PAGE NO.
DATE : / /

Week 3

Video 1: Tuning Process

Hyperparameters

Preference for tuning & importance

Preference ①
α

↓

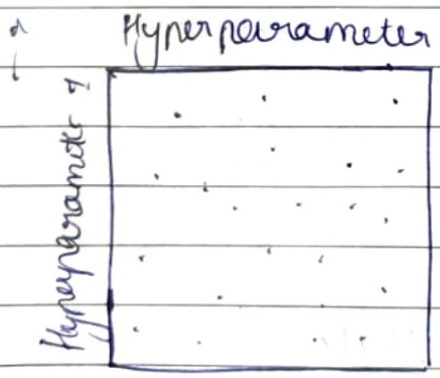
Preference ②
β, hidden units, mini batch size

↓

Preference ③
layers, learning rate

↓

Preference ④
β_1, β_2, ε
0.9 0.99 10^{-8}



Choose points at random as you don't know which hyperparameter suits you best

Coarse to fine sampling - when you find some hyperparameters values that give you a better performance - zoom into a smaller region around these values and sample more densely in this space

each hidden layers activations

This has regularisation effect.

Batch norm causes the input values to become more stable, so that the later layers of the neural network has more firm ground to stand on. And even though the input distribution changes a bit, it changes less and it allows each layer to learn by itself a little independently of the other layers and this speeds up learning.

Video 4: Batch Norm at Test time

$$\mu = \frac{1}{m} \sum_i z^{(i)}$$

$$\sigma^2 = \frac{1}{m} \sum_i (z^{(i)} - \mu)^2$$

$$z^{(i)}_{norm} = \frac{z^{(i)}}{\sqrt{\sigma^2 + \varepsilon}}$$

$$\tilde{z}^{(i)} = \gamma z^{(i)}_{norm} + \beta$$

① We compute mean & variance of mini batch
② In testing we process one example at a time. The mean & Variance of one example won't make sense
③ Thus, we compute an estimated values of mean & variance to use it in testing time

Page No.
Date

Working with mini batches

In batch normalisation, the mean of $z^{[l]}$ values in the layer gets zeroed out and is replaced by $\beta^{[l]}$ that affects the biased terms.

Algorithm:

① for t = 1 ... number of mini batches

② compute forward propagation on $X^{\{t\}}$
③ In each hidden layer use BN to replace $z^{[l]}$ with $\tilde{z}^{[l]}$
④ use backprop to compute $dw^{[l]}$, $d\beta^{[l]}$, $d\gamma^{[l]}$
⑤ update parameters:

$$w^{[l]} = w^{[l]} - \alpha\, dw^{[l]}$$
$$\beta^{[l]} = \beta^{[l]} - \alpha\, d\beta^{[l]}$$
$$\gamma^{[l]} = \ldots$$

Video 5: Why does batch norm work

Covariate shift: characteristics of the input data set change from training to test dataset.

Batch norm as regularisation:

Each mini batch is scaled by the mean/variance computed on just that mini batch.

This adds some noise to the values $z^{[l]}$ within that mini batch. So similar to dropout it adds noise to

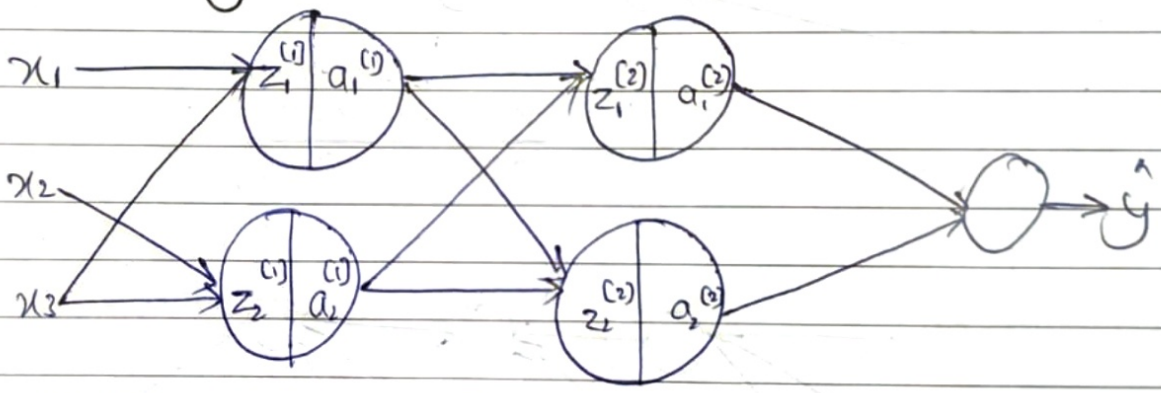
$$z^{(i)}_{norm} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \varepsilon}}$$

