Stony Brook University

IMAGE AS DATA IN SOCIAL SCIENCE

Yongjun Zhang, Ph.D.
Dept of Sociology and IACS
https://yongjunzhang.com

Today's Agenda

1. Mini Lecture on Image as Data (6:00-6:50PM)
2. Lab Tutorial on Image Processing (7:00-7:50 PM)
3. Guest Speaker Dr. Han Zhang (8:00-8:50PM)

Dr. Han Zhang
Ass. Prof. of Sociology @HKUST

Symposium on Big Data–Innovations in Computational Social Science

Sociological Methodology
2019, Vol. 49(1) 1–57

DOI: 10.1177/0081175019860244
http://sm.sagepub.com
SAGE

1

CASM: A DEEP-LEARNING APPROACH FOR IDENTIFYING COLLECTIVE ACTION EVENTS WITH TEXT AND IMAGE DATA FROM SOCIAL MEDIA

*Han Zhang**
Jennifer Pan†

Text as Data -> Image as Data

Why should we care about image as data?

Data explosion vs. Data scarcity

Twitter API

Facebook Graph API

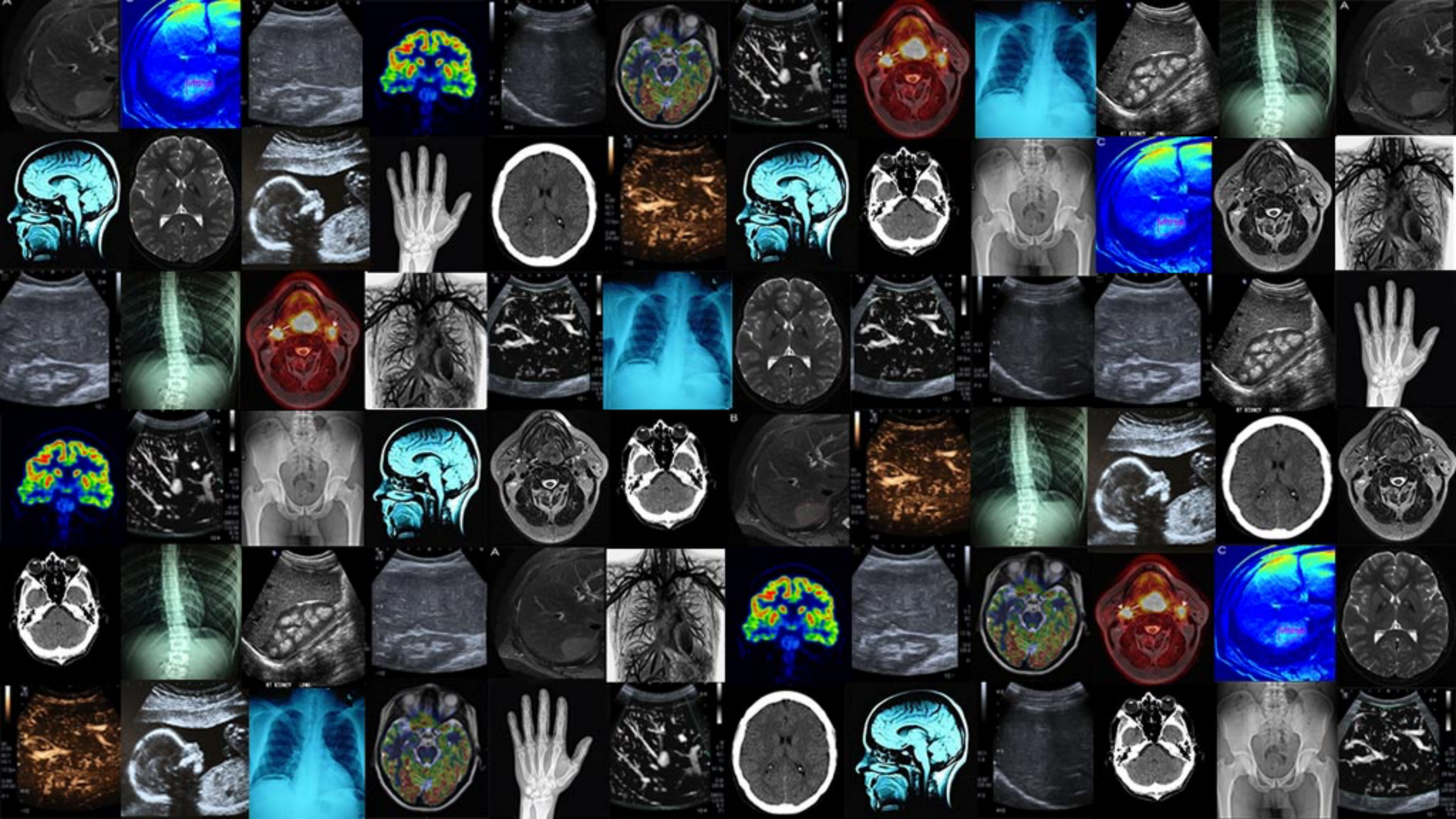
Black Lives Matter and Anti-ICE protest in Times Square, Manhattan, New York City, 1:26 PM, Saturday, September 19, 2020
Arrests first seen in third video
This is a multi tweet thread
1/
#BLM #Nyc #ny #BlackLivesMatter✊🏾✊🏿✊🏼 #ice
#TimesSquare #manhattan #newYorkCity #NewYork
#protest

3:24 PM · Sep 19, 2020 · Twitter for iPhone

BLM Protest?
Time?
Location?

Any Police Presence?

Satellite Imagery in Google Earth



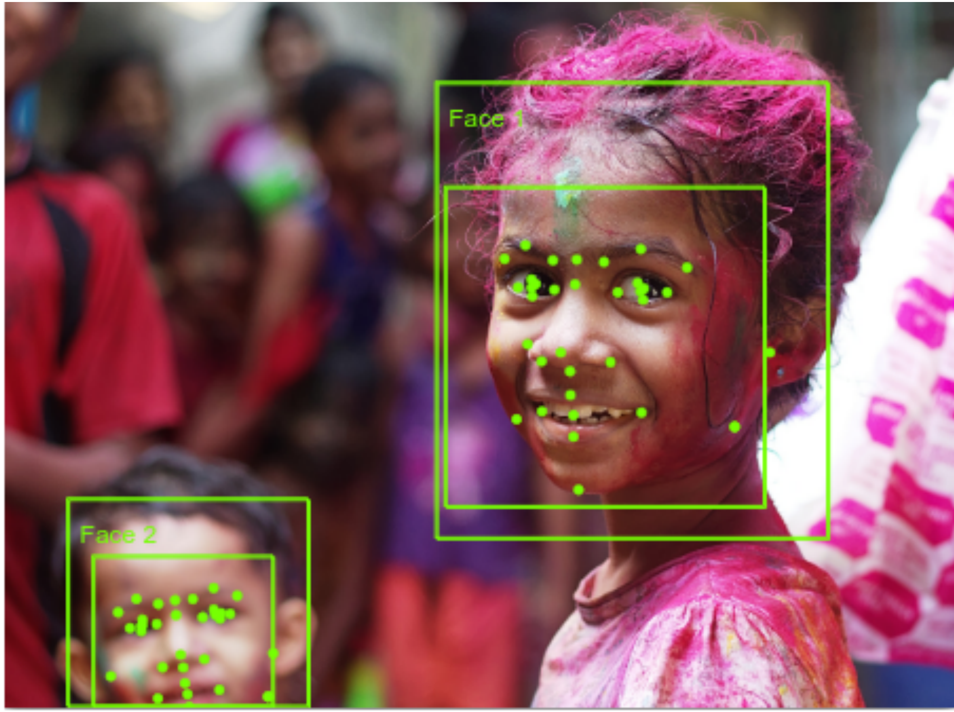


Image credit: Himanshu Singh Gurjar on Unsplash
Surveillance Camera; Smart Phone; Using Google Vision API

