

Machine learning. Neural network overview

Pooran Singh Negi

University Of Denver

August 9, 2018

Outline

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification

- Artificial neurons. Perceptron and sigmoid
- Designing Feedforward neural network for classification

What is Neural Network and deep learning

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid
Designing
Feedforward
neural network
for classification

Neural Network: Biological Neuron inspired, mathematical model. Inspiration is beautiful if you believe in connectionism.

Connectionism(wikipedia): Connectionism is a set of approaches in the fields of artificial intelligence, cognitive psychology, cognitive science, neuroscience, and philosophy of mind, that models mental or behavioral phenomena as the emergent processes of **interconnected networks of simple units**. There are many forms of connectionism, but the most common forms use neural network models.

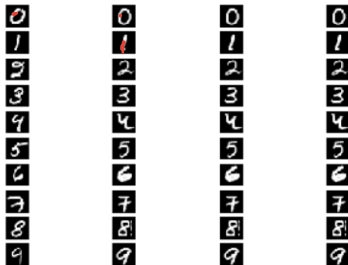
Deep learning: Set of technique for training deep neural network.

Writing Algorithm for digit recognition. Human vs Machine

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid
Designing
Feedforward
neural network
for classification



Huber

v_1 (visual cortex)
 v_2
 v_3
 v_4
 v_5

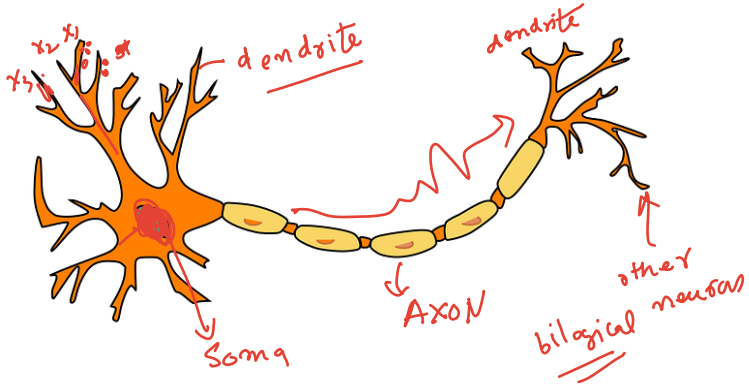
Biological neuron

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification



credit: <https://pixabay.com/en/neuron-nerve-cell-axon-dendrite-296581>

Perceptron

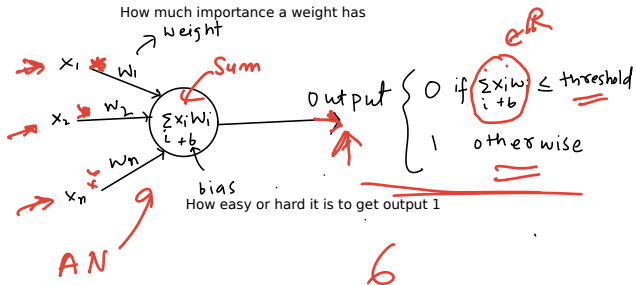
Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification

Developed in the 1950s and 1960s by the Frank Rosenblatt, inspired by earlier work by Warren McCulloch and Walter Pitts.



Perceptron as NAND

Machine learning.
Neural network overview

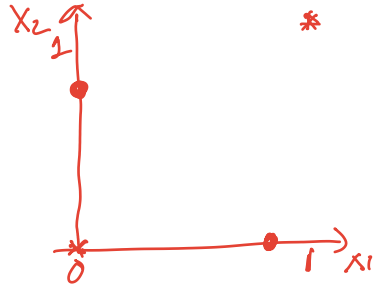
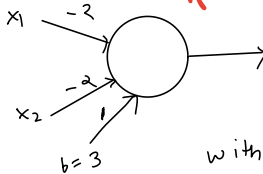
Pooran Singh Negi

Artificial neurons.
Perceptron and sigmoid

Designing Feedforward neural network for classification

XOR problem

x_1	x_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0



with threshold = 0
this perceptron is NAND gate

NAND(x_1, x_2)

x_1	x_2	NAND
0	0	1 ✓
0	1	1 ✓
1	0	1 ✓
1	1	0

Perceptron and training (How to learn w_i and b)

