Neural Network Regularization
Batch Norm

Pavlos Protopapas

Feature Normalization

Good practice to normalize features before applying learning algorithm:

Feature vector

Vector of mean feature values

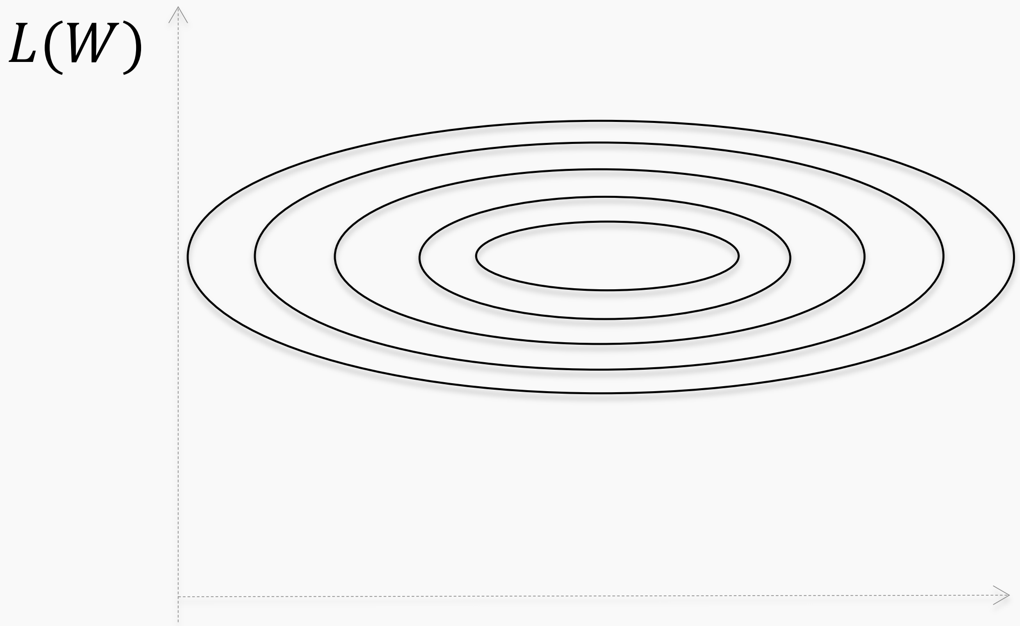
$$\boxed{x' = \frac{x - \mu}{\sigma}}$$

Vector of SD of feature values

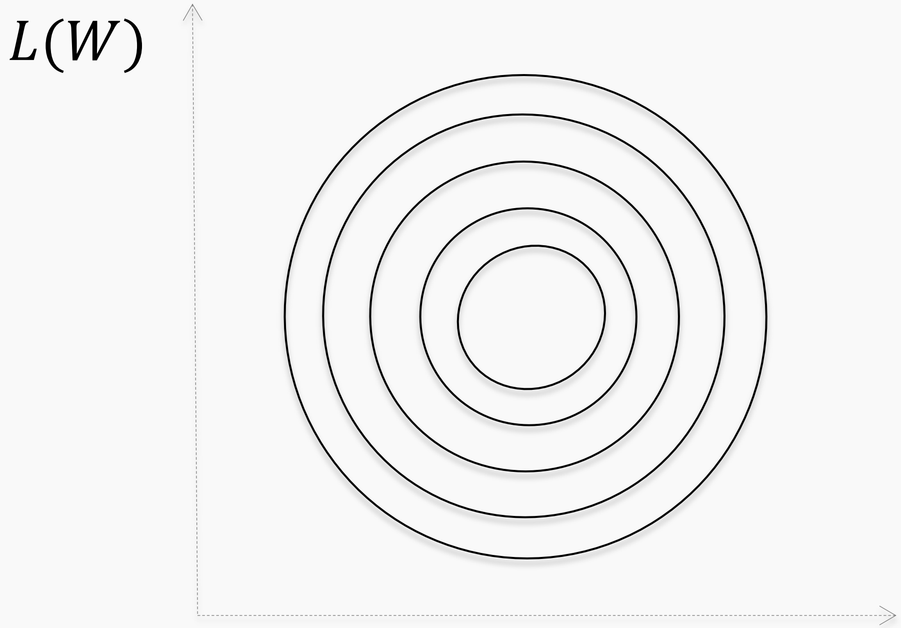
Features in same scale: mean 0 and variance 1

