Improving Deep Neural Networks

Video 3 : The Solution to Vanishing and Exploding Gradients - Part 2

Analytics Vidhya

Analytics Vidhya

Optimum Weight Initialization

Introduction

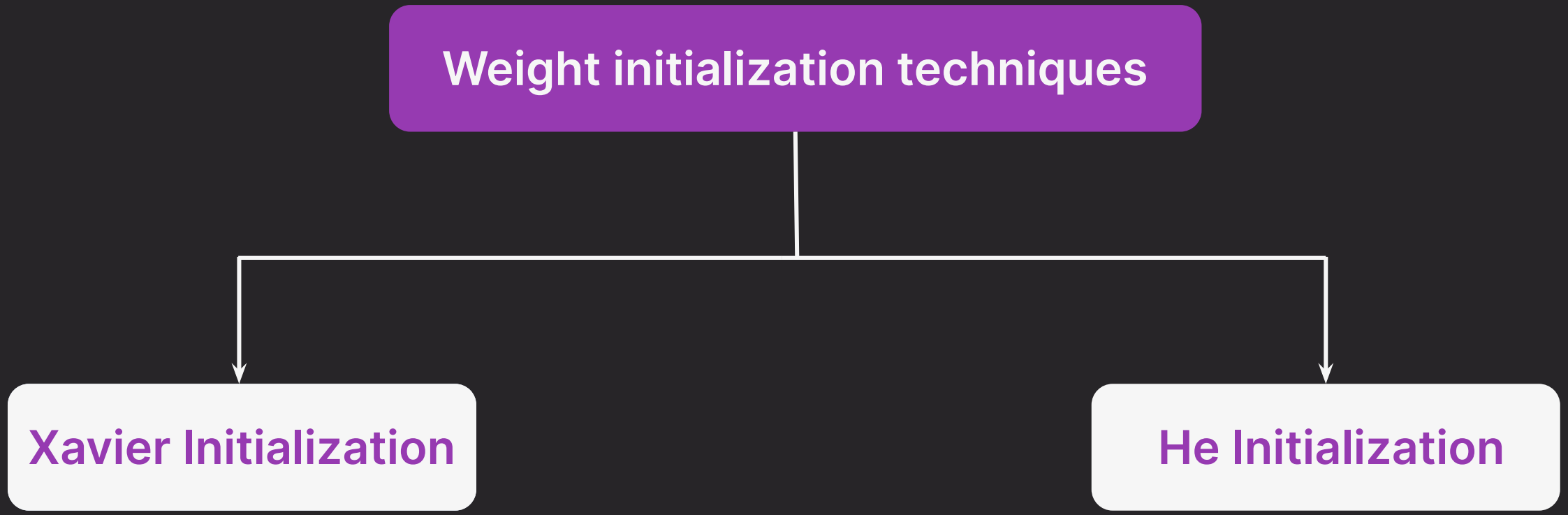
Forward propagation of the neural network

Weights assigned randomly

Analytics Vidhya

Introduction

Key concept: Keep activations and gradients variance consistent across layers



Xavier Initialization

Xavier initialization

Variance $= \frac{1}{\text{Avg (no. of input neurons, no. of output neurons)}}$

Analytics Vidhya

Xavier Initialization

Xavier initialization – – – – **TanH** activation function

Variance $= \frac{1}{\text{Avg (no. of input neurons, no. of output neurons)}}$

He Initialization

He initialization

$$\text{Variance} = \frac{2}{\text{Avg (no. of input neurons, no. of output neurons)}}$$

