# Neural Network Regularization

## Part C - Batch Normalization

CS1090B Data Science II – Spring 2025

Pavlos Protopapas, Natesh Pillai, and Chris Gumb

Isabela Yepes
Jay Peak, Vermont

# Feature Normalization

It is a good practice to normalize features before applying the learning algorithm:

Feature vector

Vector of mean feature values

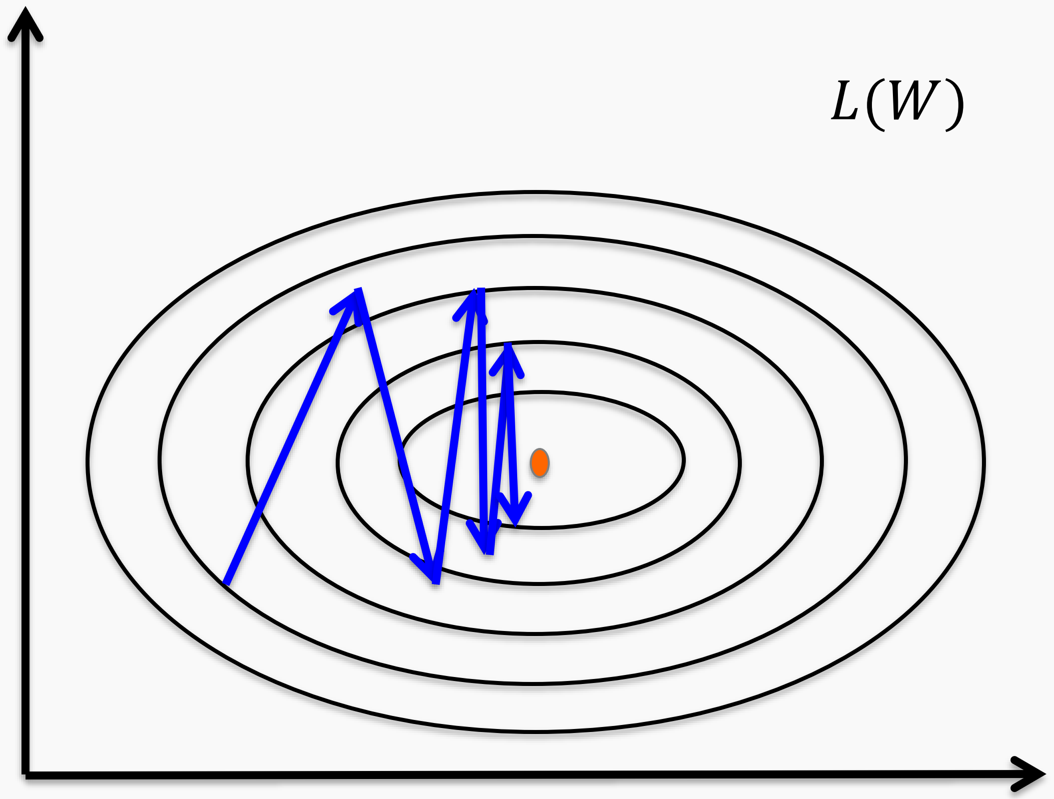
$$\boxed{x' = \frac{x - \mu}{\sigma}}$$

Vector of SD of feature values

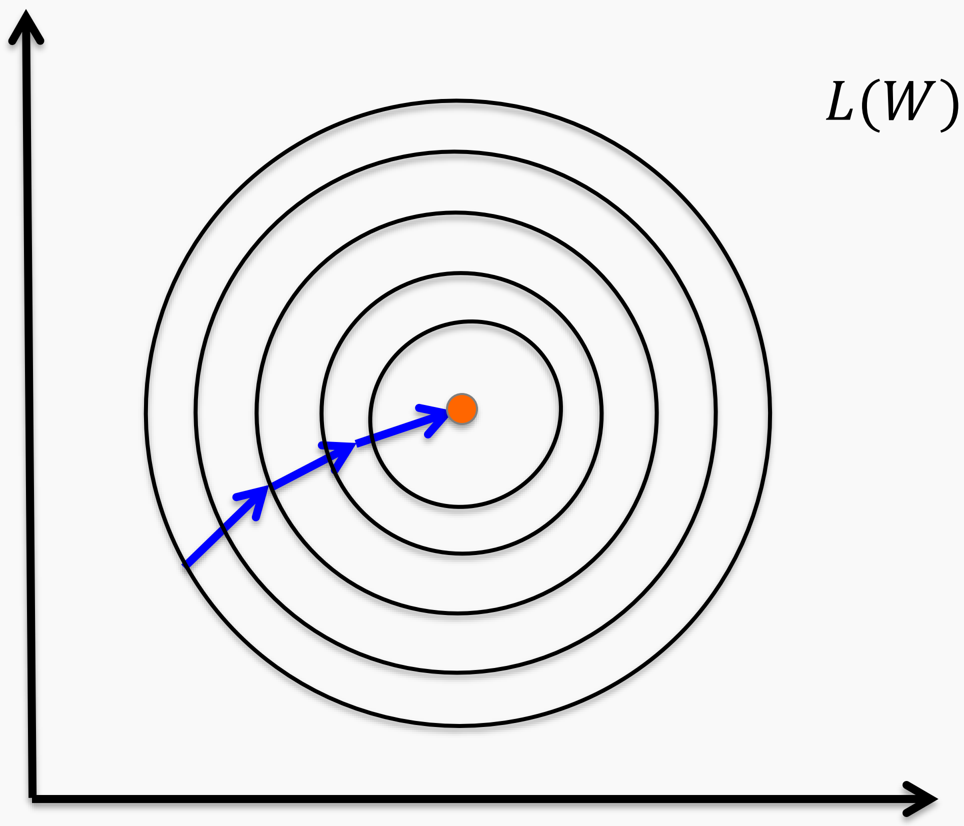
Features in the same scale: mean 0 and variance 1