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification

$$\sum_{i=1}^N w_i x_i + b < \text{thr}$$

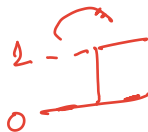
If output is one
change weight

1 As we change weights in perceptron out may flip
completely from 0 to 1.

then
after
Some
weight
combi
it become

2 Training perceptron or network of perceptron is hard

Can we model output in the range $[0, 1]$ more gradually as
a function of inputs



A New type of artificial neuron. Sigmoid

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

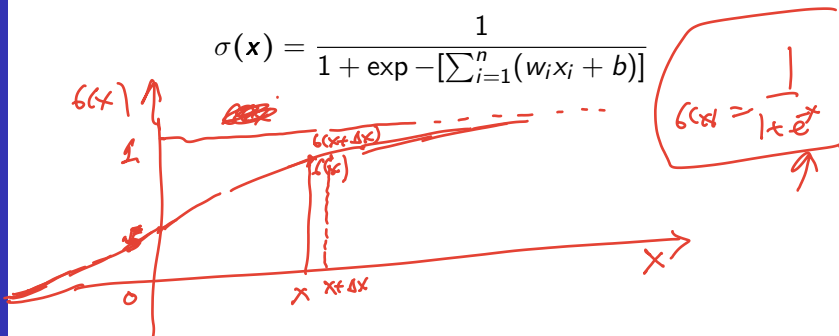
Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification


- change output smoothly
- But there are other problem

For input x , and $w \in \mathbb{R}^n$, and bias $b \in \mathbb{R}$ sigmoid function is

$$\sigma(x) = \frac{1}{1 + \exp - [\sum_{i=1}^n (w_i x_i + b)]}$$



Using calculus we can show that

$$\Delta\sigma(\mathbf{x}) \approx \sum_{i=1}^n \frac{\partial\sigma(\mathbf{x})}{\partial w_i} \Delta w_i + \frac{\partial\sigma(\mathbf{x})}{\partial b} \Delta b$$


Small change in output if small change in w_i and b

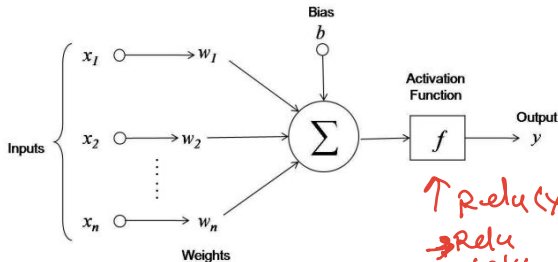
Without Brain Stuff: Artificial Neuron Activation Model

Machine learning.
Neural network overview

Pooran Singh Negi

Artificial neurons.
Perceptron and sigmoid
Designing Feedforward neural network for classification

A typical model for artificial neuron activation is



$\uparrow \text{relu}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$

$\rightarrow \text{relu}$
 $\rightarrow \text{iebu}$

$y = f(g(X; W, b))$, where f is non linear activation function.

Affine function g (pre-activation function) measures global ($\sum_i w_i x_i + b$) or local (e.g convolution $X * W + b$) similarity, correlation.

convolutions
• operation

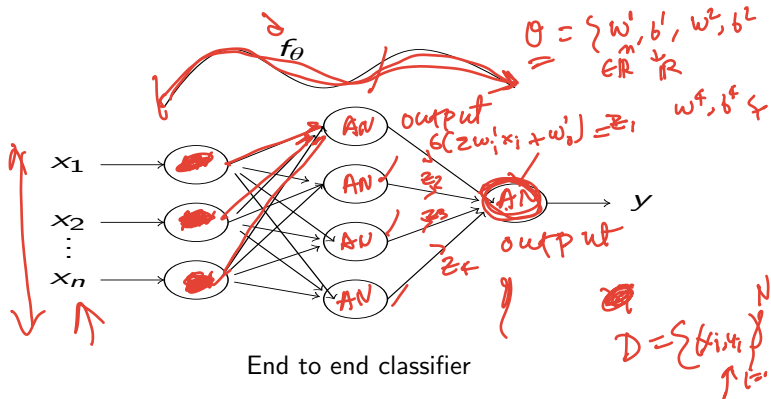
Feedforward Neural Networks(Fully connected layer)

Machine learning.
Neural network overview

Pooran Singh
Negi

Artificial neurons.
Perceptron and sigmoid

Designing Feedforward neural network for classification



End to end classifier

- Neural networks computes the function $y_i = f_{\theta}(x_i)$.
- Highly non linear end to end.
- Parameters θ can be learned via gradient descent by minimizing $\sum_{i=1}^N (y_i - f_{\theta}(x_i))^2 \approx \text{loss}$

