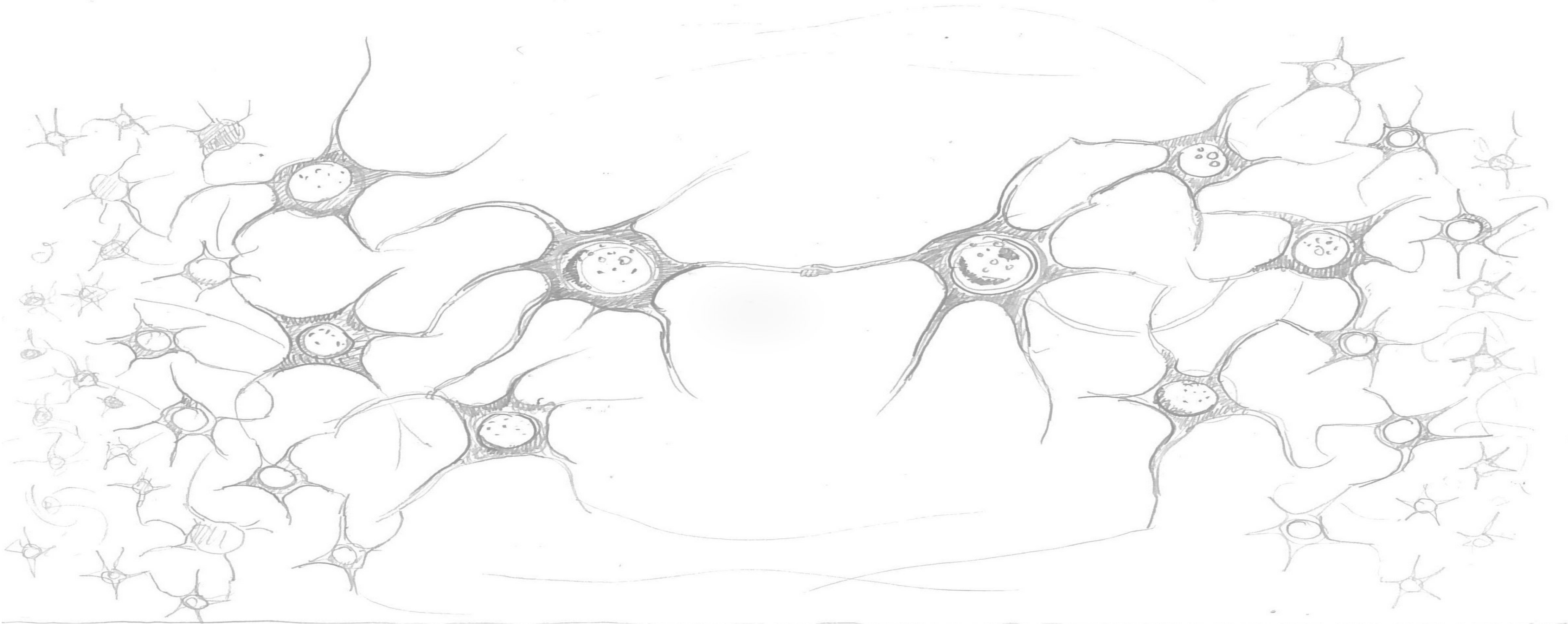
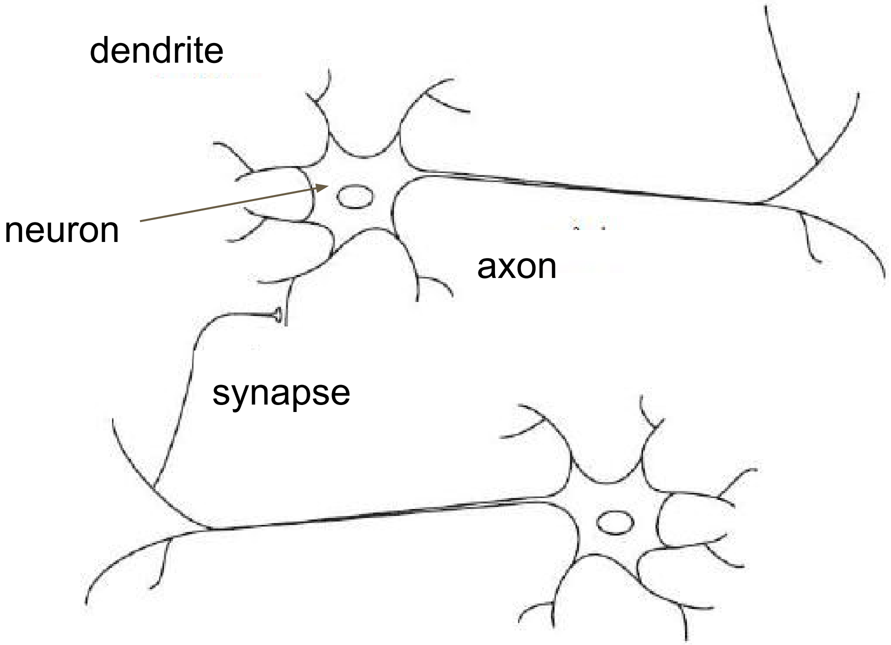
Navigating the Neural Net Terrain



The Brain Analogy

(our cartoon neuron)

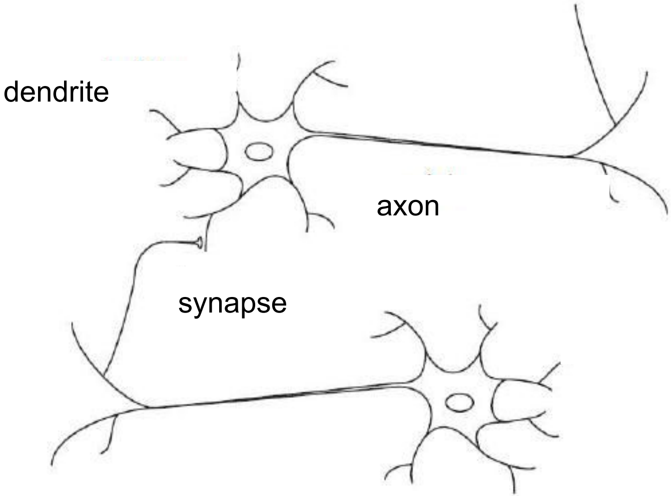


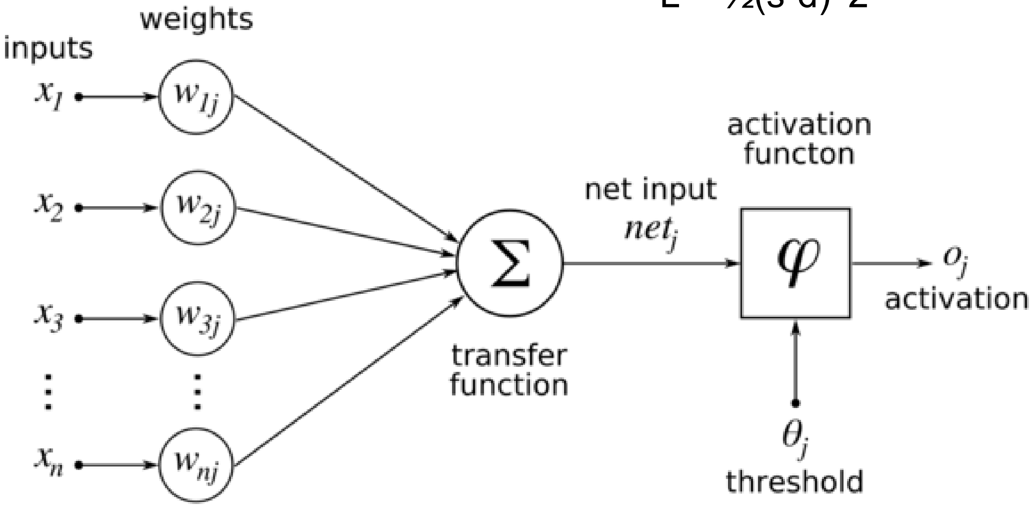
The Brain Analogy

(cartoon neuron & mathematical neuron)