## Feature Normalization

Speeds up learning



Before normalization



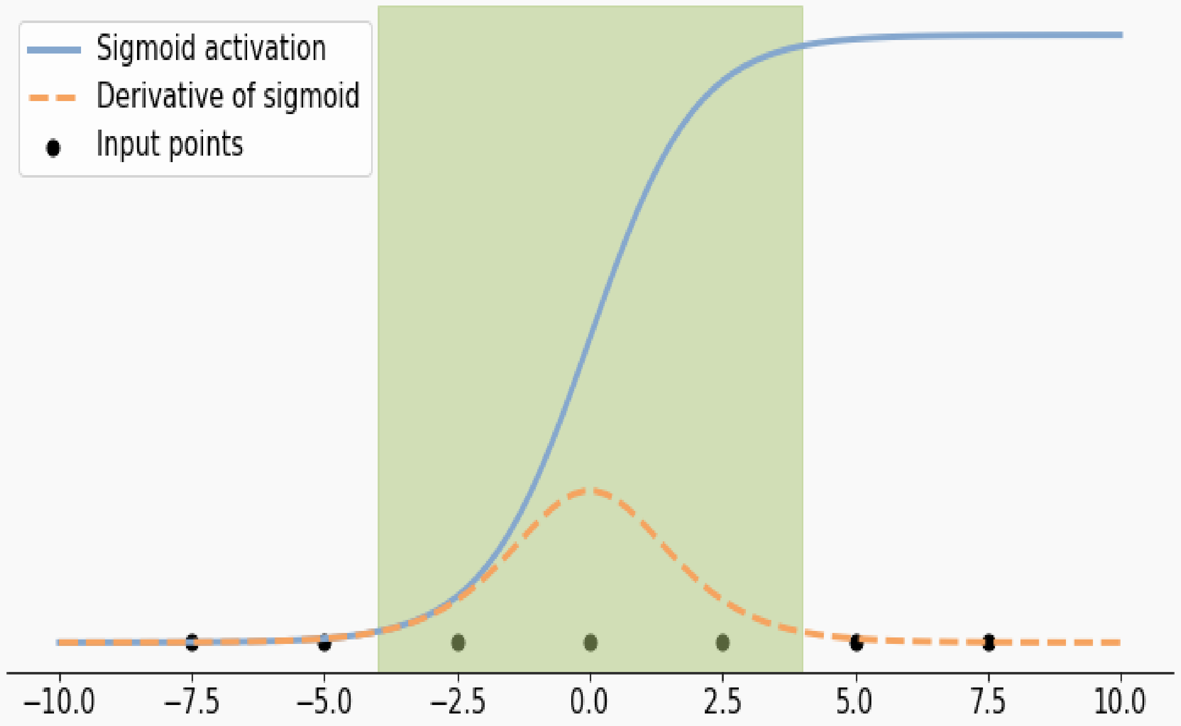
After normalization

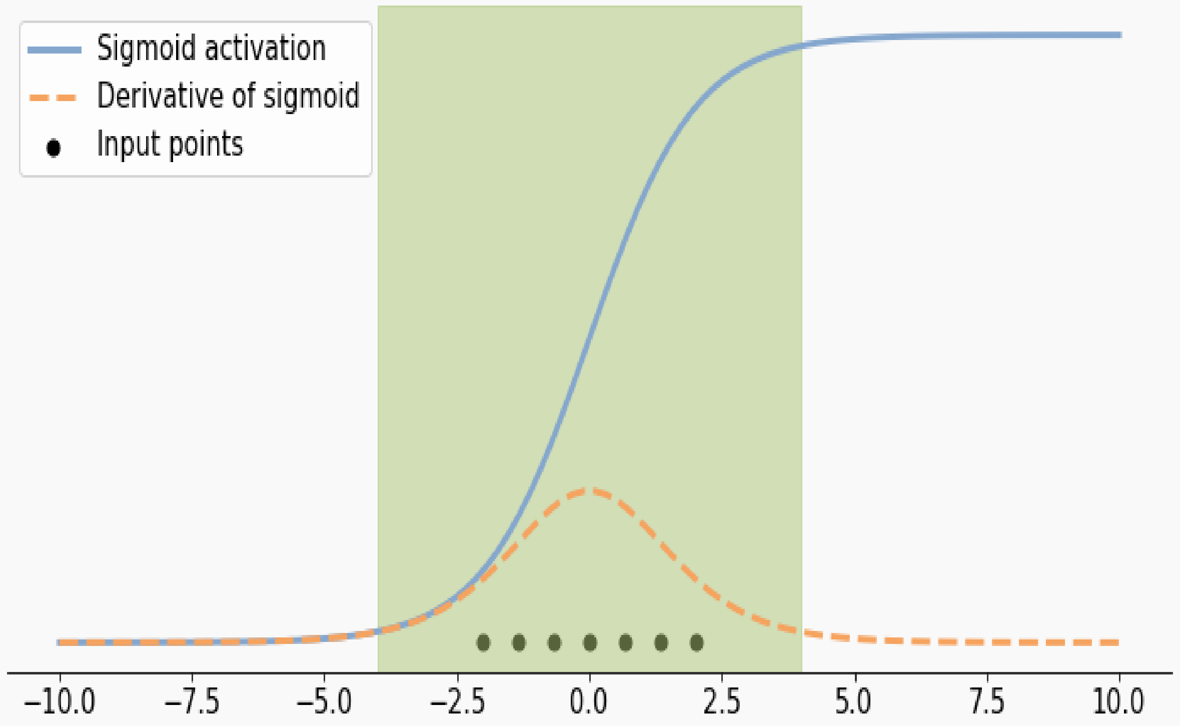
**Note:** This is an ideal case scenario. In reality, the loss landscapes are much more complex.

# How do neural networks learn efficiently?

The distribution that is fed to the layers of a network should be somewhat:

- Normalized - to avoid vanishing gradients





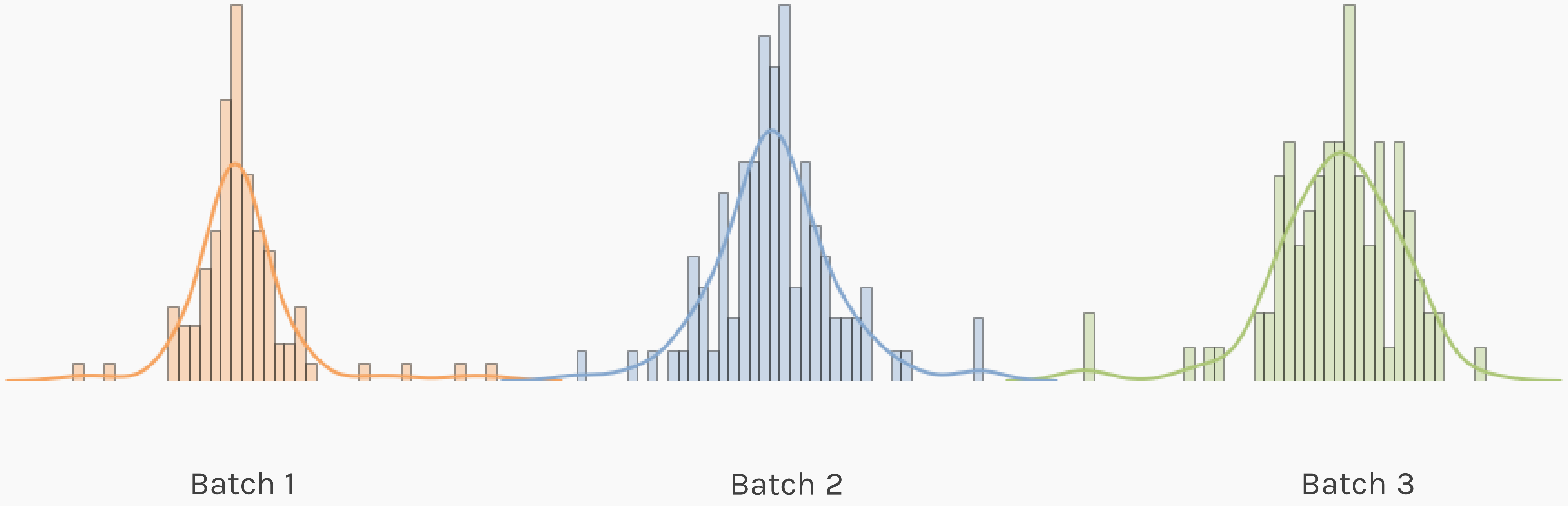
## How do neural networks learn efficiently?

