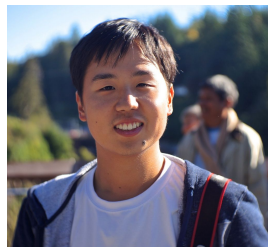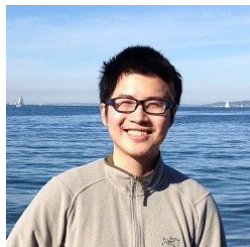# Lecture 1: Introduction to Deep Learning
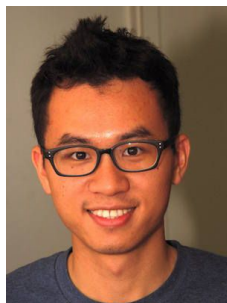
CSE599G1: Spring 2017

People

Lecturers

Teaching
Assistant

What's this course

- **Not** about Learning aspect of Deep Learning (except for the first two)

- System aspect of deep learning: faster training, efficient serving, lower memory consumption.

- Doing something that we have never done before.

Logistics

- Location/Date: Tue/Thu 1pm - 2:20pm MGH 254
- Join slack: https://uw-cse.slack.com dlsys channel

- May 3rd, 5th (Wed, Friday): Joint session with CSE 548
  - Computer Architectures for Deep Learning

- We may use other locations for invited speakers.

- Compute Resources: AWS Education, instruction sent via email.

- Office hour by appointment

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

Homeworks and Projects

- Two code assignments

- Group project
  - Two to three person team
  - Poster presentation and write-up

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

A Crash Course on Deep Learning

Elements of Machine Learning

Model

$$x_i = \begin{bmatrix} \text{feature}_0 \\ \text{feature}_1 \\ \dots \\ \text{feaure}_m \end{bmatrix}$$

$$\hat{y}_i = \frac{1}{1 + \exp\left(-w^T x_i\right)}$$

Objective

$$L(w) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \lambda \|w\|^2$$

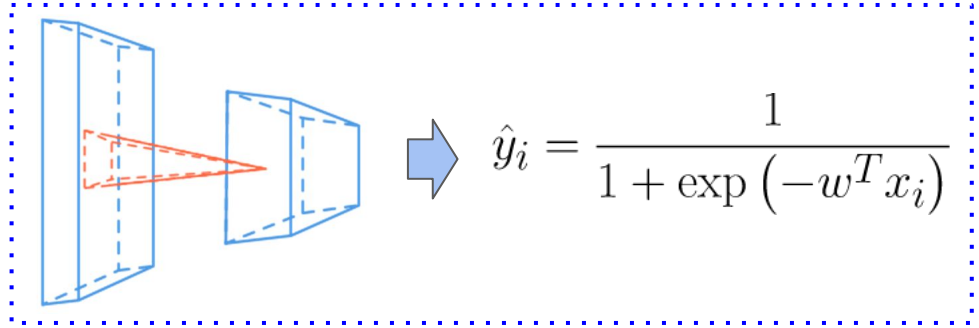**Training**

$$w \leftarrow w - \eta \nabla_w L(w)$$

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

What's Special About Deep Learning

Compositional Model

layer1 exractor

layer2 extractor

predictor

$$\hat{y}_i = \frac{1}{1 + \exp\left(-w^T x_i\right)}$$
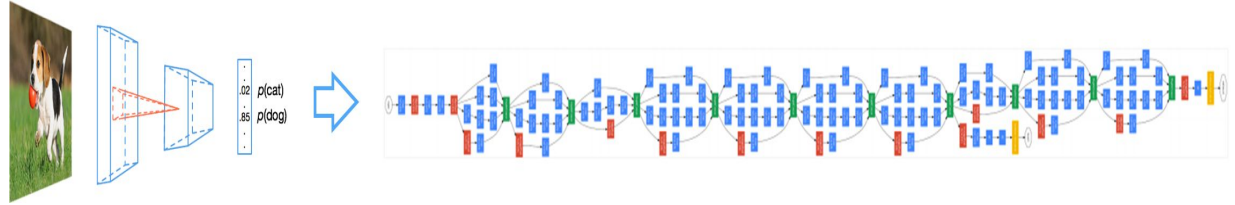
End to End Training

Ingredients in Deep Learning

- Model and architecture

- Objective function, training techniques
  - Which feedback should we use to guide the algorithm?
  - Supervised, RL, adversarial training.

