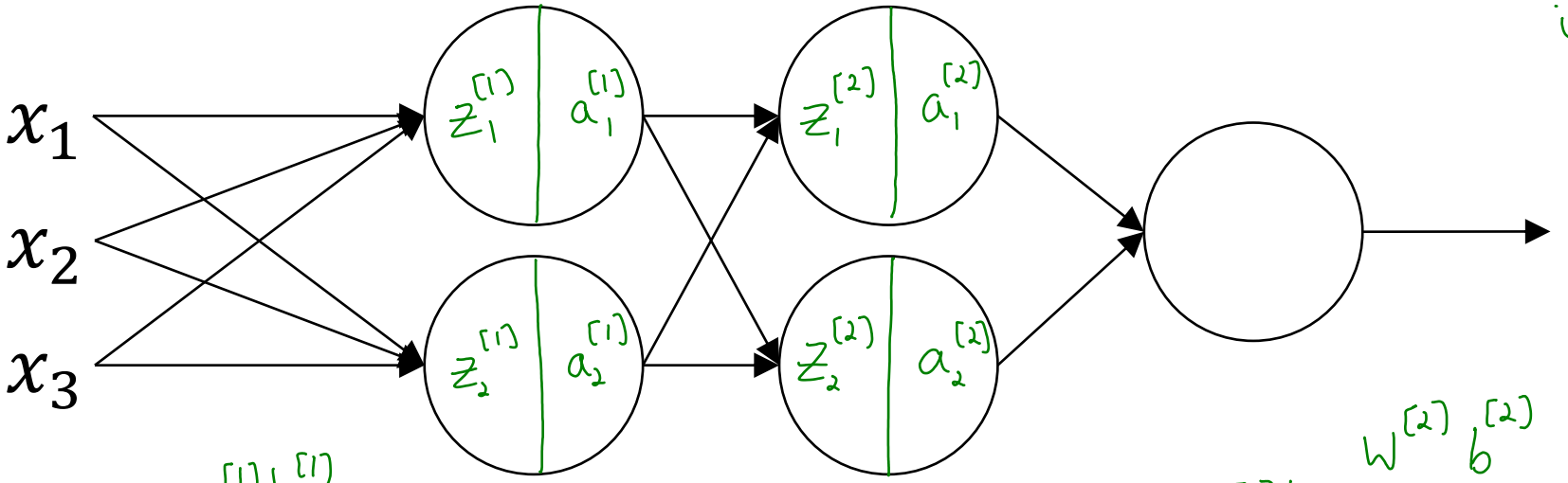
deeplearning.ai

Batch Normalization

Fitting Batch Norm into a neural network

Adding Batch Norm to a network

- performing batch normalization to inner Hidden units



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

$a^{[2]} = g^{[2]}(\tilde{z}^{[2]})$

- Basically, Batch norm happens b/w z & a's computation

Params: $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]} \ldots W^{[L]}, b^{[L]}$] For finding z

$\beta^{[1]}, \gamma^{[1]}, \beta^{[2]}, \gamma^{[2]} \ldots \beta^{[L]}, \gamma^{[L]}$] For finding $\tilde{z}$

β here has nothing to do w/ b in Gradient descent (Adam/RMS etc)