He Initialization

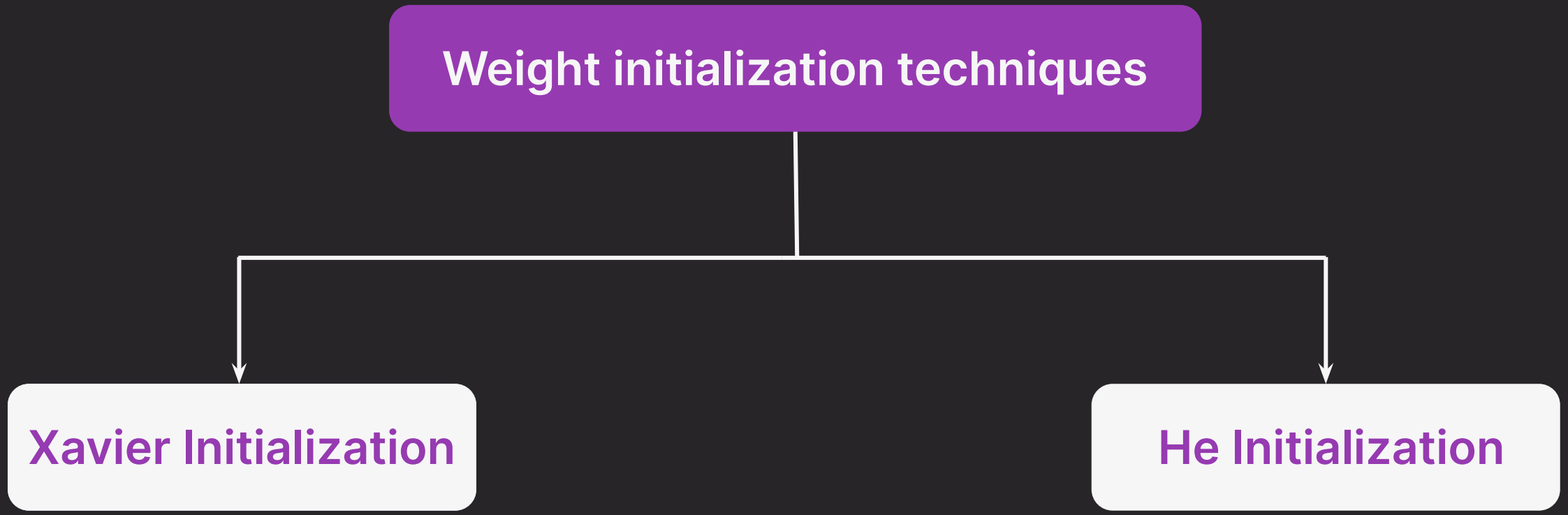
He initialization – – – – **ReLU** activation function

Variance $= \frac{2}{\text{Avg (no. of input neurons, no. of output neurons)}}$

Analytics Vidhya

Introduction

Key takeaway: PyTorch initializes weights using one of these techniques.

