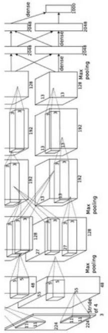
Deep Learning

Part II

Deep = Many Hidden Layers

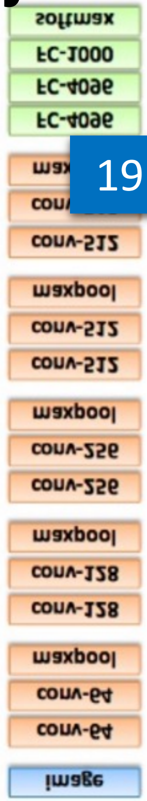
8 layers

16.4%



AlexNet (2012)

7.3%

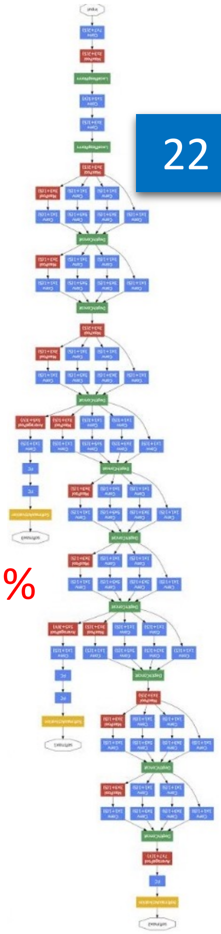


19 layers

VGG (2014)

