SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of AI &DS

Course Name – 19AD602 DEEP LEARNING

III Year / VI Semester

Unit 2-DEEP NETWORKS

Topic: BATCH NORMALIZATION

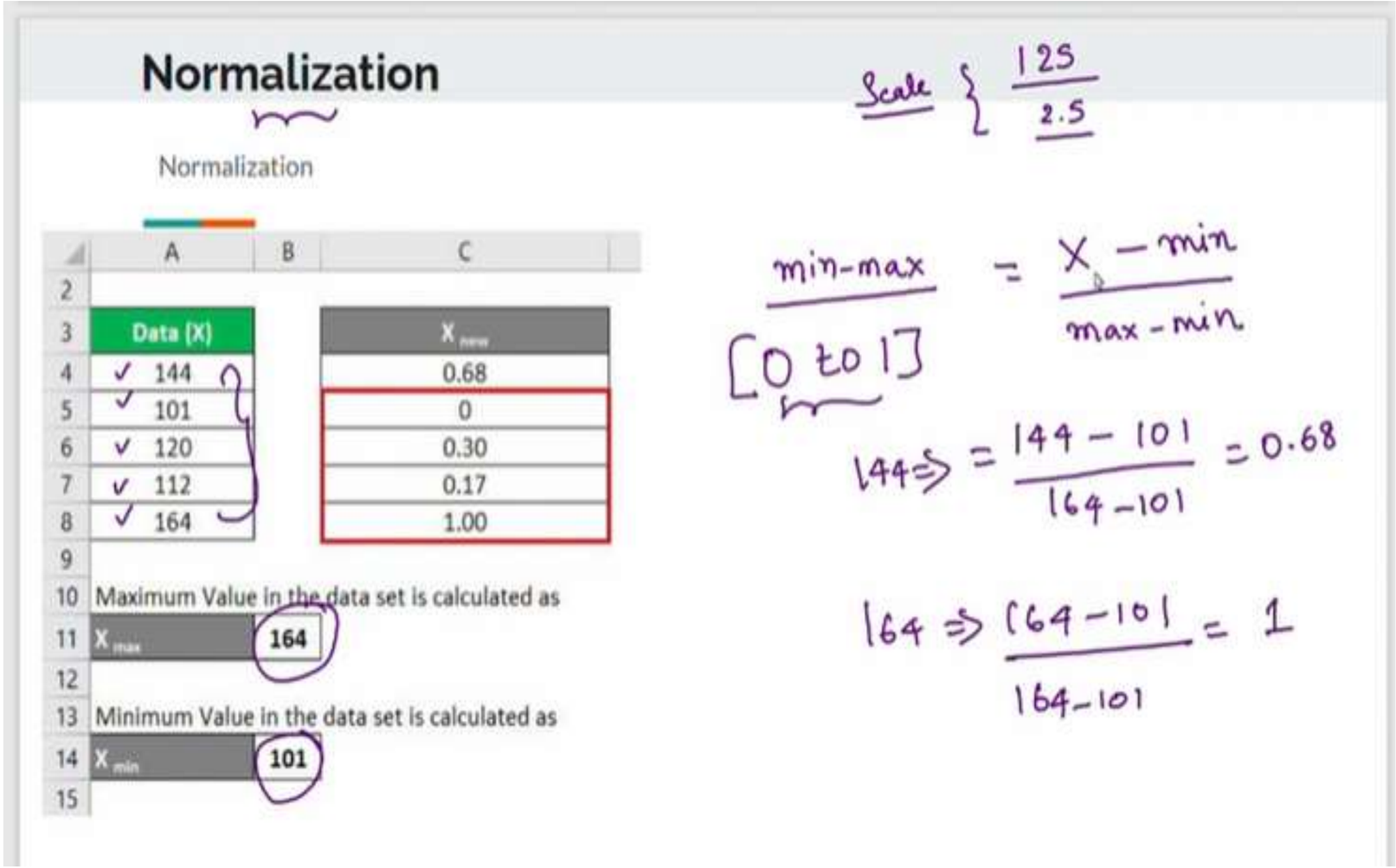
Case Study

A fintech company improved the performance of their credit risk prediction model using batch normalization. By stabilizing the distribution of activations during training, the network converged 30% faster, reducing training time while achieving higher accuracy on unseen data. This optimization allowed for quicker model updates and better real-time predictions.

