Deep Learning - Assignment

Ramavath Tharun 21219

March 27, 2024

Question 1

Ans:

Cross-entropy loss is commonly employed in classification tasks, whereas Mean Squared Error (MSE) is typically utilized in regression problems within supervised learning. Logistic Regression, a method frequently employed for binary classification, outputs probabilities ranging from 0 to 1, indicating the likelihood of an input belonging to a specific class. Cross-entropy loss offers a more informative measure of the disparity between the actual and predicted classes. MSE, on the other hand, may yield binary outcomes such as 0, 1, or -1 as losses, which fail to convey the degree of misclassification for individual examples.

Question 2

Ans:

Since the activation is linear, the network functions as a linear classifier. Consequently, employing Cross-Entropy (CE) Loss should lead to a convex optimization problem. Building on the rationale from the previous question, utilizing CE Loss enables us to gauge the extent of misclassification for each example and then aggregate these errors to derive an average, ensuring a clear path towards the true optimum.

Conversely, opting for Mean Squared Error (MSE) would only indicate whether an example was classified correctly or not. Consequently, for incorrectly classified examples, there lacks a cohesive objective to optimize the problem uniformly. This discrepancy may result in the presence of numerous local minima, complicating the optimization process.

Question 3

Ans:

Number of hidden layers: 2

Neurons per layer:

- Input: 748
- Hidden Layers: 128
- Output: 10

Activation Function: ReLU

Preprocessing images:

- Using augmentation techniques like Cropping, Random Affine, Jittering, etc.
- Transforming to grayscale, using normalization, flattening the images, etc. For this problem, the images were transformed to grayscale and flattened.

Hyperparameter tuning strategies:

- Using GridSearch to select hyperparameters such as learning rate, optimizers, loss function, epochs, etc.
- Using techniques such as early stopping and scheduled learning rate
- Using regularization to limit errors For this problem, the hyperparameters tuned were batch size, learning rate, number of neurons in the hidden layers, and epochs. Code

Question 4

Ans:

The training was done on Jupyter Notebook, and the entire dataset was used. The training was done for 5 epochs. The training dataset was split into train and validation following a 75-30 ratio. All the results can be found in the notebook Code.

Model Performance Results

LeNet-5

- Training Accuracy: 88%
- Validation Accuracy: 87%
- Testing Accuracy: 86%

AlexNet

- Training Accuracy: 43%
- Validation Accuracy: 48%
- Testing Accuracy: 43%

VGG-16

- Training Accuracy: 33%
- Validation Accuracy: 40%
- Testing Accuracy: 42%

ResNet-18

- Training Accuracy: 31%
- Validation Accuracy: 31%
- Testing Accuracy: 31%

Model Architectural Details

- LeNet-5: Implemented from scratch as pre-trained weights were unavailable. Images were converted to grayscale.
- AlexNet: Images were resized to 63x63. Only the final layer was trained.
- VGG-16: No modifications to the images were made. Only the final layer was trained.
- ResNet-18: No modifications to the images were made. Only the final layer was trained.

The reason for this performance may be due to how much the models scale to the comparatively smaller dataset. All these models except LeNet-5 were pre-trained on the huge ImageNet database, which means when using a very small SVHN dataset, there is a tendency of underfitting due to lack of data. Another reason why LeNEt-5 performs well maybe due to the fact that it is a small model and it was trained from scratch, allowing it to scale well to the SVHN dataset.