The distribution that is fed to the layers of a network should be somewhat:

- Normalized - to avoid vanishing gradients
- Constant through epochs or batches and data

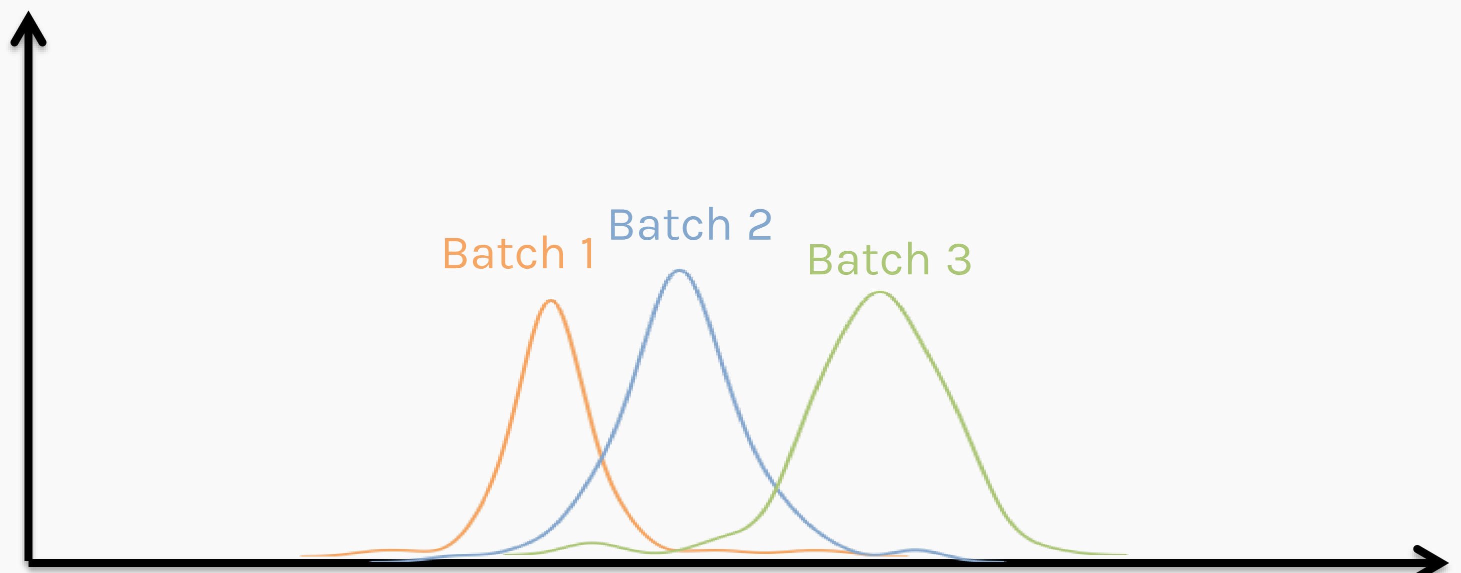
## Internal Covariance Shift (ICS)

ICS occurs when the input distribution to the hidden layers (hence “internal”) of the neural network end up fluctuating or shifting.



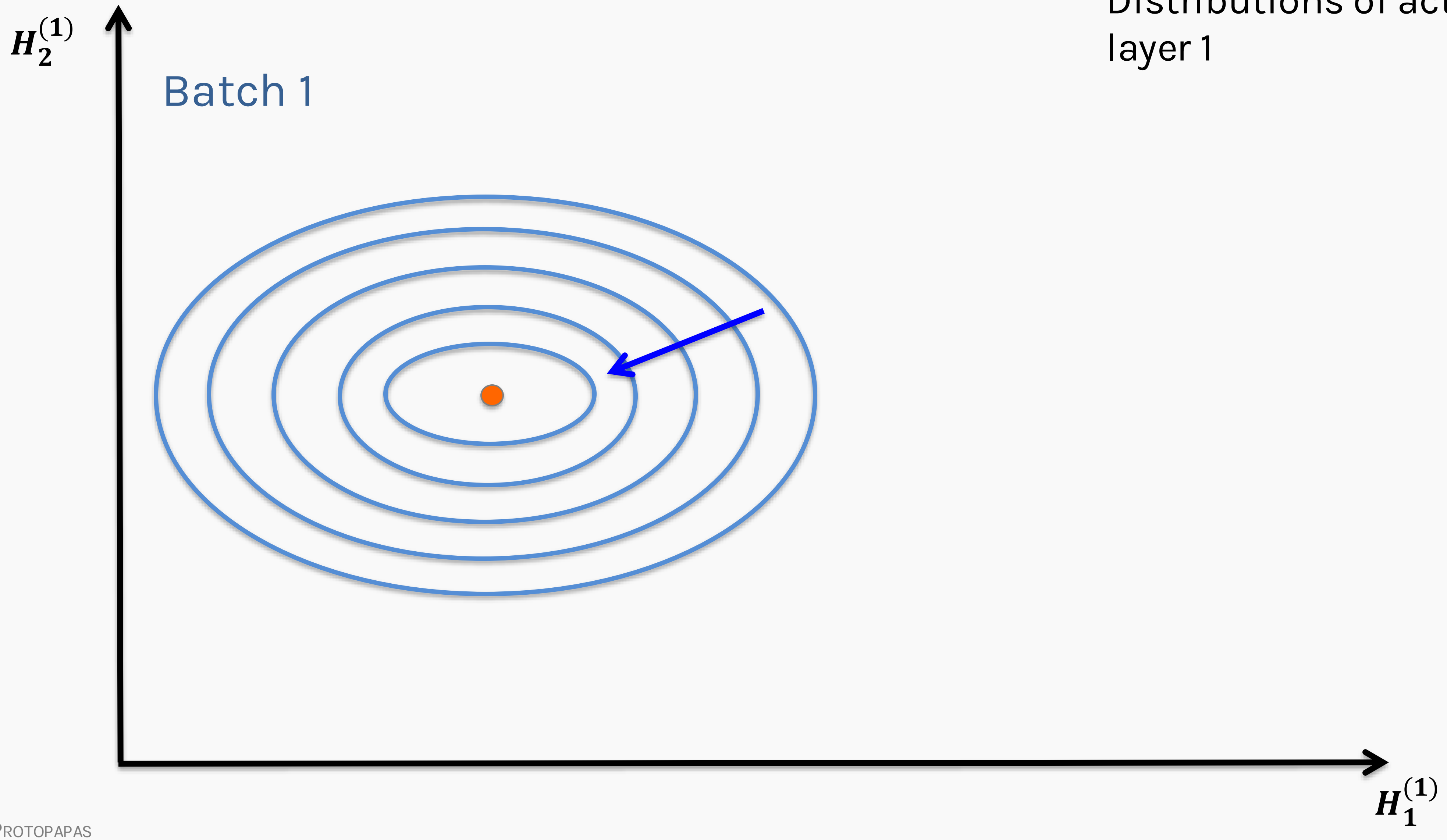
## Internal Covariance Shift (ICS)

ICS occurs when the input distribution to the hidden layers (hence “internal”) of the neural network end up fluctuating or shifting.



