REPORT - LAB ASSIGNMENT 3

PRINCY GAUTAM
(B20BB051)

Introduction

The code provided trains a pre-trained ResNet18 model on the STL10 dataset using three different optimizers:

- Adam
- Adagrad
- RMSprop.

The code defines a batch size of 64 and uses transforms to pre-process the data before loading it into the data loader. The model's last fully connected layer is replaced with a new layer with 10 output features, corresponding to the 10 classes in the dataset. The code trains the model for 10 epochs with each optimizer, and it records the training loss and accuracy for each epoch. At last the top-5 accuracy is printed for each optimizer on the test dataset.

Data Preprocessing

- The STL10 dataset is used in this code, which contains 10 classes, including birds, airplanes, and horses. The dataset is loaded using the PyTorch DataLoader class.
- The dataset is preprocessed using the transforms.Compose method.
- Two transforms are defined, one for the training set and another for the test set. The training set is transformed using Resize, CenterCrop, ToTensor, and Normalize, whereas the test set is transformed using ToTensor and Normalize.
- The Resize method resizes the image to a square of size 256x256 pixels.
- The CentreCrop method crops the center of the image to a size of 224x224 pixels, which is the input size expected by most pre-trained models. The ToTensor method converts the images to tensor format, and the Normalize method normalizes the pixel values using the specified mean and standard deviation values.

Model Building

The pre-trained ResNet18 model is loaded, and the last fully connected layer is replaced with a new one having 10 output nodes to predict the 10 classes in the STL10 dataset. The loss function used in this code is Cross-Entropy Loss.

Training

The code trains the model using three optimizers, namely Adam, Adagrad, and RMSprop, for 10 epochs each. For each epoch, the training loss and accuracy are computed and stored in separate

lists for each optimizer. The accuracy is calculated as the ratio of correctly classified images to the total number of images in the training set.

Observations and Results

- The training loss and accuracy are printed for each epoch during the training process. The training loss and accuracy for each optimizer are stored in separate lists.
- The training loss decreases, and accuracy increases as the number of epochs increases for all the optimizers.
- Among the three optimizers, Adam performs better than the other two in terms of accuracy and convergence rate.
- The training loss decreases faster and accuracy increases more quickly for Adam than for Adagrad and RMSprop.
- After ten epochs, Adam has an accuracy of 0.9992, Adagrad has an accuracy of 0.9900, and RMSprop has an accuracy of 0.9734.

Conclusion

- In conclusion, the ResNet18 model performs well on the STL10 dataset using the Adam optimizer.
- The results obtained from the three optimizers show that the choice of optimizer is crucial for obtaining good results.

Optimizers

Three different optimizers are used to train the model with the STL10 dataset. The three optimizers are Adam, Adagrad, and RMSprop.

Adam optimizer

Adam optimizer computes adaptive learning rates for each parameter and applies momentum to the updates. It is suitable for large datasets with a large number of parameters. It uses two moving averages of the gradient and the squared gradient to adjust the learning rate. It is well-suited for non-stationary problems and it works well in practice. Adam optimizer is generally preferred for most deep learning tasks.

To vary the value of attributes in the Adam optimizer and show its effect on the overall performance, we can modify the learning rate and the beta coefficients. The beta coefficients control the moving averages of the parameters' first and second moments. We can modify betas in the optim.Adam constructor to vary the beta coefficients. We can also modify lr to vary the learning rate.

Adam optimizer1 - with attributes set as lr=0.001, betas=(0.9, 0.999):

```
Epoch [1/10], Loss: 0.0701, Accuracy: 0.9774
Epoch [2/10], Loss: 0.0640, Accuracy: 0.9810
Epoch [3/10], Loss: 0.0221, Accuracy: 0.9924
Epoch [4/10], Loss: 0.0193, Accuracy: 0.9932
Epoch [5/10], Loss: 0.1097, Accuracy: 0.9646
Epoch [6/10], Loss: 0.0932, Accuracy: 0.9694
Epoch [7/10], Loss: 0.0155, Accuracy: 0.9952
Epoch [8/10], Loss: 0.0109, Accuracy: 0.9962
Epoch [9/10], Loss: 0.0408, Accuracy: 0.9864
Epoch [10/10], Loss: 0.0498, Accuracy: 0.9836
```