Feature Normalization

Speeds up learning



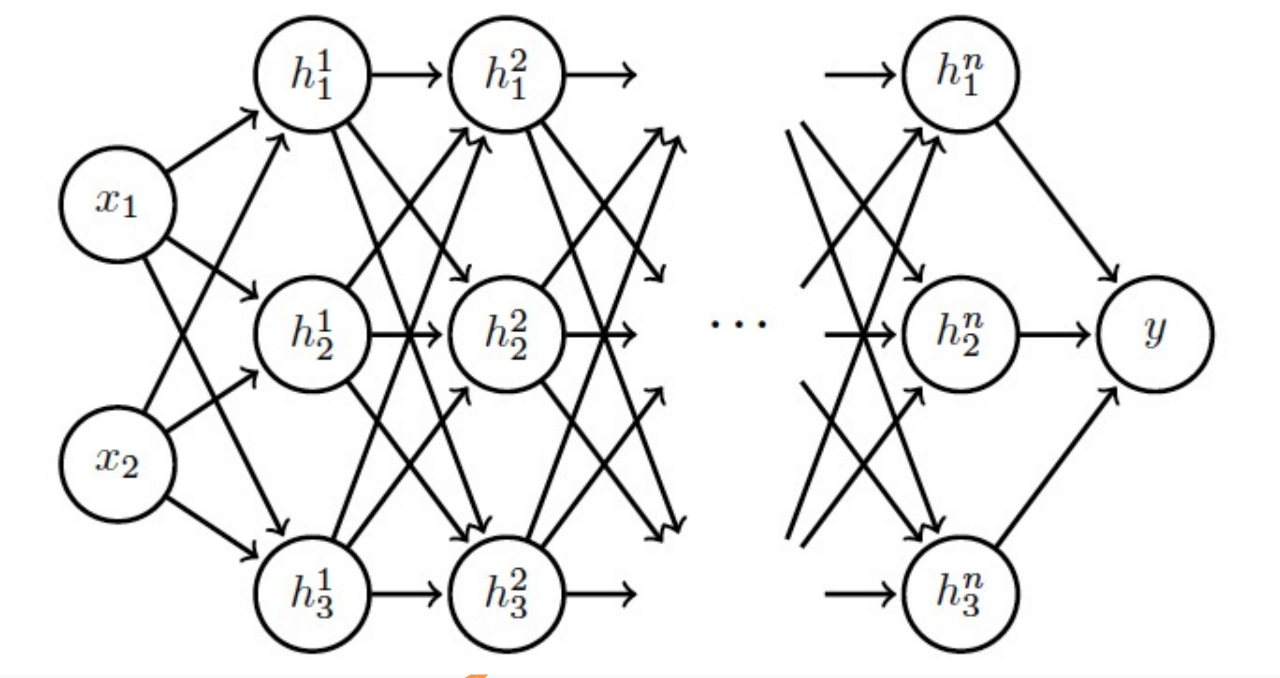
Before normalization



After normalization

Internal Covariance Shift

Each hidden layer changes distribution of inputs to next layer: *slows down learning*



Normalize inputs to layer 2

…

Normalize inputs to layer n

Batch Normalization

Training time:

Batch of activations for a layer to normalize

For a given hidden layer

$$H = \begin{bmatrix} H_{11} & \cdots & H_{1K} \\ \vdots & \ddots & \vdots \\ H_{N1} & \cdots & H_{NK} \end{bmatrix}$$

N data points in batch

K hidden units activations

Batch Normalization

Training time:

Batch of activations for a layer to normalize

$$H = \begin{bmatrix} \boxed{\begin{matrix} H_{11} \\ \vdots \\ H_{N1} \end{matrix}} & \begin{matrix} \cdots \\ \ddots \\ \cdots \end{matrix} & \begin{matrix} H_{1K} \\ \vdots \\ H_{NK} \end{matrix} \end{bmatrix}$$

$$H'_{ik} = \frac{H_{ik} - \mu_k}{\sigma_k}$$

Batch Normalization

Training time:

Mini-batch of activations for a layer to normalize

$$H = \begin{bmatrix} \boxed{\begin{matrix} H_{11} \\ \vdots \\ H_{N1} \end{matrix}} & \begin{matrix} \cdots \\ \ddots \\ \cdots \end{matrix} & \begin{matrix} H_{1K} \\ \vdots \\ H_{NK} \end{matrix} \end{bmatrix}$$

$$H'_{ik} = \frac{H_{ik} - \mu_k}{\sigma_k}$$

$$\mu_k = \frac{1}{N} \sum_i H_{ik}$$

Mean activations across mini-batch for node k.

Batch Normalization

Training time:

Mini-batch of activations for a layer to normalize

$$H = \begin{bmatrix} H_{11} & \cdots & H_{1K} \\ \vdots & \ddots & \vdots \\ H_{N1} & \cdots & H_{NK} \end{bmatrix}$$

$$H'_{ik} = \frac{H_{ik} - \mu_k}{\sigma_k}$$

$$\mu_k = \frac{1}{N}\sum_i H_{ik}$$

Mean activations across mini-batch for node k.

$$\sigma_k = \frac{1}{N}\sum_i (H_{ik} - \mu_k)^2 + \delta$$

SD of each unit across mini-batch

Batch Normalization

Training time:

Normalization can reduce expressive power

Instead use:

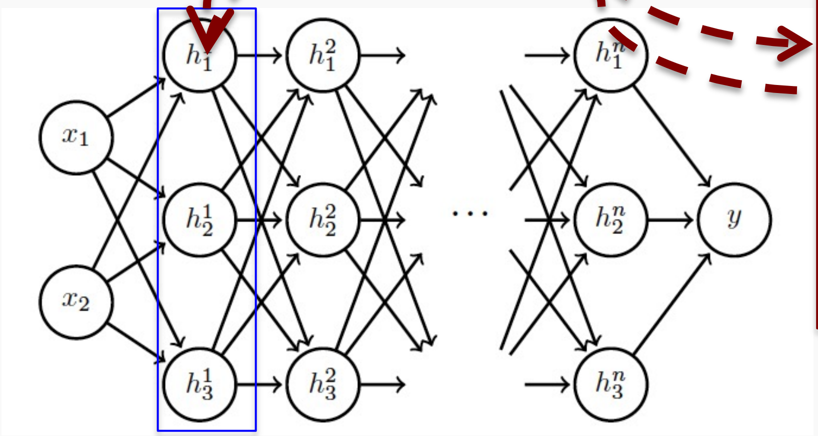
$$\boxed{H'_{ik} = \gamma H'_{ik} + \beta}$$