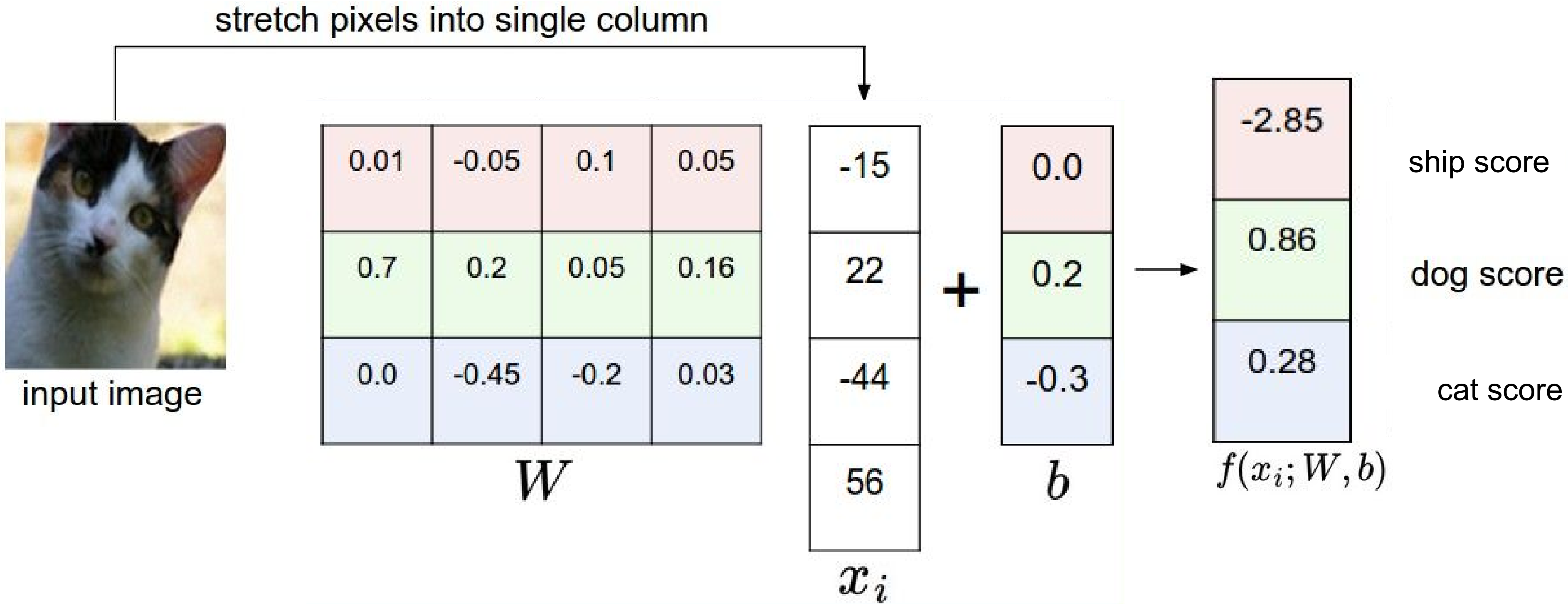
$s = f(x,w)$

$L = \frac{1}{2}(s-d)^2$

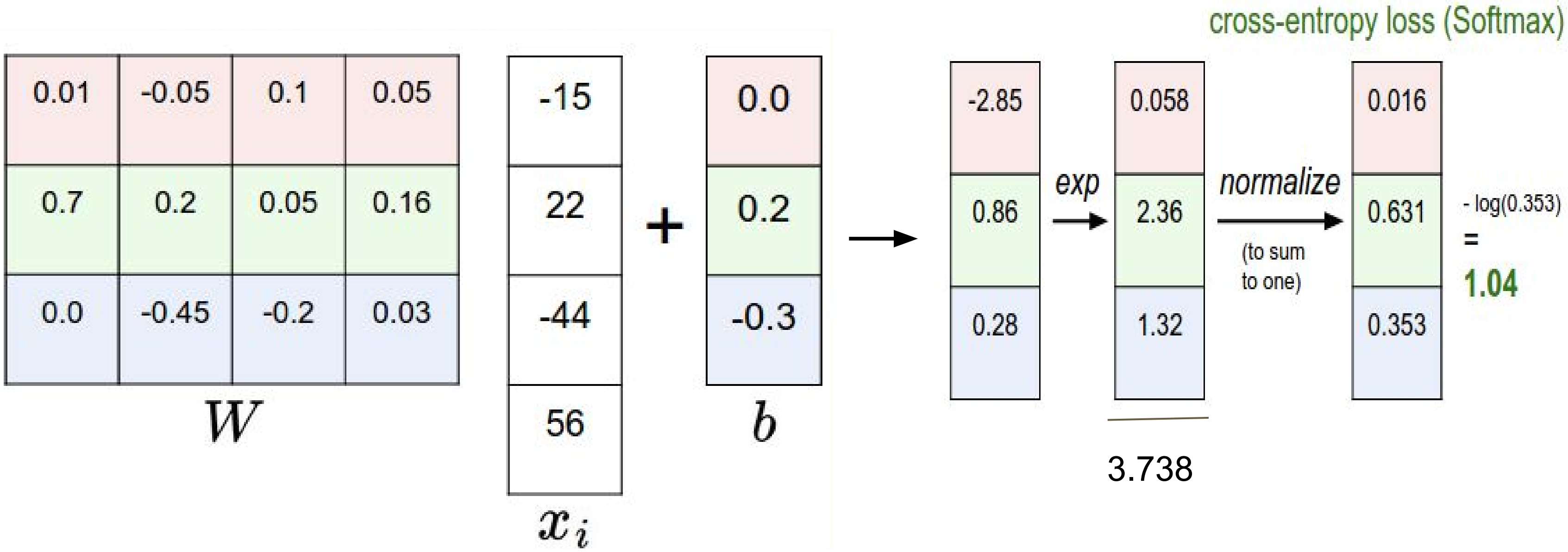




The Linear Classifier Analogy



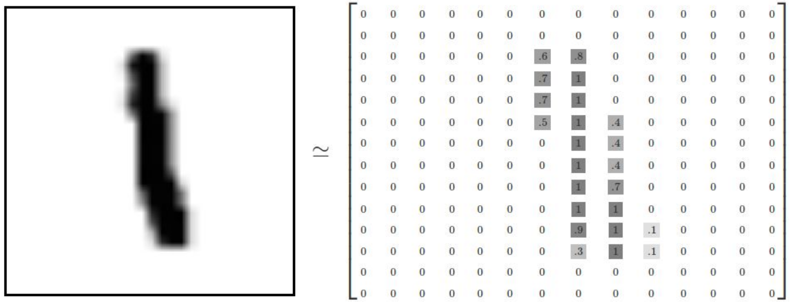
Losses:
Softmax (Cross-Entropy) Loss



Softmax: $f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$

Cross-Entropy $L_i = -\log\left(\frac{e^{z_j}}{\sum_k e^{z_k}} \right)$

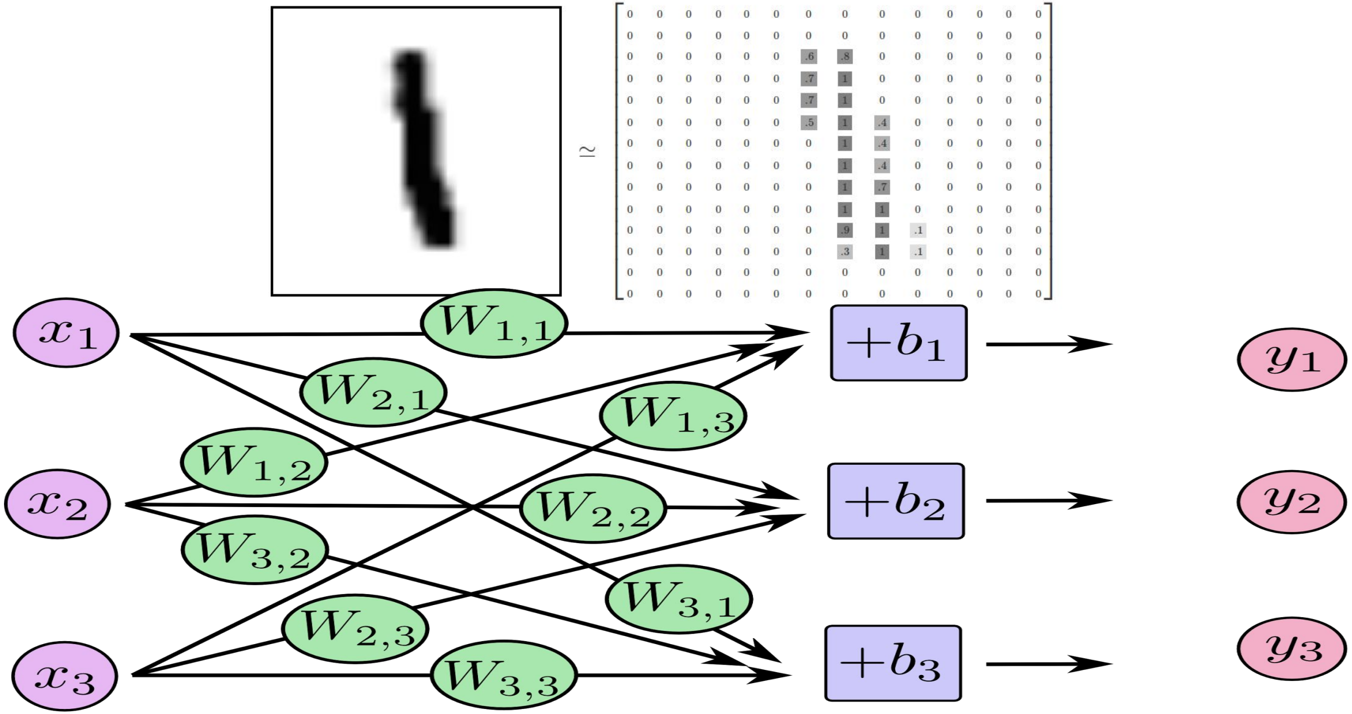
The Linear Classifier Analogy: MNIST data set