Adam optimizer2 - with attributes set as lr=0.01, betas=(0.9, 0.999):

```
Epoch [1/10], Loss: 1.9288, Accuracy: 0.3810
Epoch [2/10], Loss: 1.5613, Accuracy: 0.4346
Epoch [3/10], Loss: 1.3427, Accuracy: 0.5132
Epoch [4/10], Loss: 1.1872, Accuracy: 0.5798
Epoch [5/10], Loss: 1.2232, Accuracy: 0.5598
Epoch [6/10], Loss: 0.9312, Accuracy: 0.6662
Epoch [7/10], Loss: 0.9698, Accuracy: 0.6454
Epoch [8/10], Loss: 1.4572, Accuracy: 0.4786
Epoch [9/10], Loss: 0.9443, Accuracy: 0.6568
Epoch [10/10], Loss: 0.7718, Accuracy: 0.7162
```

Adam optimizer3 - with attributes set as lr=0.001, betas=(0.95, 0.999):

```
Epoch [1/10], Loss: 0.4291, Accuracy: 0.8464
Epoch [2/10], Loss: 0.2903, Accuracy: 0.8950
Epoch [3/10], Loss: 0.2214, Accuracy: 0.9234
Epoch [4/10], Loss: 0.1456, Accuracy: 0.9502
Epoch [5/10], Loss: 0.0951, Accuracy: 0.9694
Epoch [6/10], Loss: 0.0730, Accuracy: 0.9772
Epoch [7/10], Loss: 0.0427, Accuracy: 0.9886
Epoch [8/10], Loss: 0.0420, Accuracy: 0.9894
Epoch [9/10], Loss: 0.0305, Accuracy: 0.9918
Epoch [10/10], Loss: 0.0495, Accuracy: 0.9846
```

Adam optimizer4 - with attributes set as lr=0.001, betas=(0.9, 0.99):

```
Epoch [1/10], Loss: 0.0342, Accuracy: 0.9898
Epoch [2/10], Loss: 0.0186, Accuracy: 0.9938
Epoch [3/10], Loss: 0.0188, Accuracy: 0.9950
Epoch [4/10], Loss: 0.0341, Accuracy: 0.9896
Epoch [5/10], Loss: 0.0124, Accuracy: 0.9966
Epoch [6/10], Loss: 0.0310, Accuracy: 0.9898
Epoch [7/10], Loss: 0.0137, Accuracy: 0.9960
```

```
Epoch [8/10], Loss: 0.0169, Accuracy: 0.9960
Epoch [9/10], Loss: 0.0057, Accuracy: 0.9986
Epoch [10/10], Loss: 0.0224, Accuracy: 0.9992
```

The optimizer with learning rate = 0.001 and betas set to = (0.9,0.99) gives us lowest loss and good accuracy.

Adagrad optimizer

Adagrad optimizer adapts the learning rate to the parameters, performing smaller updates (i.e., low learning rates) for parameters associated with frequently occurring features and larger updates (i.e., high learning rates) for parameters associated with infrequent features. Adagrad is generally used for sparse data problems.

Adagrad optimizer1 - with attributes set as lr=0.01, weight decay=0.01, initial accumulator value=0.1:

```
Epoch [1/10], Loss: 0.0097, Accuracy: 0.9974
Epoch [2/10], Loss: 0.0085, Accuracy: 0.9982
Epoch [3/10], Loss: 0.0071, Accuracy: 0.9980
Epoch [4/10], Loss: 0.0073, Accuracy: 0.9992
Epoch [5/10], Loss: 0.0071, Accuracy: 0.9996
Epoch [6/10], Loss: 0.0074, Accuracy: 0.9992
Epoch [7/10], Loss: 0.0066, Accuracy: 0.9996
Epoch [8/10], Loss: 0.0095, Accuracy: 0.9986
Epoch [9/10], Loss: 0.0095, Accuracy: 0.9992
Epoch [10/10], Loss: 0.0110, Accuracy: 0.9992
```

Adagrad optimizer2 - with attributes set as lr=0.001, weight decay=0.001, initial accumulator value=0.01:

```
Epoch [1/10], Loss: 0.0087, Accuracy: 0.9998
Epoch [2/10], Loss: 0.0101, Accuracy: 0.9998
Epoch [3/10], Loss: 0.0101, Accuracy: 0.9992
Epoch [4/10], Loss: 0.0088, Accuracy: 0.9998
Epoch [5/10], Loss: 0.0106, Accuracy: 0.9992
Epoch [6/10], Loss: 0.0098, Accuracy: 0.9998
Epoch [7/10], Loss: 0.0139, Accuracy: 0.9990
Epoch [8/10], Loss: 0.0121, Accuracy: 0.9992
Epoch [9/10], Loss: 0.0104, Accuracy: 0.9994
Epoch [10/10], Loss: 0.0097, Accuracy: 0.999
```