Activity

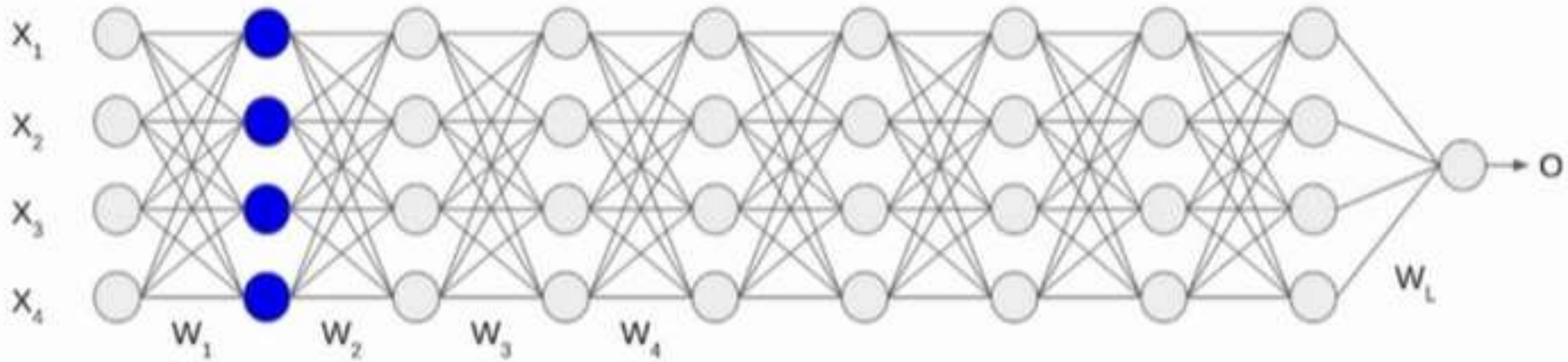
Train a neural network with and without batch normalization on a benchmark dataset, then compare training speed and model accuracy across both setups.

BATCH NORMALIZATION



BATCH NORMALIZATION

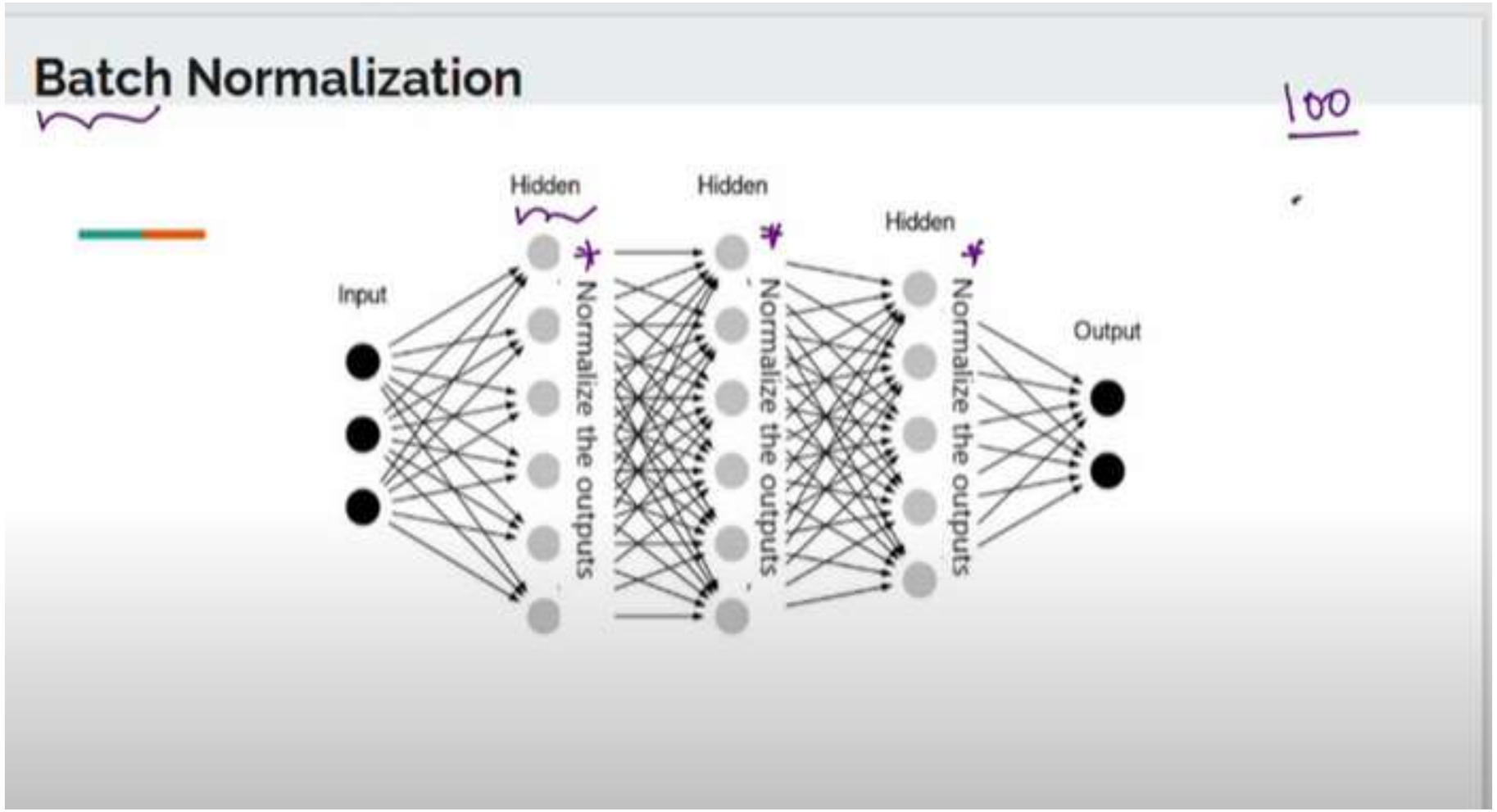
Initially, our inputs X1, X2, X3, X4 are in normalized form as they are coming from the pre-processing stage. When the input passes through the first layer, it transforms, as a sigmoid function applied over the dot product of input X and the weight matrix W.