$$\begin{pmatrix} W_{1,1}x_1 + W_{1,2}x_2 + W_{1,3}x_3 + b_1 \\ W_{2,1}x_1 + W_{2,2}x_2 + W_{2,3}x_3 + b_2 \\ W_{3,1}x_1 + W_{3,2}x_2 + W_{3,3}x_3 + b_3 \end{pmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

1) Add up evidence of input being in a certain class Evidence = $\sum W_{i,j}x_j + b_i$

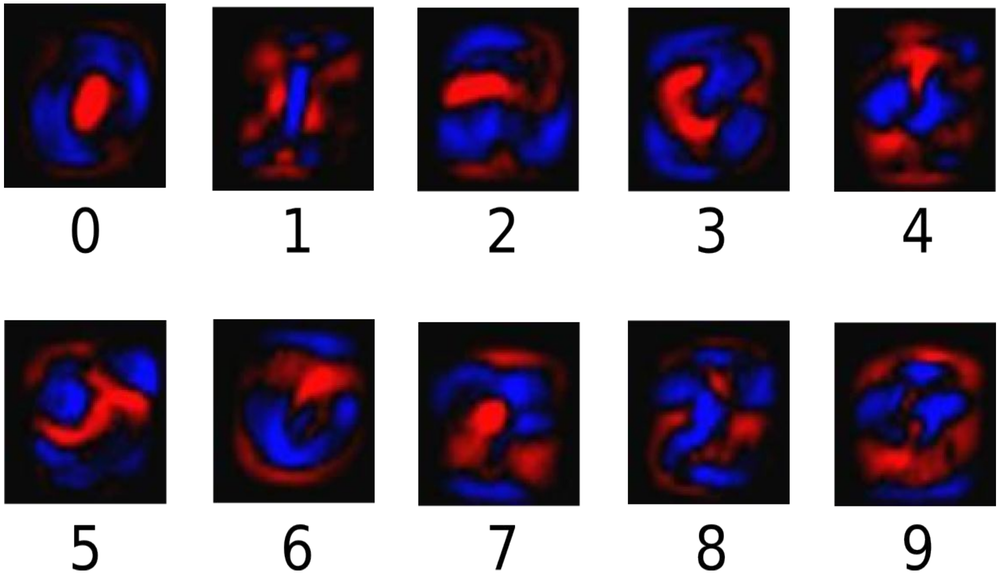
The Linear Classifier Analogy: MNIST data set



1) Add up evidence of input being in a certain class Evidence = $\sum W_{i,j}x_j + b_i$

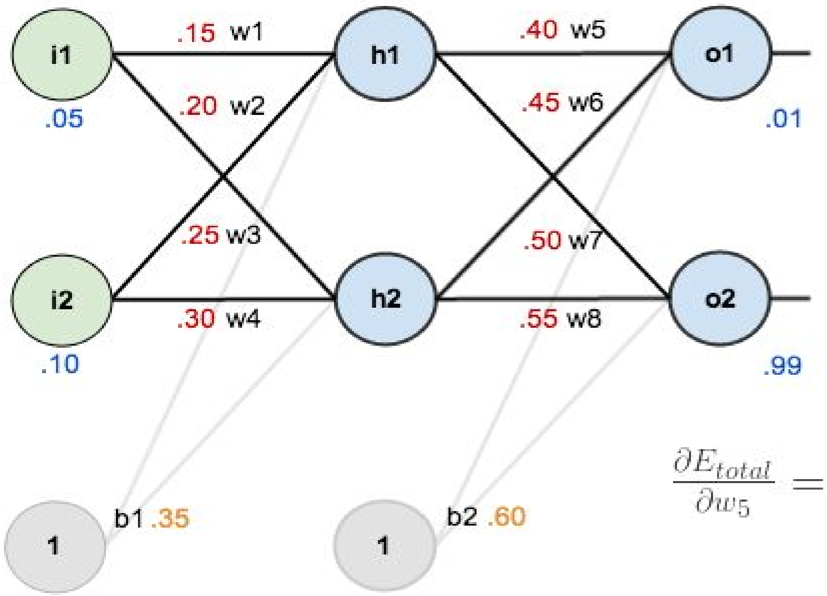
The Linear Classifier Analogy: MNIST data set

$W \simeq$



BackPropagation

How do we get there?