Locally connected

Machine learning.
Neural network overview

Pooran Singh
Negi

Artificial neurons.
Perceptron and sigmoid

Designing Feedforward neural network for classification

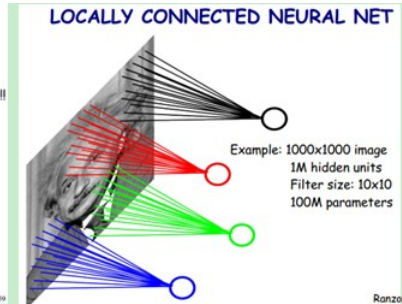
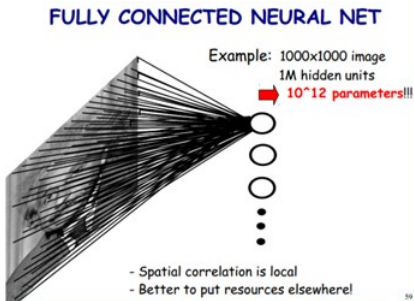


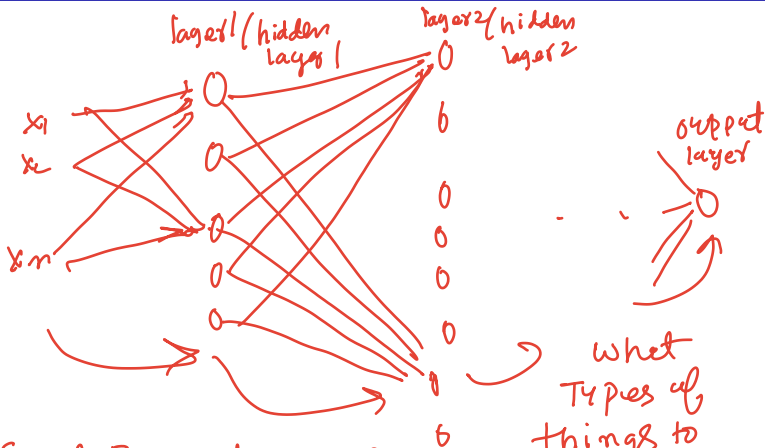
image credit:<https://irenelizihui.files.wordpress.com/2016/02/cnn1.png>

Representing operation between two layer via matrix multiplication and pointwise operation

Machine learning.
Neural network overview

Pooran Singh Negi

Artificial neurons.
Perceptron and sigmoid
Designing Feedforward neural network for classification



Feed Forward Neural

- No of layer network
- choice of activation
- no of AN in a layer

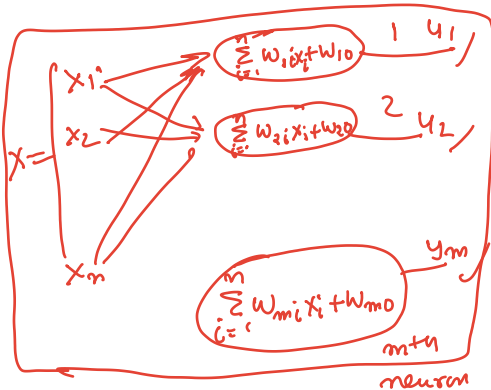
what
Types of
things to
play with
in this arch

Machine learning. Neural network overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification



$$\sigma \left(\underline{\underline{XW + W_0}} \right) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Pointwise activation function

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}^T \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{bmatrix} + \begin{bmatrix} w_{10} \\ w_{20} \\ \vdots \\ w_{n0} \end{bmatrix}$$

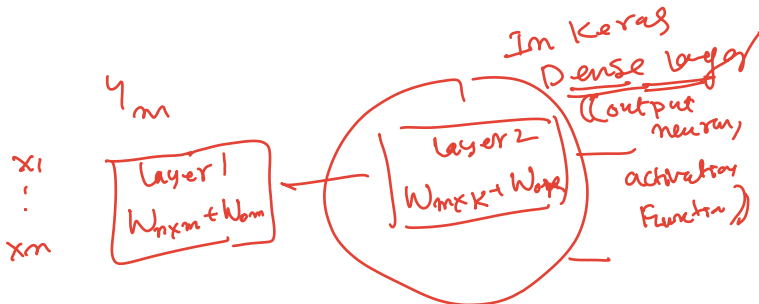
W $m \times n$ $n \times 1$

u_1

$$y = u_2$$

$$y^T W = z$$

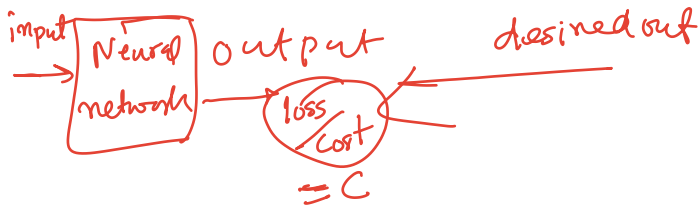
$m \times K \quad 1 \times K$



Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid
Designing
Feedforward
neural network
for classification



How to learn the weights and biases of various AN(artificial neuron in the network?)