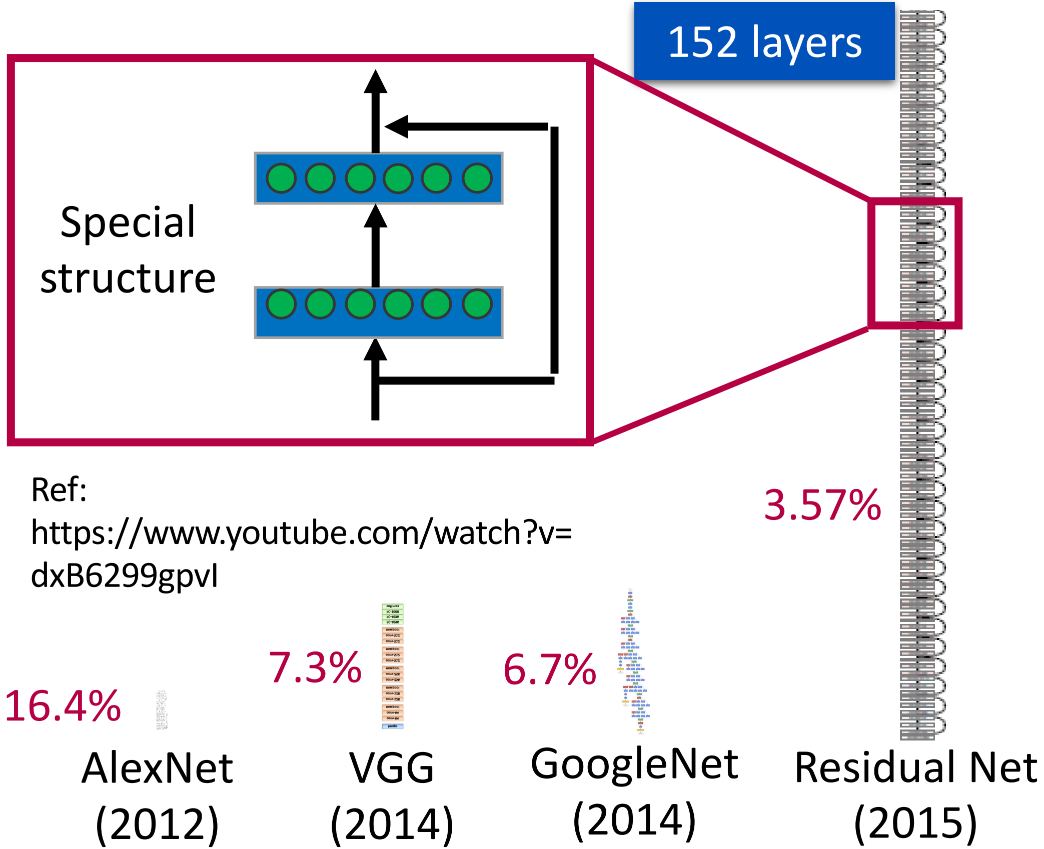
22 layers

6.7%



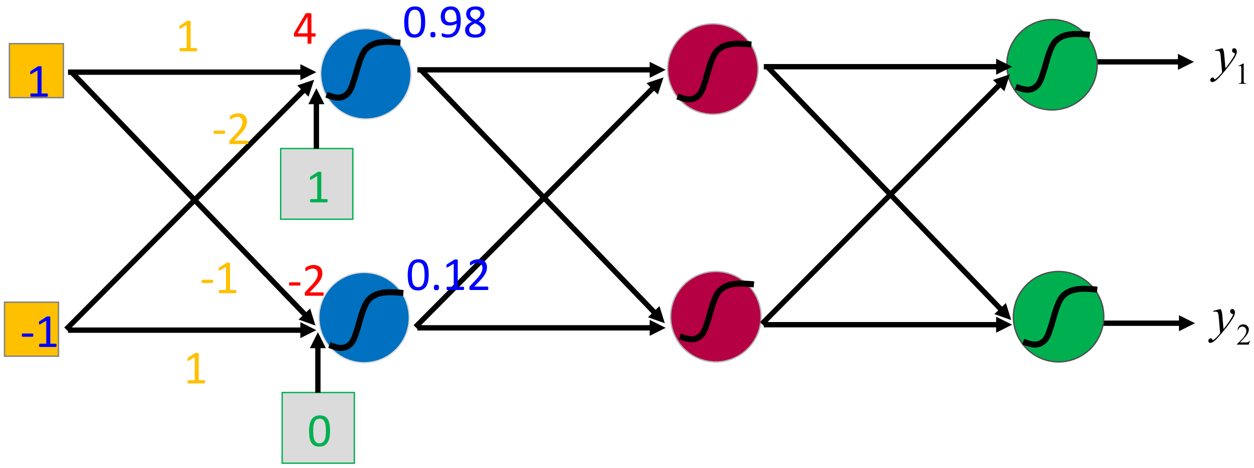
GoogleNet (2014)

Deep = Many Hidden Layers, Continued



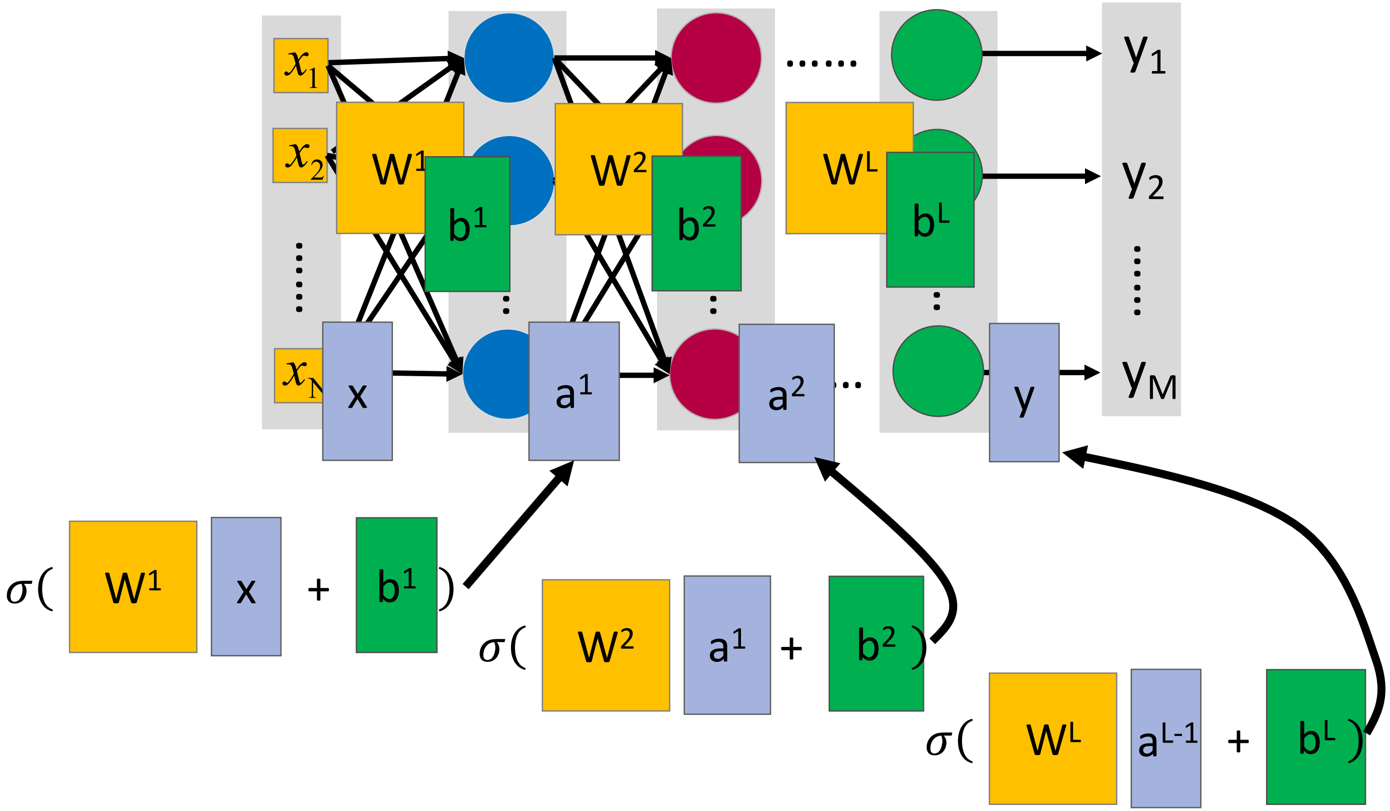
Ref:
https://www.youtube.com/watch?v=dxB6299gpvI

