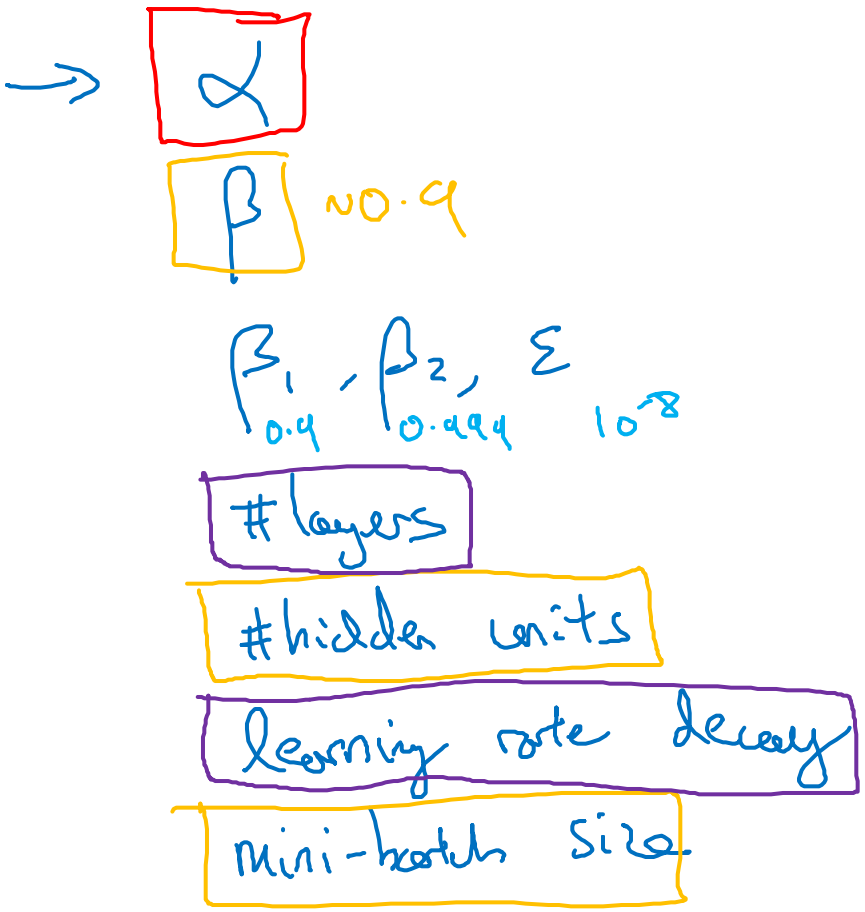
Copyright Notice

deeplearning.ai

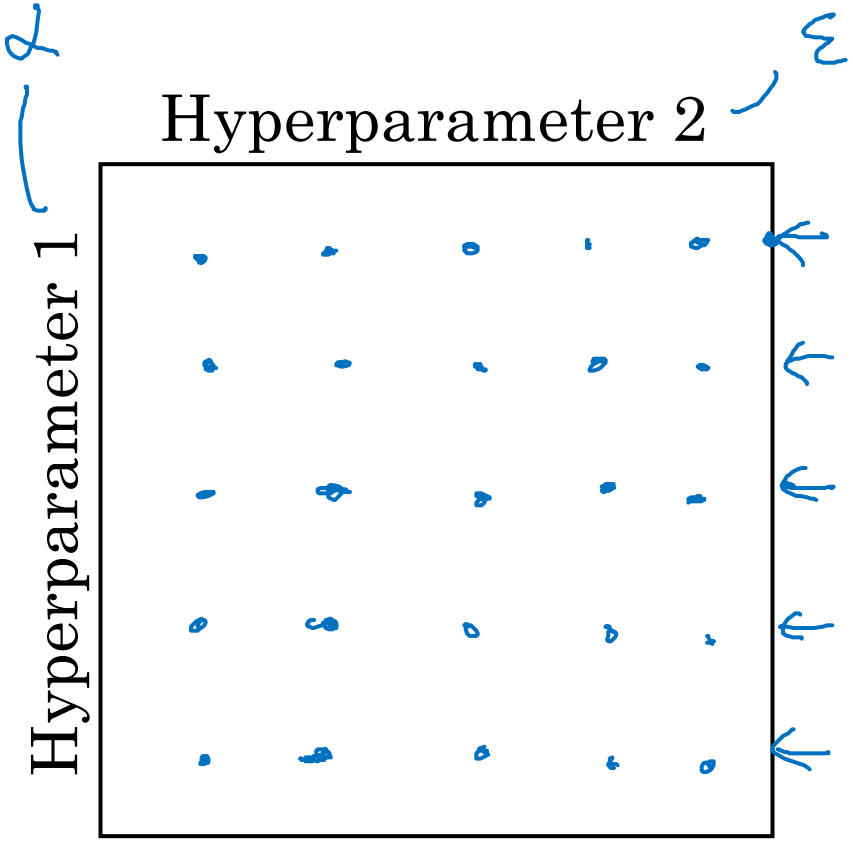
Hyperparameter tuning

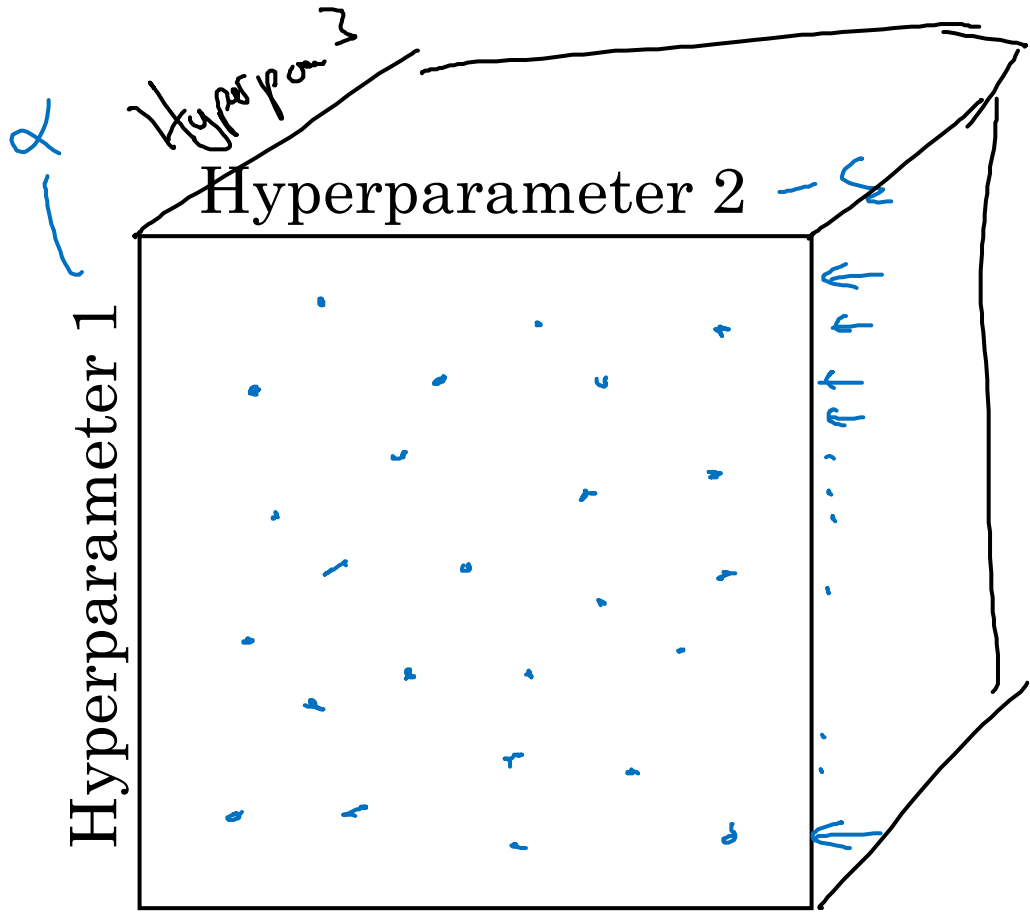
Tuning process

Hyperparameters

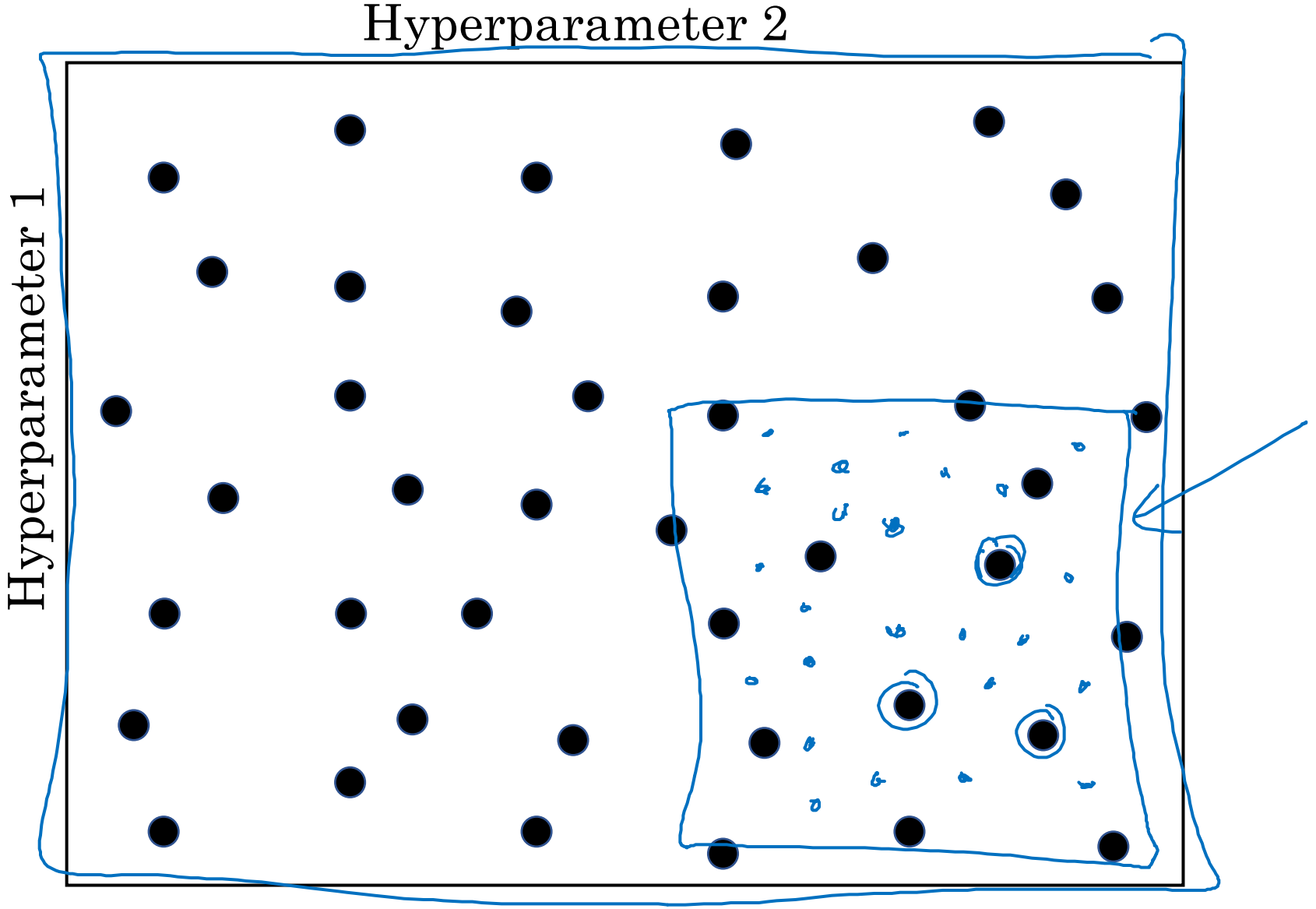


Try random values: Don't use a grid