RESEARCH

RESEARCH ARTICLES

ECONOMICS

Combining satellite imagery and machine learning to predict poverty

Neal Jean,[1,2*] Marshall Burke,[3,4,5*†] Michael Xie,[1] W. Matthew Davis,[4]
David B. Lobell,[3,4] Stefano Ermon[1]

Reliable data on economic livelihoods remain scarce in the developing world, hampering efforts to study these outcomes and to design policies that improve them. Here we demonstrate an accurate, inexpensive, and scalable method for estimating consumption expenditure and asset wealth from high-resolution satellite imagery. Using survey and satellite data from five African countries—Nigeria, Tanzania, Uganda, Malawi, and Rwanda—we show how a convolutional neural network can be trained to identify image features that can explain up to 75% of the variation in local-level economic outcomes. Our method, which requires only publicly available data, could transform efforts to track and target poverty in developing countries. It also demonstrates how powerful machine learning techniques can be applied in a setting with limited training data, suggesting broad potential application across many scientific domains.


HIGH RESOLUTION
POPULATION
DENSITY MAPS

Stony Brook University

Typical Tasks in Image as Data

Image Classification

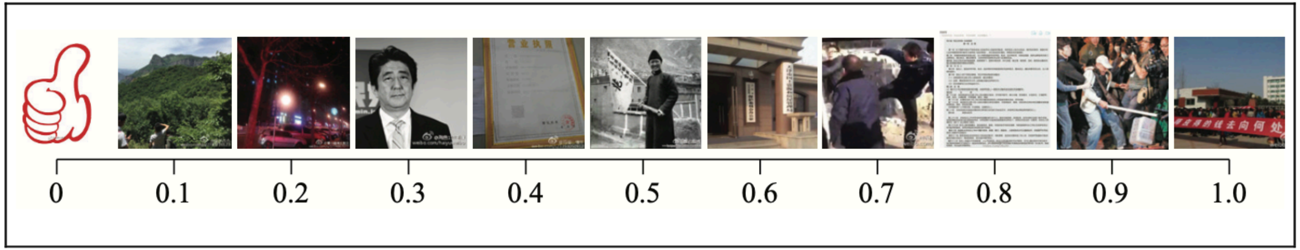


Figure 3. Images with their predicted probabilities of relating to collective action generated from the convolutional neural network in the first-stage classifier (some images are cropped).
Note. Images from Weibo.com.

Typical Tasks in Image as Data

Object Detection (Using Google Vision API) ; Image source: https://ocean.sagepub.com/

Stony Brook University

Typical Tasks in Image as Data

Face Detection and Classification

Taking a Pass

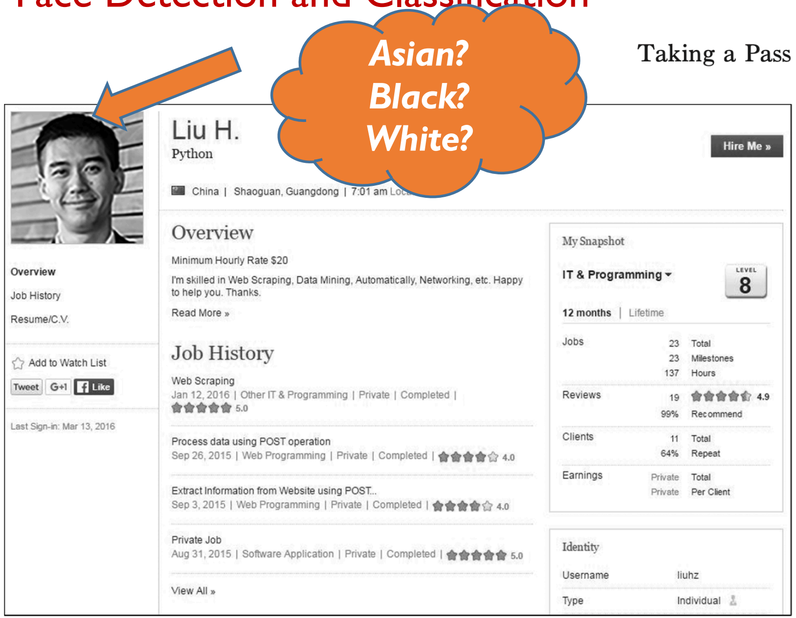


FIG. 4.—Freelancer profile page

Taking a Pass: How Proportional Prejudice and Decisions Not to Hire Reproduce Gender Segregation[1]

Ming D. Leung
University of California, Riverside

Sharon Koppman
University of California, Irvine

Why now?

The rapid development of deep learning methods!

IMAGENET

14,197,122 images, 21841 synsets indexed

Explore Download Challenges Publications Updates About

Not logged in. Login I Signup

ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently we have an average of over five hundred images per node. We hope ImageNet will become a useful resource for researchers, educators, students and all of you who share our passion for pictures.
Click here to learn more about ImageNet, Click here to join the ImageNet mailing list.

What do these images have in common? *Find out!*

Research updates on improving ImageNet data

Stony Brook University

SCIENTIFIC REPORTS
natureresearch

Check for updates

OPEN

Comparing different deep learning architectures for classification of chest radiographs

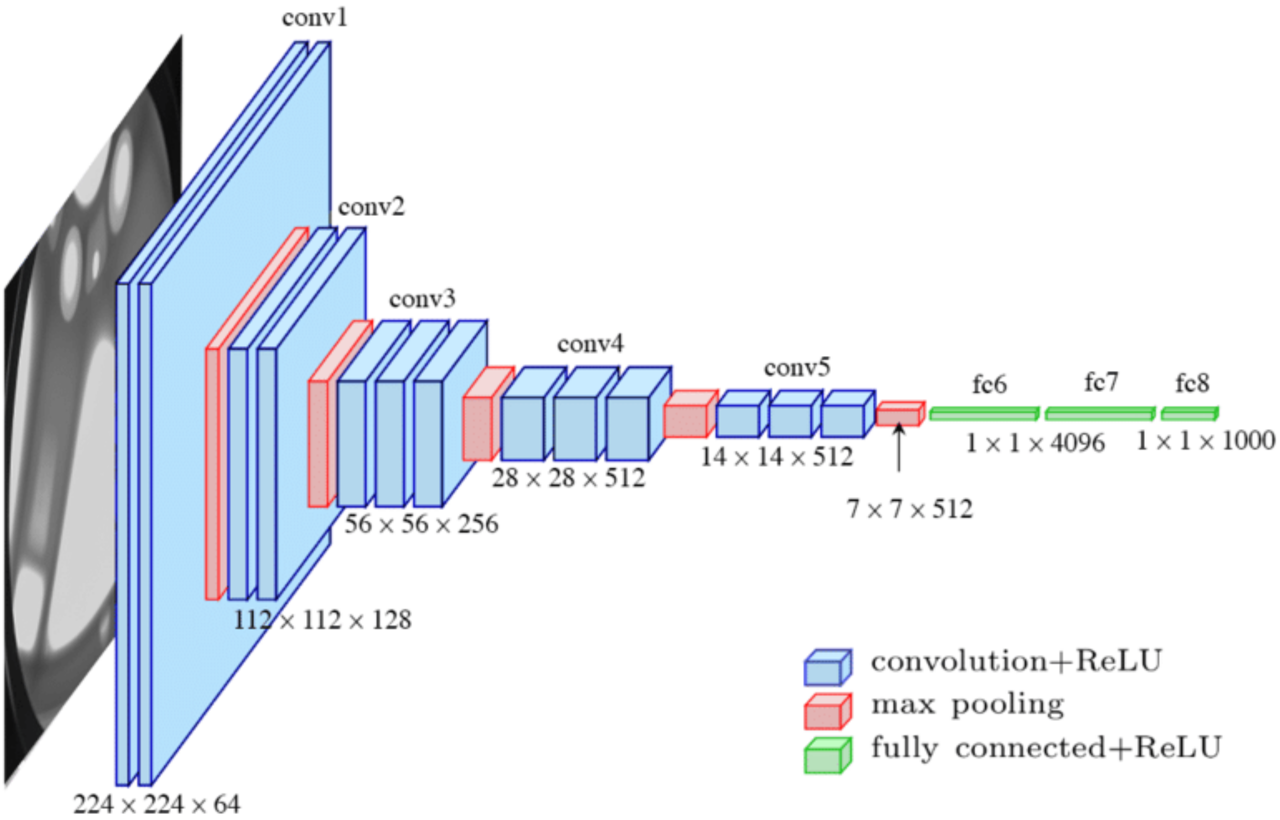
Keno K. Bressem[1✉], Lisa C. Adams[2], Christoph Erxleben[1], Bernd Hamm[1,2], Stefan M. Niehues[1] & Janis L. Vahldiek[1]