Matrix Operation



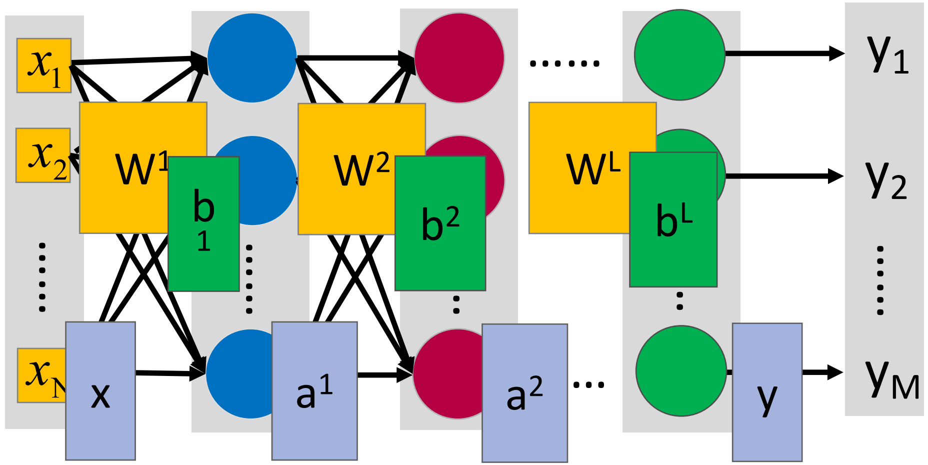
$$\sigma\left(\underbrace{\begin{bmatrix}1 & -2\\ -1 & 1\end{bmatrix}\begin{bmatrix}1\\ -1\end{bmatrix}+\begin{bmatrix}1\\ 0\end{bmatrix}}_{\begin{bmatrix}4\\ -2\end{bmatrix}}\right)=\begin{bmatrix}0.98\\ 0.12\end{bmatrix}$$

Neural Network: Part II



x_1
x_2
x_N
x
W^1
b^1
a^1
W^2
b^2
a^2
W^L
b^L
y
……
y_1
y_2
y_M
$\sigma($ W^1 x + b^1 $)$
$\sigma($ W^2 a^1 + b^2 $)$
$\sigma($ W^L a^{L-1} + b^L $)$

Neural Network: Part III

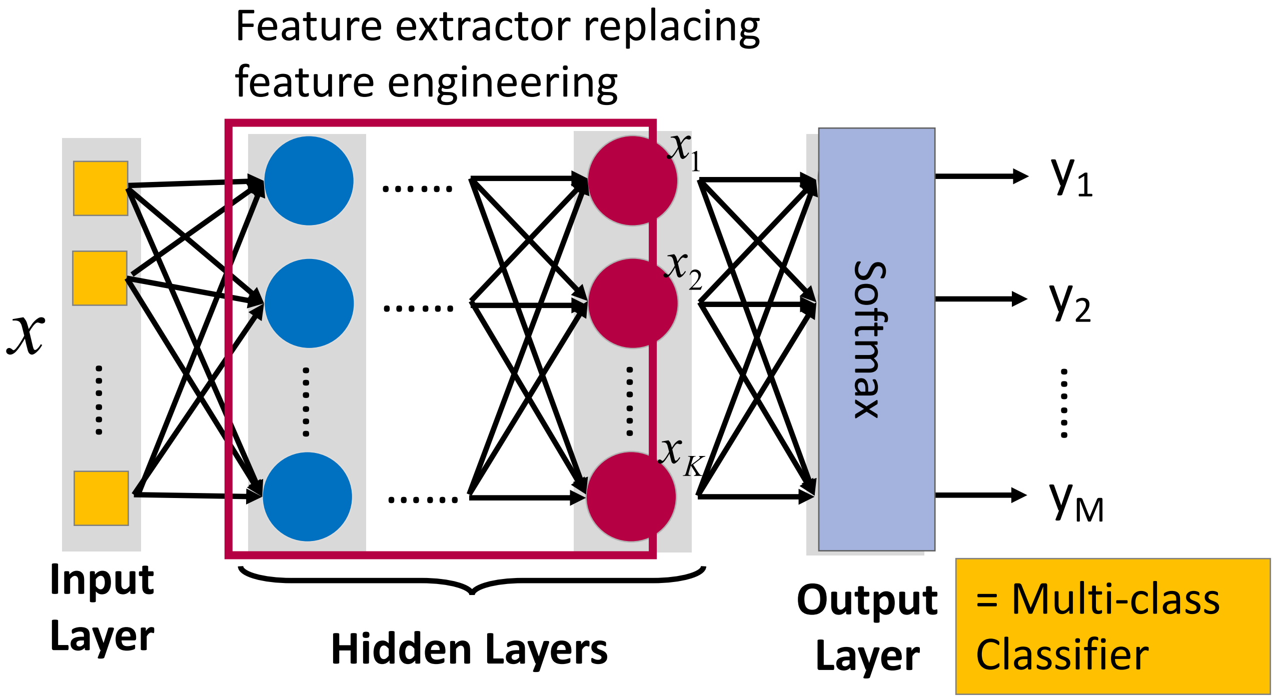


$$y = f(x)$$

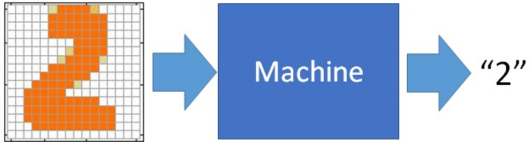
Using parallel computing techniques to speed up matrix operation

