Batch
Normalization
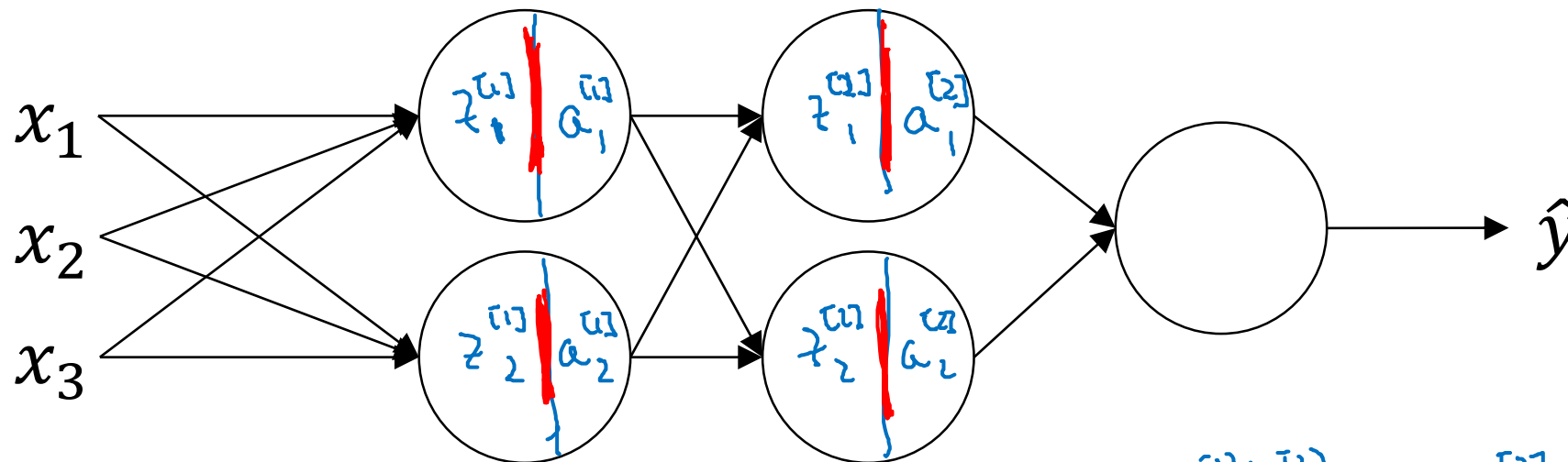
deeplearning.ai

Fitting Batch Norm
into a neural network

Adding Batch Norm to a network

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

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

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

$d\beta^{[L]}$

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

tf.nn.batch-normalization

$\rightarrow \beta$

Andrew Ng

Working with mini-batches

$$X^{\{1\}} \xrightarrow{W^{[1]}, b^{[1]}} Z^{[1]} \xrightarrow[BN]{\beta^{[1]}, \gamma^{[1]}} \tilde{Z}^{[1]} \rightarrow g^{[1]}(\tilde{Z}^{[1]}) = a^{[1]} \xrightarrow{W^{[2]}, b^{[2]}} Z^{[2]} \rightarrow \ldots$$

$$\boxed{X^{\{2\}}} \longrightarrow Z^{[1]} \xrightarrow[\boxed{BN}]{\beta^{[1]}, \gamma^{[1]}} \tilde{Z}^{[1]} \rightarrow \ldots$$

$$X^{\{3\}} \longrightarrow \ldots$$

$$\text{Parameters:} \quad W^{[l]}, \cancel{b^{[l]}}, \beta^{[l]}, \gamma^{[l]}.$$

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

$$\rightarrow Z^{[l]} = W^{[l]} a^{[l-1]} + \boxed{\cancel{b^{[l]}}}$$

$$Z^{[l]} = W^{[l]} a^{[l-1]}$$

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

$$\rightarrow \tilde{Z}^{[l]} = \gamma^{[l]} Z^{[l]}_{norm} + \boxed{\beta^{[l]}} \leftarrow$$

$$Z^{[l]}$$

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

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 $dW^{[l]}, \cancel{db^{[l]}}, d\beta^{[l]}, d\gamma^{[l]}$

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

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

$\gamma^{[l]} := \ldots$

$\Leftarrow$

Works w/ momentum, RMSprop, Adam.

Andrew Ng