Chest radiographs are among the most frequently acquired images in radiology and are often the subject of computer vision research. However, most of the models used to classify chest radiographs are derived from openly available deep neural networks, trained on large image datasets. These datasets differ from chest radiographs in that they are mostly color images and have substantially more labels. Therefore, very deep convolutional neural networks (CNN) designed for ImageNet and often representing more complex relationships, might not be required for the comparably simpler task of classifying medical image data. Sixteen different architectures of CNN were compared regarding the classification performance on two openly available datasets, the CheXpert and COVID-19 Image Data Collection. Areas under the receiver operating characteristics curves (AUROC) between 0.83 and 0.89 could be achieved on the CheXpert dataset. On the COVID-19 Image Data Collection, all models showed an excellent ability to detect COVID-19 and non-COVID pneumonia with AUROC values between 0.983 and 0.998. It could be observed, that more shallow networks may achieve results comparable to their deeper and more complex counterparts with shorter training times, enabling classification performances on medical image data close to the state-of-the-art methods even when using limited hardware.

Pre-Trained Models

Network	S	COVID-19 pneumonia		Non-COVID-19 pneumonia		No pneumonia	
		Sensitivity	Specificity	Sensitivity	Specificity	Sensitivity	Specificity
AlexNet	6	0.93 (0.68–1.00)	0.91 (0.90–0.92)	0.94 (0.93–0.95)	0.91 (0.89–0.93)	0.90 (0.88–0.92)	0.95 (0.94–0.96)
DenseNet-121	6	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.97 (0.96–0.98)	0.97 (0.96–0.98)	0.97 (0.96–0.98)	0.97 (0.96–0.97)
DenseNet-161	6	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.98 (0.98–0.99)	0.96 (0.95–0.97)	0.96 (0.95–0.97)	0.98 (0.98–0.99)
DenseNet-169	6	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.98 (0.97–0.98)	0.96 (0.95–0.97)	0.96 (0.94–0.97)	0.98 (0.97–0.98)
DenseNet-201	6	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.98 (0.97–0.98)	0.97 (0.95–0.98)	0.96 (0.95–0.97)	0.98 (0.97–0.98)
Inception v4	6	0.93 (0.68–1.00)	0.95 (0.94–0.95)	0.93 (0.92–0.94)	0.95 (0.94–0.96)	0.95 (0.93–0.96)	0.93 (0.92–0.94)
ResNet-18	6	0.93 (0.68–1.00)	0.99 (0.99–1.00)	0.97 (0.96–0.98)	0.95 (0.93–0.96)	0.95 (0.93–0.96)	0.97 (0.96–0.98)
ResNet-34	6	0.93 (0.68–1.00)	1.00 (1.00–1.00)	[illegible]	[illegible]	0.96 (0.94–0.97)	0.97 (0.96–0.98)
ResNet-50	6	[illegible]	[illegible]	[illegible]	[illegible]	0.96 (0.94–0.97)	0.98 (0.97–0.98)
ResNet-101	6	[illegible]	[illegible]	[illegible]	[illegible]	0.95 (0.94–0.96)	0.98 (0.97–0.98)
ResNet-152	6	[illegible]	[illegible]	[illegible]	[illegible]	0.97 (0.96–0.98)	0.97 (0.96–0.98)
SqueezeNet-1.0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.90 (0.88–0.92)	0.96 (0.95–0.97)
SqueezeNet-1.1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.92 (0.91–0.94)
VGG-13	6	[illegible]	[illegible]	[illegible]	[illegible]	0.94 (0.92–0.95)	0.97 (0.96–0.97)
VGG-16	6	[illegible]	[illegible]	[illegible]	[illegible]	0.95 (0.93–0.96)	0.97 (0.96–0.98)
VGG-19	6	[illegible]	[illegible]	[illegible]	[illegible]	0.95 (0.94–0.97)	0.97 (0.96–0.98)
AlexNet	2	[illegible]	[illegible]	[illegible]	0.92 (0.90–0.94)	0.92 (0.91–0.94)	0.94 (0.93–0.95)
DenseNet-121	2	0.93 (0.68–1.00)	[illegible]	[illegible]	0.97 (0.96–0.98)	0.97 (0.96–0.98)	0.98 (0.97–0.98)
DenseNet-161	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.98 (0.98–0.99)	0.98 (0.97–0.99)	0.98 (0.97–0.98)	0.98 (0.98–0.99)
DenseNet-169	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.99 (0.98–0.99)	0.97 (0.96–0.98)	0.97 (0.96–0.98)	0.99 (0.98–0.99)
DenseNet-201	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.99 (0.98–0.99)	0.98 (0.97–0.99)	0.98 (0.97–0.99)	0.99 (0.98–0.99)
Inception v4	2	0.93 (0.68–1.00)	0.94 (0.93–0.94)	0.94 (0.93–0.95)	0.95 (0.93–0.96)	0.95 (0.93–0.96)	0.94 (0.93–0.95)
ResNet-18	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.97 (0.96–0.97)	0.96 (0.94–0.97)	0.96 (0.94–0.97)	0.97 (0.96–0.97)
ResNet-34	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.99 (0.98–0.99)	0.95 (0.94–0.96)	0.95 (0.93–0.96)	0.99 (0.98–0.99)
ResNet-50	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.98 (0.97–0.98)	0.96 (0.95–0.97)	0.96 (0.94–0.97)	0.98 (0.97–0.99)
ResNet-101	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.99 (0.98–0.99)	0.96 (0.94–0.97)	0.96 (0.94–0.97)	0.99 (0.98–0.99)
ResNet-152	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.98 (0.97–0.98)	0.96 (0.94–0.97)	0.96 (0.94–0.97)	0.98 (0.97–0.99)
SqueezeNet-1.0	2	0.93 (0.68–1.00)	0.96 (0.95–0.96)	0.96 (0.95–0.97)	0.91 (0.89–0.93)	0.93 (0.92–0.95)	0.93 (0.92–0.95)
SqueezeNet-1.1	2	0.93 (0.68–1.00)	0.99 (0.98–0.99)	0.95 (0.94–0.96)	0.92 (0.90–0.93)	0.92 (0.91–0.94)	0.95 (0.94–0.96)
VGG-13	2	0.93 (0.68–1.00)	1.00 (0.99–1.00)	0.95 (0.94–0.96)	0.97 (0.96–0.98)	0.95 (0.94–0.97)	0.96 (0.95–0.97)
VGG-16	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.98 (0.97–0.98)	0.95 (0.94–0.97)	0.95 (0.94–0.96)	0.98 (0.97–0.98)
VGG-19	2	0.93 (0.68–1.00)	1.00 (1.00–1.00)	0.96 (0.95–0.97)	0.98 (0.97–0.98)	0.98 (0.96–0.98)	0.96 (0.95–0.97)