Distribution of the layer outputs across batches

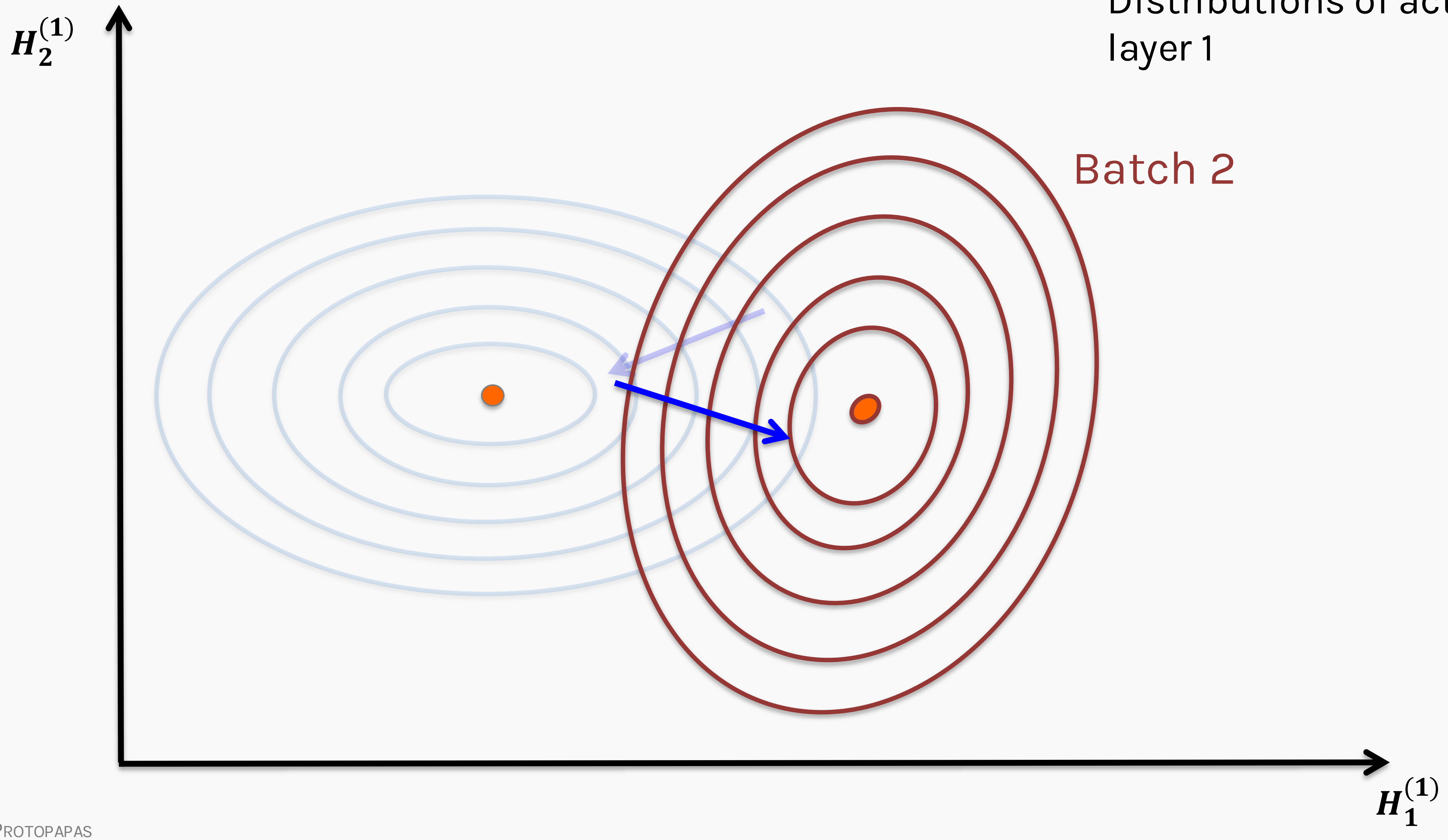
# Internal Covariance Shift



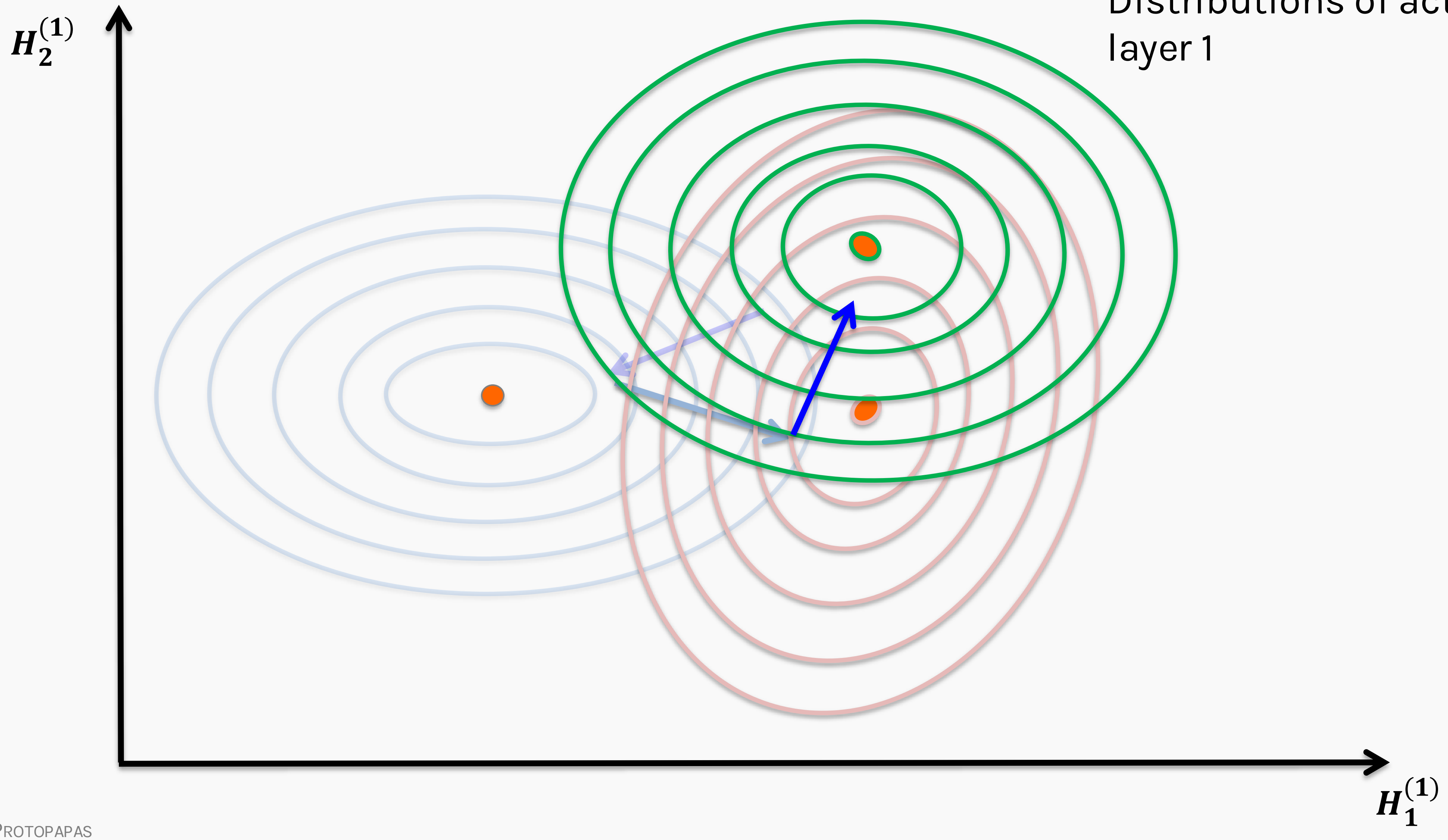
Distributions of activations of hidden layer 1

