Regularization Techniques: Dropout and Batch Normalization

1. What is Regularization?

Regularization means adding certain techniques during training to stop a model from **overfitting** — that is, performing very well on training data but poorly on unseen data.

In simple terms: Regularization makes the model simpler, more general, and less sensitive to noise.

2. Dropout

2.1 What is Dropout?

Dropout is a technique where, during training, we randomly turn off (set to zero) some neurons. This prevents the model from depending too much on certain neurons and helps it learn better general patterns.

2.2 Why Use Dropout?

- Prevents overfitting.
- Makes the model more robust.
- Acts like training multiple smaller networks together.

2.3 Mathematical Explanation

Let the neuron outputs be $a_1, a_2, a_3, \ldots$

We create a random mask m_i as:

$$m_i = \begin{cases} 1, & \text{with probability } p \\ 0, & \text{with probability } 1 - p \end{cases}$$

Then the new output becomes:

$$\tilde{a}_i = \frac{m_i \times a_i}{p}$$

This ensures that the expected output remains the same:

$$E[\tilde{a}_i] = a_i$$

2.4 Example

Let $a = [0.8, 0.5, 0.3, 0.9]$ and keep probability $p = 0.5$.

Random mask $m = [1, 0, 1, 0]$

Then:

$$\tilde{a} = \frac{m \times a}{p} = [1.6, 0, 0.6, 0]$$

Two neurons are dropped, and the remaining outputs are scaled by $1/p$.

3. Batch Normalization (BN)

3.1 What is Batch Normalization?

Batch Normalization helps stabilize and speed up training by normalizing the outputs of a layer so that they have a similar range (mean ≈ 0, variance ≈ 1).

3.2 Why Use Batch Normalization?

- Keeps training stable.
- Allows higher learning rates.
- Reduces vanishing and exploding gradients.
- Adds mild regularization.

3.3 Step-by-Step Mathematical Process

Given a batch of data $x_1, x_2, \ldots, x_m$:

1. **Compute the mean and variance:**

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i, \quad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

2. **Normalize:**

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

Now, the output has mean 0 and variance 1.

3. **Scale and shift (trainable parameters):**

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

where γ (scale) and β (shift) are learned during training.

3.4 Example

Suppose $x = [1, 2, 3, 4]$

Mean:

$$\mu_B = 2.5$$

Variance:

$$\sigma_B^2 = 1.25$$

Normalized:

$$\hat{x} = \frac{x - 2.5}{\sqrt{1.25}} = [-1.34, -0.45, 0.45, 1.34]$$

Now, the activations are centered and scaled for stable learning.

4. Comparison Table

Feature	**Dropout**	**Batch Normalization**
Goal	Reduce overfitting	Stabilize training
Method	Randomly drops neurons	Normalizes activations
When used	Training only	Training and testing (with saved stats)
Main control	Keep probability (p)	Scale (γ) and Shift (β)
Effect	Adds noise, improves robustness	Keeps learning fast and stable

5. Summary

- **Regularization** helps prevent overfitting and improves generalization.
- **Dropout** randomly removes neurons during training to make the model robust.
- **Batch Normalization** normalizes activations for stable and faster learning.
- Both techniques are important in modern deep learning networks.