- Regularization, initialization (coupled with modeling)
  - Dropout, Xavier

- Get enough amount of data

Major Architectures

**Image Modeling
Convolutional Nets**

**Language/Speech
Recurrent Nets**

output

state

input

hello   world   !
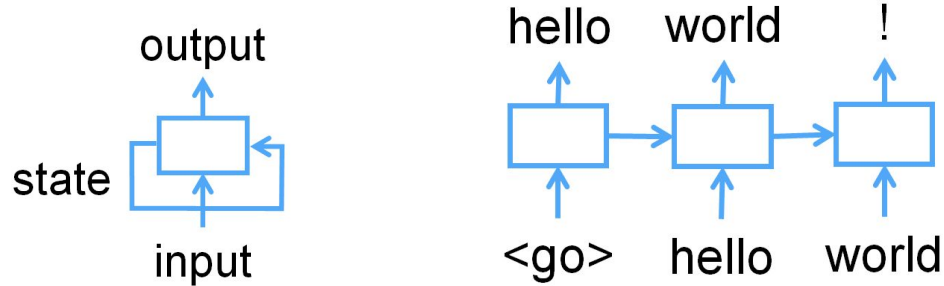
<go>   hello   world

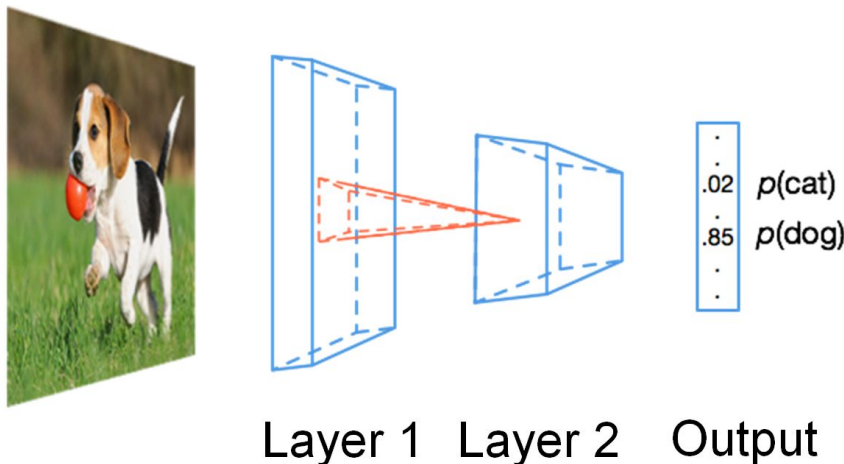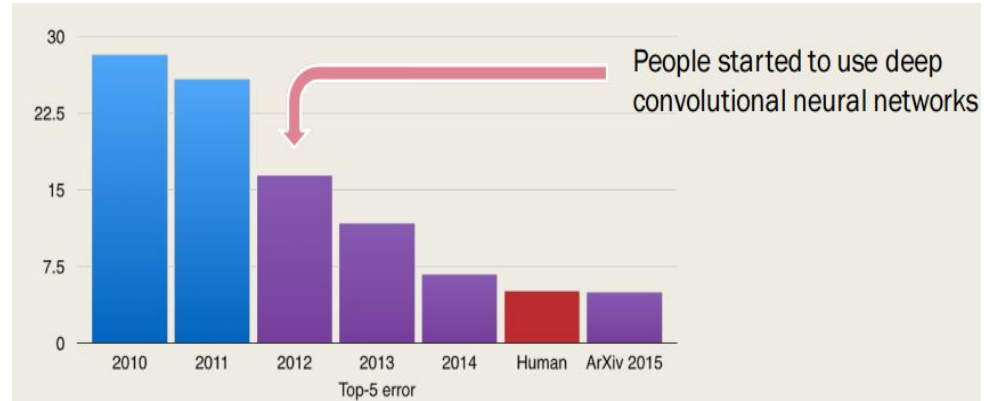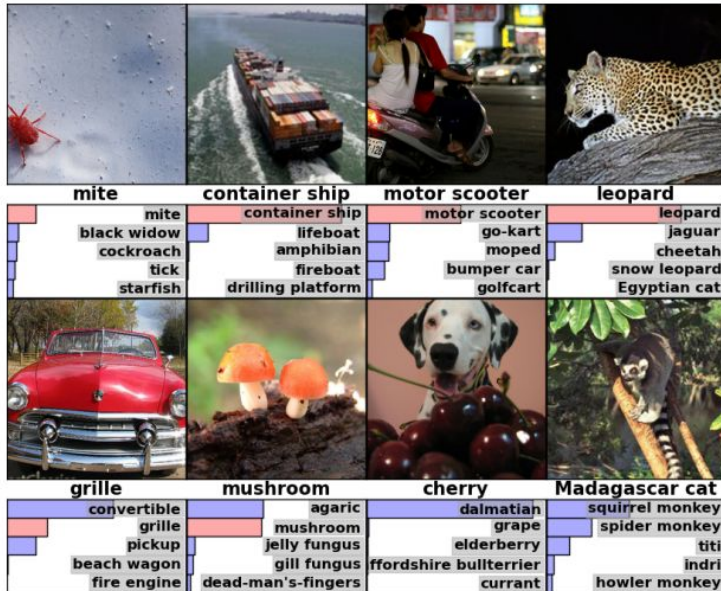# Image Modeling and Convolutional Nets