Table 5. Sensitivity and specificity for the detection of COVID-19 or non-COVID-19 pneumonia. This table gives Sensitivity and specificity, as well as the corresponding confidence intervals for the COVID-19 Image Data Collection. *BS* Batchsize.


conv1
conv2
conv3
conv4
conv5
fc6
fc7
fc8
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU

Stony Brook University

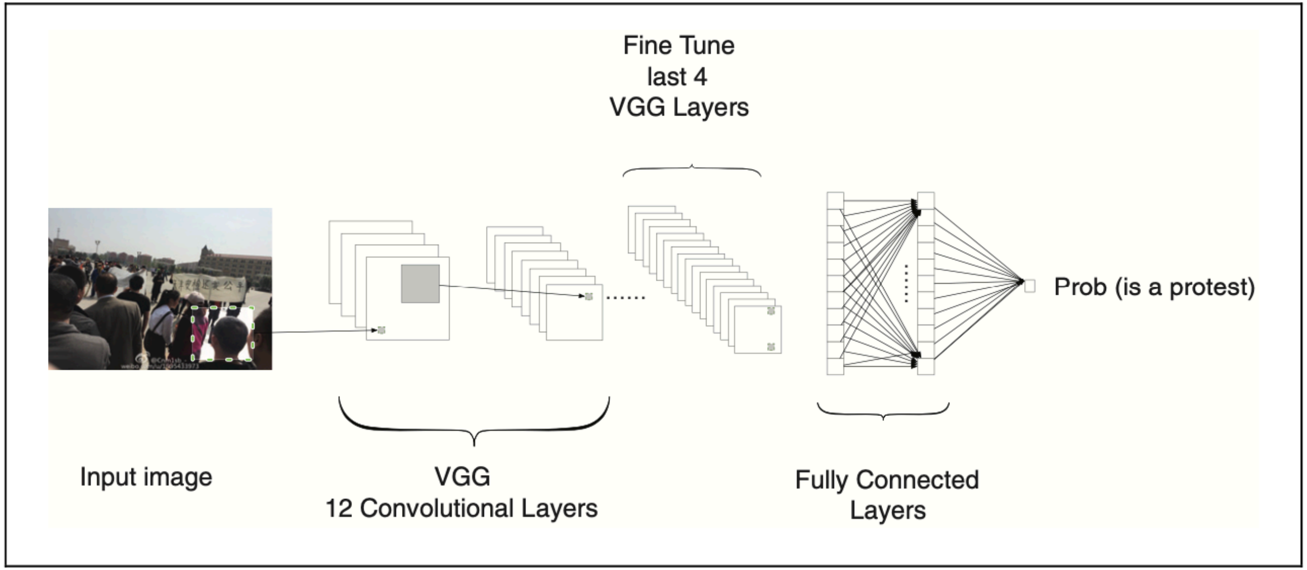
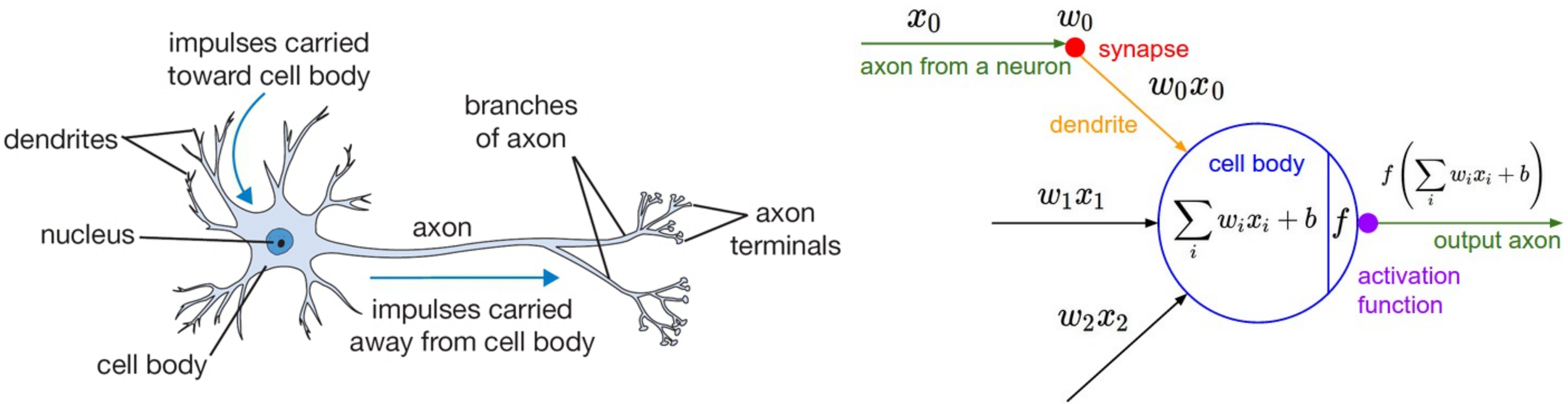


Figure 1. Illustration of our convolutional neural network architecture for image classification.
Note. Input image from Weibo.com.

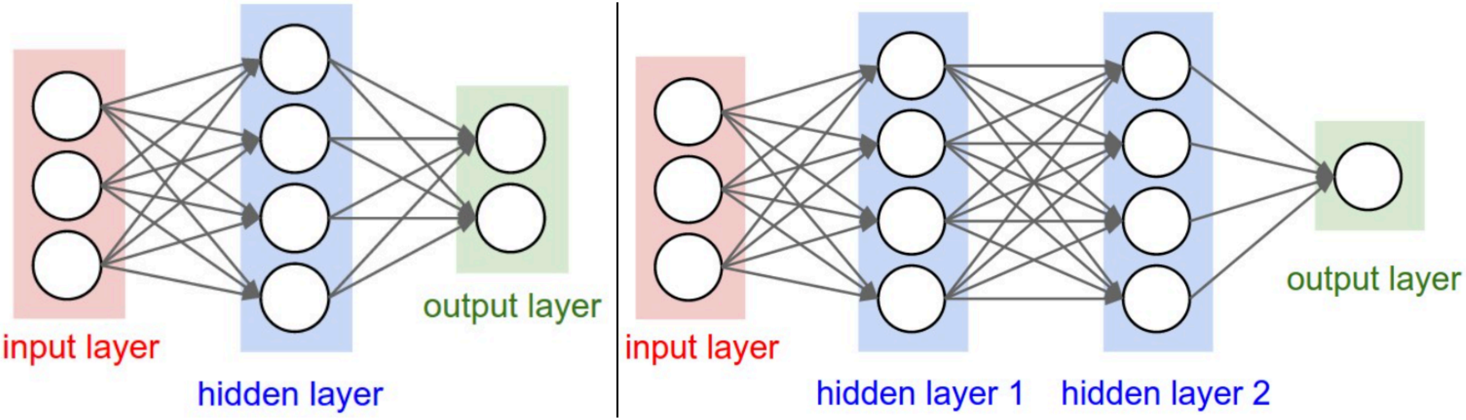
Basics on Neural Network



A cartoon drawing of a biological neuron (left) and its mathematical model (right).