Coarse to fine



Andrew Ng

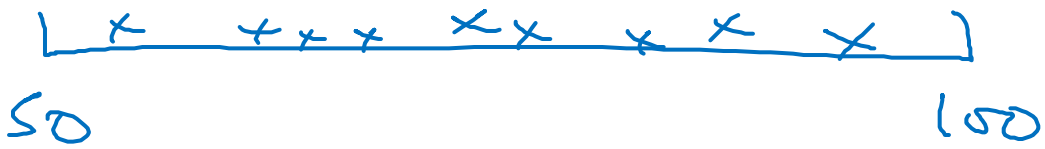
deeplearning.ai

Hyperparameter tuning

Using an appropriate scale to pick hyperparameters

Picking hyperparameters at random

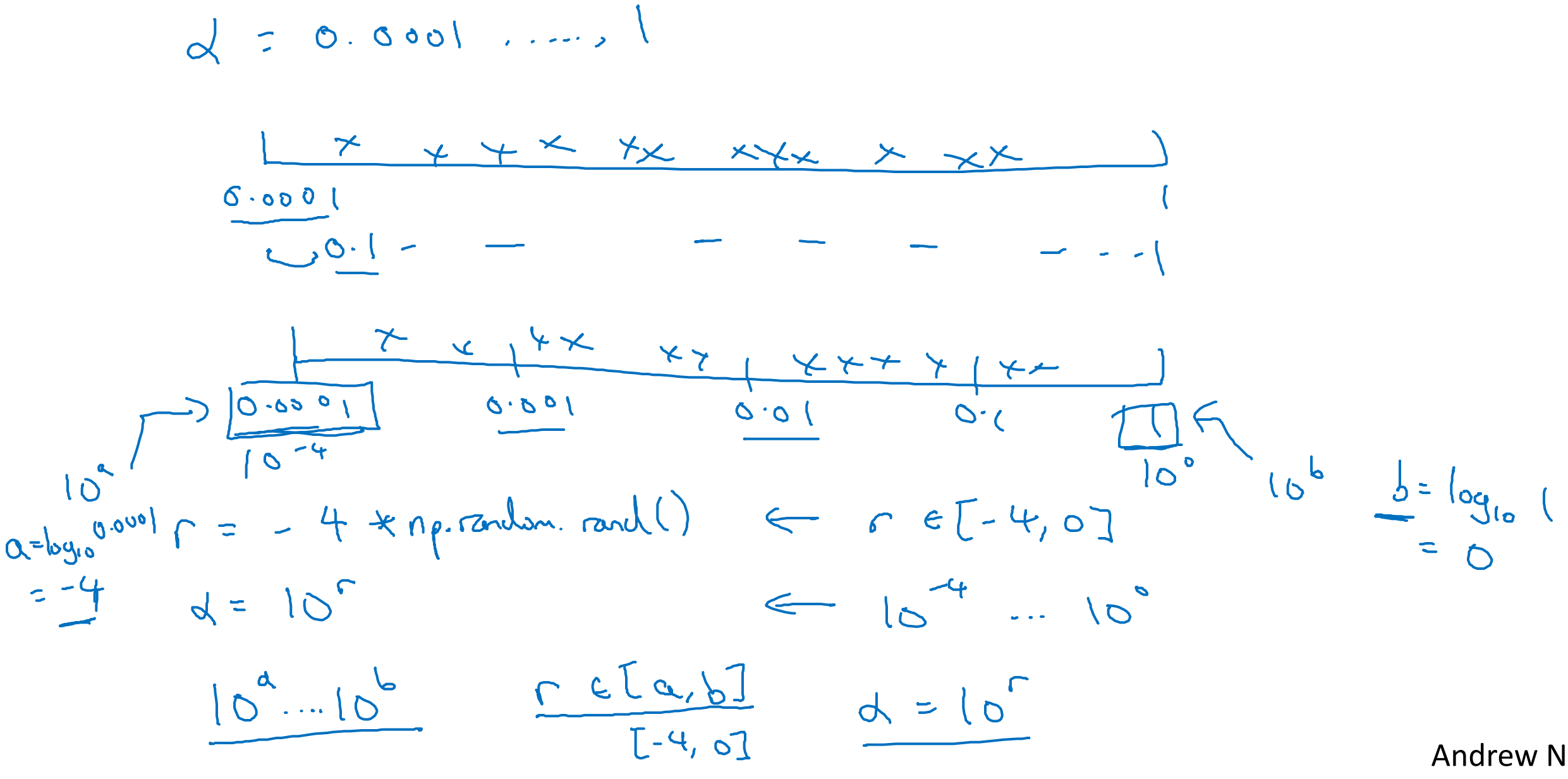
→ $n^{[l]} = 50, \ldots, 100$