explore spatial information with convolution layers

Breakthrough of Image Classification

Evolution of ConvNets

- LeNet (LeCun, 1998)
  - Basic structures: convolution, max-pooling, softmax
- Alexnet (Krizhevsky et.al 2012)
  - ReLU, Dropout
- GoogLeNet (Szegedy et.al. 2014)
  - Multi-independent pass way (Sparse weight matrix)
- Inception BN (Ioffe et.al 2015)
  - Batch normalization
- Residual net (He et.al 2015)
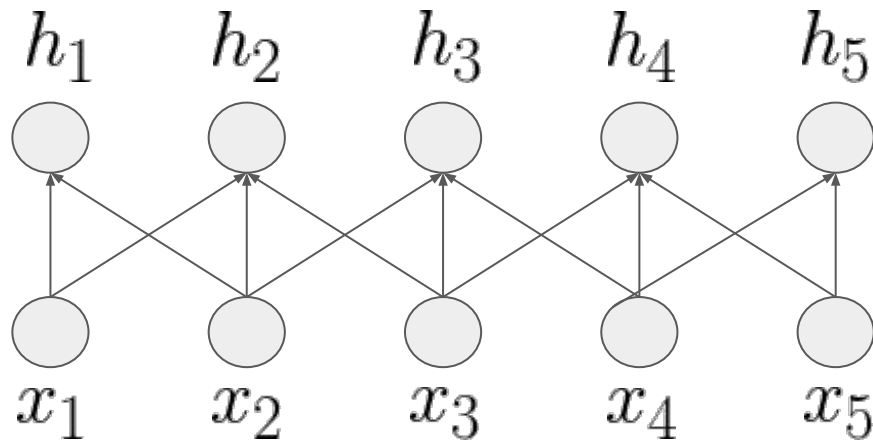 - Residual pass way

Fully Connected Layer

Output h_1 h_2 h_3 h_4 h_5

Input x_1 x_2 x_3 x_4 x_5
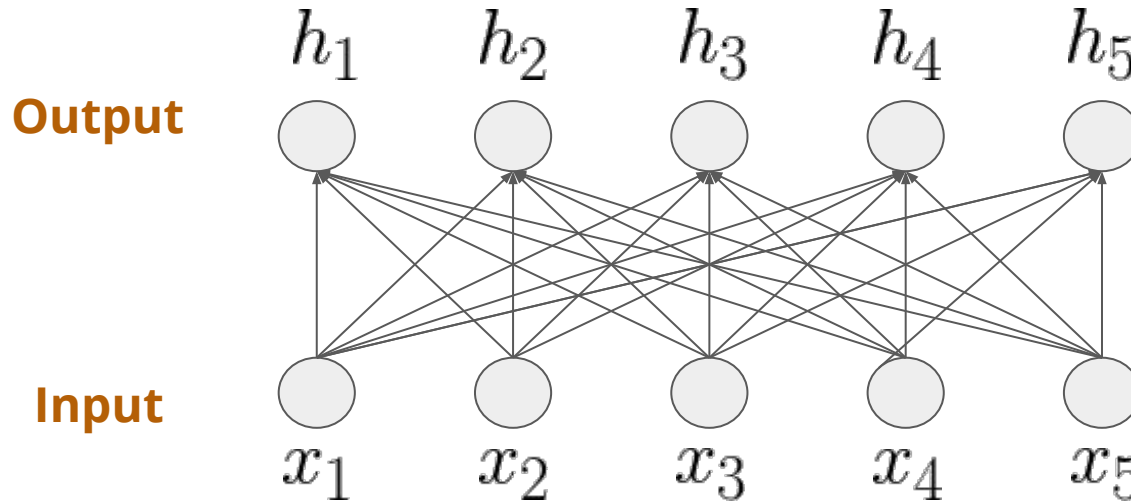
$$h_i = \sum_{j=1}^{5} W_{ij} x_i$$