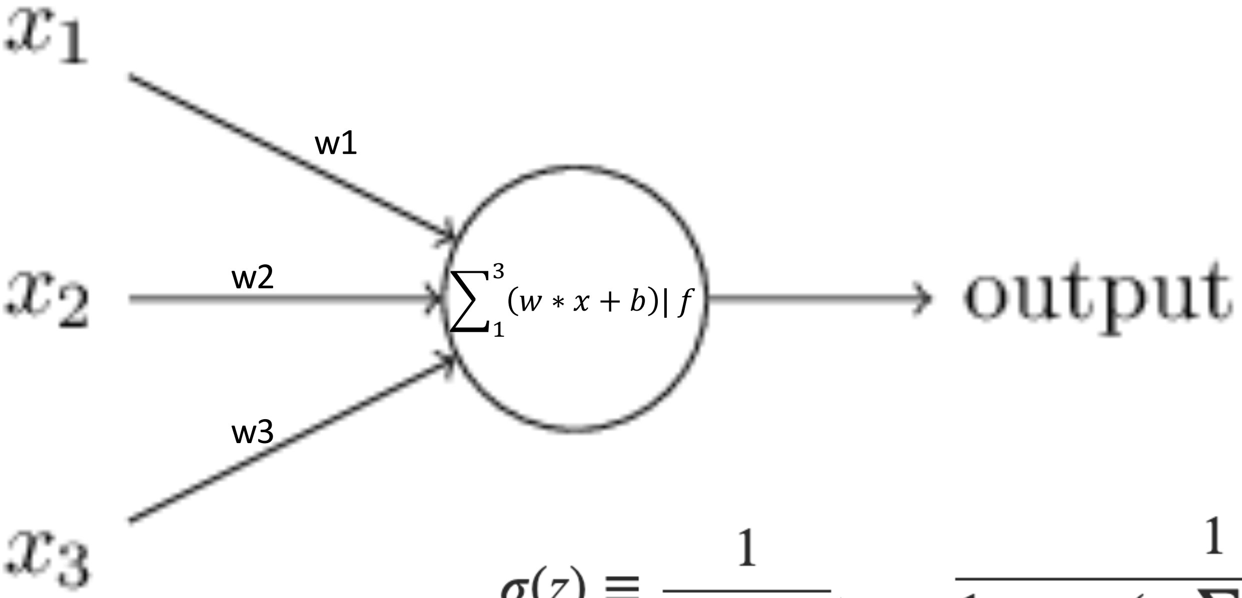
Each neuron performs a dot product with the input and its weights, adds the bias and applies the non-linearity (or activation function)

https://cs231n.github.io/neural-networks-1/



Left: A 2-layer Neural Network (one hidden layer of 4 neurons (or units) and one output layer with 2 neurons), and three inputs. **Right:** A 3-layer neural network with three inputs, two hidden layers of 4 neurons each and one output layer. Notice that in both cases there are connections (synapses) between neurons across layers, but not within a layer.

https://cs231n.github.io/



$$\sigma(z) \equiv \frac{1}{1 + e^{-z}}, \qquad \frac{1}{1 + \exp(-\sum_j w_j x_j - b)}.$$

Let us use binary classification as an example.

If you have some stats background, you know we use logistic regression…

Y = alpha + beta1*X1+ beta2*X2+…+ error

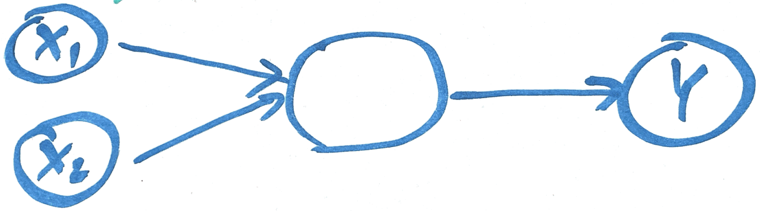
How we estimate these parameters?

Maximize likelihood estimation

$$Y_{Trump} = \alpha + \beta_1 \cdot Money + \beta_2 Media + \cdots + \varepsilon$$

let us say, you have one training dp.

Money = 1B Media = 1M display



$$\begin{cases} 1 & \geq 0.5 \\ 0 & < 0.5 \end{cases}$$

Input Hidden Output

w_1

$w_1 x_1 + w_2 x_2 + b_1 + b_2$

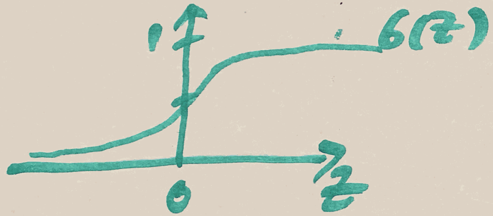
Sigmoid

w_2

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Given X, want $\hat{y} = P(y=1|x)$

output $\hat{y} = \sigma(w^T x + b)$



if z large → 1

if z small → 0

Job: train weights and bias

$$\hat{y}^{(i)} = \sigma(w^T x^{(i)} + b)$$

loss (error) function

$$\mathcal{L}(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$$

$$\mathcal{L}(\hat{y}, y) = -(y \log \hat{y} + (1-y) \log(1-\hat{y}))$$

if $y=1$, $\mathcal{L}(\hat{y}, y) = -\log \hat{y}$

if $y=0$, $\mathcal{L}(\hat{y}, y) = -\log(1-\hat{y})$

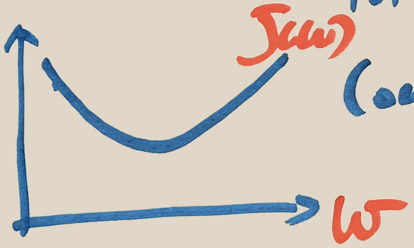
Cost Function:

$$J(w, b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$$

Measure how well your w, b perform predicting the outcome in your entire training dataset

We want to find w, b that minimize

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$



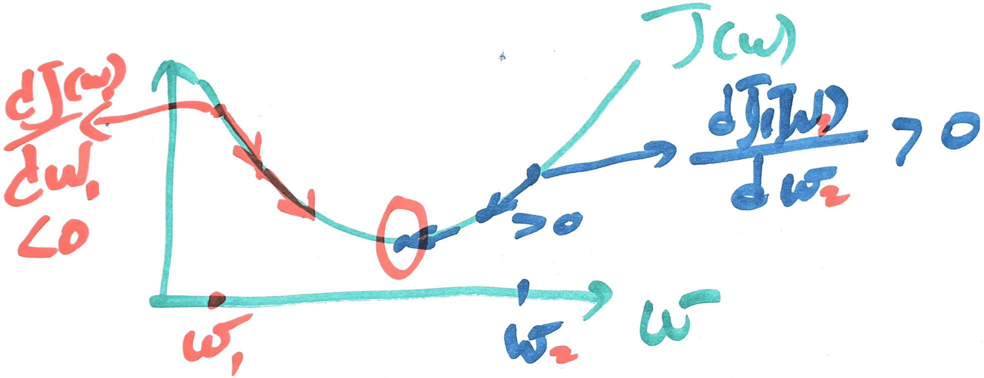
Convex

Repeat:

$$w := w - \alpha \frac{dJ(w)}{dw}$$

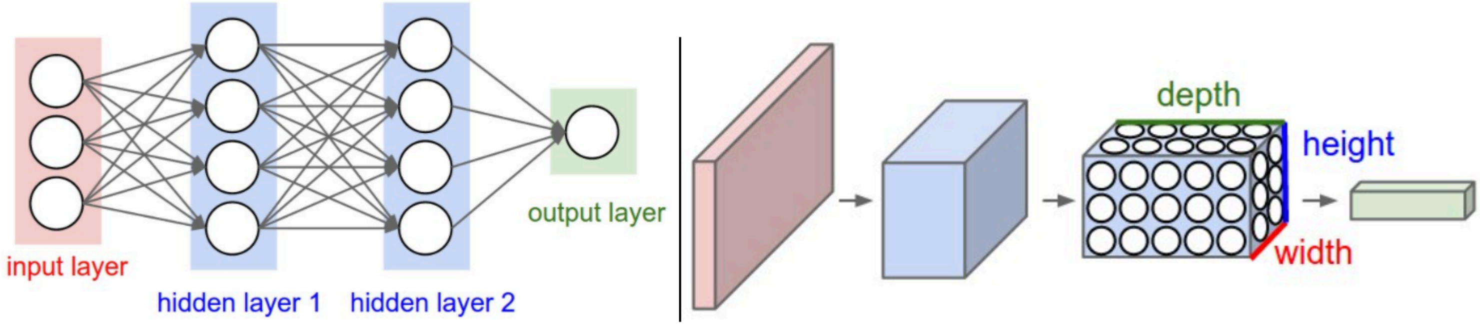
learning Rate

Gradient descent



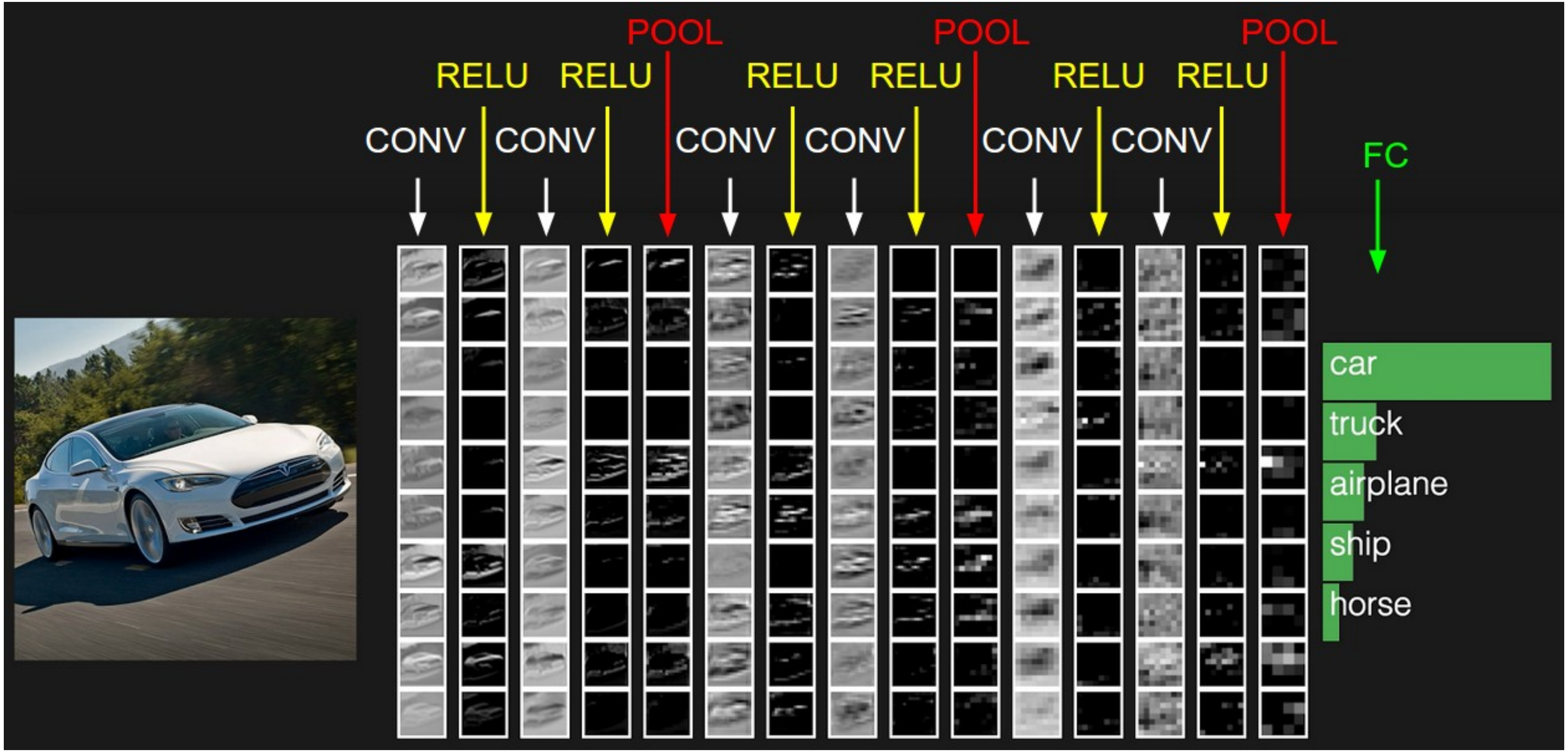
$$w := w - \alpha \frac{dJ(w)}{dw}$$

Convolutional Neural Network



Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

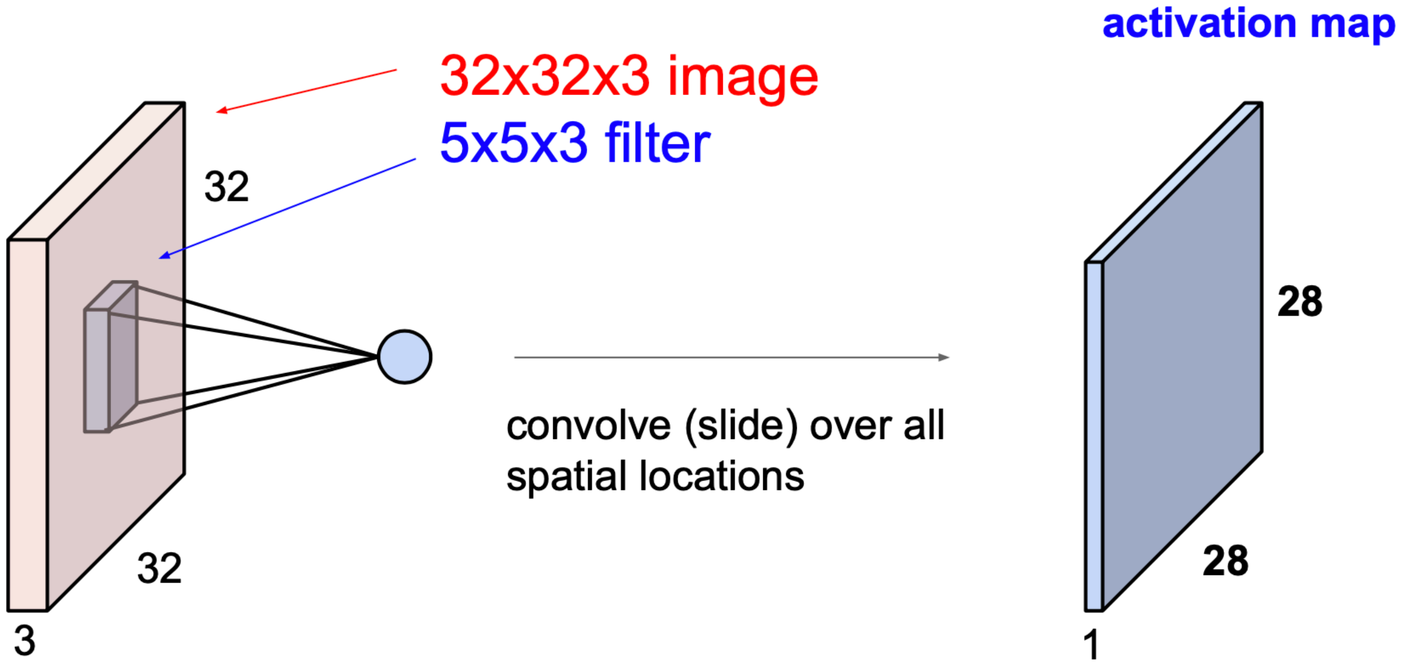
https://cs231n.github.io/



https://cs231n.github.io/

CNN stacks conv layers, pooling layers, and fully connected layers together…

Conv Layers

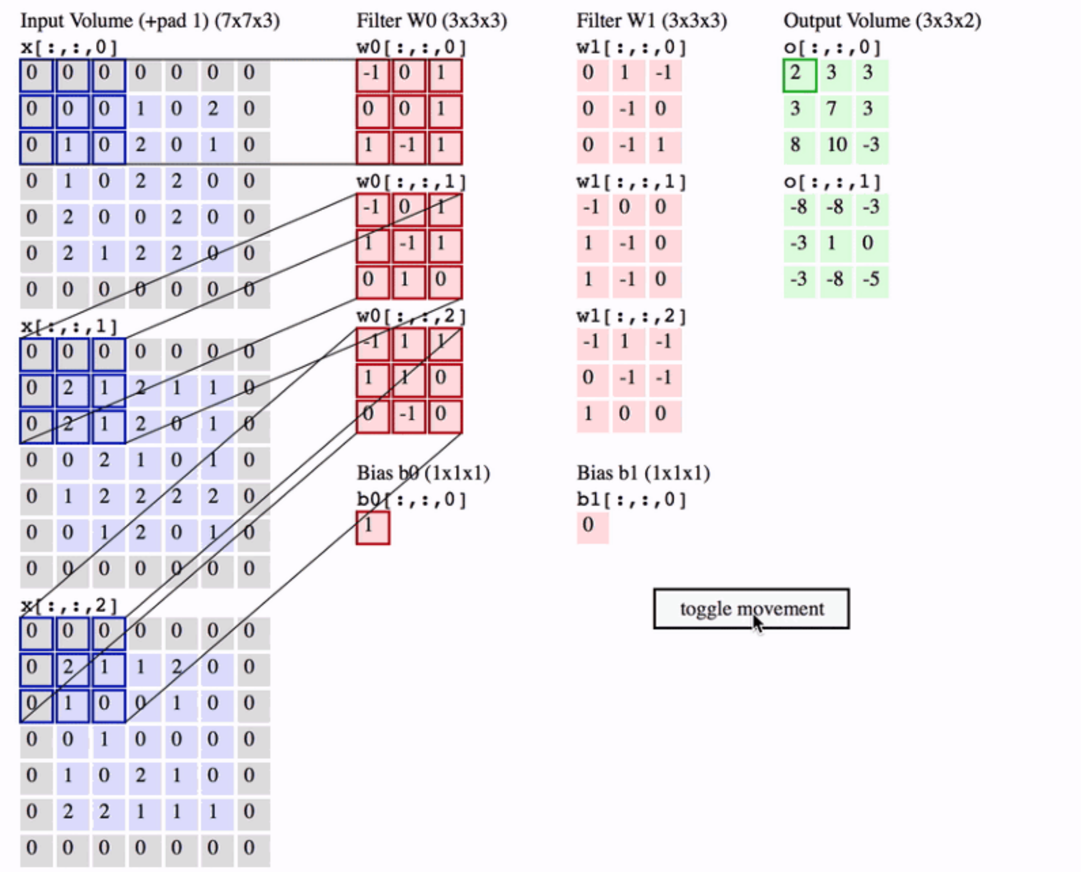


https://cs231n.github.io/

1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		

https://cs231n.github.io/



https://cs231n.github.io/

Convolution layer: summary

Let's assume input is W_1 x H_1 x C
Conv layer needs 4 hyperparameters:

- Number of filters **K**
- The filter size **F**