$$= \sigma(W^L \cdots \sigma(W^2 \sigma(W^1 x + b^1) + b^2) \cdots + b^L)$$

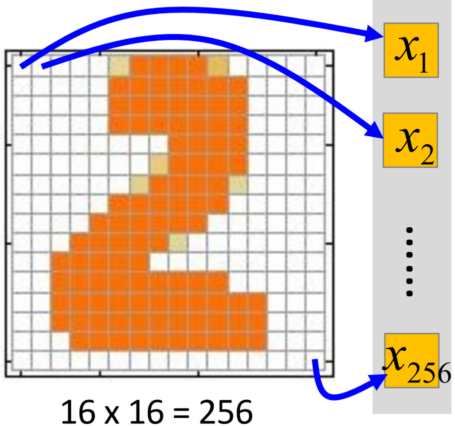
Output Layer as Multi-Class Classifier



Example Application: Part I



Input

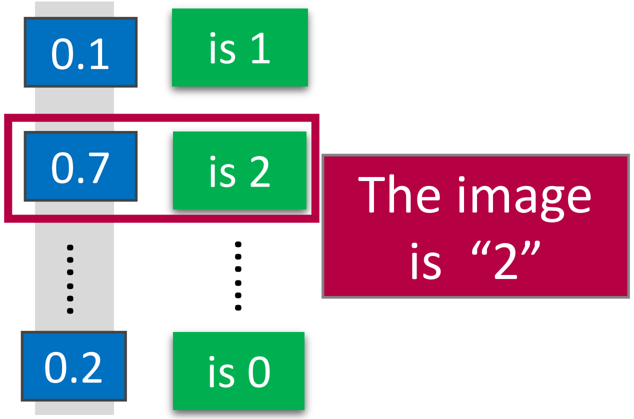


16 x 16 = 256

Ink → 1

No ink → 0

Output



0.1 is 1

0.7 is 2

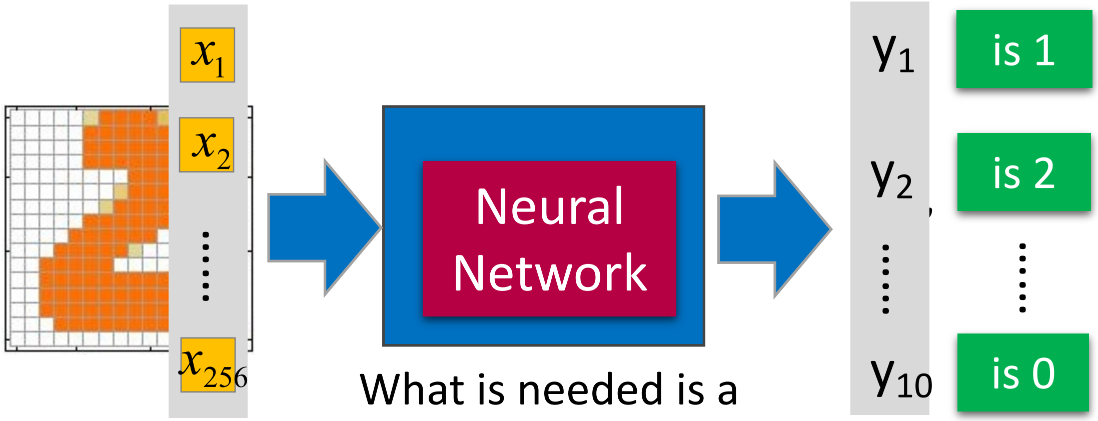
0.2 is 0

The image is “2”

Each dimension represents the confidence of a digit.