$$h_1 = W_{11}x_1 + W_{21}x_2 + W_{31}x_3 + W_{41}x_4 + W_{51}x_5$$

Convolution = Spatial Locality + Sharing

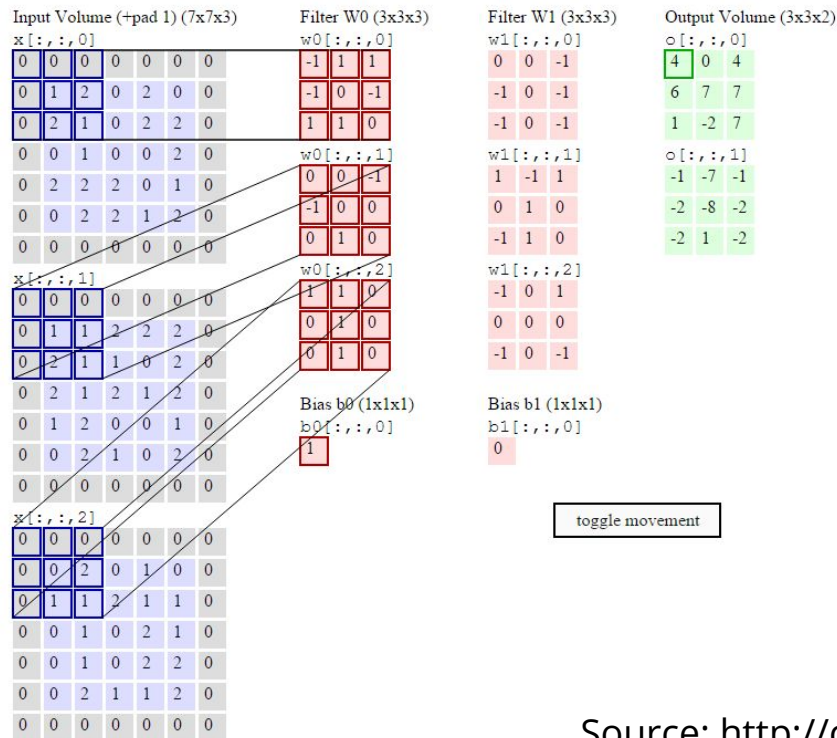$$h_1 \quad h_2 \quad h_3 \quad h_4 \quad h_5$$