→ #layers L: 2 – 4

2, 3, 4

Appropriate scale for hyperparameters



Andrew Ng

Hyperparameters for exponentially weighted averages

$\beta = 0.9 \quad \ldots \quad 0.999$

$10 \qquad\qquad 1000$

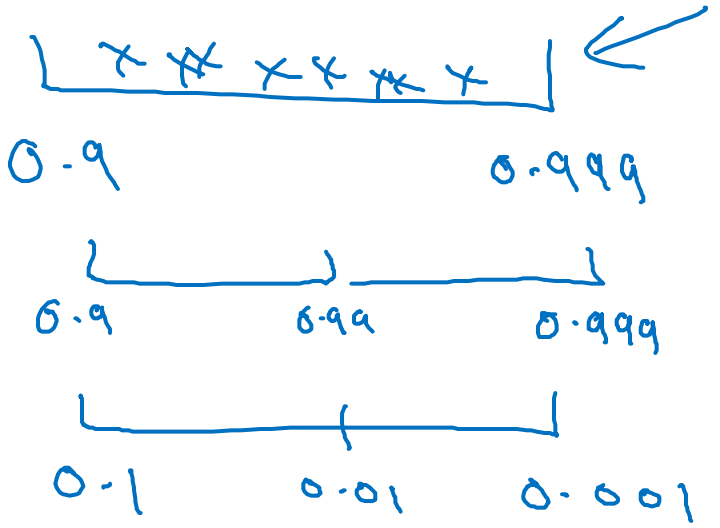
$1-\beta = 0.1 \quad \ldots \quad 0.001$