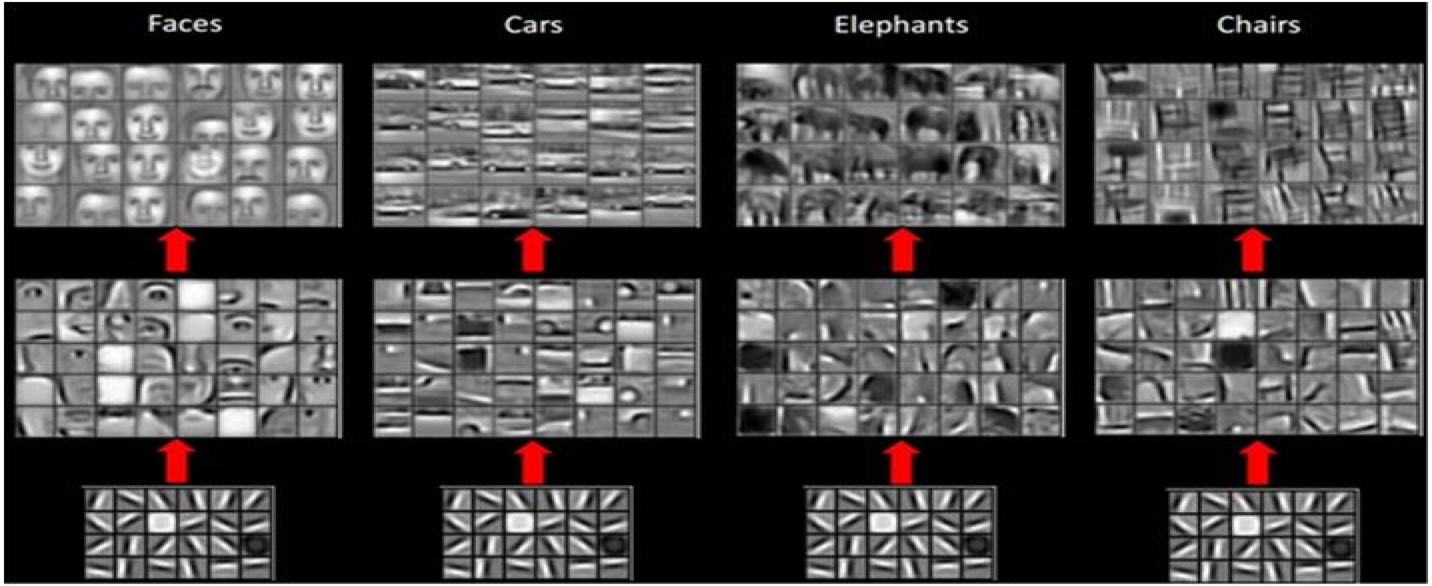


$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

Chain Rule!

Convolutional Neural Nets

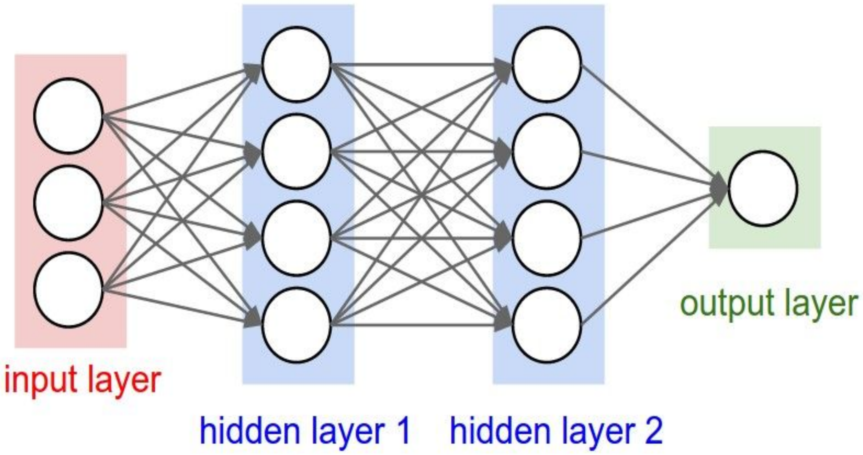
Very similar to Neural Nets.. But how are they different ?

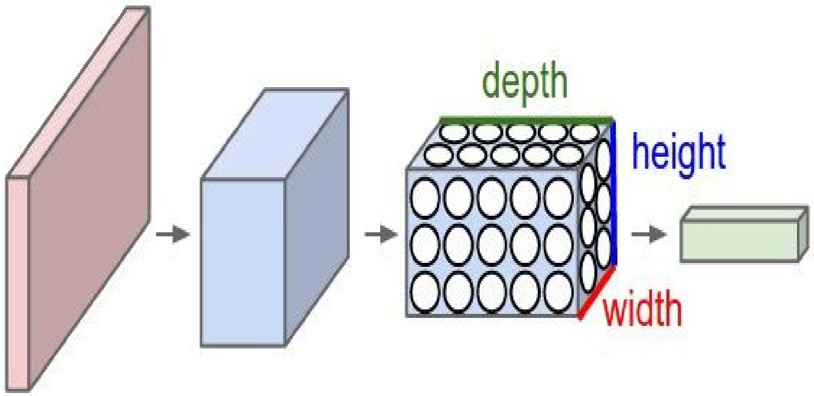


Convolutional Neural Nets

Vs. Neurals Nets

- Input is an image: Leverage 3D Structure
- Fully Connected ?   Not really





The CNN Family

Winners of the ILSVRC ImageNet challenges

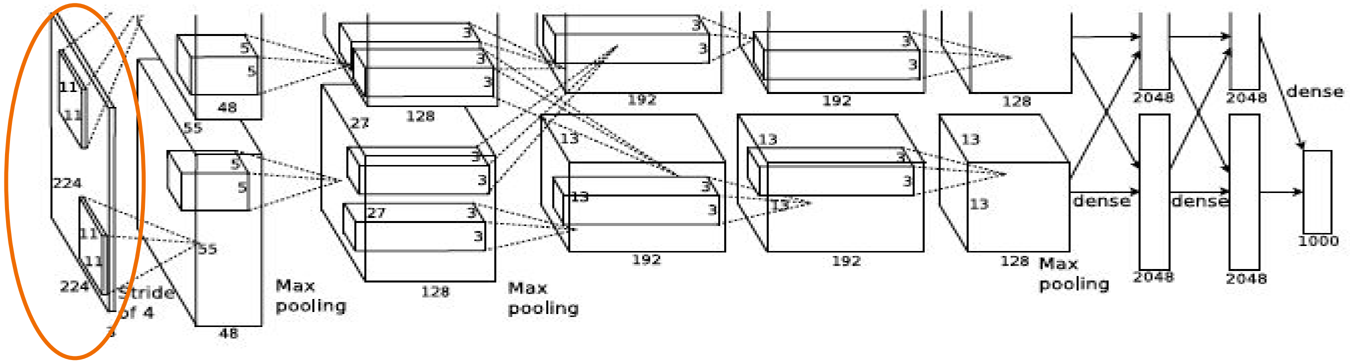
AlexNet (2012, Krizhevsky): Popularized CNNs - 1st to incorporate consecutive convolutional layers

GoogleNet / Inception (2014, Szegedy): Drastically reduced the # of parameters used (from 60 million to 4 million)

ResNet (2015, Kaiming He): Residual Network : famous for skip-connections and heavy use of batch-normalization; also removes some fully connected layers (at end of network)