Spatial Locality

$$x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5$$

Without Sharing $\quad h_i = W_{1,i}x_{i-1} + W_{2,i}x_i + W_{3,i}x_{i+1}$

With Sharing $\quad h_i = W_1 x_{i-1} + W_2 x_i + W_3 x_{i+1}$

Convolution with Multiple Channels

Source: http://cs231n.github.io/convolutional-networks/

Pooling Layer

Can be replaced by strided convolution

Source: http://cs231n.github.io/convolutional-networks/
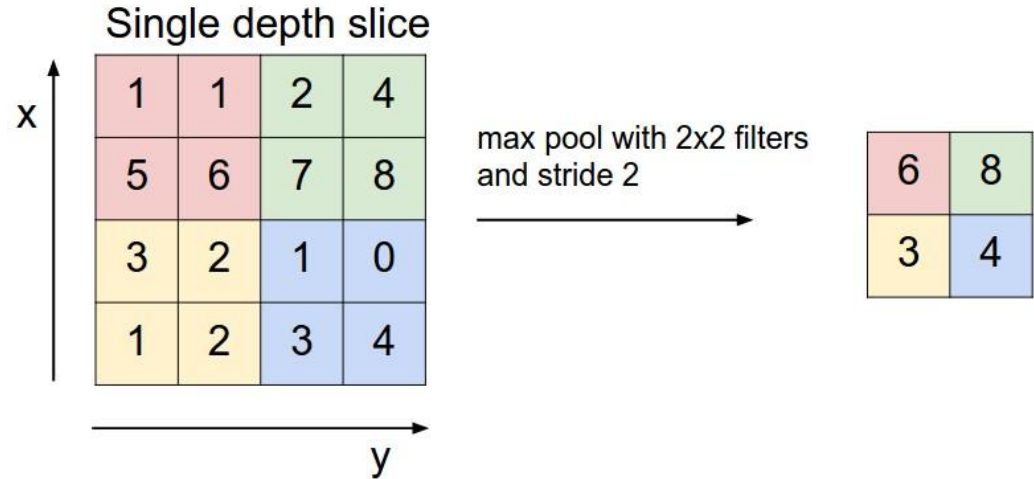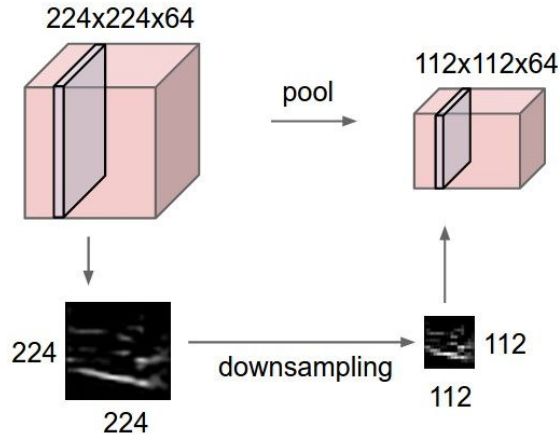
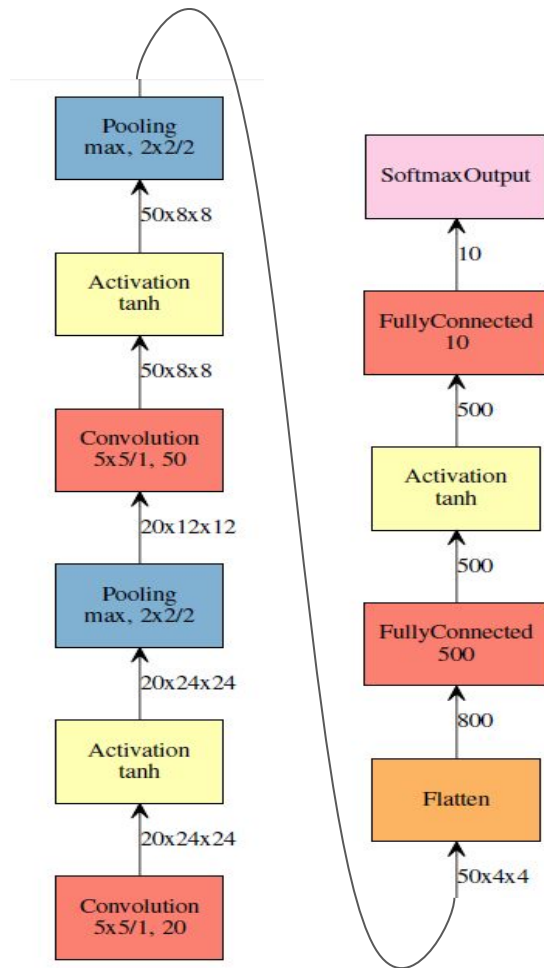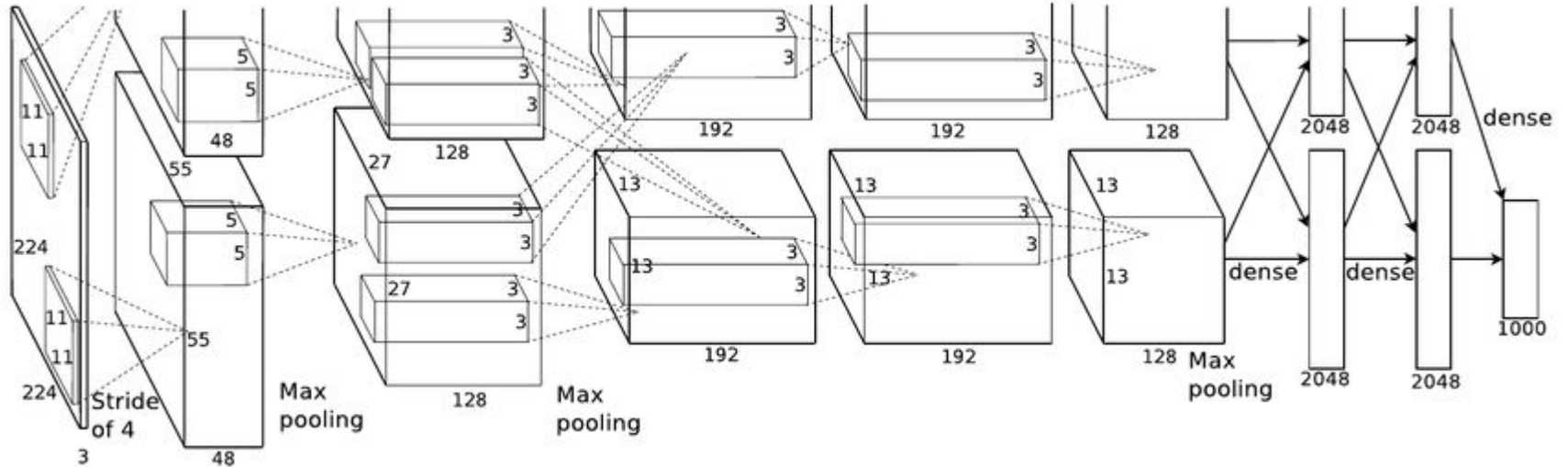# LeNet (LeCun 1998)