$\beta: 0.900 \rightarrow 0.9005$ } ~ 10

$\beta: 0.999 \rightarrow 0.9995$

$\sim 1000 \qquad \sim 2000$

$\frac{1}{1-\beta}$



$10^{-1} \qquad\qquad 10^{-3}$

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

$1-\beta = 10^{r}$

$\beta = 1 - 10^{r}$

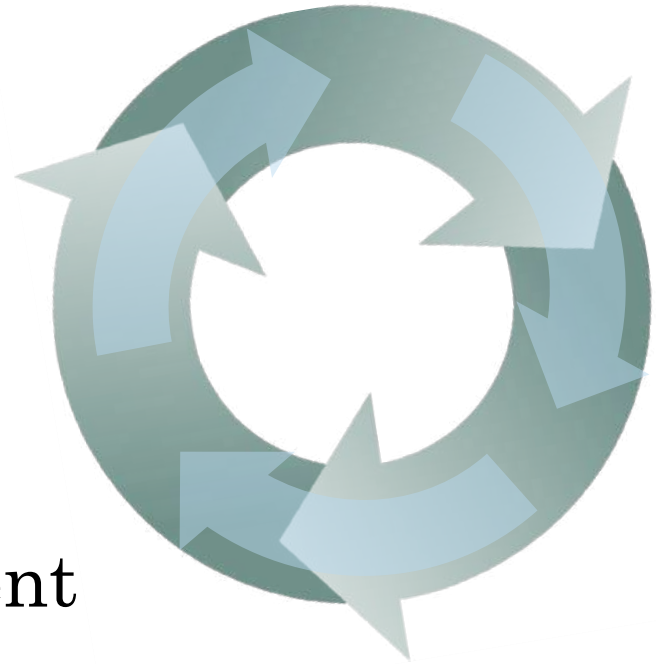
deeplearning.ai

Hyperparameters tuning

Hyperparameters tuning in practice: Pandas vs. Caviar

Re-test hyperparameters occasionally

Idea



Experiment

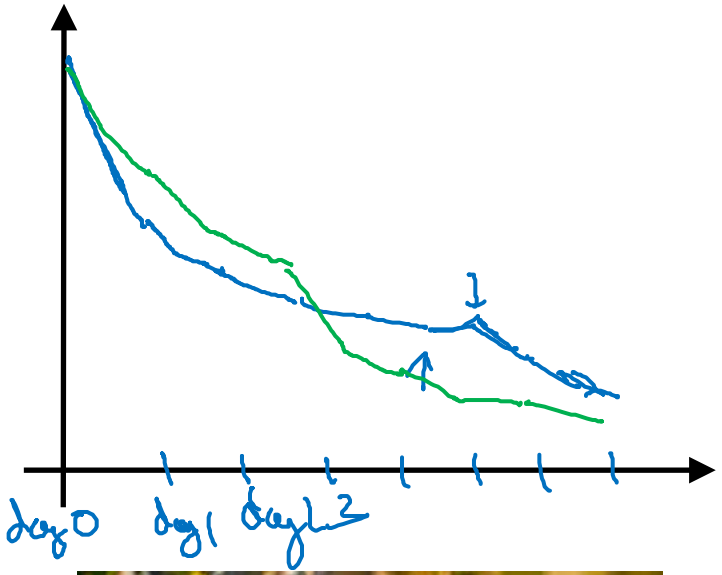
Code

- NLP, Vision, Speech, Ads, logistics, ...

- Intuitions do get stale. Re-evaluate occasionally.