## Internal Covariance Shift

Distributions of activations of hidden layer 1



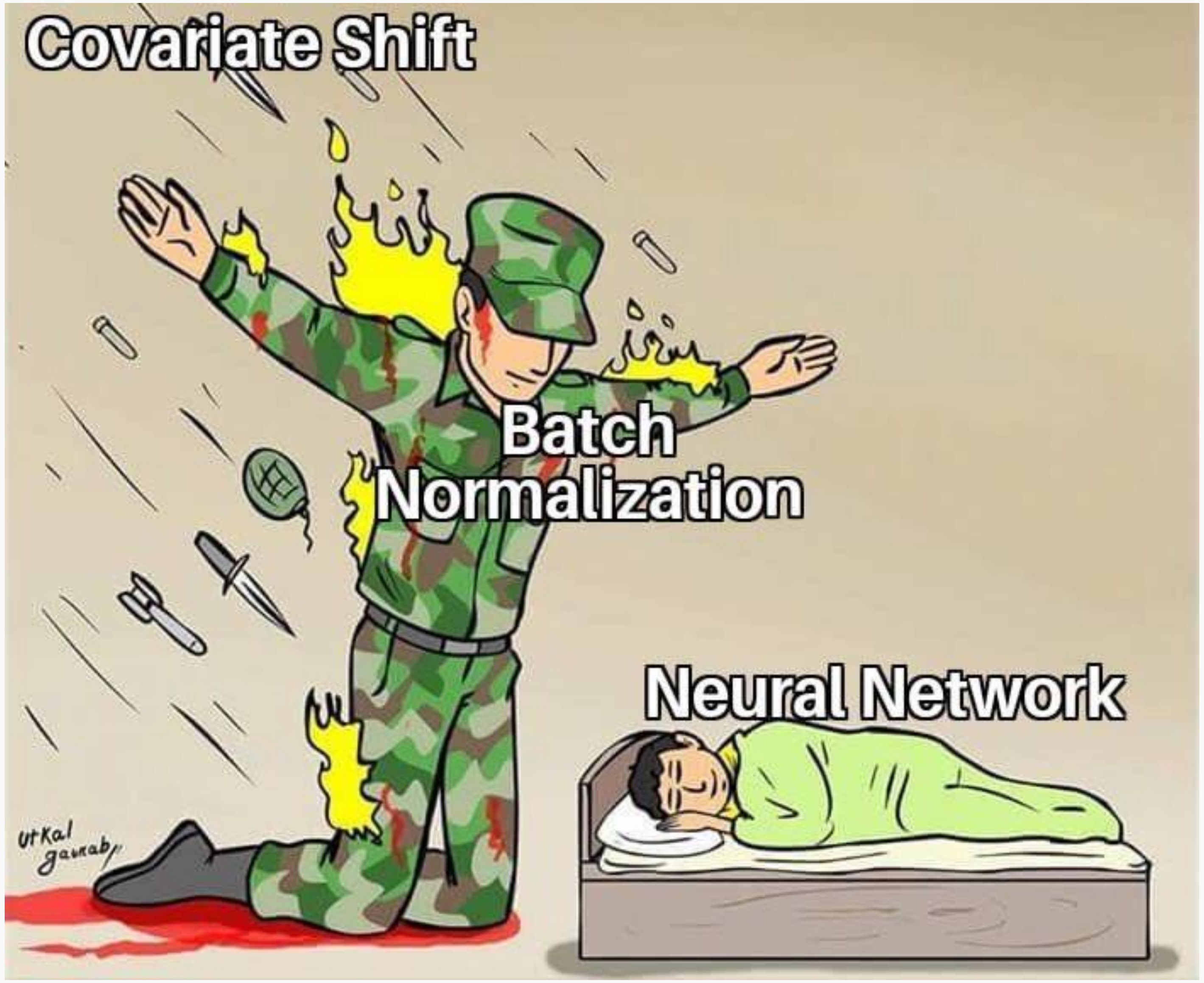
## Internal Covariance Shift



Distributions of activations of hidden layer 1

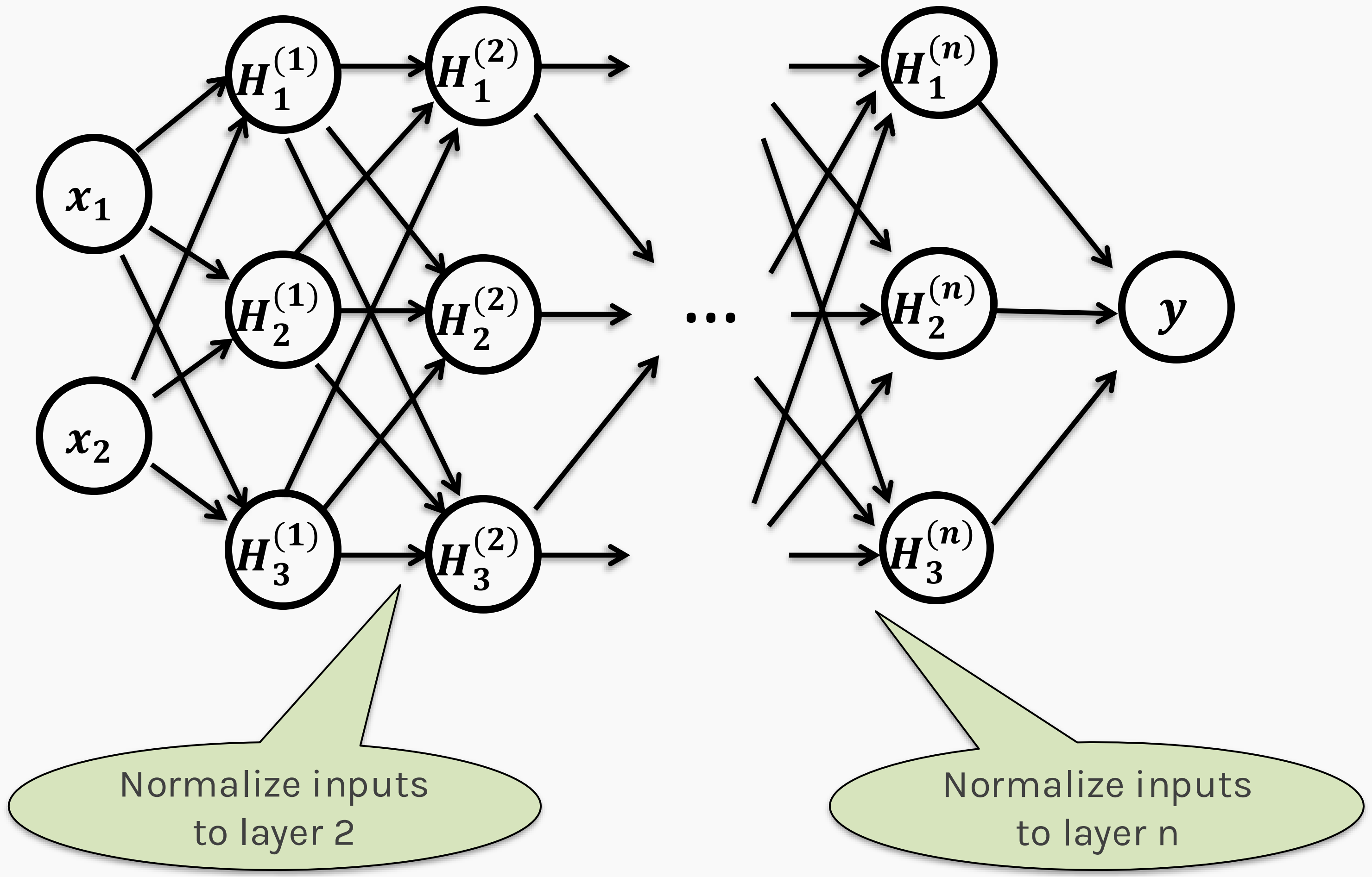
## Internal Covariance Shift

How can this problem be solved?


Covariate Shift
Batch Normalization
Neural Network
Utkal gaurab