- Convolution
- Pooling
- Flatten
- Fully connected
- Softmax output

$$p(i) = \frac{e^{\frac{f(i)}{T}}}{\sum_j e^{\frac{f(j)}{T}}}$$

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

AlexNet (Krizhevsky et.al 2012)

# Challenges: From LeNet to AlexNet

- Need much more data: ImageNet
- A lot more computation burdens: GPU

- Overfitting prevention
  - Dropout regularization

- Stable initialization and training
  - Explosive/vanishing gradient problems
  - Requires careful tuning of initialization and data normalization
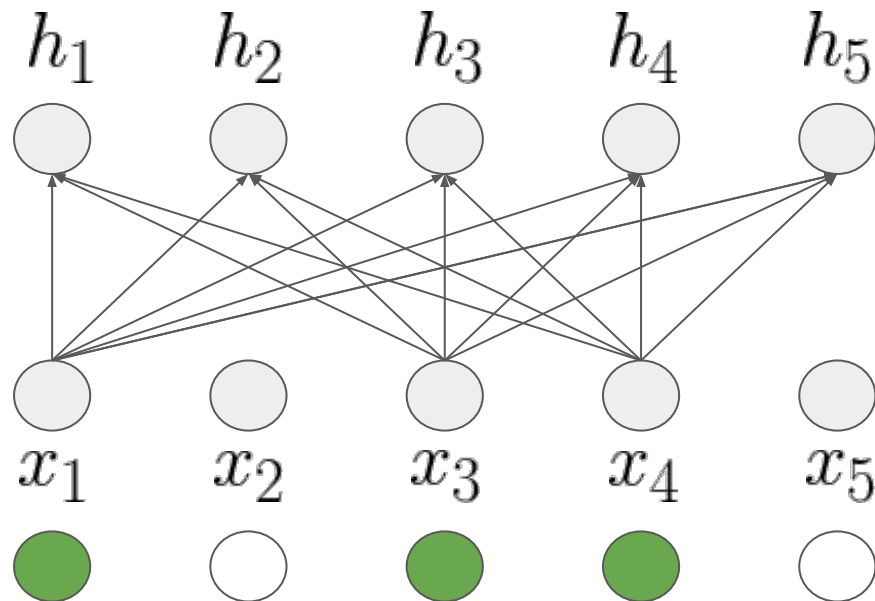
ReLU Unit

- ReLU $\quad y = max(x, 0)$

- Why ReLU?
  - Cheap to compute
  - It is roughly linear..
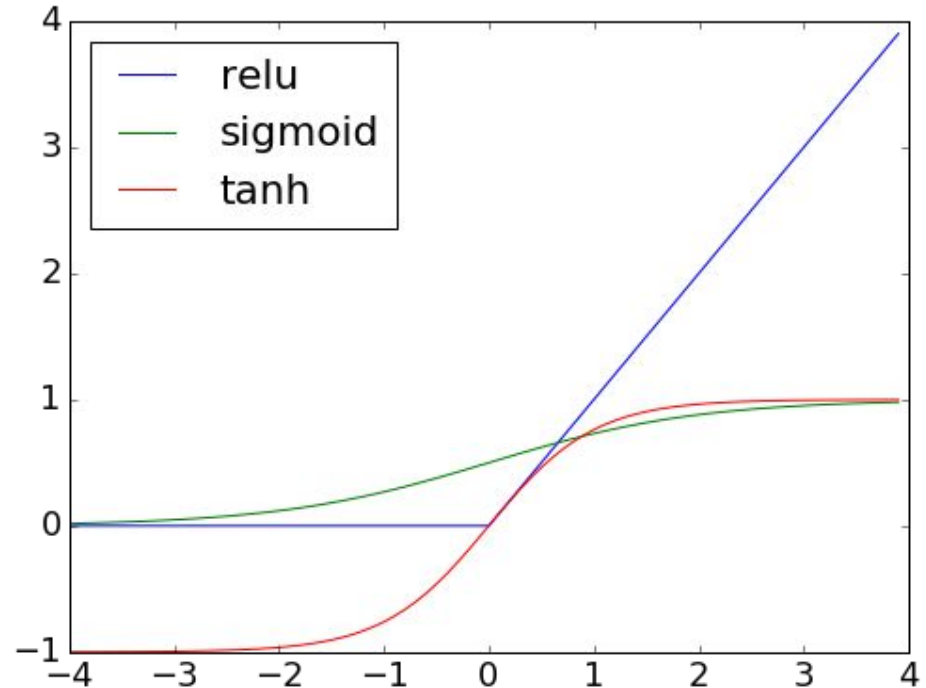
Dropout Regularization