Example Application: Part II

Handwriting Digit Recognition

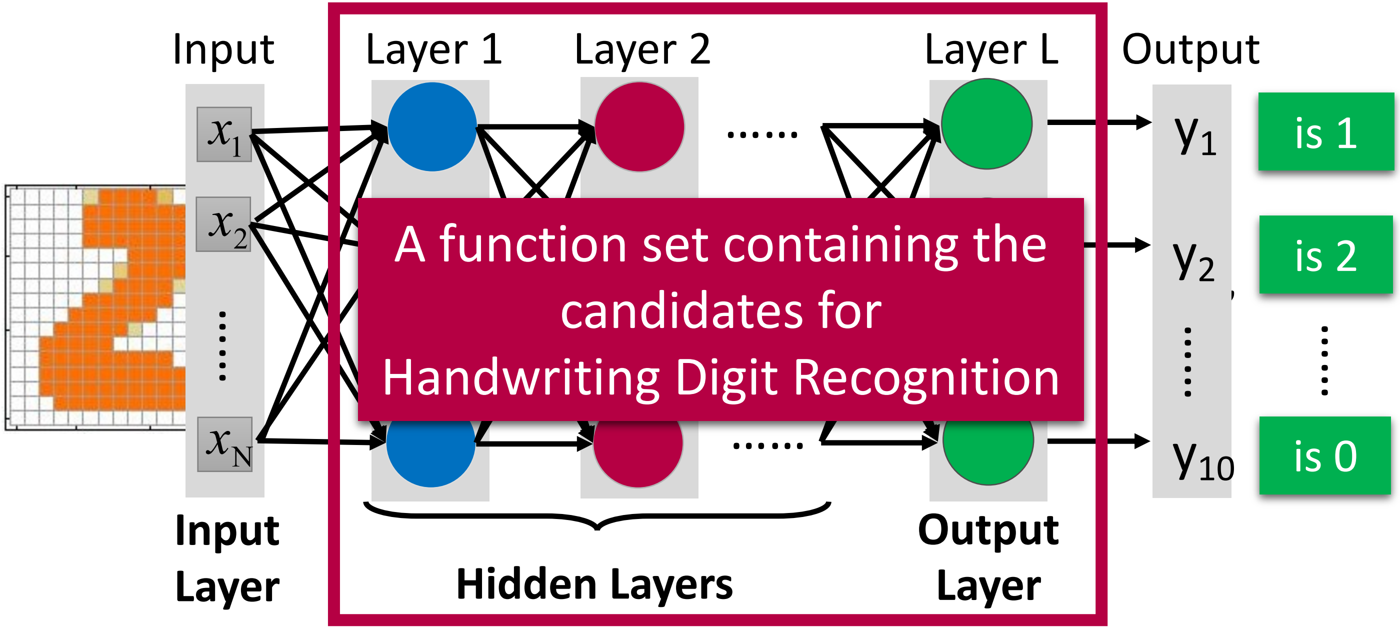


What is needed is a function ......

Input:
256-dim vector

Output:
10-dim vector

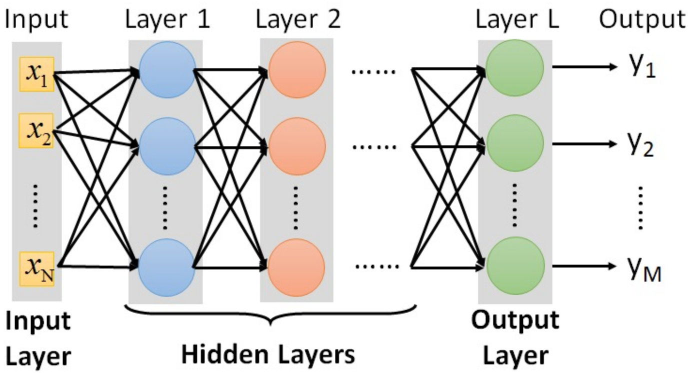
Example Application: Part III



A function set containing the candidates for Handwriting Digit Recognition

You need to decide the network structure to let a good function in your function set.

FAQ



- Q: How many layers? How many neurons for each layer?

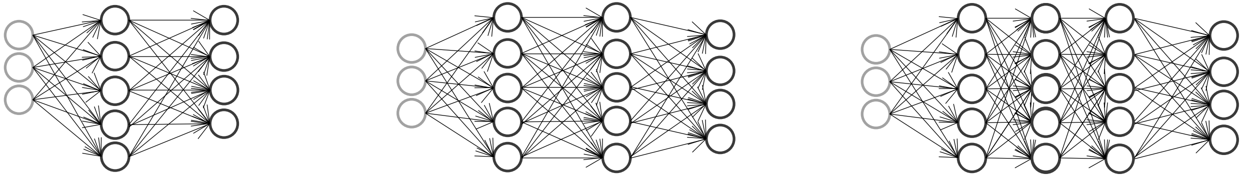
Trial and Error + Intuition

- Q: Can the structure be automatically determined?
  - E.g. Evolutionary Artificial Neural Networks
- Q: Can we design the network structure?

Convolutional Neural Network (CNN)

Training a Neural Network

Pick a network architecture (connectivity pattern between neurons)