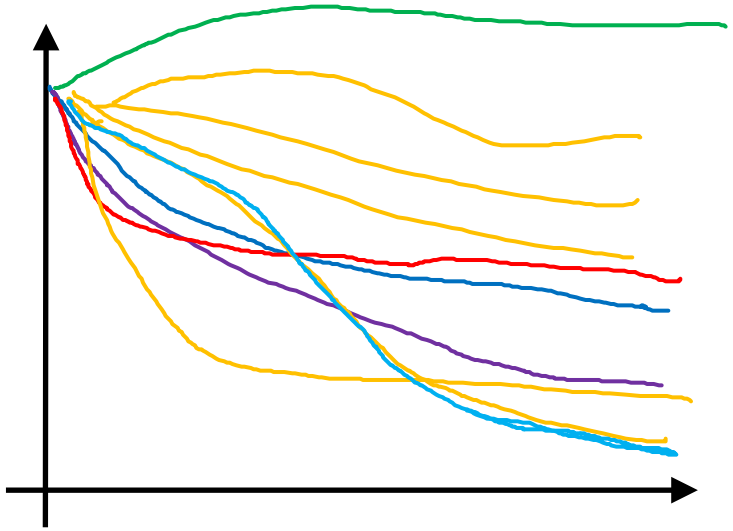
Andrew Ng

Babysitting one model



Panda

Training many models in parallel



Caviar

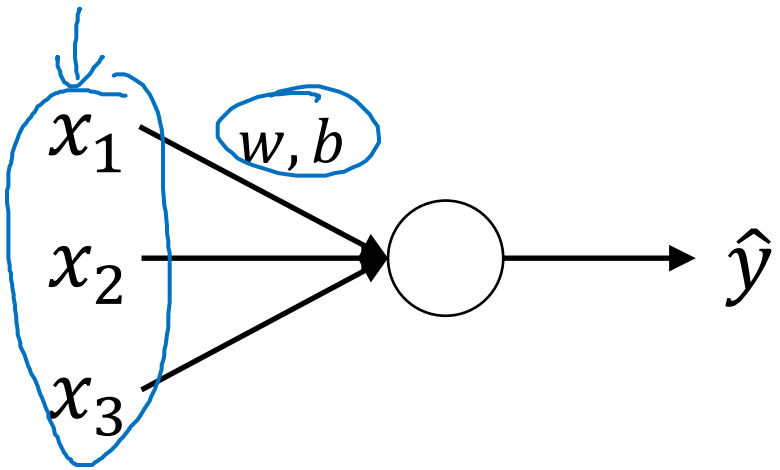
Andrew Ng

deeplearning.ai

Batch Normalization

Normalizing activations in a network

Normalizing inputs to speed up learning

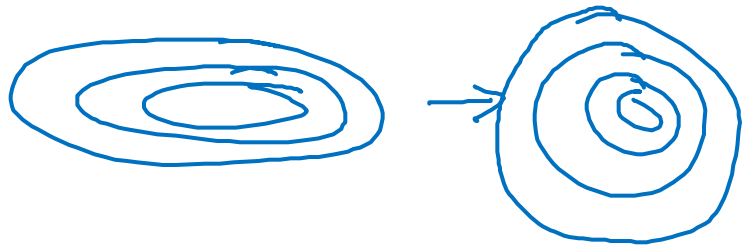


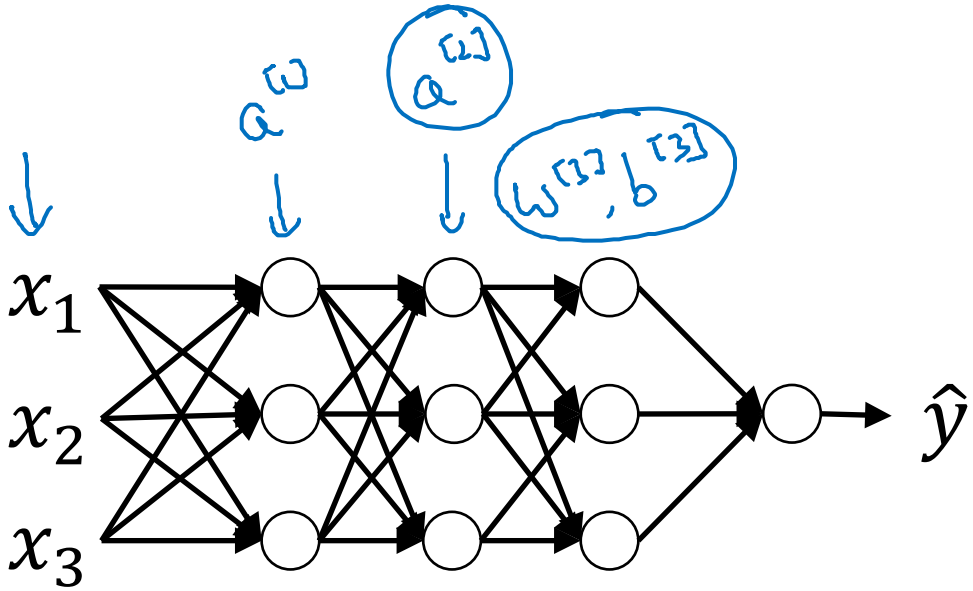
$$\mu = \frac{1}{m}\sum_i x^{(i)}$$

$$X = X - \mu \quad \text{← element-wise}$$

$$\sigma^2 = \frac{1}{m}\sum_i x^{(i)2}$$

$$X = X / \sigma^2$$





Can we normalize $a^{[2]}$ so as to train $w^{[3]}, b^{[3]}$ faster?

Normalize $z^{[2]}$

Andrew Ng

Implementing Batch Norm

Given some intermediate values in NN $z^{(1)}, \ldots, z^{(m)}$ — $z^{[l](i)}$

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

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

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

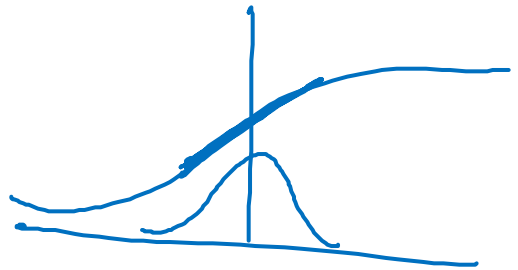
$$\tilde{z}^{(i)} = \gamma z^{(i)}_{norm} + \beta$$

γ, β: learnable parameters of model.

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

$X \leftarrow$

$z^{(i)} \leftarrow$



Use $\tilde{z}^{[l](i)}$ instead of $z^{[l](i)}$.

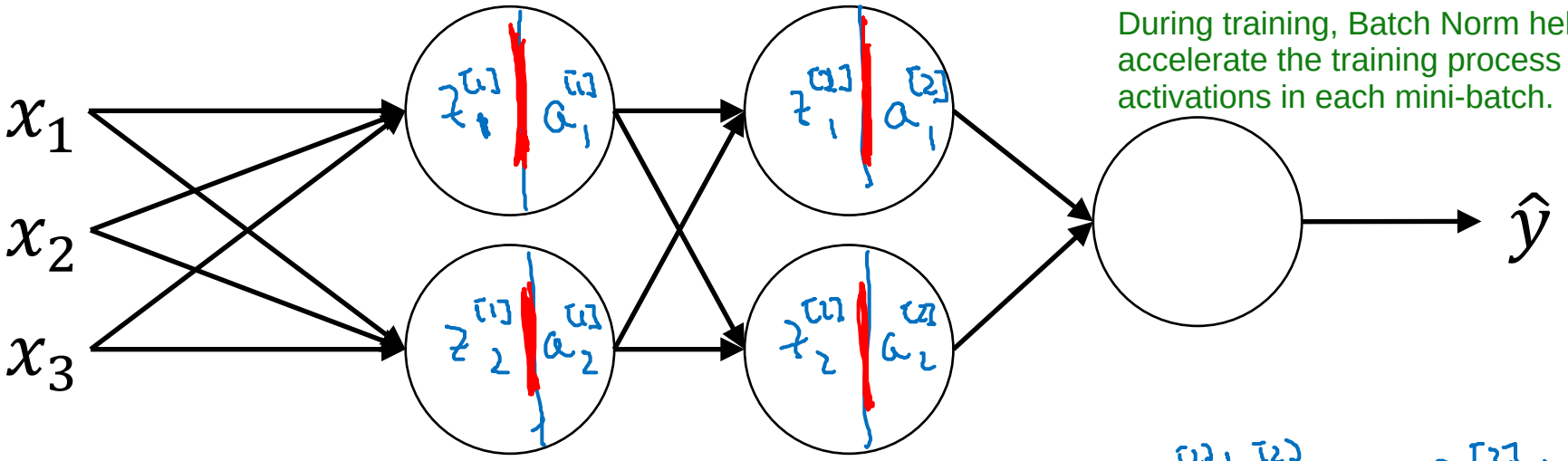
deeplearning.ai

Batch Normalization

Fitting Batch Norm into a neural network

Adding Batch Norm to a network

During training, Batch Norm helps stabilize and accelerate the training process by normalizing the activations in each mini-batch.