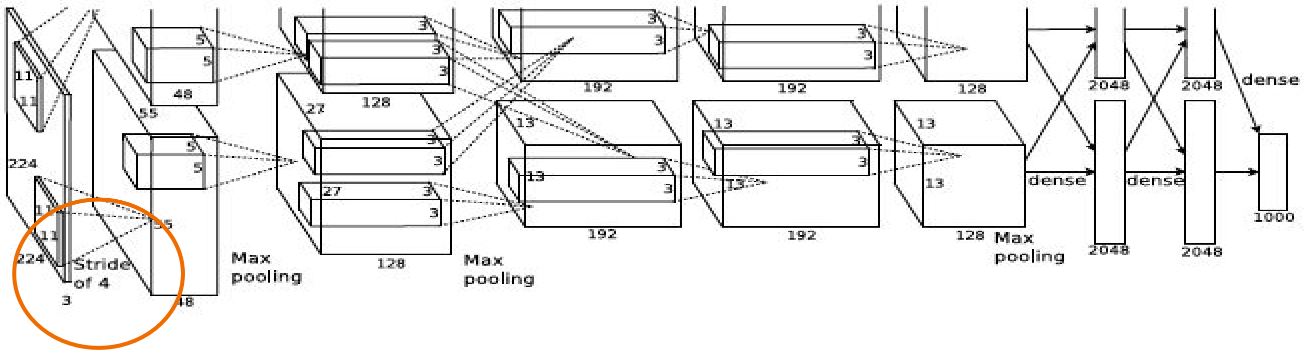
Convolutional Neural Nets : Architecture

1) **Input Layer: Raw pixel values of the image**
   (ex: 224 x 224 x 3 (3 ~ color channels (RGB))
2) Conv Layer
3) Pool Layer
4) ReLU Layer
5) FC (Fully Connected Layer)



Convolutional Neural Nets : Architecture

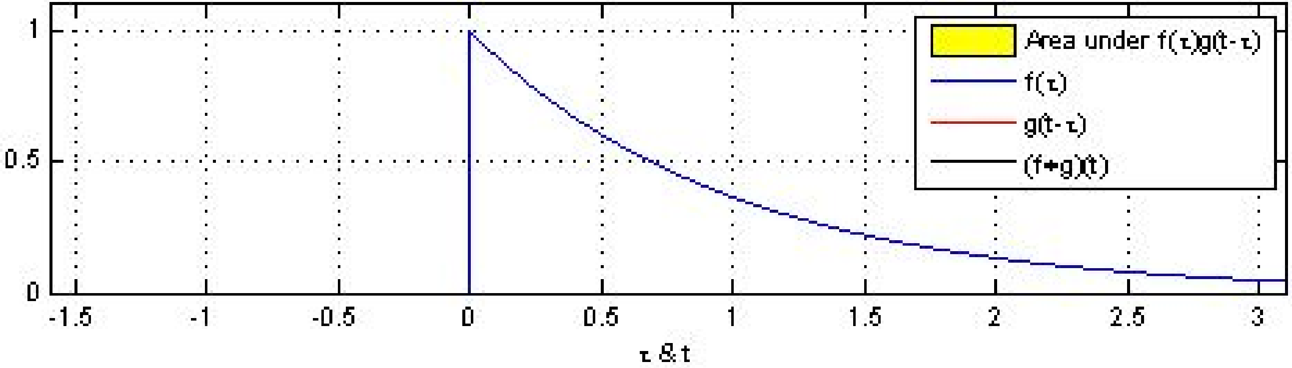
1) Input Layer: Raw pixel values of the image
2) **Conv Layer: Dot product between weights and the small region of input volume (ex: 11 x 11 x 3 filters)**
3) Pool Layer
4) ReLU Layer
5) FC (Fully Connected Layer)



Convolutional Neural Nets : Architecture

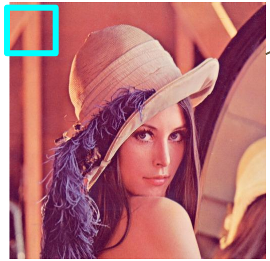
What is a Convolution?

$$f*g = \int f(t - \tau)g(\tau)d\tau$$



Convolutional Neural Nets : Architecture

What is a Convolution?



$$\begin{matrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{matrix} \qquad \begin{matrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{matrix} \longrightarrow 0$$

$$\begin{matrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{matrix} \qquad \begin{matrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{matrix} \longrightarrow -3$$

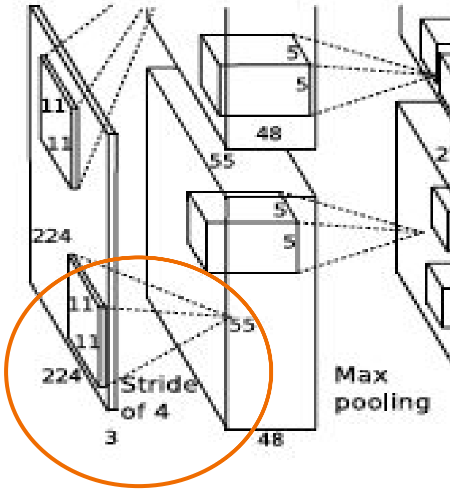
Convolutional Neural Nets : Architecture

What is a Convolution?

Convolutional Layer: (W-F + 2P)/S +1

- **W : Input Volume size**
- **F: Receptive Field size of the Conv Layer Neuron**
- **P: Zero- Padding**
- **S: Stride**

(224 - 11 +2(3))/4 + 1 = 55

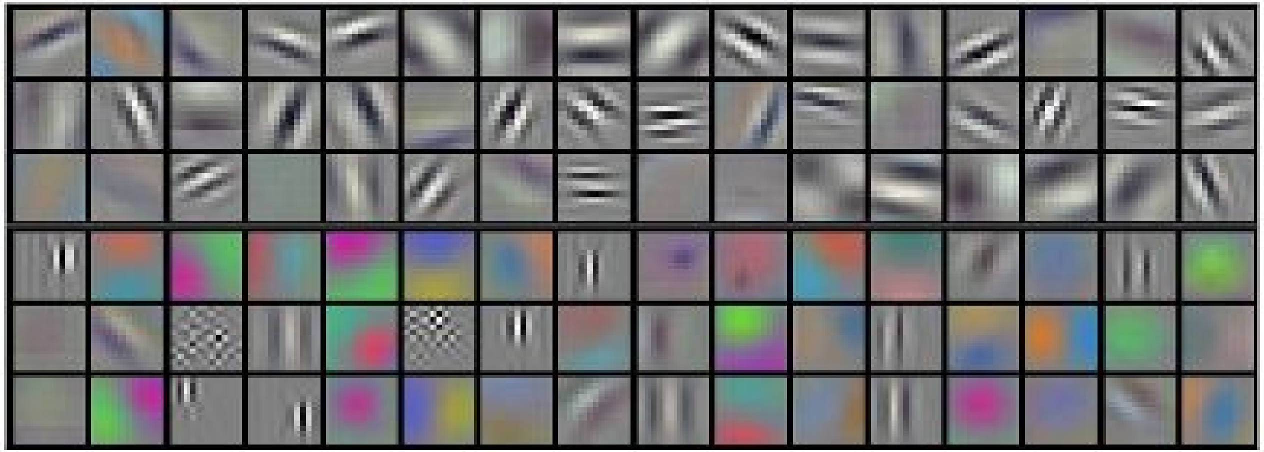


Conv Layer Output ~ 55 x 55 x 96 (ie : 55^2 neurons in each layer)

Convolutional Neural Nets : Architecture

What is a Convolution?

Voila. We have 96 filters.



