Deep Learning Assignment

Name:	**Suraj Kanu**
Registration No./Roll No.:	2311007
Institute/University Name:	IISER Bhopal
Program/Stream:	PhD DSE

Q1: Which loss function, out of Cross Entropy and Mean Squared Error, works best with logistic regression because it guarantees a single best answer (no room for confusion)? Explain why this is important and maybe even show how it affects the model's training process.

Answer: Cross Entropy loss works best with logistic regression. For classification problem we use logistic regression and it has output probability between zero to one representing the likelihood of a particular instance belonging to a certain class.
It is important because

- It penalizes the model more heavily for confidently incorrect predictions, thus pushing the model to learn better representations.
- It is easy to implement and optimize. Most neural network frameworks provide built-in functions for cross-entropy loss and its gradients. Cross-entropy loss also has a smooth and convex shape, which makes it easier for gradient-based optimization methods to find the global minimum.
- it is invariant to scaling and shifting of the predicted probabilities. This means that multiplying or adding a constant to the predicted probabilities does not affect the cross-entropy loss value, as long as they are still between 0 and 1. This can be useful for regularization and calibration of the model's outputs.

Q2: For a binary classification task with a deep neural network (containing at least one hidden layer) equipped with linear activation functions, which of the following loss functions guarantees a convex optimization problem? Justify your answer with a formal proof or a clear argument. (a) CE (b) MSE (c) Both (A) and (B) (d) None

Answer: Mean Squared Error loss functions guarantees a convex optimization problem, as we know that mean square error is define by the square of the difference between the predicted and actual value, the squared term and the affine function inside the norm ensure that the loss function is convex. As the loss function is convex so it guarantees a convex optimization.
But in cross entropy we have log function which introduces non convexity so doesn't guarantee a convex optimization.

Q3: Dense Neural Network: Implement a feed-forward neural network with dense layers only. Specify the number of hidden layers, neurons per layer, and activation functions. How will you preprocess the input images? Consider hyperparameter tuning strategies.

Answer: Dense Neural Network:

- Number of hidden layers = 3

- Number of hidden Neuron = [32, 64, 128]
- Activation function = Relu, Tanh, Sigmoid.

Preprocess the input images: I have used MNIST dataset which consists of 28×28 pixel grayscale images of handwritten digits from 0 to 9 . for preprocessing I have converted each sample is represented as a 3D array with dimensions $28 \times 28 \times 1$

Activation Function	**Neurons per Layer**	**Mean Test Score**
ReLU	32	0.925867
	64	0.936767
	128	0.945133
	256	0.951883
Tanh	32	0.921500
	64	0.930183
	128	0.935700
	256	0.937667
Sigmoid	32	0.905850
	64	0.914533
	128	0.919650
	256	0.922817

Table 1: Results for Different Activation Functions

The test accuracy of the best model: 0.966

Q4: Build a classifier for Street View House Numbers (SVHN) (Dataset) using pre-trained model weights from PyTorch. Try multiple models like LeNet-5, AlexNet, VGG, or ResNet(18, 50, 101). Compare performance comment why a particular model is well suited for SVHN dataset.

Answer: The SVHN dataset consists of images of house numbers collected from Google Street View. Each image contains digits , and the task is to classify these digits accurately. **Preprocessing:** The image is resized to 32×32, random rotation, randomcroping and random horizontal flip.
Model Selection:

- LeNet-5
- VGG-16
- ResNet-18
- ResNet-50
- ResNet-101

Metric	**Value**
Test Accuracy	83.92%
Precision	0.8457
Recall	0.8196
F1-score	0.8278

Table 2: Performance Report for ResNet-101

These models exhibit greater depth and complexity compared to LeNet-5. While they excel in handling extensive image classification tasks, their suitability diminishes when confronted with datasets featuring small-sized images such as SVHN. The increased depth of these models may exhibits overfitting issues or encounter difficulty in extracting important features from the images, thereby resulting in suboptimal performance.