$X \rightarrow z^{[1]} \xrightarrow[\text{Batch Norm (BN)}]{\beta^{[1]}, \gamma^{[1]}} \tilde{z}^{[1]} \rightarrow a^{[1]} = g^{[1]}(\tilde{z}^{[1]}) \rightarrow z^{[2]} \xrightarrow[\text{BN}]{\beta^{[2]}, \gamma^{[2]}} \tilde{z}^{[2]} \rightarrow a^{[2]} \rightarrow \dots$

$W^{[1]}, b^{[1]}$ $W^{[2]}, b^{[2]}$

Parameters: $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}, \dots, W^{[L]}, b^{[L]},$

$\beta^{[1]}, \gamma^{[1]}, \beta^{[2]}, \gamma^{[2]}, \dots, \beta^{[L]}, \gamma^{[L]}$

β

Gamma and Beta are added parameters in this list

$d\beta^{[l]}$

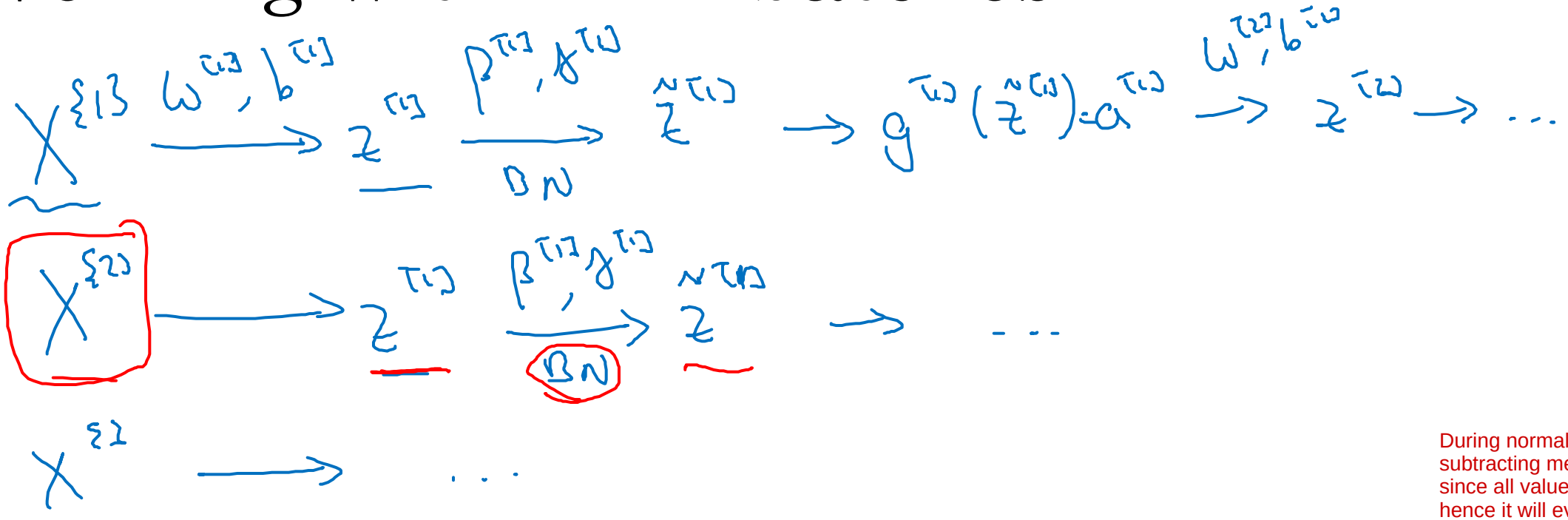
$\beta^{[l]} = \beta^{[l]} - \alpha\, d\beta^{[l]}$

tf.nn.batch_normalization

To implement batch norm in tensorflow

Andrew Ng

Working with mini-batches



Parameters: $w^{[l]}$, ~~$b^{[l]}$~~, $\beta^{[l]}$, $\gamma^{[l]}$.

$(n^{[l]}, 1)$ $(n^{[l]}, 1)$ $(n^{[l]}, 1)$

$z^{[l]}$

$(n^{[l]}, 1)$

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

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

$z^{[l]}_{norm}$

$z^{[l]} = \gamma^{[l]} z^{[l]}_{norm} + \beta^{[l]}$

During normalization, we will be subtracting mean of Z[l] from Z[l], but since all values of b[l] will remain same hence it will eventually cancel out, so no need to store the values of b[L]

Or you may think of it as : Since in the normalization process the values of z[l] are re-centered at the origin, it is irrelevant to add the b[l] parameter.

Andrew Ng

Implementing gradient descent

for $t = 1 \ldots$ numMiniBatches

Compute forward prop on $X^{\{t\}}$.

In each hidden layer, use BN to REPLACE $z^{[l]}$ with $\tilde{z}^{[l]}$.

Use backprop to compute $dW^{[l]}$, ~~$db^{[l]}$~~, $d\beta^{[l]}$, $d\gamma^{[l]}$

Update parameters $W^{[l]} := W^{[l]} - \alpha\, dW^{[l]}$

$\beta^{[l]} := \beta^{[l]} - \alpha\, d\beta^{[l]}$

$\gamma^{[l]} := \ldots$ } ←

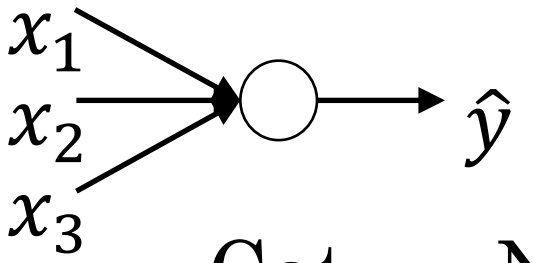
Works w/ momentum, RMSprop, Adam.

deeplearning.ai

Batch Normalization

Why does Batch Norm work?

Learning on shifting input distribution

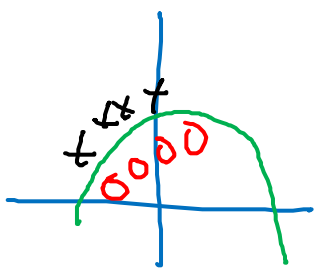
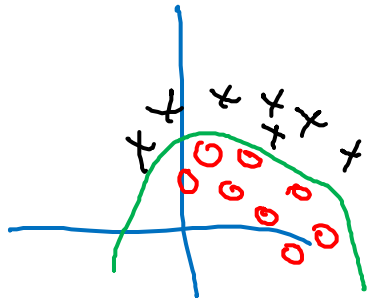


Cat $y = 1$




Non-Cat $y = 0$



It reduces internal covariate shift, making the network more robust and allowing for faster convergence.
The mean and variance statistics are computed per mini-batch, which introduces some randomness.

