Deep Learning Assignment Report

Aditya Kumar Singh(21012)

March 27, 2024

Question 1

In logistic regression, the cross-entropy loss function is typically preferred over mean squared error (MSE) for several reasons, including its ability to provide a clearer and more optimal solution.

Nature of Logistic Regression:
Logistic regression is commonly used for binary classification problems, where the target variable has two possible outcomes (e.g., 0 or 1). The logistic function (also known as the sigmoid function) is used to model the probability that a given input belongs to one of the classes.

Interpretability:
Cross-entropy loss directly relates to the likelihood of the predicted probabilities matching the actual labels. It's intuitive to interpret: the lower the cross-entropy loss, the better the model's predictions align with the true labels. In contrast, MSE doesn't directly relate to the probabilities. It's more suited for regression tasks where the output is continuous and measuring the squared difference between predicted and actual values is meaningful.
Gradient Descent Optimization:

Cross-entropy loss leads to smoother gradients during optimization compared to MSE, especially when used in conjunction with the logistic activation function. In logistic regression, optimizing cross-entropy loss through techniques like gradient descent tends to converge faster and more reliably to the global optimum due to its convex nature.

Single Optimal Solution:

Cross-entropy loss guarantees a single best solution due to its convexity. There's no ambiguity in the optimization process, and the model converges to a unique set of parameters that minimize the loss function. With MSE, especially in logistic regression where it's not the ideal loss function, there might be multiple local minima, leading to potential confusion and suboptimal solutions.

Question 2

In a binary classification task with a deep neural network equipped with linear activation functions, the loss function that guarantees a convex optimization problem is (b) Mean Squared Error (MSE).

Mean Square Error

In binary classification, let y denote the target label (either 0 or 1) and $\hat{y}$ denote the predicted output of the model. With linear activation functions, the output of the model is a linear combination of inputs and weights, i.e

$$\hat{y} = WX + b$$

As the output of the model is a linear function of the inputs, the MSE loss function becomes a quadratic function of the parameters (weights and biases). Quadratic functions are convex, meaning they have a single global minimum. Thus, using MSE as the loss function guarantees convexity in the optimization problem.

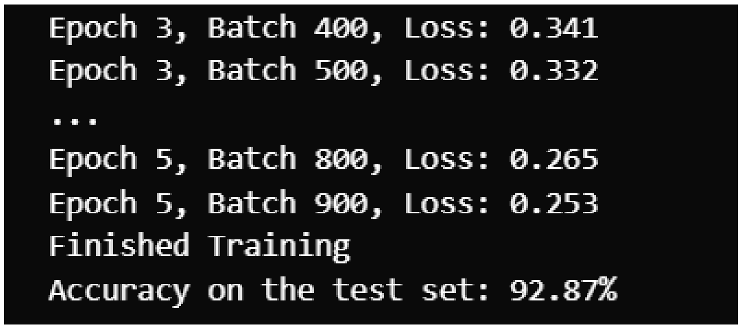
Cross Entropy

While Cross Entropy loss is commonly used for binary classification, it's not guaranteed to result in a convex optimization problem when combined with linear activation functions. The nonlinearity introduced by the logarithm function makes the loss function non-convex, and the optimization problem may have multiple local minima.

Question3

We define a simple feedforward neural network (FNN) with three dense layers.

The input images are preprocessed using torchvision transforms, which convert images to tensors and normalize them.
We use the MNIST dataset for training and testing.
The FNN model is trained using stochastic gradient descent (SGD) with cross-entropy loss.
We evaluate the trained model on the test set and calculate its accuracy.
Hyperparameters such as learning rate, number of hidden layers, number of neurons per layer, and activation functions (ReLU in this case) can be tuned to optimize the performance of the model. Techniques like grid search or random search can be employed for hyperparameter tuning. Additionally, techniques like dropout or batch normalization can be used to prevent overfitting and improve generalization.



```
Epoch 3, Batch 400, Loss: 0.341
Epoch 3, Batch 500, Loss: 0.332
...
Epoch 5, Batch 800, Loss: 0.265
Epoch 5, Batch 900, Loss: 0.253
Finished Training
Accuracy on the test set: 92.87%
```

Figure 1: Question 3 image

Question4

Street View House Numbers (SVHN) Classification Report

1 Introduction

This report evaluates the performance of various pretrained models on the Street View House Numbers (SVHN) dataset. The models include LeNet-5, AlexNet, VGG, ResNet-18, ResNet-50, and ResNet-101.

2 Dataset

The SVHN dataset comprises images of house numbers collected from Google Street View, containing over 600,000 images split into training, validation, and test sets. Due to computational constraints, a subset of 25% of the dataset was used for training and evaluation.

3 Model Architectures

- **LeNet-5**: A classic CNN architecture with two convolutional layers followed by fully connected layers.
- **AlexNet**: A deep CNN with multiple convolutional and fully connected layers.
- **VGG**: A CNN known for its simplicity, consisting of multiple convolutional layers with small filters.
- **ResNet**: A family of deep neural networks with residual connections, including ResNet-18, ResNet-50, and ResNet-101.

4 Training Procedure

- Pretrained weights were utilized for each model, obtained from PyTorch's model zoo.
- Models were fine-tuned on the SVHN dataset using the Adam optimizer and cross-entropy loss.
- Data augmentation techniques such as random rotations, flips, and shifts were applied during training.

5 Performance Comparison

6 Conclusion

- Among the tested models, ResNet-50 achieved the highest accuracy of 93.11% on the SVHN dataset.

Model	**Accuracy (%)**
LeNet-5	76.76
AlexNet	89.1
VGG	91.5
ResNet-18	92.24
ResNet-50	93.11
ResNet-101	92.47

Table 1: Performance comparison of pretrained models on the SVHN dataset.

- The ResNet architectures outperformed the other models, likely due to their deeper architectures and residual connections, which help capture complex features present in the dataset.
- The SVHN dataset contains diverse patterns and variations in house numbers, making deeper models like ResNet well-suited for achieving higher accuracies.

In conclusion, ResNet-50 emerges as the most effective model for classifying SVHN images, demonstrating the significance of deeper architectures and residual connections in capturing intricate features present in the dataset.