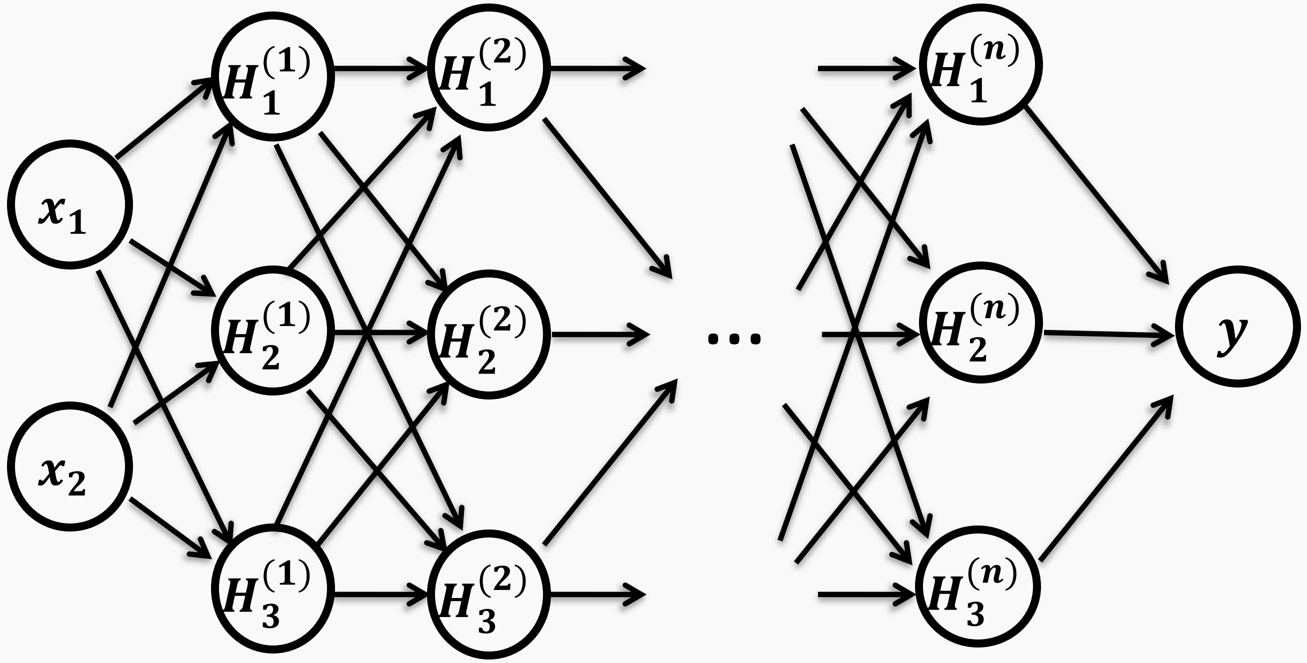
# Internal Covariance Shift Solution

We normalize the inputs to every hidden layer.



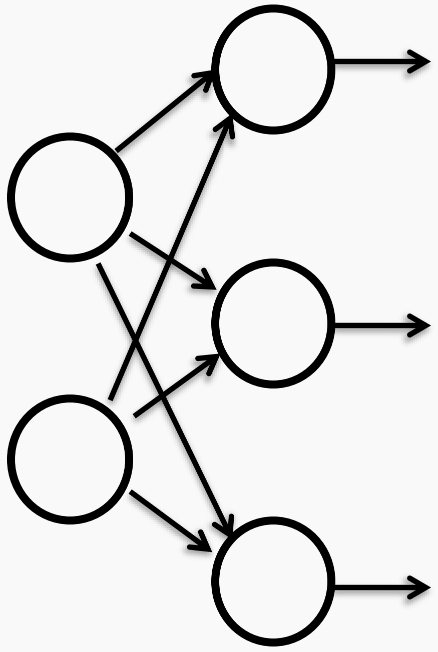
# Internal Covariance Shift Solution

We normalize inputs to every hidden layer.

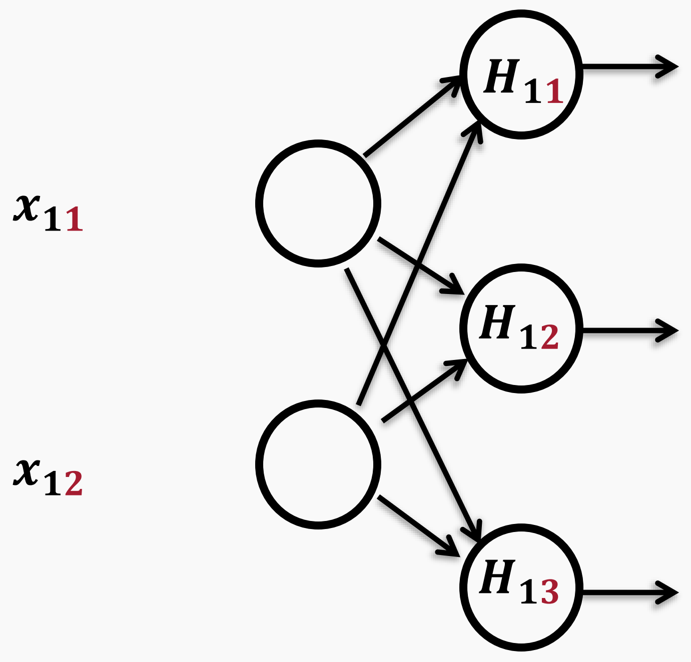


## Batch Normalization

We get the outputs from the first hidden layer.


