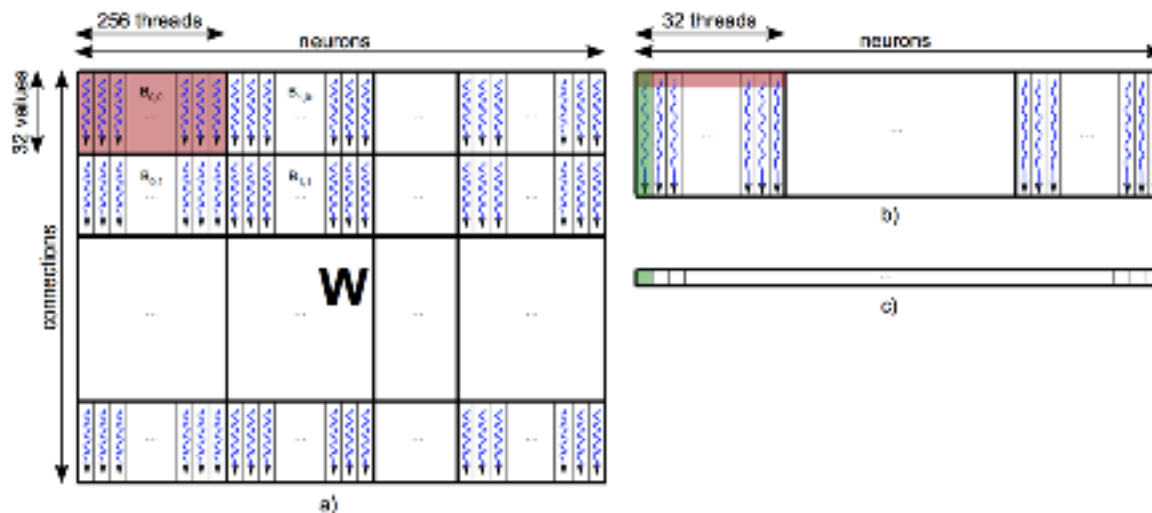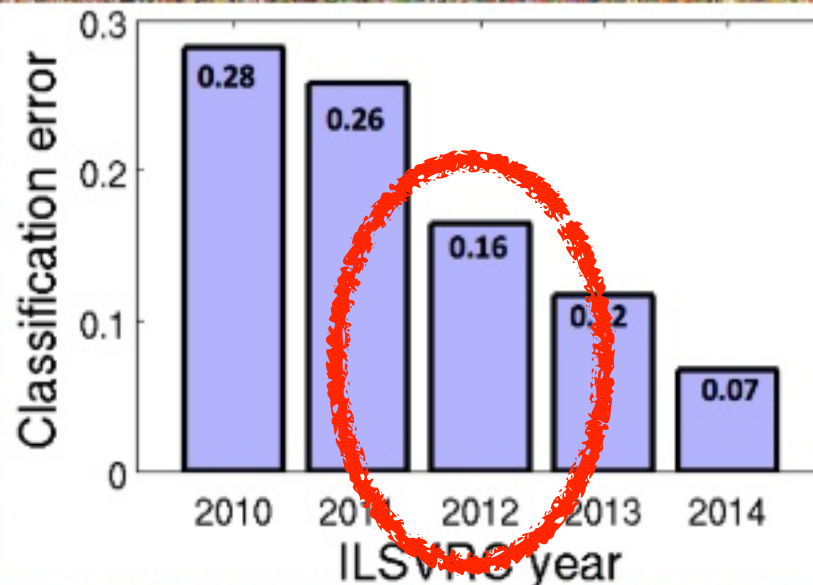- Major contribution: NVIDIA parallelized training algorithms

Dan
Ciresan

AI Researcher
IDSA, Switzerland

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)

Types of CNN, 2012

AlexNet diagram (right side):
- FC 1000
- FC 4096 / ReLU
- FC 4096 / ReLU
- Max Pool 3x3s2
- Conv 3x3s1, 256 / ReLU
- Conv 3x3s1, 384 / ReLU
- Conv 3x3s1, 384 / ReLU
- Max Pool 3x3s2
- Local Response Norm
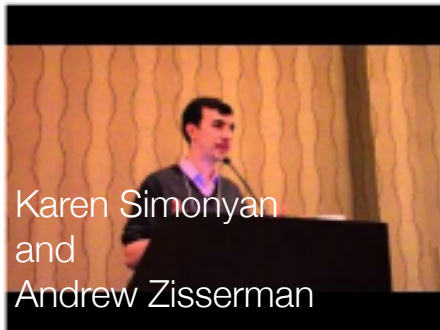- Conv 5x5s1, 256 / ReLU
- Max Pool 3x3s2
- Local Response Norm
- Conv 11x11s4, 96 / ReLU