- Randomly zero out neurons with probability 0.5

- During prediction, use expectation value (keep all neurons but scale output by 0.5)

Dropout Mask

Dropout Regularization

- Randomly zero out neurons with probability 0.5

- During prediction, use expectation value (keep all neurons but scale output by 0.5)

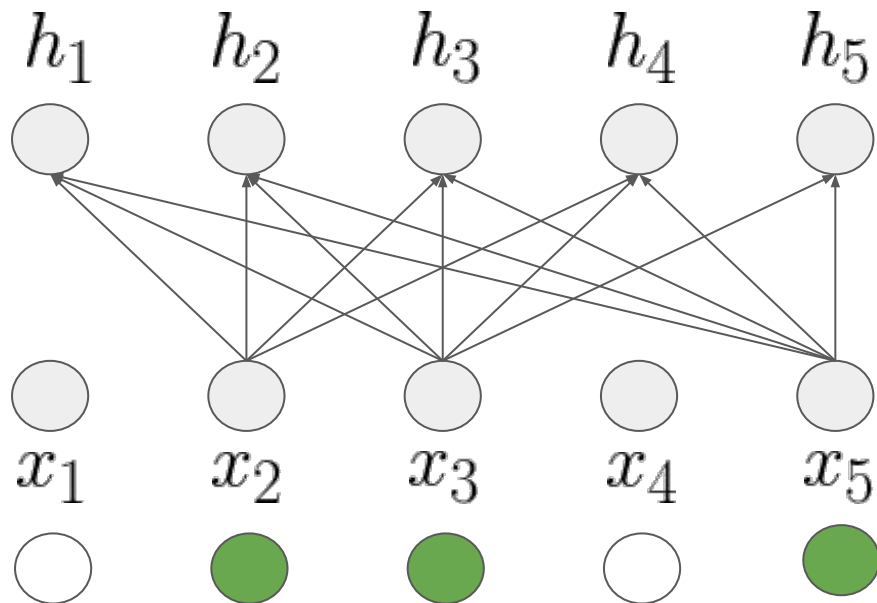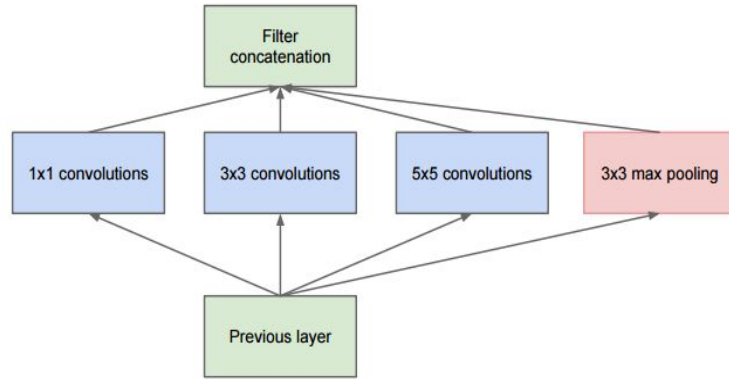h_1 h_2 h_3 h_4 h_5

x_1 x_2 x_3 x_4 x_5

Dropout Mask

GoogleNet: Multiple Pathways, Less Parameters

(a) Inception module, naïve version
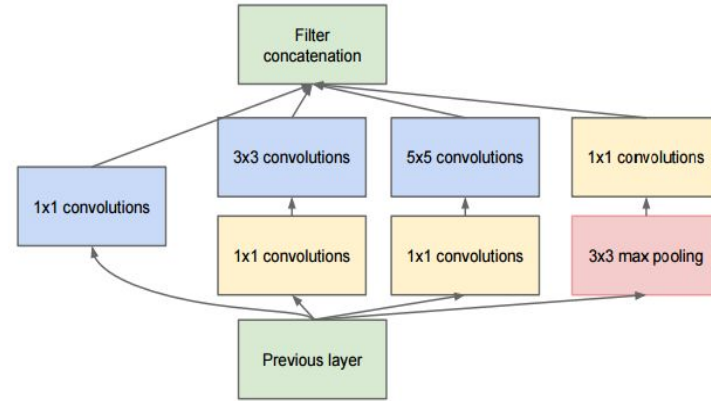
(b) Inception module with dimension reductions