$$h_1 = \sigma(W_1 X)$$

Similarly, this transformation will take place for the second layer and go till the last layer

BATCH NORMALIZATION



BATCH NORMALIZATION

1. Normalization is a data pre-processing tool used to bring the numerical data to a common scale without distorting its shape.
2. Generally, when we input the data to a machine or deep learning algorithm we tend to change the values to a balanced scale. The reason we normalize is partly to ensure that our model can generalize appropriately.
3. Now coming back to Batch normalization, it is a process to make neural networks faster and more stable through adding extra layers in a deep neural network. The new layer performs the standardizing and normalizing operations on the input of a layer coming from a previous layer.
4. In a standard feedforward neural network, the batch normalization layer takes the activations of a hidden layer as input, and outputs the normalized and transformed activations, which are then passed to the activation function. During training, the mean and standard deviation of the activations are computed over each mini-batch of data, and the learnable parameters gamma and beta are updated using gradient descent to minimize the loss function of the network.

BATCH NORMALIZATION

Here's how batch normalization works:

1. Compute the mean and standard deviation of the activations of each layer in a mini-batch of data.
2. Normalize the activations of each layer by subtracting the mean and dividing by the standard deviation.
3. Scale and shift the normalized activations by learnable parameters, known as gamma and beta, which are updated during training.
4. Pass the normalized and transformed activations to the next layer.

During training, the mean and standard deviation of the activations are computed over each mini-batch of data, and the gamma and beta parameters are updated using gradient descent to minimize the loss function of the network.

BATCH NORMALIZATION

We apply a batch normalization layer as follows for a minibatch $\mathcal{B}$:

m → no. of neurons

(1) $$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} x_i$$

(2) $$\sigma_{\mathcal{B}}^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2$$

(3) $$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$ ← smoothing

(4) $$y_i = \gamma \hat{x}_i + \beta = \mathrm{BN}_{\gamma,\beta}(x_i)$$

Where γ and β are learnable parameters.

BATCH NORMALIZATION

Example - Mini batch of 4 examples with 1 feature

X = [1, 2, 3, 4]

1. mu = 2.5 # mean of X

   sigma = 1.118 # standard deviation of X

2. X_norm = (X - mu) / sigma

   X_norm = [-1.3416, -0.4472, 0.4472, 1.3416]

3. gamma = 1, beta = 0

   Z = gamma * X_norm + beta

   Z = [-1.3416, -0.4472, 0.4472, 1.3416] pass to next layer

BATCH NORMALIZATION

2.5.1 Difference Between a Shallow Net & Deep Learning Net:

Sl.No	Shallow Net's	Deep Learning Net's
1	One Hidden layer(or very less no. of Hidden Layers)	Deep Net's has many layers of Hidden layers with more no. of neurons in each layers
2	Takes input only as VECTORS	DL can have raw data like image, text as inputs
3	Shallow net's needs more parameters to have better fit	DL can fit functions better with less parameters than a shallow network
4	Shallow networks with one Hidden layer (same no of neurons as DL) cannot place complex functions over the input space	DL can compactly express highly complex functions over input space
5	The number of units in a shallow network grows exponentially with task complexity.	DL don't need to increase it size(neurons) for complex problems
6	Shallow network is more difficult to train with our current algorithms (e.g. it has issues of local minima etc)	Training in DL is easy and no issue of local minima in DL

THANK YOU