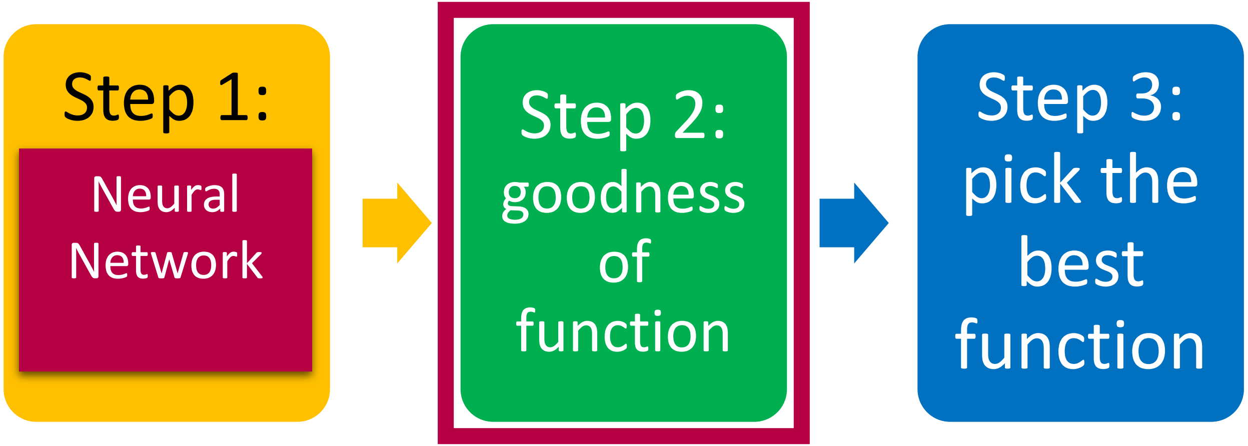


No. of input units: Dimension of features $x^{(i)}$

No. output units: Number of classes

Reasonable default: 1 hidden layer, or if >1 hidden layer, have same no. of hidden units in every layer (usually the more the better)

Three Steps for Deep Learning: Part II



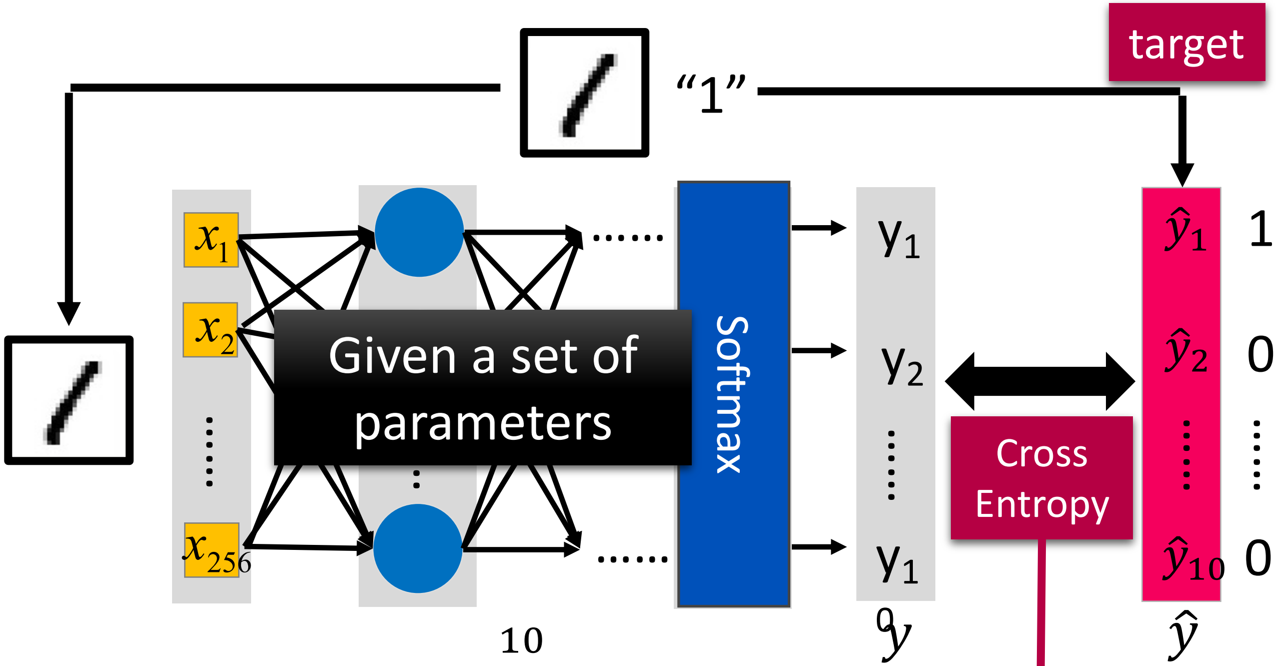
Step 1:

Neural Network

Step 2: goodness of function

Step 3: pick the best function

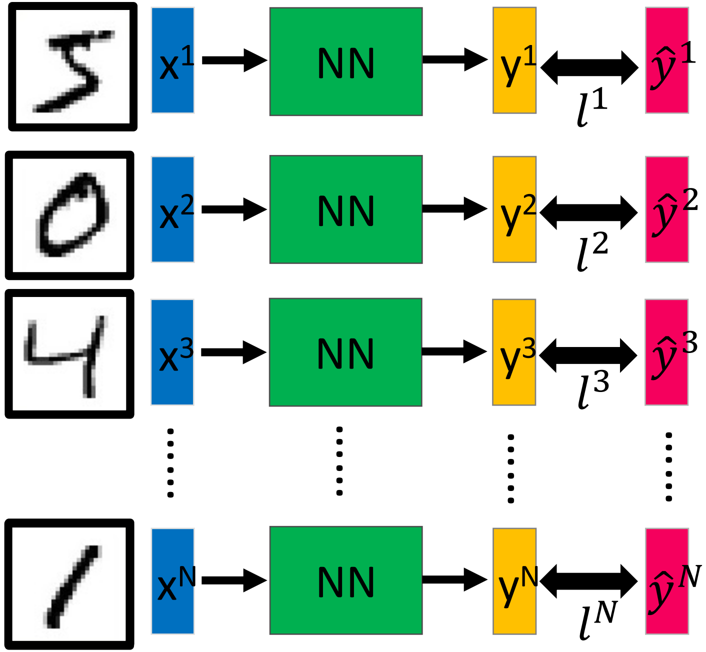
Loss for an Example



$$l(y\,,\hat{y}\,) = -\sum_{i=1}^{10} \hat{y}_i \ln y_i$$

Total Loss

For all training data …



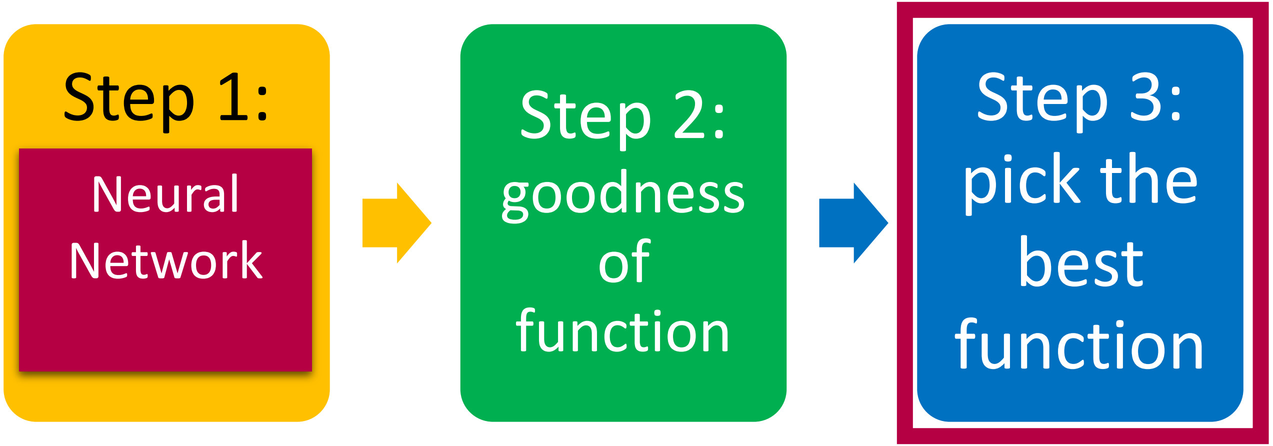
Total Loss:

$$\boxed{L = \sum_{n=1}^{N} l^n}$$

Find ***a function in function set*** that minimizes total loss L

Find ***the network parameters θ^**** that minimize total loss L

Three Steps for Deep Learning: Part III

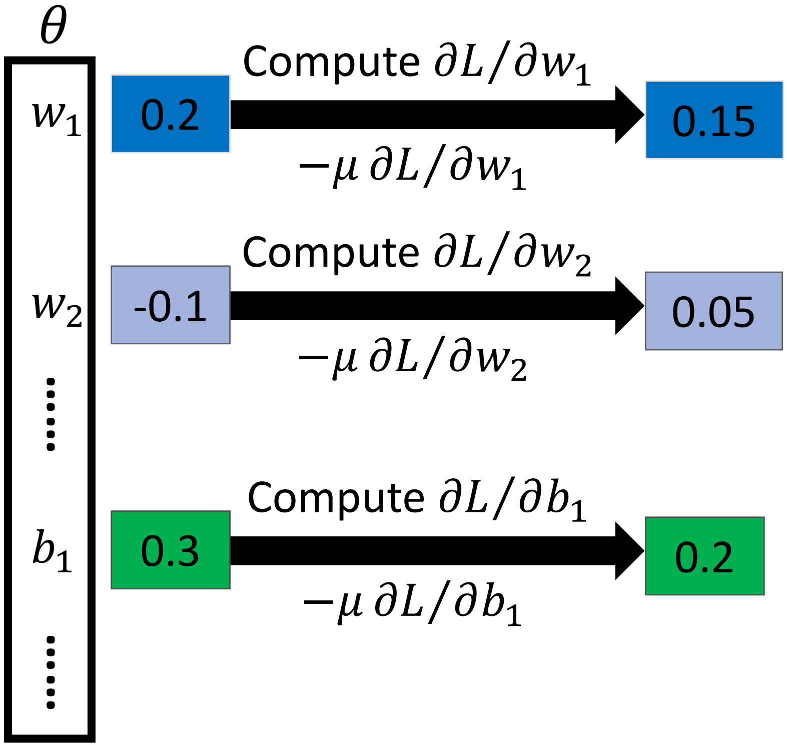


Step 1: Neural Network

Step 2: goodness of function

Step 3: pick the best function

Gradient Descent



θ

w_1 0.2 Compute $\partial L / \partial w_1$ $-\mu \, \partial L / \partial w_1$ 0.15