Learnable parameters

Allows network to control range of normalization

Batch Normalization

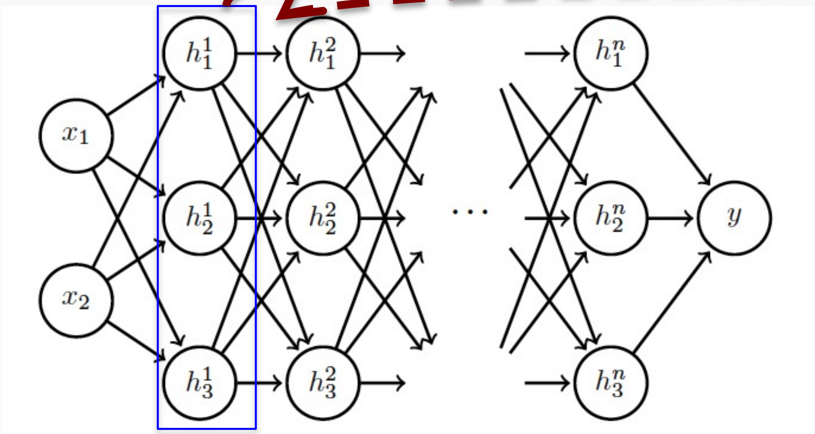
Batch 1



$$\mu^1 = \frac{1}{m}\sum_i H_{i,:}$$

$$\sigma^1 = \sqrt{\frac{1}{m}\sum_i (H-\mu)_i^2 + \delta}$$

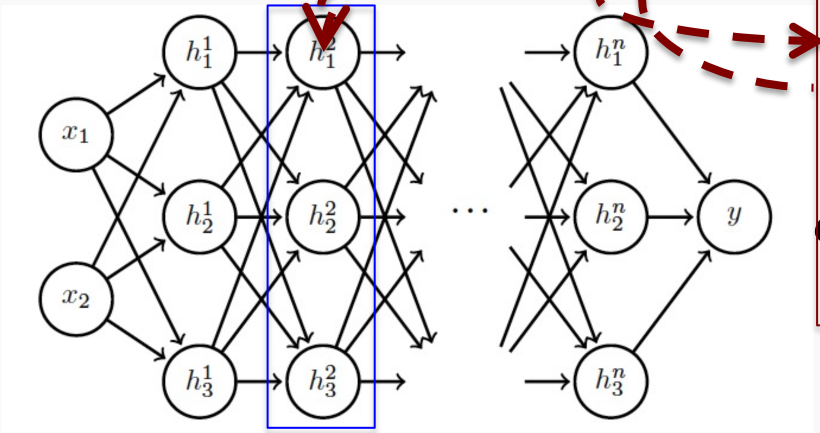
Batch *N*



Add normalization operations for layer 1

Batch Normalization

Batch 1

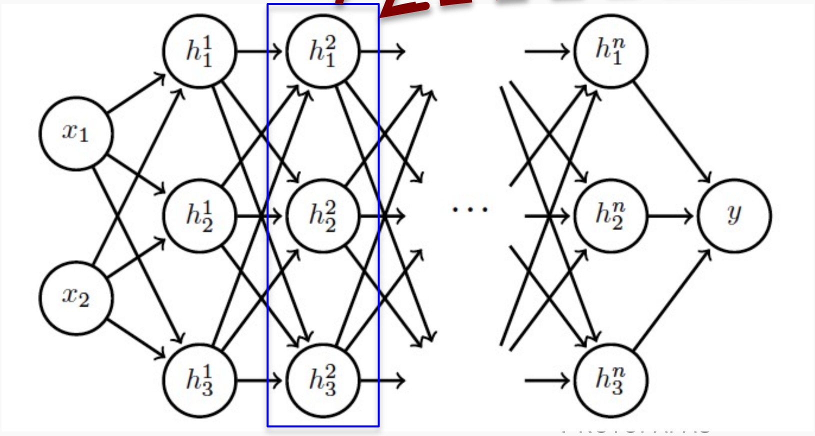


$$\mu^2 = \frac{1}{m}\sum_i H_{i,:}$$

$$\sigma^2 = \sqrt{\frac{1}{m}\sum_i (H-\mu)_i^2 + \delta}$$

⋮

Batch *N*



Add normalization operations for layer 2 and so on …

Evaluation

We saw how batch normalization works during training, but what about evaluation phase when we do not have a complete batch!

Evaluation

?

- Store the different means and standard deviations calculated during training .
- Calculate the average mean and standard deviation.

$$\mu_{global} = \frac{\mu_{batch1} + \mu_{batch2} + \ \dots\dots\dots + \mu_{batch\,n}}{n}$$

Use this for evaluation

$$\sigma_{global} = \frac{\sigma_{batch1} + \sigma_{batch2} + \ \dots\dots\dots + \sigma_{batch\,n}}{n}$$