"Covariate shift"

$X \rightarrow y$

$y = 1$

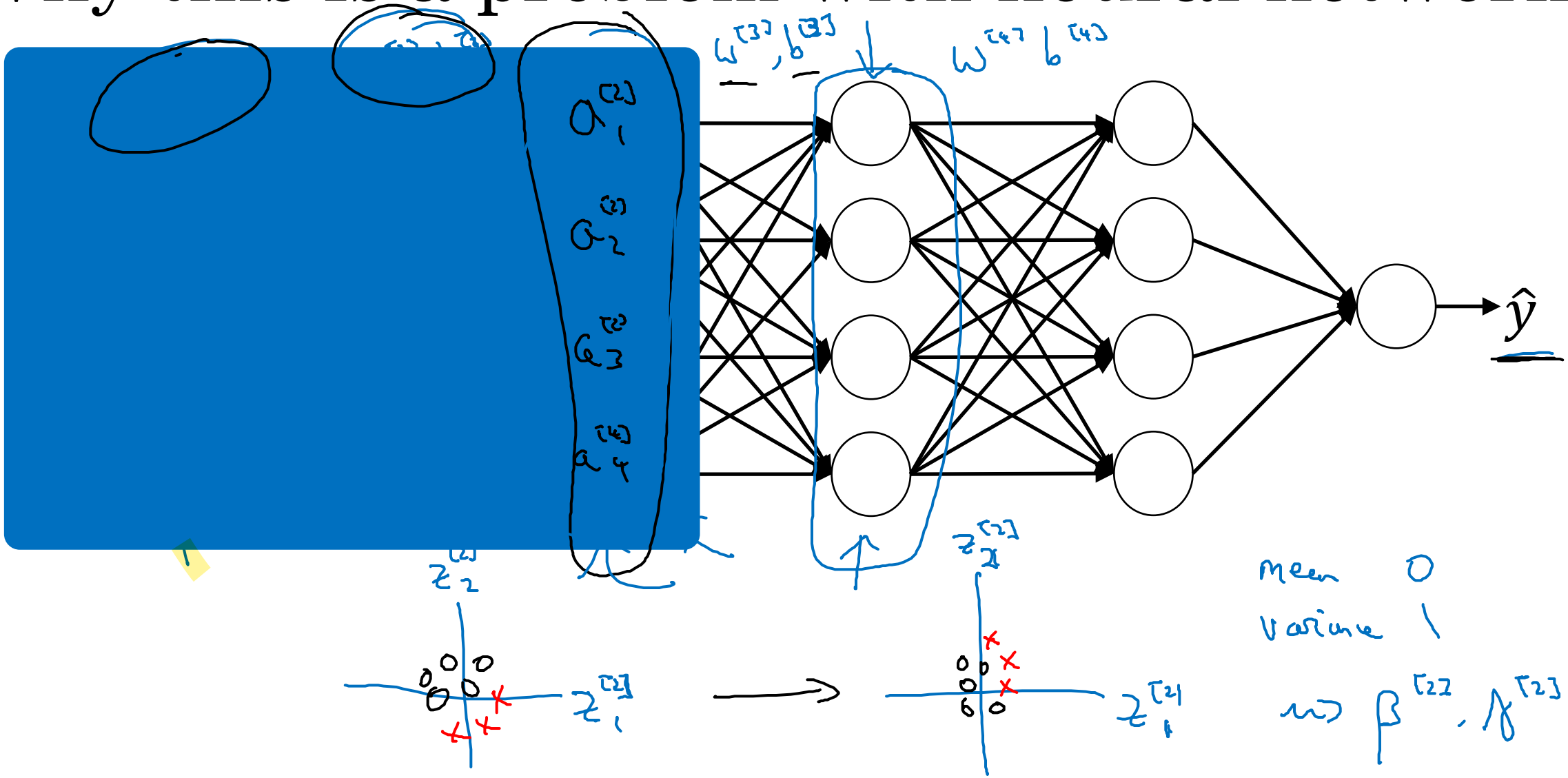



$y = 0$




Andrew Ng

Why this is a problem with neural networks?



Batch Norm as regularization

- 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 minibatch. So similar to dropout, it adds some noise to each hidden layer's activations.
- This has a slight regularization effect.

Test Phase (Inference):

- When evaluating the neural network on unseen data (test/validation set or during deployment), we want consistent and deterministic predictions.

- If Batch Norm were applied during inference, it would use the mini-batch statistics from the test data, leading to different normalization parameters than those used during training.

Solution :

During the test, the parameters u(mue) and sigma^2 are estimated using an exponentially weighted average across mini-batches used during training.