- The stride **S**
- The zero padding **P**

This will produce an output of W_2 x H_2 x K
where:

- $W_2 = (W_1 - F + 2P)/S + 1$
- $H_2 = (H_1 - F + 2P)/S + 1$

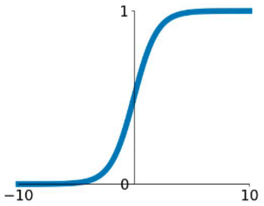
Number of parameters: F^2CK and K biases

https://cs231n.github.io/

Activation Functions

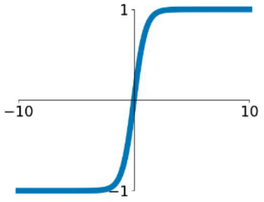
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



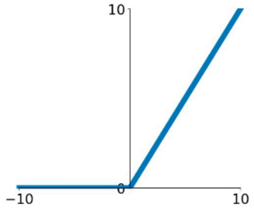
tanh

$\tanh(x)$



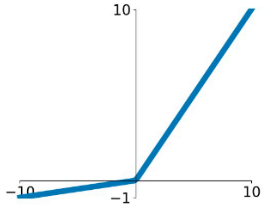
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

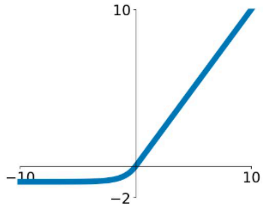


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

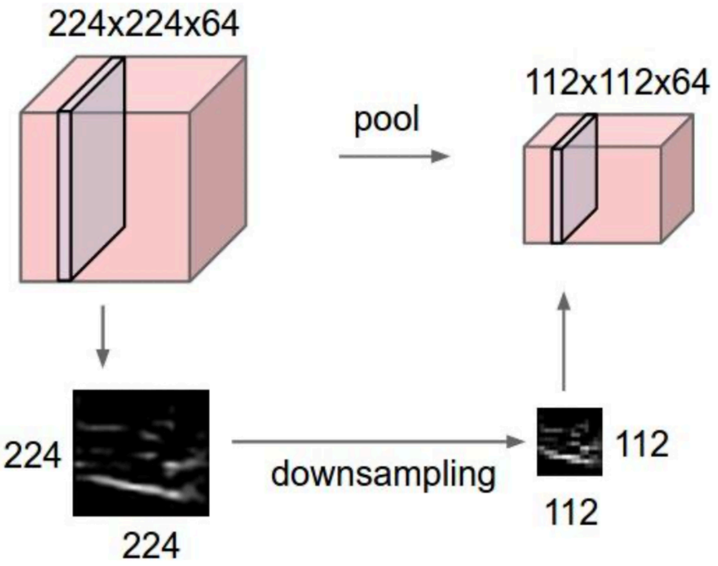
ELU

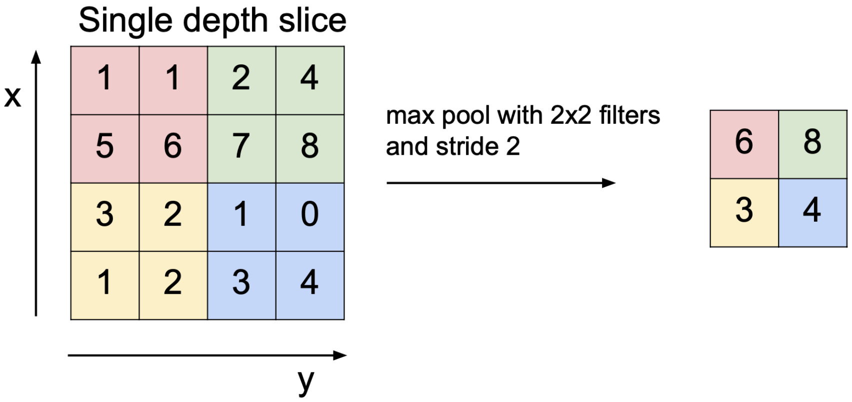
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



