**Assignment – 03**

1. Is it OK to initialize all the weights to the same value as long as that value is selected randomly using He initialization?

Ans: Initializing all weights to the same value, even if randomly selected using He initialization, is not recommended. While using He initialization helps in preventing the vanishing or exploding gradients problem, initializing all weights to the same value can lead to symmetry breaking issues. It's better to initialize weights with small random values drawn from a suitable distribution, such as a normal distribution with zero mean and variance based on He initialization.

1. Is it OK to initialize the bias terms to 0?

Ans: Initializing bias terms to 0 is generally acceptable, especially when using activation functions like ReLU. However, in some cases, initializing bias terms to non-zero values can help speed up convergence or improve learning dynamics, particularly when dealing with imbalanced datasets or asymmetric activation functions.

1. Name three advantages of the SELU activation function over ReLU.

Ans: Advantages of the SELU (Scaled Exponential Linear Unit) activation function over ReLU include:

SELU can maintain a mean of 0 and unit variance in the activations, helping in preventing vanishing or exploding gradients during training.

SELU is self-normalizing, meaning the activations tend to converge towards a stable distribution over time, which can lead to faster convergence and improved generalization.

SELU allows for deeper networks without the need for additional normalization techniques, such as batch normalization, thereby simplifying model architectures.

1. In which cases would you want to use each of the following activation functions: SELU, leaky ReLU (and its variants), ReLU, tanh, logistic, and softmax?

Ans:

Use cases for different activation functions:

SELU: Suitable for deep neural networks where self-normalization and stable activations are desired.

Leaky ReLU (and its variants): Useful when preventing dead neurons and encouraging gradient flow through the network, especially in deeper architectures.

ReLU: Widely used as a default choice due to its simplicity and computational efficiency. Suitable for most deep learning tasks.

Tanh: Suitable for hidden layers in neural networks where outputs need to be bounded between -1 and 1, and where vanishing gradients are a concern.

Logistic (Sigmoid): Typically used in binary classification tasks where outputs need to be squashed between 0 and 1, such as the output layer of logistic regression models.

Softmax: Appropriate for multi-class classification problems, where the output layer needs to produce probability distributions over multiple classes.

1. What may happen if you set the momentum hyperparameter too close to 1 (e.g., 0.99999) when using an SGD optimizer?

Ans: Setting the momentum hyperparameter too close to 1 (e.g., 0.99999) in an SGD optimizer can lead to unstable training dynamics and oscillations in the weight updates. This can cause the optimizer to overshoot or diverge from the optimal solution, resulting in poor convergence and training performance.

1. Name three ways you can produce a sparse model.

Ans: Ways to produce a sparse model include:

L1 Regularization (Lasso): Adding an L1 penalty term to the loss function encourages sparsity by driving many weights towards zero.

Dropout: Randomly dropping out a fraction of units (neurons) during training can induce sparsity in the network, as it prevents co-adaptation of neurons.

Pruning: Iteratively removing connections or units with small weights or low importance based on certain criteria, such as magnitude-based pruning or sensitivity-based pruning.

1. Does dropout slow down training? Does it slow down inference (i.e., making predictions on new instances)? What about MC Dropout?

Ans: Dropout can slow down training as it introduces randomness and uncertainty, requiring the model to learn robust representations. During inference (making predictions on new instances), dropout is typically turned off, so it doesn't slow down the process. However, dropout can affect the prediction uncertainty. MC Dropout (Monte Carlo Dropout) involves performing multiple forward passes with dropout enabled during inference and averaging the predictions, which can increase computation time but provide more reliable uncertainty estimates.