Batch normalization

- Normalize values on layers (Z-score normalization) => Train Weights and biaises faster

- For each layer
  - Mean $\mu = \frac{1}{m}\sum_{i=1}^{m} z^{(i)}$
  - Standard deviation $\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}\left(z^{(i)} - \mu\right)^2$
  - Normalized value: $Z_{norm}^{(i)} = \frac{z^{(i)}-\mu}{\sqrt{\sigma^2+\epsilon}}$ with $\mu_{norm} = 0$ and $\sigma_{norm} = 1$
  - The ε is for numerical stability and to avoid dividing by 0 in some cases
  - $\tilde{z}^{(i)} = \gamma Z_{norm}^{(i)} + \beta$ ($\tilde{z}^{(i)}$ is used in algorithm to compute $a^{(i)} = g(\tilde{z}^{(i)})$)

Learnable parameters and some tips

- $w^{[l]}, b^{[l]}, \beta^{[l]}, \gamma^{[l]}$ by gradient descent optimization or ADAM optimization
- Each mini batch should have its unique batch norm
- Z[l] will have some noise = dropout noise => Regularisation effect
- To reduce noise => Bigger mini-batch size (2^n)
- At test time => Single example at a time

Multi class classification

- Use softmax function in last layer or outputlayer
- Softmax give probability of prediction for each class
- Softmax function

$$a^{[L]} = \frac{e^{z^{[L]}}}{\sum_{i=1}^{C} t_i} \text{ with C number of classes}$$

$$t = e^{z^{[L]}} \text{ element wize exponential}$$

Softmax classifier

- Train it:

$$z^{[L]} \Rightarrow t \Rightarrow g^{[L]}(.) = \frac{t}{\sum_{i=1}^{C} t_i} \Rightarrow a^{[L]}$$

- Loss function

$$L(\hat{y}, y) = -\sum_{j=1}^{C} y_j \log(\hat{y}_j)$$

$$J\left(w^{[i]}, b^{[i]}\right) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$