Batch Normalization

Batch Normalization is a popular technique in deep learning which is used to improve the training of deep neural networks. It was introduced by Sergey Ioffe and Christian Szegedy in 2015 in their paper "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shifts".

Batch Normalization is a method to normalize the inputs of each layer in a neural network for each mini-batch during training. This helps reduce the problem of internal covariate shift.

* Internal Covariate Shift is a phenomenon where the distribution of each layer's input changes during training due to updates in the parameters of the preceding layers.

How Batch Normalization Works

The core idea is to normalize the activations or intermediate outputs of a layer to have a mean of 0 and variance of 1. However, BatchNorm does more than this ~~for~~ by introducing learnable parameters that allows the model to scale and shift the normalized values.

For a mini-batch $X = \{x_1, x_2, x_3, ---, x_m\}$ with m samples:

1. Compute Mean and Variance

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{(Mean of the batch)}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2 \quad \text{(Variance of the batch)}$$

2. Normalize the batch:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

ϵ → a very small value so that denominator never becomes 0.

3. Scale and Shift using (Learnable Parameters):

$$\boxed{y_i = \gamma \hat{x}_i + \beta}$$

γ → Scaling factor, β → shifting factor } both are learned during training

Key Components of Batch Normalization

⇒ Normalization: Ensures that inputs to a layer have consistent distribution, improving training stability and reducing sensitivity to initialization.

⇒ Learnable Parameter (γ, β): Allow the network to recover original distribution if necessary, making BatchNorm more flexible than simple standardization.

⇒ Mini-batch dependency: Normalization statistics (μ_B, σ_B^2) are computed per mini-batch during training.

Benefits of Batch Normalization

(1) Improved Convergence: It reduces the problem of vanishing or exploding gradients by keeping intermediate values in a stable range; It also allows the network to use higher learning rates, speeding up training process.

(2) Reduced Dependence on Initialization: Less sensitivity to weight initialization, as BatchNorm reduces the importance of initial distribution.

(3) Handles Covariate Shift: Reduces Internal Covariate Shift by stabilizing the distribution of activations during training.

(4) Regularization Effect: By adding noise due to mini-batch statistics, BatchNorm acts as a regularizer, reducing the need for dropout or other regularization techniques.

Challenges and Alternatives

(1) Performance may degrade with very small mini-batches.

(2) The computation of batch statistics adds some more overhead.

(3) Alternatives like Layer Normalization or Group Normalization are better suited for RNNs.

Variants of Batch Normalization

(1) Layer Normalization ⇒ Normalizes accross features instead of batch.

(2) Group Normalization ⇒ Divides features into groups and normalizes within each group.

(3) Instance Normalization ⇒ Normalizes each sample individually, commonly used in style transfer.

Batch Normalization during Inference

During Inference, the model uses the population (global) statistics for mean and variance instead of the mini-batch statistics.
These global statistics are computed as running averages during training.
Suppose we have 100 rows of data and we take mini-batch of size 20 then for each mini-batch we have separate mean and variance

1-20	21-40	41-60	61-80	81-100
μ_{B1}	μ_{B2}	μ_{B3}	μ_{B4}	μ_{B5}
σ^2_{B1}	σ^2_{B2}	σ^2_{B3}	σ^2_{B4}	σ^2_{B5}

Using all these separate mean and variance, we calculate a population mean and variance by EWMA (Estimated Weighted Moving Average) method for predictions.