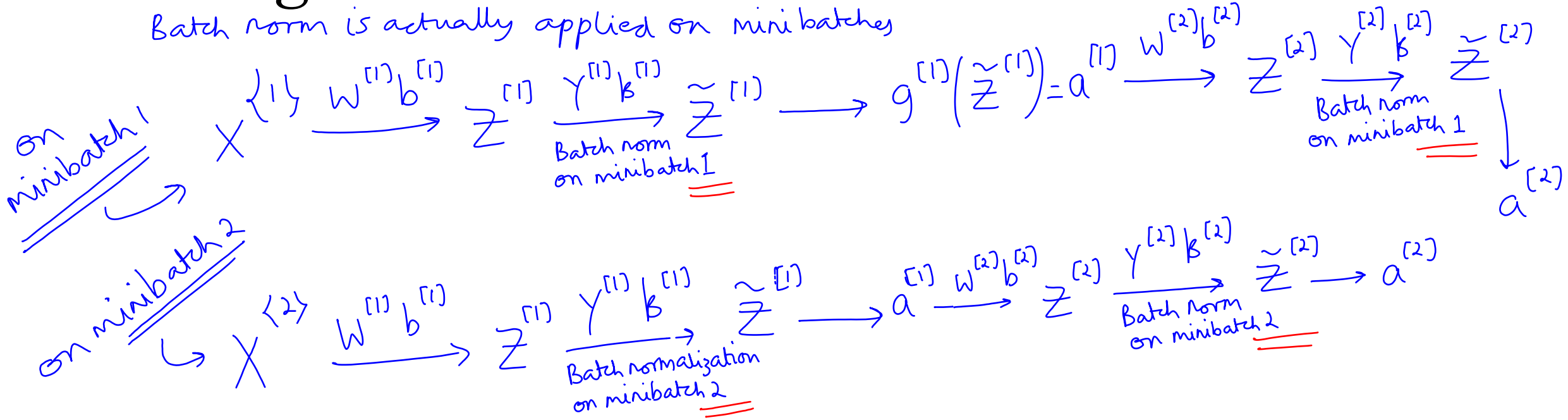
∴ Now we will have to find $d\beta$ & $d\gamma$ & update the model

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

$\gamma^{[l]} = \gamma^{[l]} - \alpha \cdot d\gamma^{[l]}$

Andrew Ng

Working with mini-batches



There is one catch

We said Params were $W^{[l]}, b^{[l]}, \beta^{[l]}, \gamma^{[l]}$

In reality, it is $W^{[l]}, \beta^{[l]}, \gamma^{[l]}$ (no $b^{[l]}$) - why?

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

Now when we find $Z^{[l]}_{norm}$, we do $Z^{[l]}_{norm} = \frac{Z^{[l]} - \mu}{\sigma^2}$

where $\mu = \frac{1}{m}\sum_{i=1}^{m} Z^{(i)}$ for layer l

$$\mu = \frac{1}{m}\sum_{i=1}^{m}\left(W^{[l]} a^{[l-1]} + b^{[l]}\right)$$

$$\mu = \frac{1}{m}\sum_{i=1}^{m}\left(W^{[l]} a^{[l-1]}\right) + \frac{1}{m} \times m\, b^{[l]}$$

∴ when we do $Z^{[l]}_{norm}$, each $Z^{[l]}$ has a $b^{[l]}$ component which cancels out because we do $Z^{[l]} - \mu$

∴ No need of optimizing for $b^{[l]}$, which is never used, now from $Z^{[l]}_{norm}$, we find $\tilde{Z}^{[l]}$

dim $b^{[l]}$ & $\gamma^{[l]}$ $= (n^{[l]}, 1)$

Why?

$\tilde{Z}^{[l]} = \gamma^{[l]} \cdot Z^{[l]}_{norm} + b^{[l]}$

$Z^{[l]}_{norm}$ is same as $Z^{[l]}$

$Z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]}$ → dim $= (n^{[l]} \times 1)$

Andrew Ng

Implementing gradient descent

for $t = 1$ to Minibatches

compute fwd prop on $X^{\{t\}}$

In each hidden layer, use batch norm to replace $z^{[l]}$ with $\tilde{z}^{[l]}$

use backprop, to find $dw^{[l]}$, ~~$db^{[l]}$~~, $d\beta^{[l]}$, $dy^{[l]}$

update the params of Gradient descent

$$w^{[l]} = w^{[l]} - \alpha \cdot dw^{[l]}$$
$$\beta^{[l]} = \beta^{[l]} - \alpha \cdot d\beta^{[l]}$$
$$y^{[l]} = y^{[l]} - \alpha \cdot dy^{[l]}$$

- No need to find $b^{[l]}$
- Can use RMS prop / Adam to update weights.