1st Hidden Layer

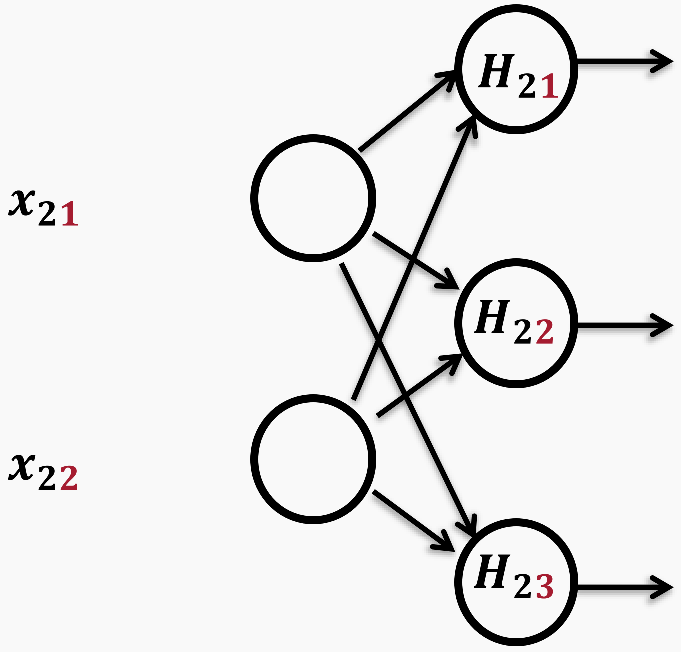
# Batch Normalization



1st Hidden Layer

$$\left[\begin{array}{c} \\ \\ \\ \end{array}\right]$$

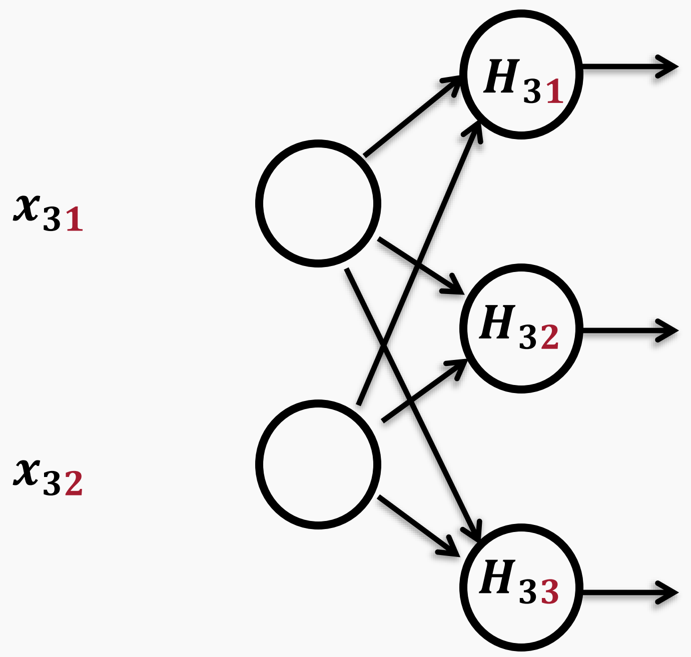
## Batch Normalization



1st Hidden Layer

$$\begin{bmatrix} H_{11} & H_{12} & H_{13} \\ & & \\ & & \end{bmatrix}$$

# Batch Normalization



$3$ data points in batch

$$\begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ & & \end{bmatrix}$$

$3$ hidden units activations

We do the same for $N$ data points in batch and $K$ hidden units activations in the next slide

## Batch Normalization

Training time:

Batch of activations for a given layer to normalize

For a given hidden layer

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

*N* data points in batch

*K* hidden units' activations

## Batch Normalization

Training time:

Batch of activations for a given layer to normalize

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

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

## Batch Normalization

Training time:

Batch of activations for a given 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 batch for node k.

## Batch Normalization

When calculating the variance, we add a small constant to the variance to prevent potential divisions by zero.

Training time:

Batch of activations for a given layer to normalize

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

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

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

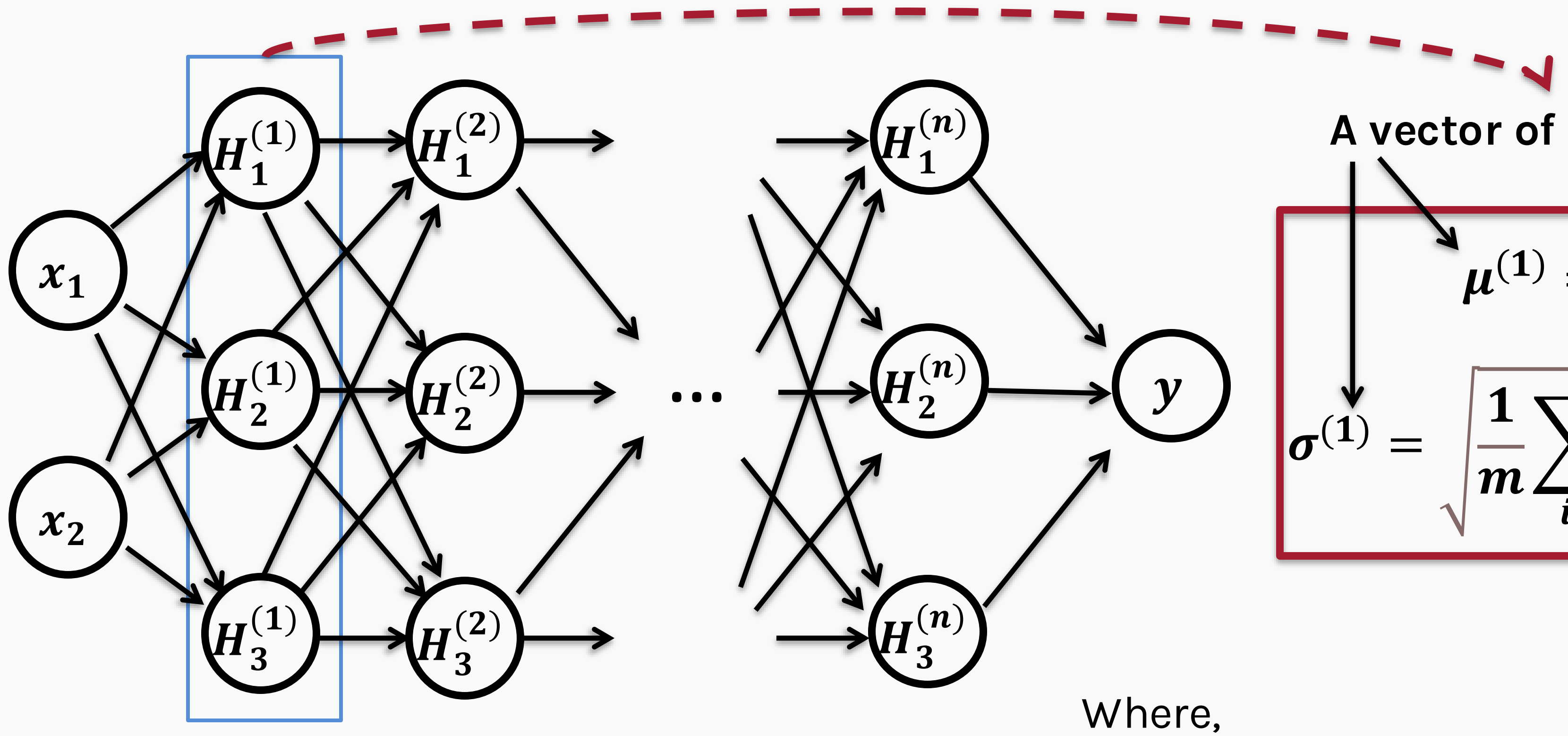
Mean activation across batch for node

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

Standard deviation across batch, $k$

## Batch Normalization

Batch 1



**A vector of 3 values**

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

$$\sigma^{(1)} = \sqrt{\frac{1}{m}\sum_i \left(H_i^{(1)} - \mu^{(1)}\right)^2 + \delta}$$