Figure 2: Inception module

Vanishing and Explosive Value Problem

- Imagine each layer multiplies
  Its input by same weight matrix
  - W > 1: exponential explosion
  - W < 1: exponential vanishing

$$x_0 \qquad x_1 = W x_0 \qquad x_k = W^k x_0$$

- In ConvNets, the weight are not tied, but
  their magnitude matters
  - Deep nets training **was** initialization sensitive

Batch Normalization: Stabilize the Magnitude

- Subtract mean
- Divide by standard deviation
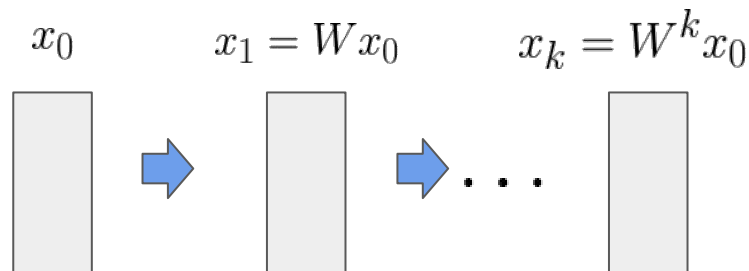- Output is invariant to input scale!
  - Scale input by a constant
  - Output of BN remains the same

- Impact
  - Easy to tune learning rate
  - Less sensitive initialization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$; Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \text{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

(Ioffe et.al 2015)

The Scale Normalization (Assumes zero mean)
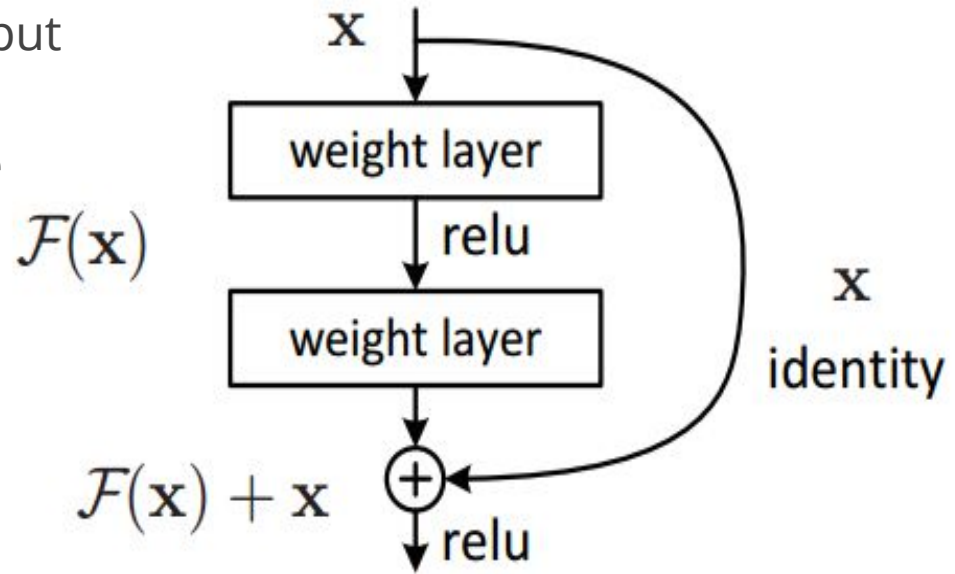
Scale Normalization

$$BN(x)_i = \frac{x_i}{\sqrt{\sum_{j=1}^{m} x_j^2}}$$

Invariance to Magnitude!

$$BN(\alpha x)_i = \frac{\alpha x_i}{\sqrt{\sum_{j=1}^{m} (\alpha x_j)^2}} = BN(x)_i$$

Residual Net (He et.al 2015)

- Instead of doing transformation add transformation result to input

- Partly solve vanishing/explosive value problem

$\mathbf{x}$

weight layer

$\mathcal{F}(\mathbf{x})$ relu

weight layer

$\mathbf{x}$ identity

$\mathcal{F}(\mathbf{x}) + \mathbf{x}$ $\oplus$

relu

Evolution of ConvNets

- LeNet (LeCun, 1998)
    - Basic structures: convolution, max-pooling, softmax
- Alexnet (Krizhevsky et.al 2012)
    - ReLU, Dropout
- GoogLeNet (Szegedy et.al. 2014)
    - Multi-independent pass way (Sparse weight matrix)
- Inception BN (Ioffe et.al 2015)
    - Batch normalization
- Residual net (He et.al 2015)
    - Residual pass way

More Resources

- Deep learning book (Goodfellow et. al)

- Stanford CS231n: Convolutional Neural Networks for Visual Recognition

- Recurrent nets: to be continued next lecture..