INFO7375

HW8

Jiani Lyu

1. To which values initialize parameters (W, b) in a neural network and why?

Initializing parameters (weights W and biases b) in neural networks is crucial for effective learning. Different strategies are used:

- Zero Initialization: Generally avoided due to symmetry problems.
- Random Initialization: Small random values to break symmetry but must be carefully scaled to avoid vanishing or exploding gradients.
- Xavier/Glorot Initialization: Suitable for sigmoid and tanh activations, balances the variance of inputs and outputs.
- He Initialization: Designed for ReLU activations, helps avoid the dead ReLU problem by keeping the variance of activations higher.
- Biases: Often initialized to zero as it doesn't contribute to symmetry breaking.

2. Describe the problem of exploding and vanishing gradients?

The problems of exploding and vanishing gradients refer to issues in deep neural networks where gradients become either too large or too small, complicating the training process:

- Vanishing Gradients: Occur when gradients become very small, slowing down the training or stopping it altogether. Common in deep networks with sigmoid or tanh activations.
- Exploding Gradients: Happen when gradients grow excessively large, causing numerical instability and hindering learning. More likely in very deep networks or with poor initialization.

Solutions include improved weight initialization (Xavier/Glorot, He initialization), non-saturating activation functions (ReLU), gradient clipping, batch normalization, and skip connections, all aimed at stabilizing and facilitating effective training of deep neural networks.

3. What is Xavier initialization?

Xavier initialization, also known as Glorot initialization, is a weight initialization method designed to help mitigate the vanishing/exploding gradient problem in neural networks. It initializes the weights of a layer by drawing them from a distribution with zero mean and a

variance of $2/(n_{in}+n_{out})$, where n_{in} and n_{out} are the number of input and output units for the layer. This approach aims to keep the variance of activations consistent across layers at the start of training, promoting stable and faster convergence. It is particularly suited for layers with sigmoid or tanh activation functions.

4. **Describe training, validation, and testing data sets and explain their role and why all they are needed.**

Training Dataset: Used to train the model. It helps the model learn the patterns and relationships in the data.
Validation Dataset: Used for tuning model parameters and preventing overfitting. It provides a way to validate model performance during training.
Testing Dataset: Used to evaluate the model's performance after training and tuning. It gives an unbiased assessment of how well the model generalizes to new data.

5. **What is training epoch?**

A training epoch means one complete pass of the entire training dataset through the algorithm or model. During an epoch, the model sees and learns from every example in the training set once.

6. **How to distribute training, validation, and testing sets?**

Based on the dataset size, if the dataset is small, we can do 70% training, 15% validation, and 15% testing. If the dataset is medium, we can do 90% training, 5% validation, and 5% testing, if the dataset is large, we can do 98% training, 1% validation, and 1% testing

7. **What is data augmentation and why may it needed?**

Data augmentation is a technique used to increase the diversity of a training dataset by applying various transformations to the data, such as rotation, scaling, flipping, or adding noise. It's needed to prevent overfitting, enhance the model's generalization ability, and improve performance, especially when the original dataset is small or lacks variability.