Alex Krizhevsky
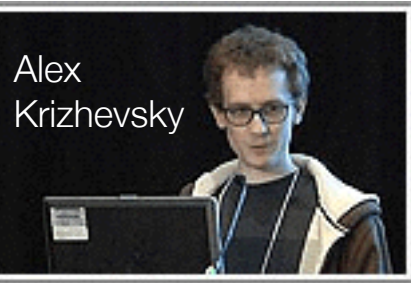
Google

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

Warning

DON'T TRY WHAT YOU'RE ABOUT TO SEE AT HOME

WE'RE WHAT YOU CALL EXPERTS

Types of CNN, 2013

Karen Simonyan
and
Andrew Zisserman

UNIVERSITY OF OXFORD

Google DeepMind

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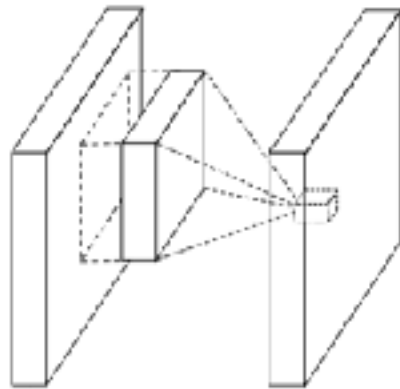 - trained on NVIDIA GPUs for 2-3 weeks

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 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 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 conv1-256	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 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 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

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

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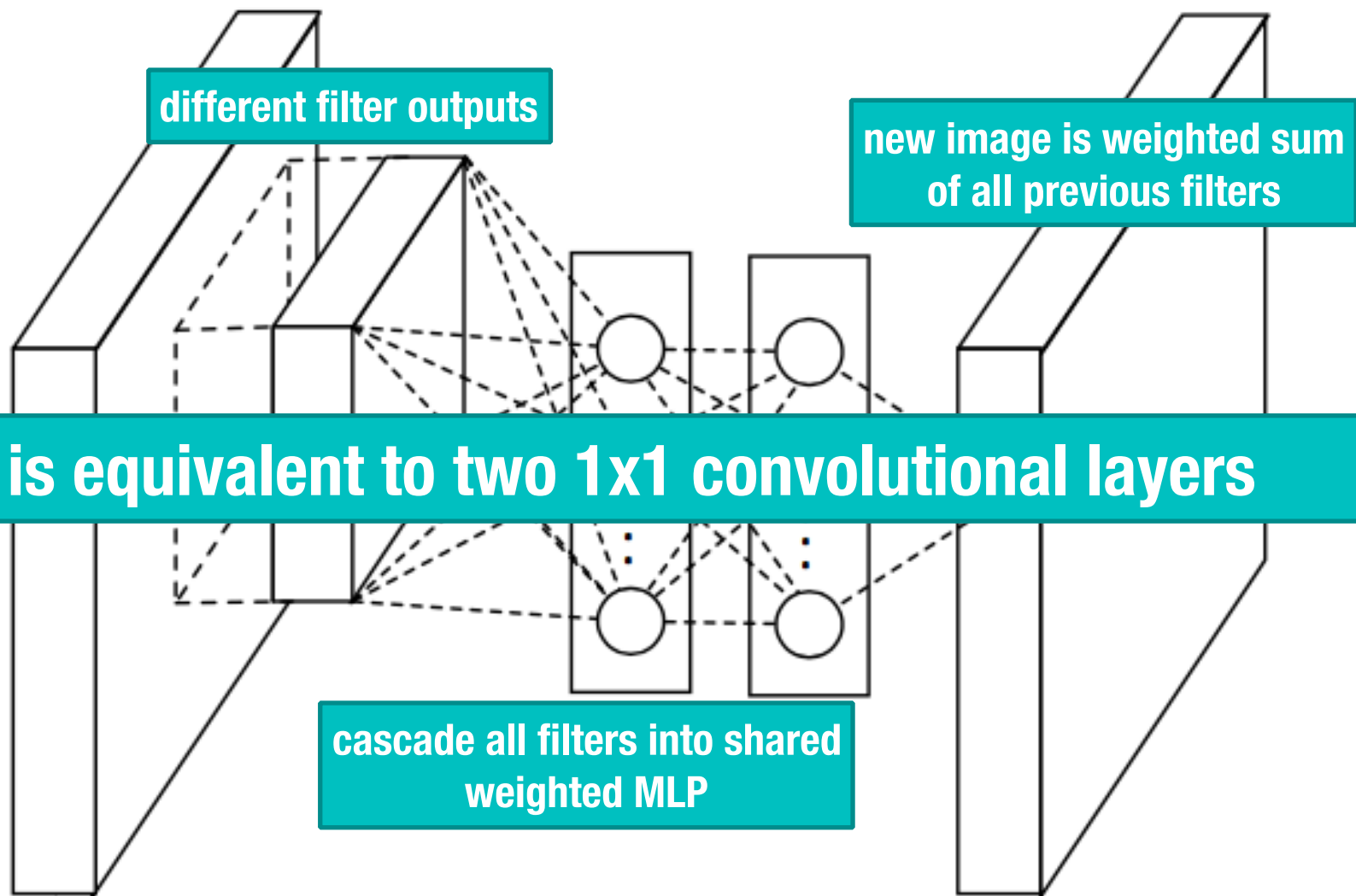
- Network in Network **NiN**
  - or MLPConv