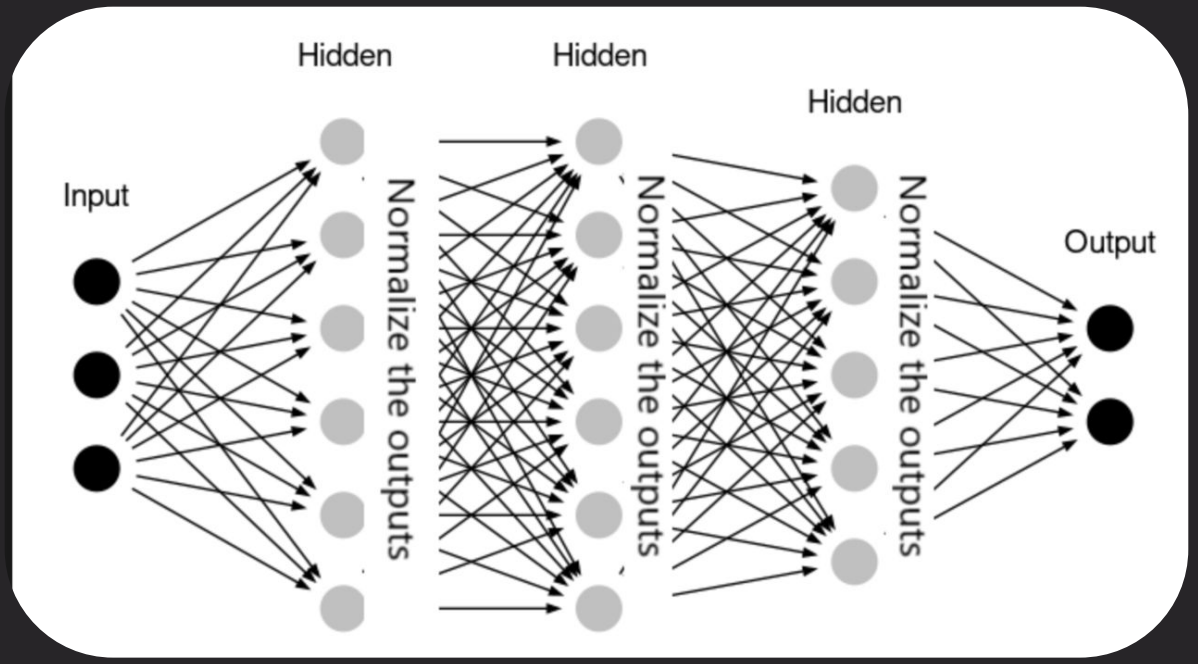


Analytics Vidhya

Batch Normalization

Key concept: Normalize previous outputs before sending it to the next layer.