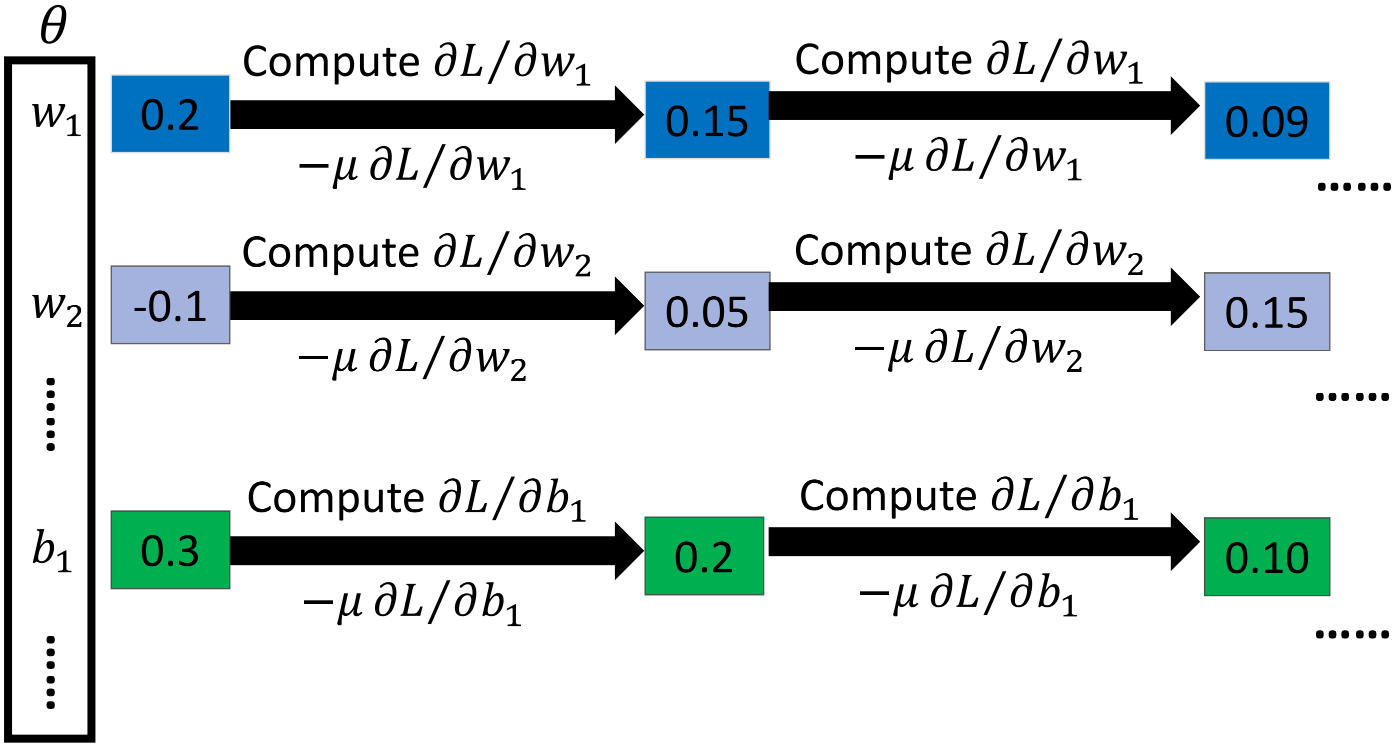
w_2 -0.1 Compute $\partial L / \partial w_2$ $-\mu \, \partial L / \partial w_2$ 0.05

b_1 0.3 Compute $\partial L / \partial b_1$ $-\mu \, \partial L / \partial b_1$ 0.2

$$\nabla L = \begin{bmatrix} \dfrac{\partial L}{\partial w_1} \\ \dfrac{\partial L}{\partial w_2} \\ \vdots \\ \dfrac{\partial L}{\partial b_1} \\ \vdots \end{bmatrix}$$

gradient

Gradient Descent, Continued


θ
w_1
w_2
b_1
0.2
Compute $\partial L/\partial w_1$
$-\mu\,\partial L/\partial w_1$
0.15
Compute $\partial L/\partial w_1$
$-\mu\,\partial L/\partial w_1$
0.09
……
-0.1
Compute $\partial L/\partial w_2$
$-\mu\,\partial L/\partial w_2$
0.05
Compute $\partial L/\partial w_2$
$-\mu\,\partial L/\partial w_2$
0.15
……
0.3
Compute $\partial L/\partial b_1$
$-\mu\,\partial L/\partial b_1$
0.2
Compute $\partial L/\partial b_1$
$-\mu\,\partial L/\partial b_1$
0.10
……

Backpropagation

Backpropagation: an efficient way to compute $\partial L / \partial w$ in neural network

TensorFlow

torch

theano

Caffe

Microsoft CNTK

Chainer

Deep Learning library produced by Amazon
DSSTNE

mxnet