https://cs231n.github.io/

Pooling Layer

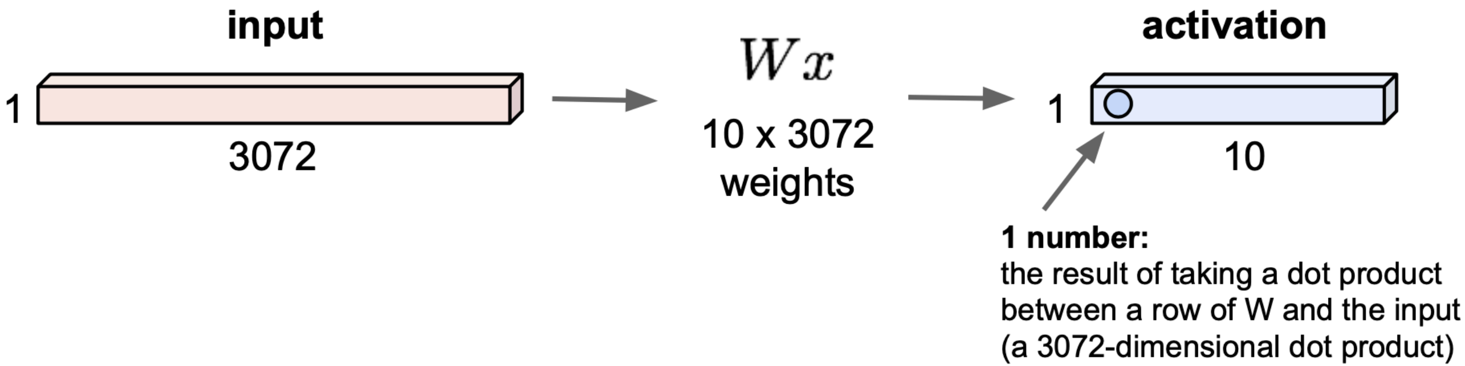




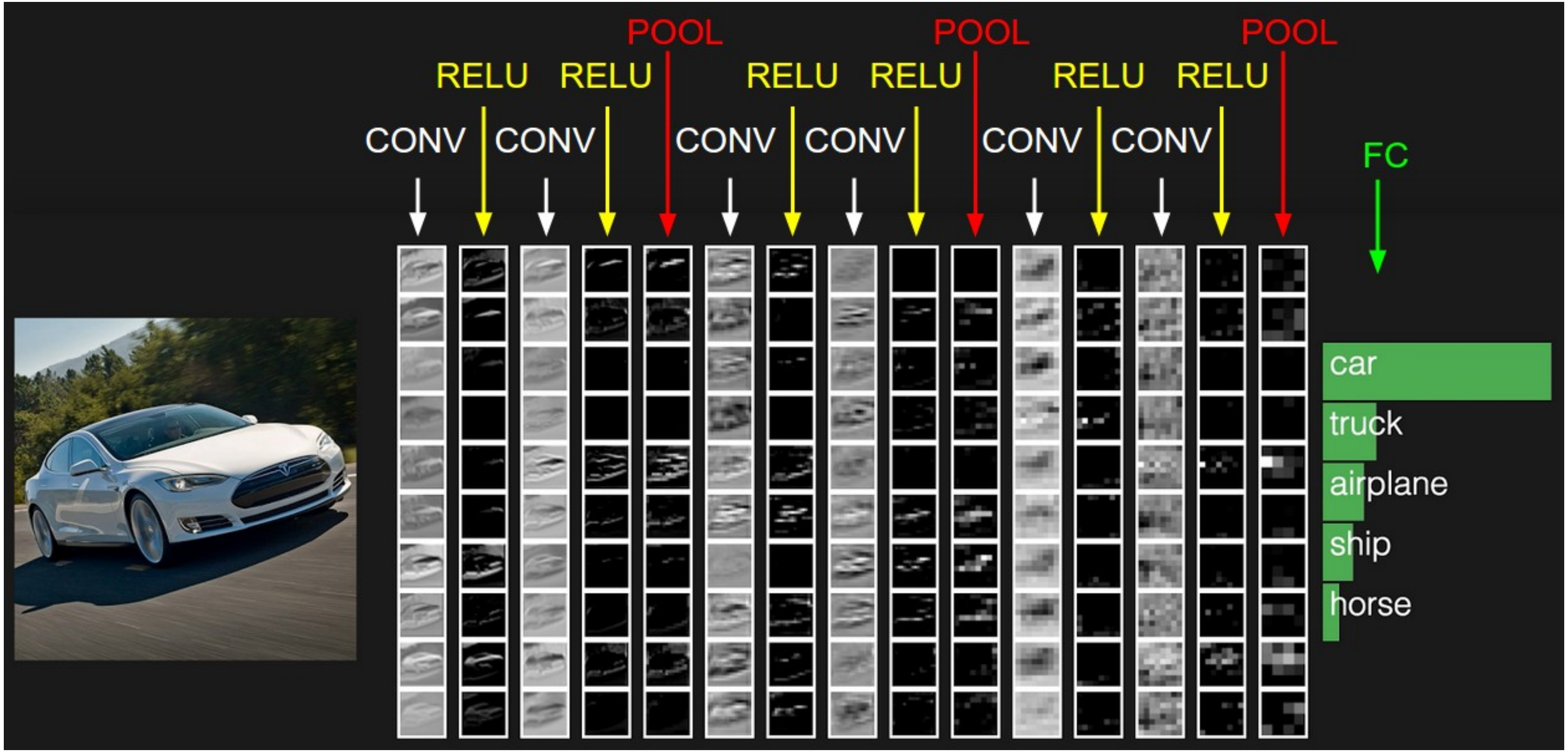
https://cs231n.github.io/

Fully Connected Layer

32x32x3 image -> stretch to 3072 x 1



https://cs231n.github.io/



https://cs231n.github.io/

A zoo of frameworks!

Caffe
(UC Berkeley)

Caffe2
(Facebook)
mostly features absorbed by PyTorch

Torch
(NYU / Facebook)

PyTorch
(Facebook)

Theano
(U Montreal)

TensorFlow
(Google)

PaddlePaddle
(Baidu)

Chainer
(Preferred Networks)
The company has officially migrated its research infrastructure to PyTorch

MXNet
(Amazon)
Developed by U Washington, CMU, MIT, Hong Kong U, etc but main framework of choice at AWS

CNTK
(Microsoft)

JAX
(Google)

And others...

https://cs231n.github.io/

For more details on CNN architecture

https://cs231n.github.io/

https://cs231n.github.io/convolutional-networks/

http://cs231n.stanford.edu/slides/2020/lecture_9.pdf

Thank you!

Yongjun Zhang, Ph.D
Assistant Professor
Dept of Sociology and
Institute for Advanced Computational Science
Stony Brook University
Yongjun.Zhang@stonybrook.edu
https:yongjunzhang.com