Adagrad optimizer3 - with attributes set as lr=0.0001, weight decay=0.0001, initial accumulator value=0.001:

```
Epoch [1/10], Loss: 0.0091, Accuracy: 0.9998
Epoch [2/10], Loss: 0.0092, Accuracy: 0.9998
Epoch [3/10], Loss: 0.0115, Accuracy: 0.9994
```

```
Epoch [4/10], Loss: 0.0090, Accuracy: 0.9996
Epoch [5/10], Loss: 0.0106, Accuracy: 0.9988
Epoch [6/10], Loss: 0.0094, Accuracy: 0.9994
Epoch [7/10], Loss: 0.0083, Accuracy: 0.9998
Epoch [8/10], Loss: 0.0089, Accuracy: 0.9998
Epoch [9/10], Loss: 0.0109, Accuracy: 0.9994
Epoch [10/10], Loss: 0.0086, Accuracy: 0.9994
```

The optimizer with varied attributes gave almost the same accuracy with minimal loss.

RMSprop optimizer

RMSprop optimizer is a variation of Adagrad that performs well on problems with non-stationary or noisy gradients. RMSprop uses an exponentially decaying average to give more weight to recent gradients and less weight to past gradients.

RMSprop optimizer1 - with attributes set to lr=0.01, alpha=0.9, weight decay=0.01:

```
Epoch [1/10], Loss: 2.1017, Accuracy: 0.3326
Epoch [2/10], Loss: 1.9456, Accuracy: 0.2358
Epoch [3/10], Loss: 1.7963, Accuracy: 0.2686
Epoch [4/10], Loss: 1.7699, Accuracy: 0.2832
Epoch [5/10], Loss: 1.7581, Accuracy: 0.2858
Epoch [6/10], Loss: 1.7462, Accuracy: 0.2898
Epoch [7/10], Loss: 1.7387, Accuracy: 0.2862
Epoch [8/10], Loss: 1.7445, Accuracy: 0.2756
Epoch [9/10], Loss: 1.7212, Accuracy: 0.2930
Epoch [10/10], Loss: 1.7289, Accuracy: 0.2878
```

RMSprop optimizer2 - with attributes set to lr=0.001, alpha=0.95, weight decay=0.001:

```
Epoch [1/10], Loss: 1.6503, Accuracy: 0.3196
Epoch [2/10], Loss: 1.6107, Accuracy: 0.3246
Epoch [3/10], Loss: 1.5860, Accuracy: 0.3434
Epoch [4/10], Loss: 1.5741, Accuracy: 0.3442
Epoch [5/10], Loss: 1.5402, Accuracy: 0.3784
Epoch [6/10], Loss: 1.4745, Accuracy: 0.4050
Epoch [7/10], Loss: 1.4277, Accuracy: 0.4326
Epoch [8/10], Loss: 1.3835, Accuracy: 0.4500
Epoch [9/10], Loss: 1.3275, Accuracy: 0.4920
Epoch [10/10], Loss: 1.2979, Accuracy: 0.4924
```

RMSprop optimizer3 - with attributes set to lr=0.0001, alpha=0.99, weight decay=0.0001:

```
Epoch [1/10], Loss: 1.1965, Accuracy: 0.5482
Epoch [2/10], Loss: 1.1301, Accuracy: 0.5782
Epoch [3/10], Loss: 1.1175, Accuracy: 0.5818
```

```
Epoch [4/10], Loss: 1.0913, Accuracy: 0.5912
Epoch [5/10], Loss: 1.0756, Accuracy: 0.6006
Epoch [6/10], Loss: 1.0591, Accuracy: 0.6038
Epoch [7/10], Loss: 1.0342, Accuracy: 0.6202
Epoch [8/10], Loss: 1.0035, Accuracy: 0.6296
Epoch [9/10], Loss: 0.9602, Accuracy: 0.6498
Epoch [10/10], Loss: 0.9370, Accuracy: 0.6620
```

Our model performs best with Adam optimizer with a training accuracy of 99.92% and a loss of 0.0056. Other models with Adagrad and RMSprop optimizers gave relatively less accuracy.