Min Lin[1,2], Qiang Chen[2], Shuicheng Yan[2]
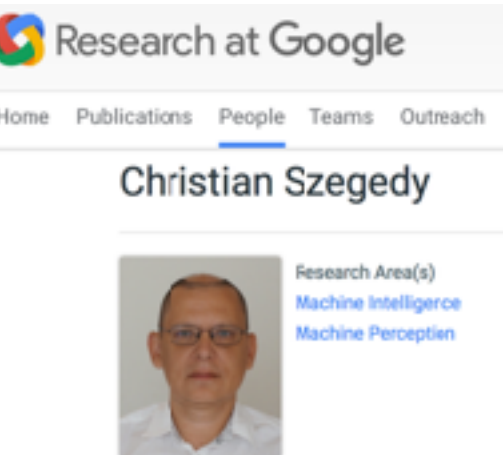[1]Graduate School for Integrative Sciences and Engineering
[2]Department of Electronic & Computer Engineering
National University of Singapore, Singapore
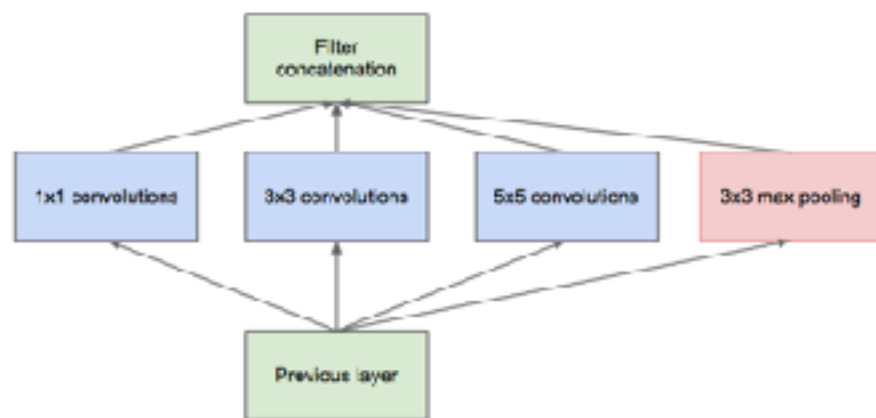{linmin,chenqiang,eleyans}@nus.edu.sg