Where,

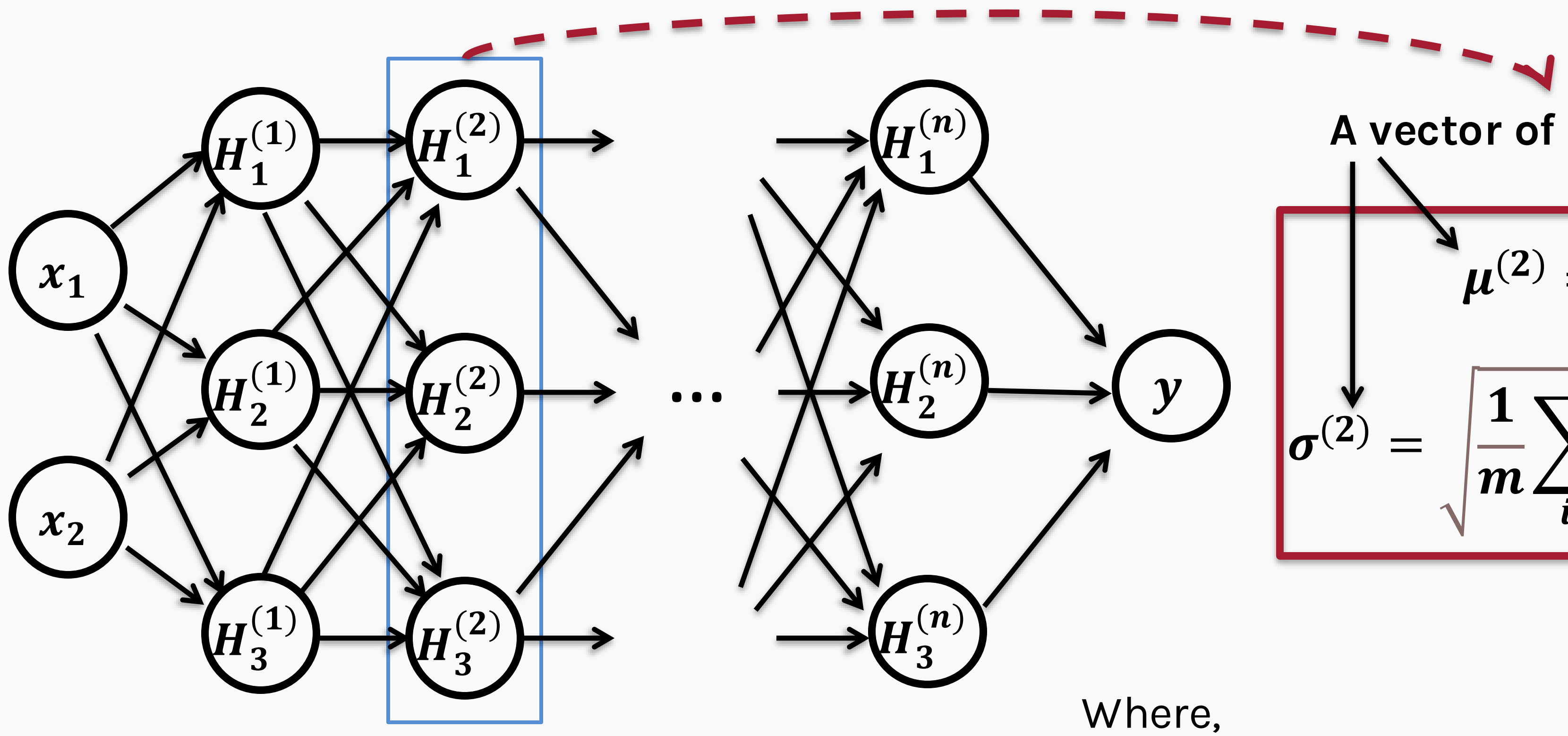
m: Number of training examples in the batch

$H_i^{(1)}$: Hidden layer activation of the first hidden layer for the $i^{th}$ training example

$\boldsymbol{\delta}$: A small constant

# Batch Normalization

Batch 1



A vector of 3 values

$$\mu^{(2)} = \frac{1}{m}\sum_i H_i^{(2)}$$

$$\sigma^{(2)} = \sqrt{\frac{1}{m}\sum_i \left(H_i^{(2)} - \mu^{(2)}\right)^2 + \delta}$$

Where,

m: Number of training examples in the batch

$H_i^{(2)}$: Hidden layer activation of the second hidden layer for the $i^{th}$ training example

$\boldsymbol{\delta}$: A small constant

## Batch Normalization

Training time:

- Normalization can reduce expressive power
- Instead use:

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

## Batch Normalization

Training time:

- Normalization can reduce expressive power
- Instead use:

Normalized Activation

Final Transformed Activation

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

Learnable parameters

## Batch Normalization

When do we apply batch normalize: before or after activation?

**Before Activation**

## Batch Normalization

We have the equation

$$h^{(2)} = Wa^{(1)} + b$$

where

$a^{(1)}$ : Activation of the first hidden layer

$h^{(2)}$: the output of the second hidden layer w/o activation

If we do batch normalization **after** activation:

$a^{(1)}$ will be affined transformed after this and therefore there is not need to transform again.

## Batch Normalization

We have the equation

$$h^{(2)} = Wa^{(1)} + b$$

where

$a^{(1)}$ : Activation of the first hidden layer

$h^{(2)}$: the output of the second hidden layer w/o activation

If we do batch normalization **before** activation:

$Wa^{(1)} + b$ is very likely to have a symmetric, non-sparse distribution; normalizing it is likely to produce activations with a stable distribution.

## Evaluation

We saw how batch normalization works during training, but what about **prediction**, when we might not have a complete batch!

## Evaluation

### Evaluation time:

- Calculate the running average of the mean and standard deviation.
- For every batch:

Decay parameter

Use this for evaluation

$$\mu_{global} = \alpha\mu_{global} + (1-\alpha)\mu_k$$

$$\sigma_{global} = \alpha\sigma_{global} + (1-\alpha)\sigma_k$$

## Batch Normalization

Evaluation time:

Hidden activations will be a vector as there are no batches.

$$\boxed{H = \begin{bmatrix} H_1 & \dots & H_K \end{bmatrix}}$$

# Batch Normalization

## Evaluation time:

Use the global statistics to normalize the node activations.

$$\boxed{H = [H_1 \quad \dots \quad H_K]}$$

For each hidden node $k$:

$$H'_k = \frac{H_k - \mu_{global}}{\sigma_{global}}$$

Estimated global mean of each unit activation.

Estimated global SD of each unit activation.

Thank you (again)