Convolutional Neural Nets : Architecture

1) Input Layer
2) Conv Layer
3) **Pooling Layer: Performs downsampling operation**
4) ReLU Layer
5) FC (Fully Connected Layer)

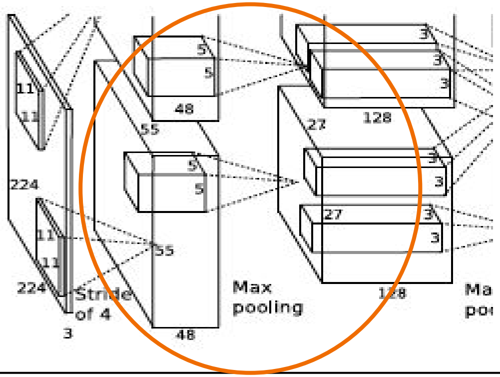
12	**23**	**9**
8	**11**	**13**
22	**5**	**10**

$\rightarrow \boxed{23}$

Our Eqn : O = (W- F)/S +1

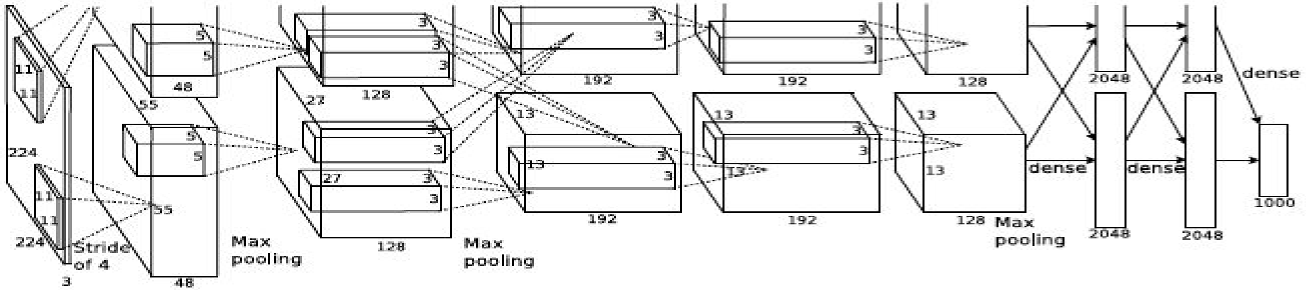
AlexNet: use 3 x 3 MaxPooling w/ stride = 2

O =(55-3)/2 + 1 = 27



Convolutional Neural Nets : Architecture

1) Input Layer: Raw pixel values of the image
2) Conv Layer:
3) Pool Layer:
4) **ReLU Layer: Apply an elementwise activation function (ex : max(0,x) thresholding output dimension ~ same as input)**
5) FC (Fully Connected Layer)

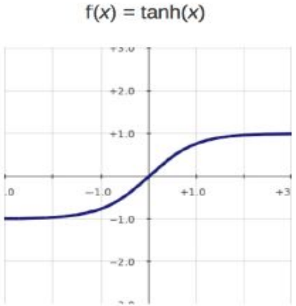


*The ReLU non-linearity is applied to the output of every convolutional and fully-connected layer.

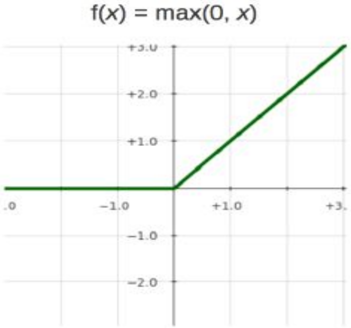
Convolutional Neural Nets : Architecture

ReLU Layer:

Tradionally:
f(x) = tanh(x) or fx= (1+e-x)^-1   (Very slow to train)

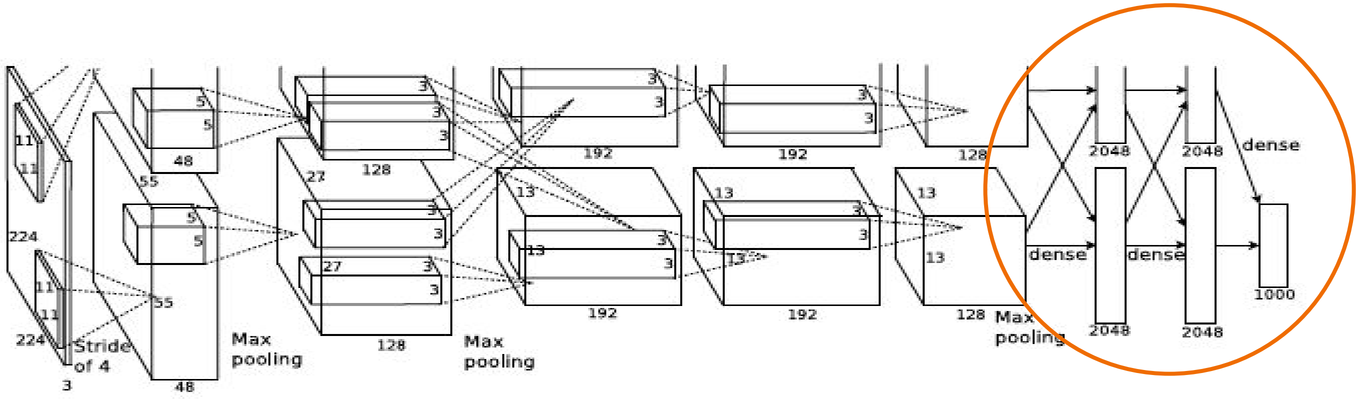


Now:
f(x) = max(0,x)   (Faster to train)



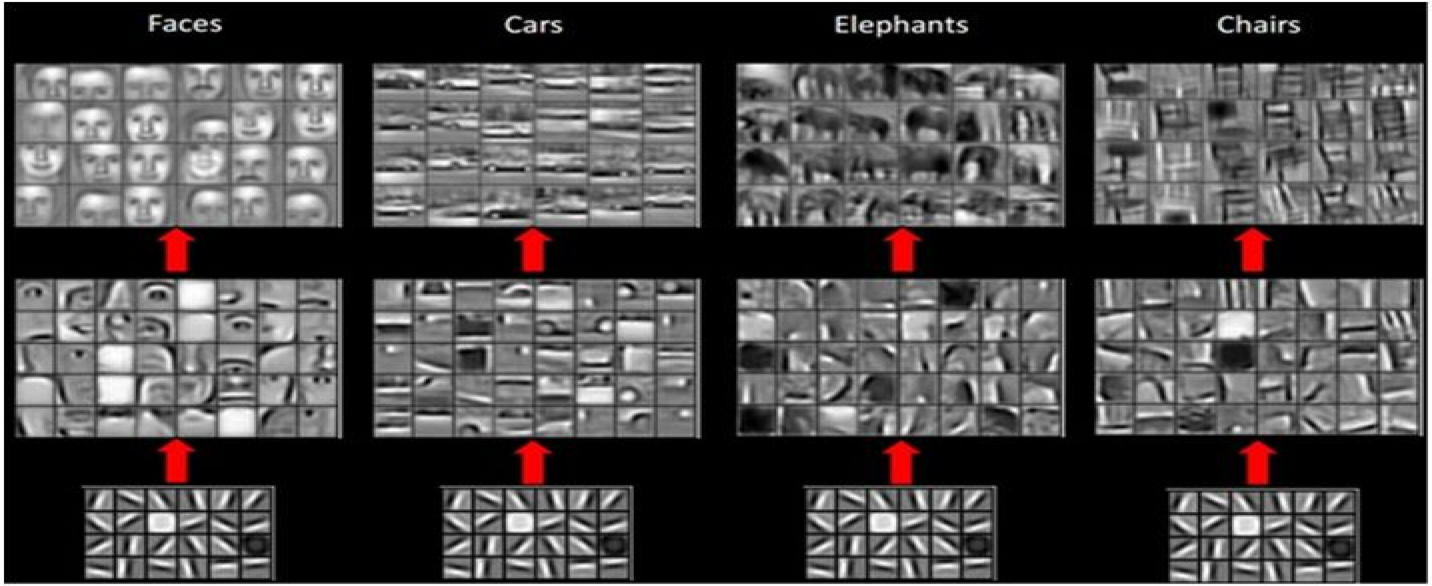
Convolutional Neural Nets : Architecture