(a) Linear convolution layer

(b) Mlpconv layer

36

Types of CNN, 2014

different filter outputs

new image is weighted sum of all previous filters

this is equivalent to two 1x1 convolutional layers

cascade all filters into shared weighted MLP

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

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
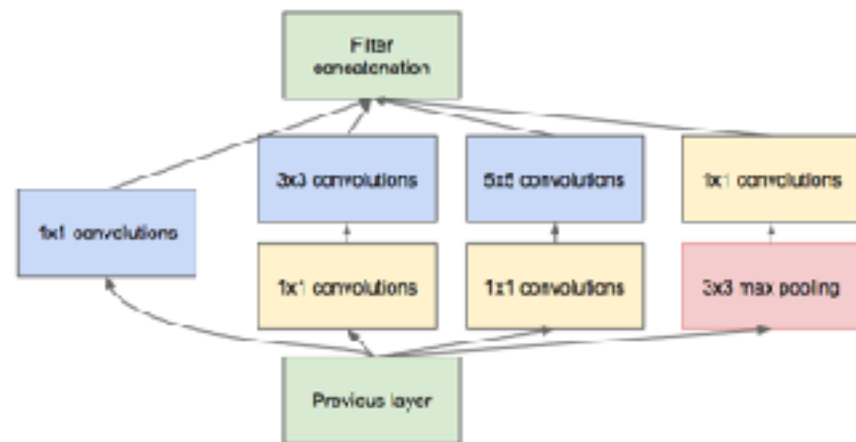
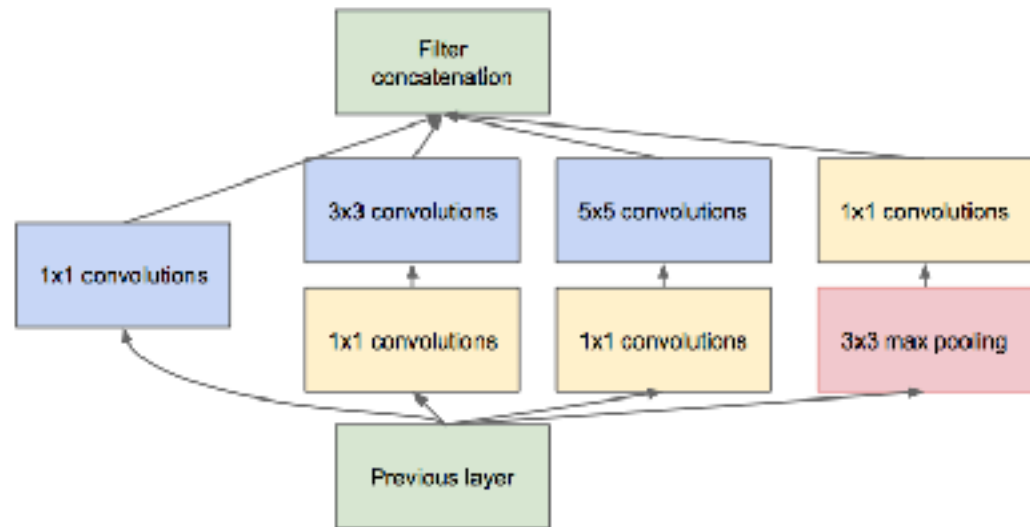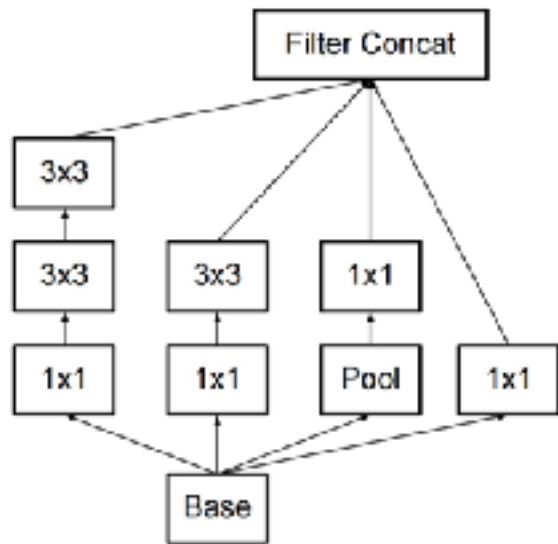(b) Inception module with dimension reductions