- It is highly efficient and acts as a catalyst for faster convergence



Analytics Vidhya

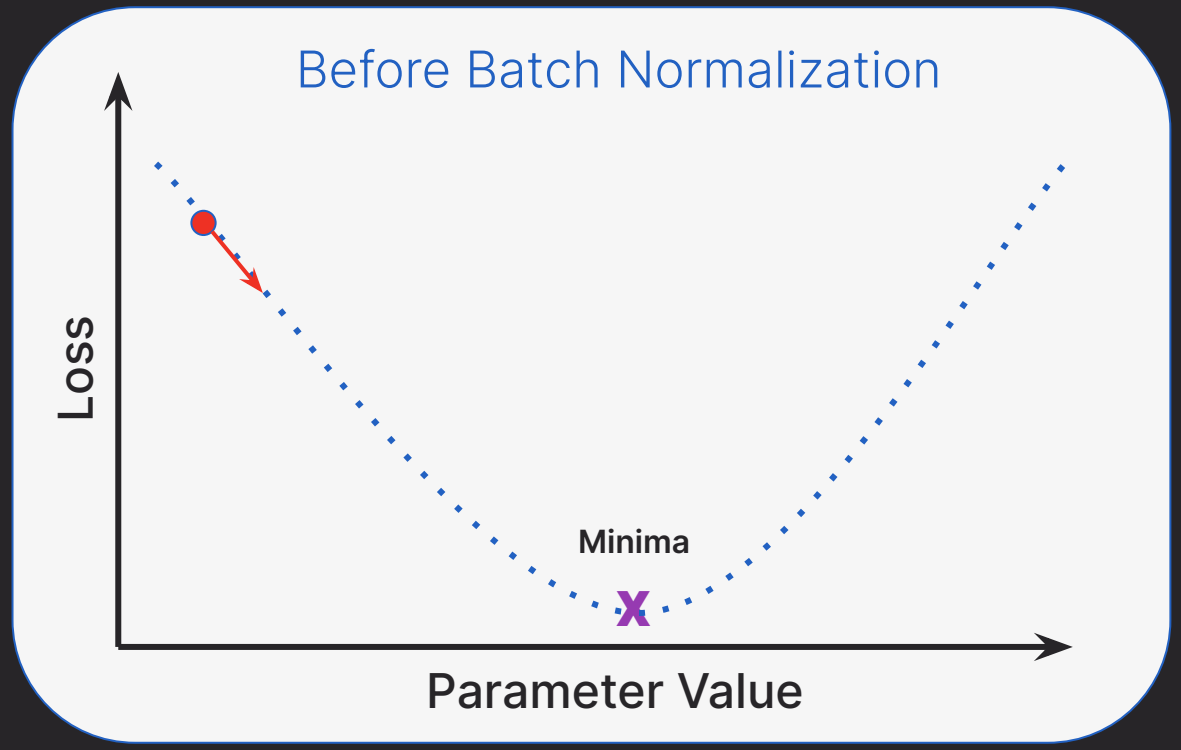
Batch Normalization

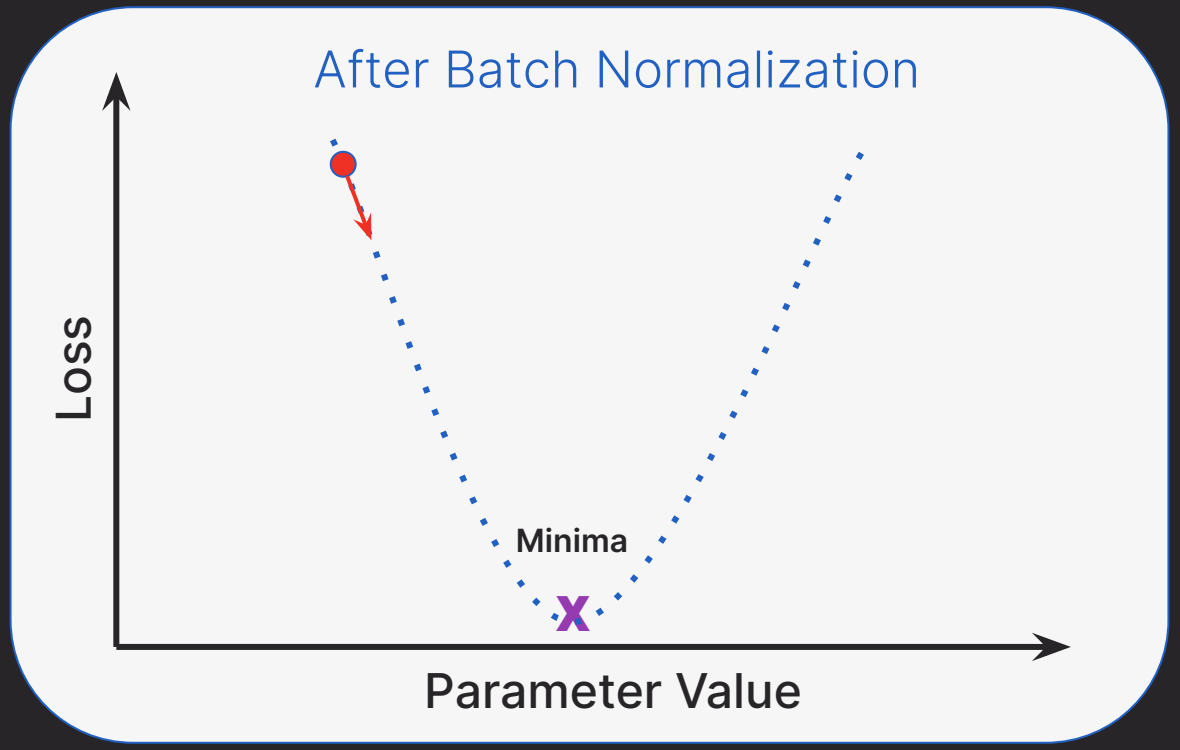
Why normalize the input features?

- Equal importance to all features
- Fast conversion

Analytics Vidhya

Batch Normalization: Analogy





Analytics Vidhya

What is “Batch” in Batch Normalization?

Batch Normalization

Neural networks and batch normalization both operate in batches

How does Batch Normalization work?

Analytics Vidhya

Batch Normalization: Working

Activations of an input layer are normalized for each mini-batch of data:

$$NA_i = (ai - mean) / sqrt(variance + \varepsilon)$$

Analytics Vidhya

Batch Normalization: Working

Activations of an input layer are normalized for each mini-batch of data:

$$NA_i = (ai - mean) / sqrt(variance + \varepsilon)$$

Mean of activation function in a layer = 0

Standard Deviation of activation function in a layer = 1

Batch Normalization: Working

Activations of an input layer are normalized for each mini-batch of data:

$$NA_i = (ai - mean) / sqrt(variance + \varepsilon)$$

Adjustment: $z_i = \gamma NA_i + \beta$

Analytics Vidhya

Fun Facts: Batch Normalization

- Helps in reducing overfitting marginally
- Batch normalization often replaces the neuron's bias with its beta term

Adjustment: $z_i = \gamma NA_i + \beta$

Analytics Vidhya

Gradient Clipping

Gradient clipping prevents exploding gradients by capping their absolute value to a set threshold.

Clipped Gradient = **min(threshold , gradient)**

Analytics Vidhya

Up-Next: Hands-On Vanishing and Exploding Gradients