1) Input Layer: Raw pixel values of the image
2) Conv Layer:
3) Pool Layer:
4) ReLU Layer:
5) **FC (Fully Connected) Layer : Each neuron will be connected to all activations of the previous volume. The output layer will compute class scores (ex: [1 x 1 x 1000])**



Convolutional Neural Nets : Architecture

\

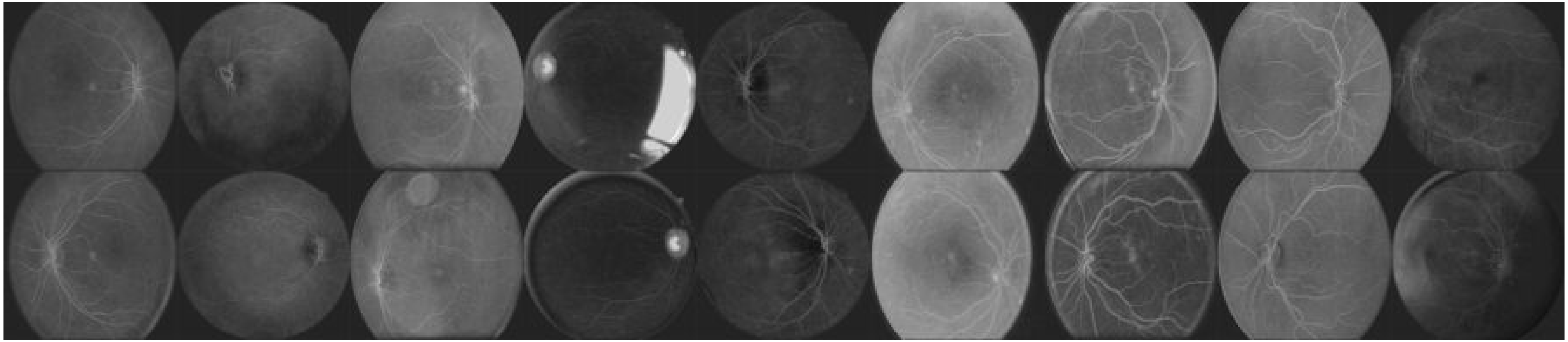


Real Life example #1:

Train Retinopathy data via AlexNet

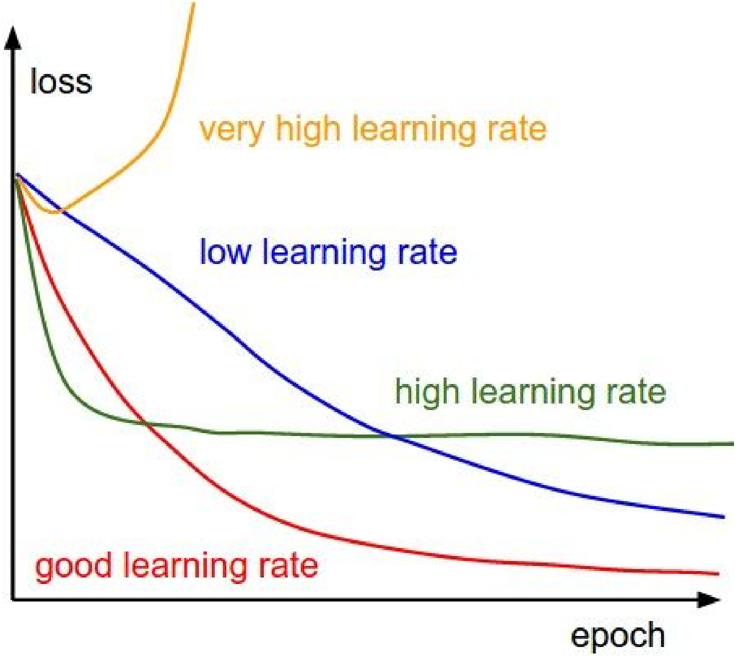
Kaggle Dataset:

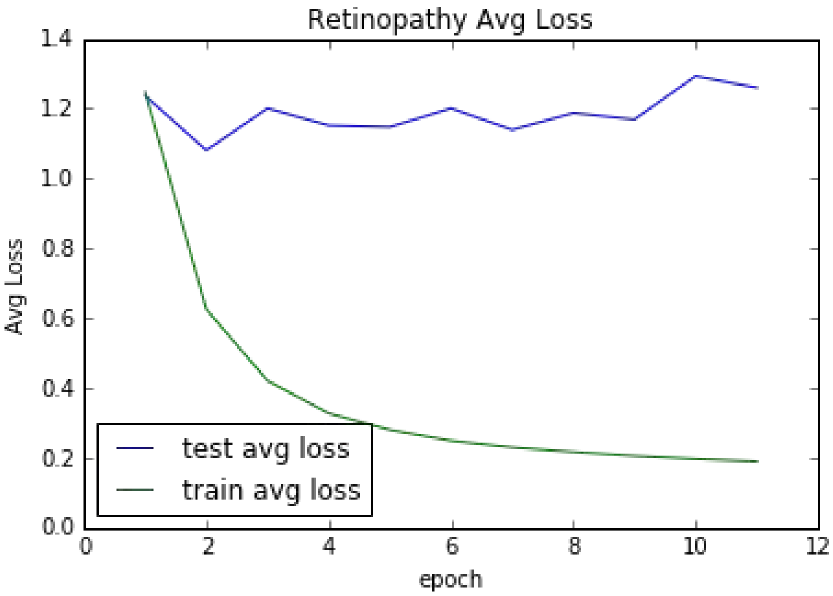
Diabetic Retinopathy Detection



Learning from the Learning Process

1) Loss functions

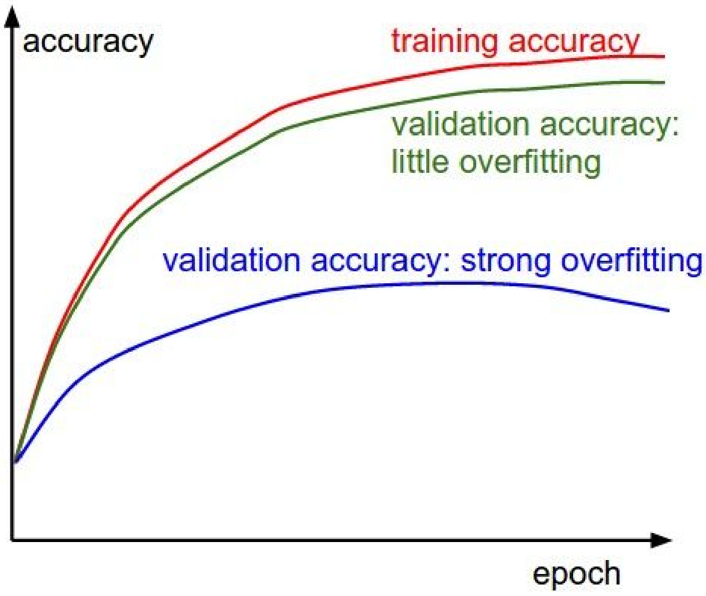


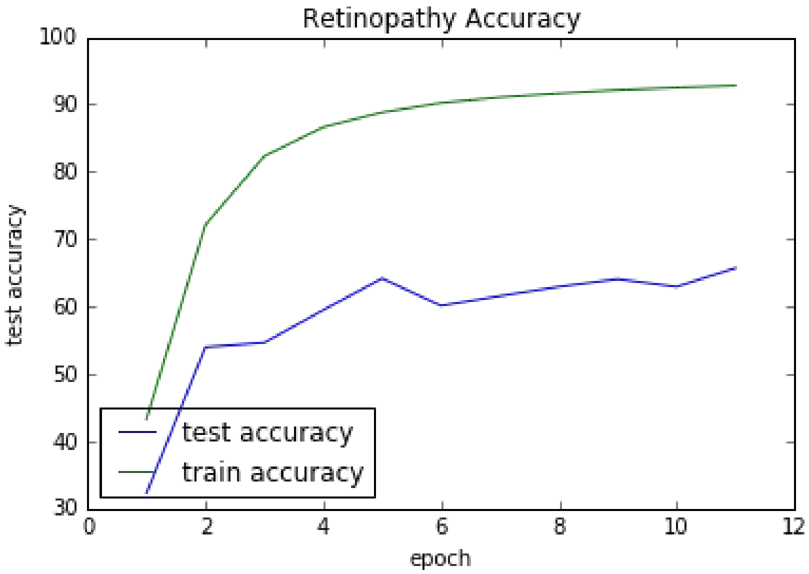


* Tip: Change the learning rate!

Learning from the Learning Process

2) Accuracy





* Tip: Increase L2 weight penalty , Increase Drop-Out, More Data (possibly with jitter) - -try batch norm ?

Learning from the Learning Process:

3) Weight Ratios

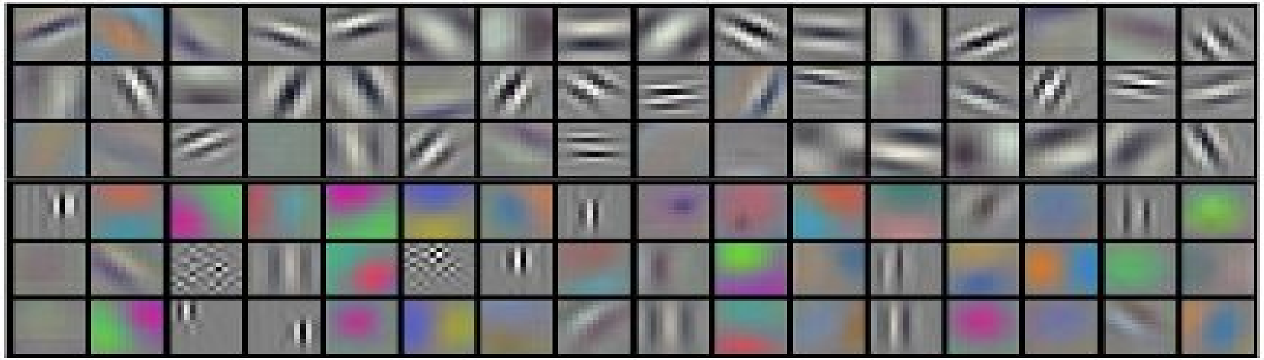
- update / weight ratio : should be roughly about 1e-3

(larger ratio~ learning rate may be too high, lower ratio~ learning rate may be too low)

4) First-layer Visualizations

Visualized weights from the 1st layer of the network:

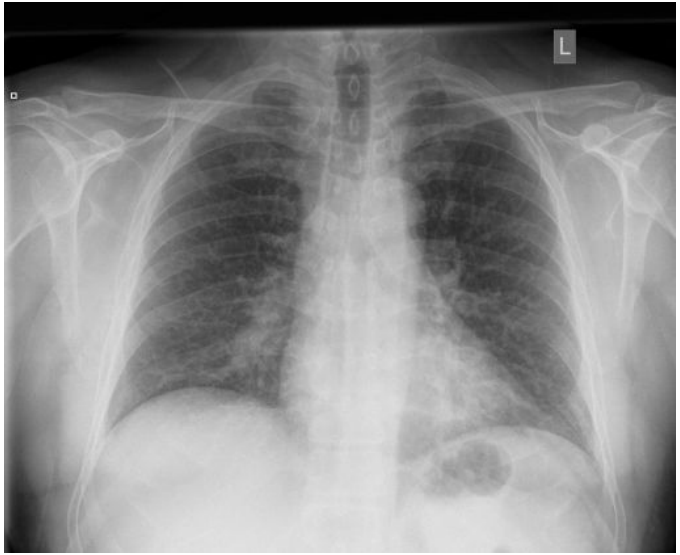
(smooth, diverse features indicate that training is going well)



Real Life example #2:

Can I leverage the Inception model to decipher different lung diseases?

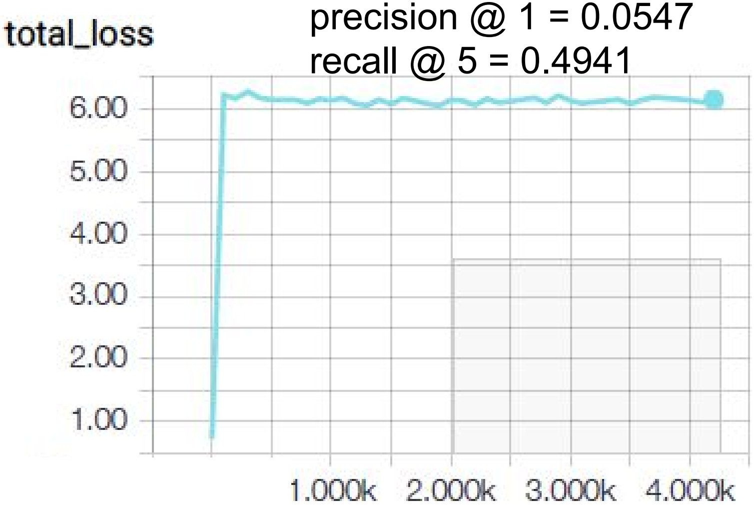
Train XRAY data via Inception-V3



Opacity

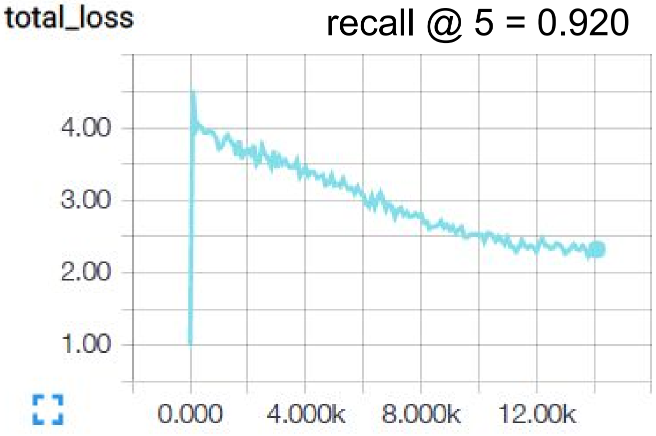
Train XRAY data via TF's Inception-V3

precision @ 1 = 0.0547
recall @ 5 = 0.4941



Train using Inception-V3 from scratch
(not looking good)

precision @ 1 = 0.587
recall @ 5 = 0.920



Retrain using Inception-V3 from pretrained model
(much better!)

Why did that work so well ???