$$\tilde{z}^{(i)} = \gamma z^{(i)}_{norm} + \beta$$

γ & β would update weights - learnable parameters

If $\gamma = \sqrt{\sigma^2 + \varepsilon}$ & if $\beta = \mu$
then $\tilde{z}^{(i)} = z^{(i)}_{norm}$

Video 2: Fitting Batch Norm Into a Neural Network



$$X \xrightarrow{w^{[1]}\ b^{[1]}} z^{[1]} \xrightarrow[\text{Batch Norm BN}]{\beta^{[1]}\ \gamma^{[1]}} \tilde{z}^{[1]} \rightarrow a^{[1]} = g^{[1]}(\tilde{z}^{[1]})$$

$$\rightarrow w^{[2]}\ b^{[2]} \rightarrow \cdots \rightarrow a^{[2]} \leftarrow \tilde{z}^{[2]} \leftarrow \beta^{[2]}\ \gamma^{[2]}$$

Parameters:

$$w^{[1]}\ b^{[1]}\ w^{[2]}\ b^{[2]}, \ldots, w^{[L]}\ b^{[L]}$$
$$\beta^{[1]}, \gamma^{[1]}, \beta^{[2]}, \ldots, \beta^{[L]}, \gamma^{[L]}$$

$d\beta^{[2]}$

$\beta^{[2]} = \beta^{[2]} - \alpha d\beta^{[2]}$

PAGE NO.
DATE: / /

Video 2: Using an appropriate scale to pick hyperparameters

Use logarithmic scale:

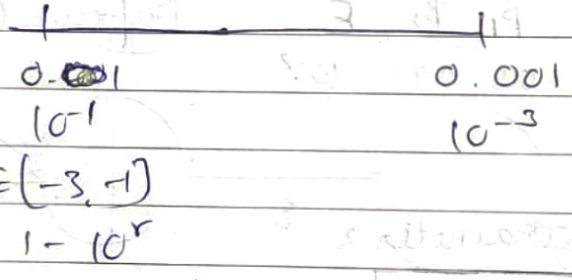
$r = -4 * np.random.rand()$ ← $r \in [-4, 0]$

$\alpha = 10^r$ ← $\alpha \in [10^{-4}, 10^0]$

$10^a \ldots 10^b$

$\beta \in [0.9, 0.999]$

$1-\beta \in [0.001, 0.1]$



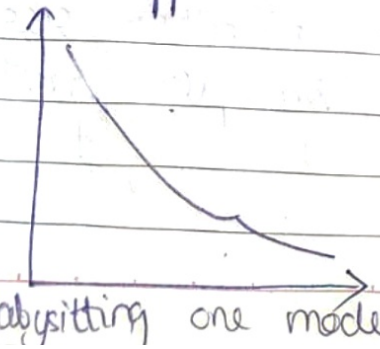
$r \in [-3, -1]$

$\beta = 1 - 10^r$

Video 3:

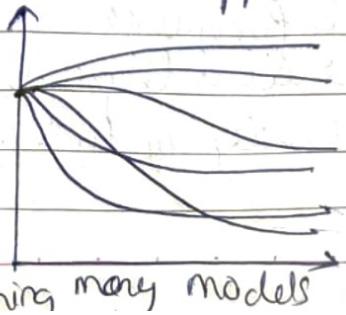
Hyperparameters can get stale
Thus, revaluate