Figure 2: Inception module

Types of CNN, 2015 February and December

Research at Google

Home Publications People Teams Outreach

Christian Szegedy

Research Area(s)
Machine Intelligence
Machine Perception

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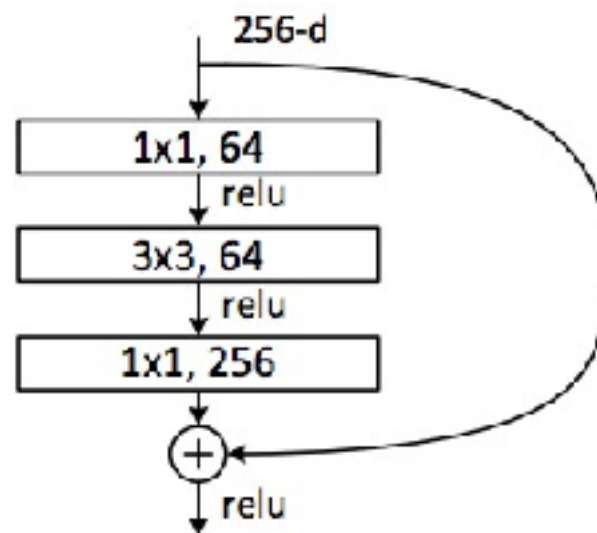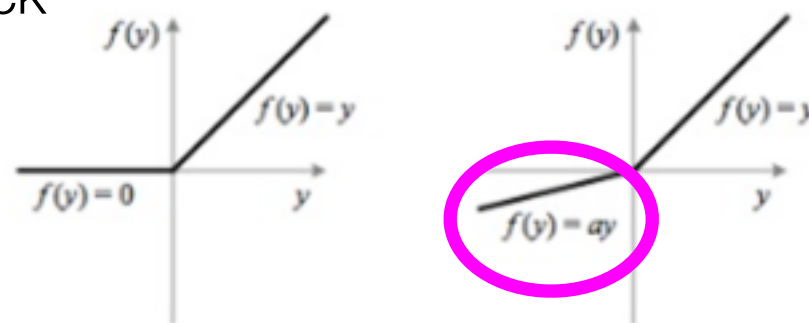
Microsoft
Research

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

- **ResNet**
  - PReLU: adaptive trained slope
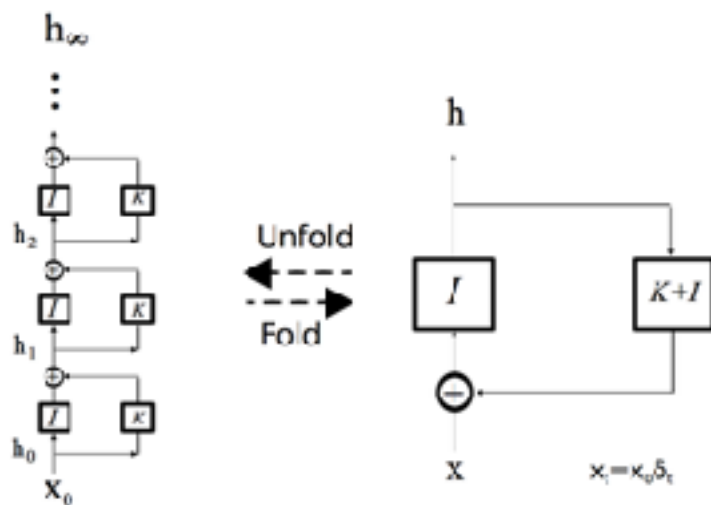
(A) ResNet with shared weights   (B) ResNet in recurrent form