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification

Let say we want to classify a digit $\mathbf{x} \in R^{28 \times 28 = 784}$ in MNIST dataset. So we build a Feedforward neural network f giving us output $\mathbf{y} = f(\mathbf{x})$ where \mathbf{y} is 10 dimensional output like $[1, 0, 0, 0, 0, 0, 0, 0, 0, 0]$

We use gradient descent or variant [▶ Link](#) of it to find weights and biases in AN such that network predicts correct output for training example or minimizes some cost function.
Main ides of gradient descent is

$$w_{k+1} = w_k - \eta \frac{\partial \text{cost or loss function}}{\partial w_k}$$

↑ learning rate

like **squared loss** i.e.

$$\ell(D; \theta) = \sum_{i=1}^N (\mathbf{y}_i - f_{\theta}(\mathbf{x}_i))^2$$

given that network outputs continous value vector/scalar. or
cross entropy

$$\ell(D; \theta) = - \sum_{i=1}^N \mathbf{y}_i^T f_{\theta}(\mathbf{x}_i)$$

Handwritten notes: cross entropy (with an arrow pointing to the equation), KL loss

given that network outputs probability of different classes and class label \mathbf{y}_i is **onehot** encoded. Here θ represents all the weights and biases of all the ANs and other paramters in the neural network f .

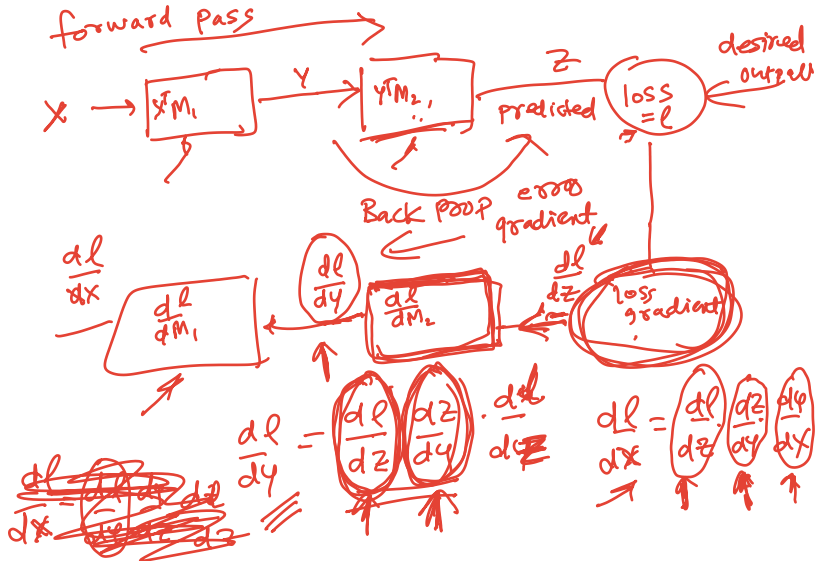
Backpropagation

Machine learning.
Neural network overview

Pooran Singh
Negi

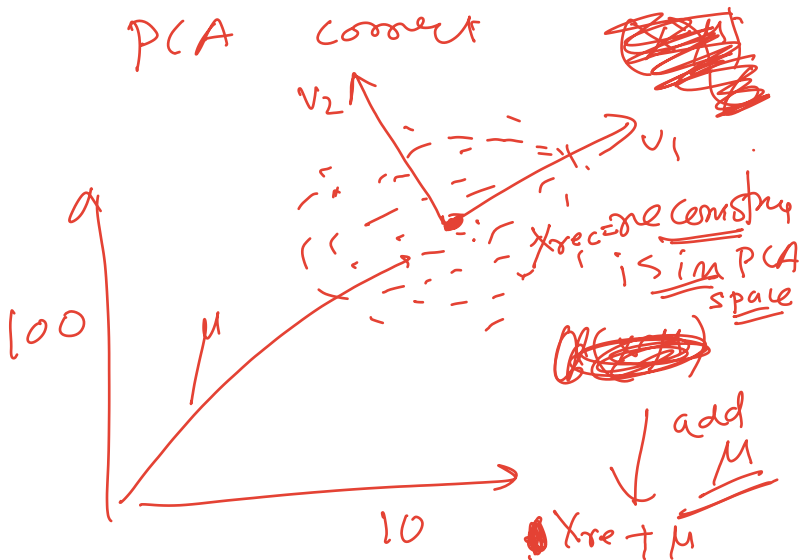
Artificial neurons.
Perceptron and sigmoid

Designing Feedforward neural network for classification



$$\neq X_{std} = std.$$

Stand. Scals
(X)



Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification

Thank you!

Machine
learning.
Neural
network
overview

Pooran Singh
Negi

Artificial
neurons.
Perceptron and
sigmoid

Designing
Feedforward
neural network
for classification



Thank you!