Andrew Ng

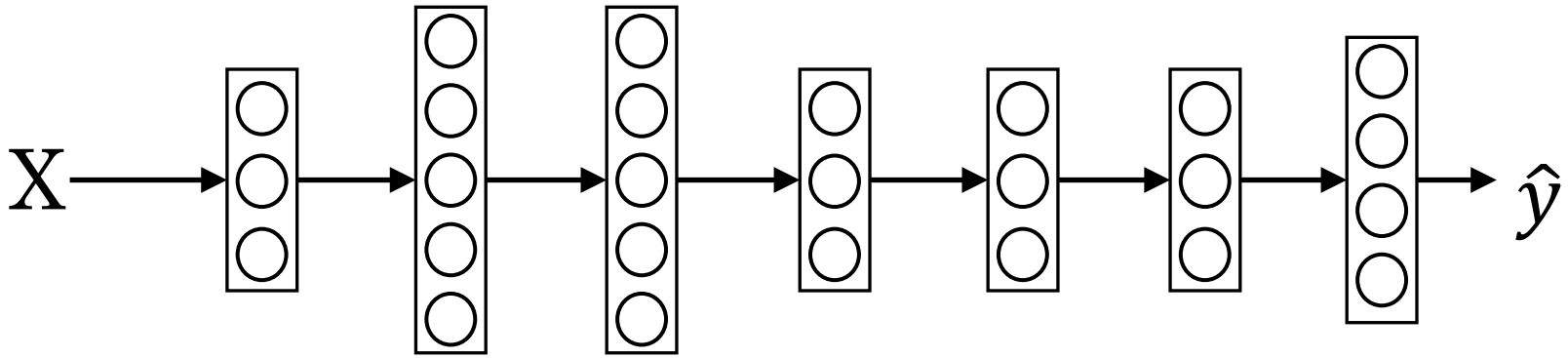
deeplearning.ai

Multi-class classification

Softmax regression

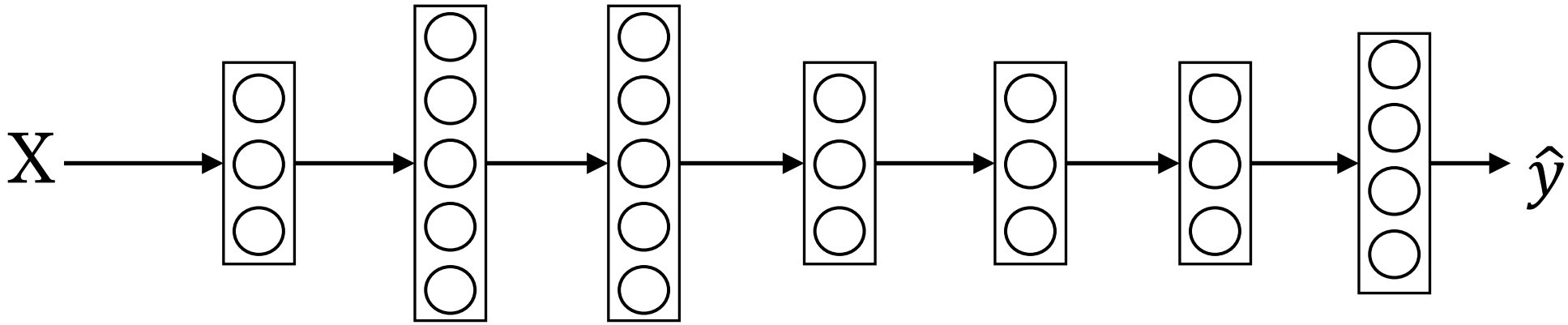
Recognizing cats, dogs, and baby chicks

3 1 2 0 3 2 0 1

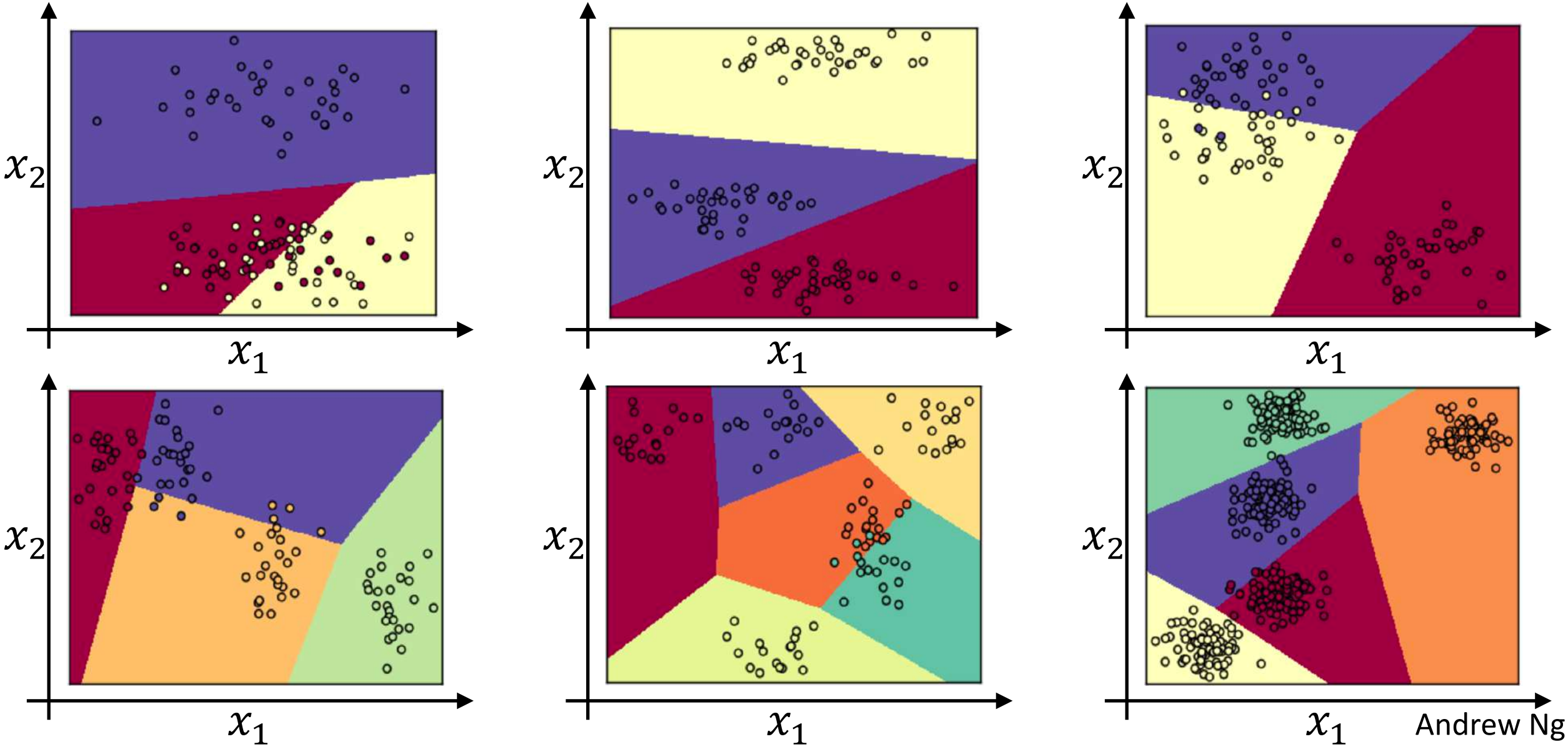


Andrew Ng

Softmax layer



Softmax examples



deeplearning.ai

Programming Frameworks

Deep Learning frameworks

Deep learning frameworks

- Caffe/Caffe2
- CNTK
- DL4J
- Keras
- Lasagne
- mxnet
- PaddlePaddle
- TensorFlow
- Theano
- Torch

Choosing deep learning frameworks

- Ease of programming (development and deployment)
- Running speed
- Truly open (open source with good governance)

deeplearning.ai

Programming Frameworks

TensorFlow

Motivating problem

$J(w) = w^2 - 10w + 25$

(cost)

$(w-5)^2$

$w = 5$

$J(w, b)$