# Homework-3 (Theory)
## (Neural Network)

5 No we can't model a XOR operation truth tables with a neural network having arbitrary depth & linear activations.

Basically a neural net is a function mapping $f: R \to R$ or

$$f(x) = B(Ax + b)$$

↳ some activation function

$$\cong \text{sum with linear function}$$

$$\cong BAx + Bb$$
$$= \text{const} + \text{const}$$
$$= \text{const}$$

hence neural net of any arbitrary length would not be able to classify XOR function Truth table.

4 learning is slow in neural network when Squared Error is used for classification because for example if we use sigmoid activation

then $\hat{y} = \sigma(z) = \sigma(Wx)$

then Error $= (y - \hat{y})^2 = ((y - \sigma(z))^2$

now $\frac{\partial L}{\partial \theta} = -(y - \sigma(z)) \cdot \sigma'(z) \cdot x$

now when $\sigma(z)$ tends to 0 or 1 $\sigma'(z)$ gets close to zero & when $\sigma(z)$ is close to 0.5, $\sigma'(z)$ will reach maximum. In this case, when the difference between $y$ & $\hat{y}$ is large $\sigma'(z)$ will reach to zero, thereby decreasing convergence speed.

when we cross entropy, loss is

$$L = \frac{-1}{n} \sum_{i=1}^{n} \left[ y^i \log(\hat{y}^{(i)}) + (1-y^{(i)}) \log(1-\hat{y}^{(i)}) \right]$$

It measures the divergence between 2 probability distributions. Now if cross entropy is large which means difference in distribution is large. while if cross entropy small, means two distributions are similar (approximately) hence we can see that the convergence problem is not here in cross entropy.

$$\frac{\partial L}{\partial w_j} = \frac{-1}{n} \sum_{x} \left( \frac{y}{\sigma(z)} - \frac{(1-y)}{1-\sigma(z)} \right) \sigma'(z)\, y$$

$$= \frac{1}{n} \sum_{x} x_j \left( \sigma(z) - y \right)$$

3 neural network of n layers
input is m dimensional array with each value in range [50, 1000].

activation → sigmoid $\left(\sigma(z) = \frac{1}{1+\exp(-z)}\right)$

The range of sigmoid is between 0 & 1, so at tail of sigmoid (at time of saturation), either 0 or 1, the gradients become zero. So during backpropogration, zero is multiplied to the Error of this layer for whole objective. Therefore no signal will flow through the neuron to its weight & recursively to its data.

Moreover sigmoid outputs are not zero centered, hence the gradient will always become either positive or negative. So this could produce undesirable zig-zagging in the gradient updates for the weight.

These are the above problems that can be faced during neural network training.

now when we using RELU as an activation layer, the convergence of gradients accelerates, all expensive exponential operations can be avoided & the problem of

vanishing gradients ~~is also~~ can also be avoided.

We can use various preprocessing techniques like zero-center the data and then normali-ze them.

for initializing the weights of neural network, it is not reasonable to assign all weights to zero, ~~as~~ because if every neuron in the network computes the same output, then they will also compute the same gradient during backpropogation and undergo the exact same parameter updates, i.e. there is no asymetry between neurons, if the weights are initialized to be the same.

He should use cross entropy loss for classification problem as Mean Squared Error may have slow convergence rate.