Panda approach



babysitting one model

Caviar approach



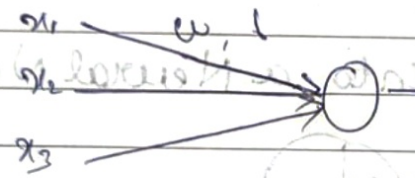
Training many models on parallel

Batch Normalisation

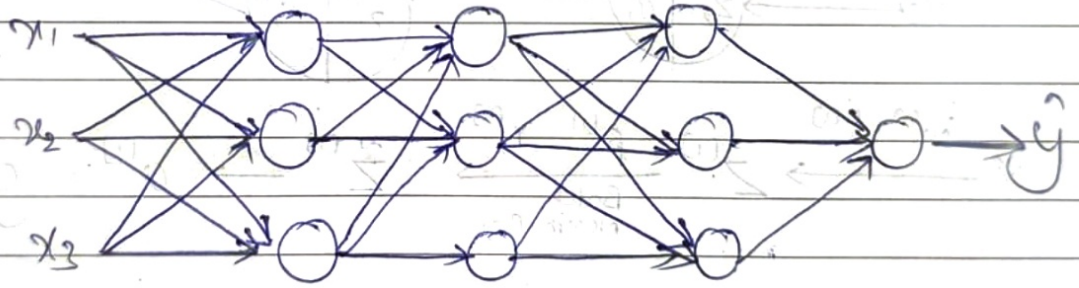
Video 1:

* Advantages of batch normalisation -
  (i) It makes your hyperparameter search problem easier.
  (ii) neural network becomes more robust
  (iii) You have a lot of suitable options for hyperparameters
  (iv) Helps train deep networks.

*



$$x' = \frac{x}{\sigma} = \frac{x-\mu}{\sigma}$$



we normalise $z^{(2)}$

implementing batch norm

given some intermediate values in NN ... $z^{(1)}, \ldots, z^{(m)}$

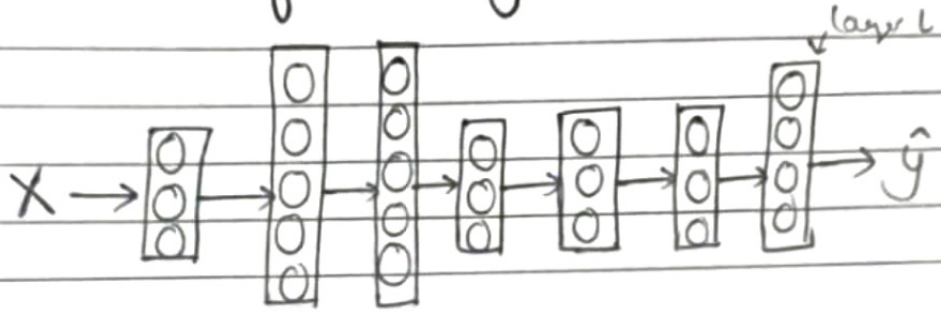
$$\mu = \frac{1}{m} \sum_i z^{(i)}$$

$$\sigma^2 = \frac{1}{m} \sum_i (z^{(i)} - \mu)^2$$

- We can use weighted average across mini batches.
- We will use estimated values of mean & variance to Test.

Multi Class Classification

Video 1: Softmax Regression



$\hat{y}$ is (4,1)

$$z^{[l]} = w^{[l]} a^{[l-1]} + b^{[l]}$$

Activation function for softmax

(4,1) ← $t = e^{(z^{[l]})}$ (4,1)

$$a^{[l]} = \frac{e^{z^{[l]}}}{\sum_{j=1}^{4} t_i}$$

Video 2: Training a Softmax Classifier

Softmax is a generalization of logistic ~~regression~~ activation function to C classes. If C=2, softmax reduces to logistic regression.

$$L(y, \hat{y}) = -\sum_{j} y_j \log(\hat{y}_j)$$