256-d
1×1, 64
relu
3×3, 64
relu
1×1, 256
relu

- NiN: triple bypass layer
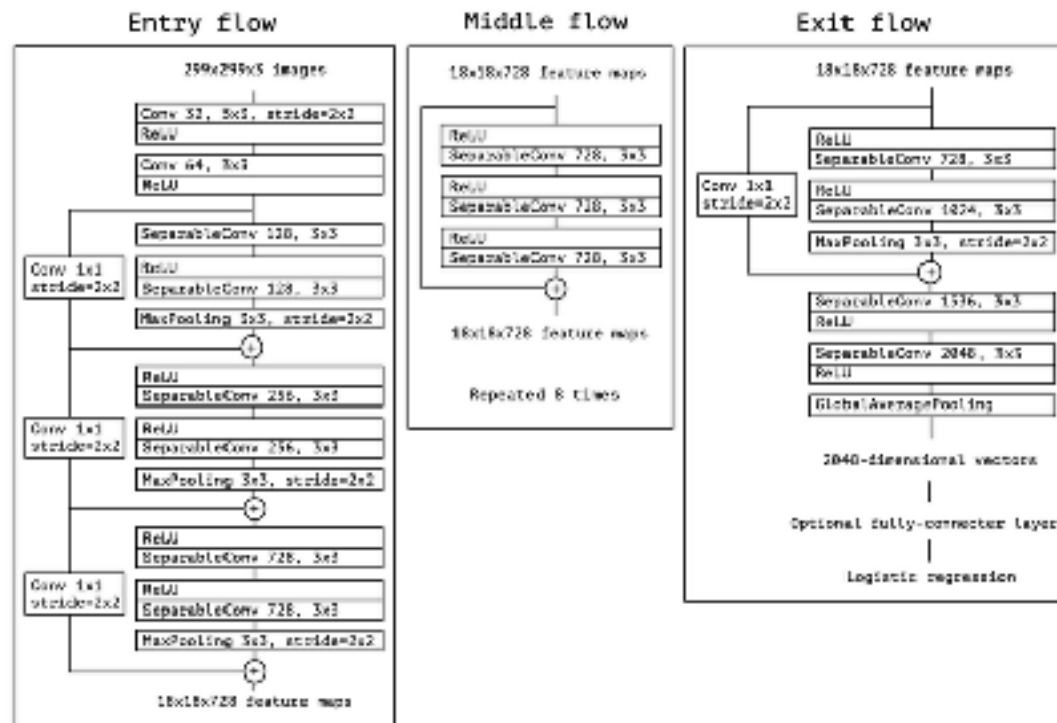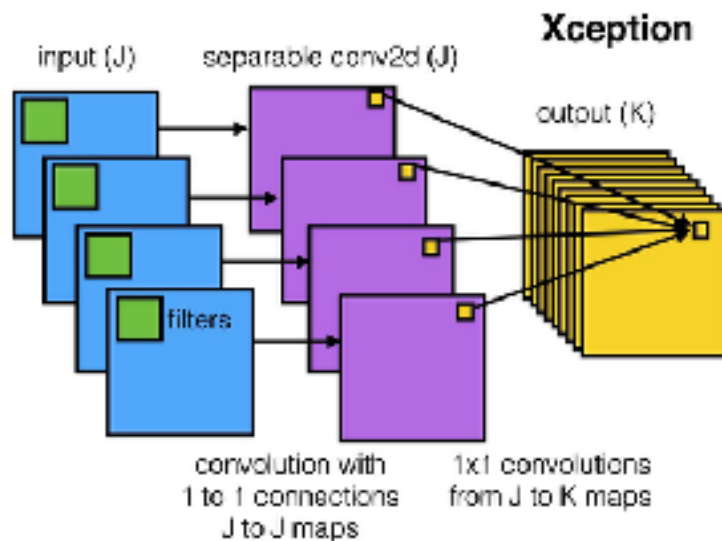 - similar to bottelneck

Types of CNN, 2017 April

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

Xception

Francois Chollet
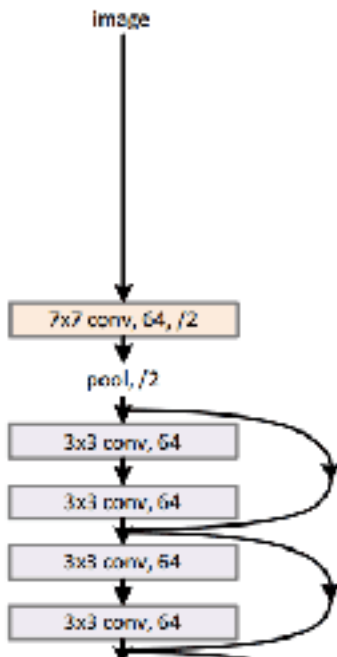**Google**

input (J)   separable conv2d (J)   outout (K)

filters

convolution with
1 to 1 connections
J to J maps

1x1 convolutions
from J to K maps

Xception

Entry flow   Middle flow   Exit flow

How big are these networks?

How big are these networks?

AlexNet, 8 layers
(ILSVRC 2012)

VGG, 19 layers
(ILSVRC 2014)

GoogleNet, 22 layers
(ILSVRC 2014)

ResNet, 152 layers
(ILSVRC 2015)

34-layer residual

image

7x7 conv, 64, /2

pool, /2

3x3 conv, 64

3x3 conv, 64

3x3 conv, 64

3x3 conv, 64

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

More Modern CNN Architectures

Demo

Even more Convolutional Neural Networks
…in TensorFlow
…with Keras

Next Time:

- CNN Lab Discussion (Town Hall)

- Intro to Recurrent Neural Network Architectures
  - RNNs, GRUs, LSTMs
  - Training for characters