JOSHUA ADEYEMO (21077839) Report on CIFAR-10 Image Classification Model

Introduction

In this project, we trained a convolutional neural network (CNN) to classify images in the CIFAR-10 dataset. The dataset consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. We used a CNN architecture with six convolutional layers, three max pooling layers, and two fully connected layers to train the model.

TASK 1

Model Performance

The model was trained for 50 epochs using the SGD optimizer with a batch size of 128. The default learning rate and Sparse Categorical Crossentropy loss function were used, while the accuracy metric was used to monitor model performance during training.

During the first epoch, the model had an accuracy of 13.22% on the training set and an accuracy of 17.21% on the validation set. As training continued, the accuracy on both sets gradually improved. After 50 epochs, the model achieved a training accuracy of 92.45% and a validation accuracy of 60.70%. The loss on both the training and validation sets decreased as training progressed, indicating that the model was learning from the data. The model had a training loss of 0.2251 and a validation loss of 1.9163 after 50 epochs.

The model appeared to plateau in terms of accuracy after approximately 30 epochs, as the improvement in validation accuracy slowed significantly after this point, while the train accuracy increased progressively. The model has achieved a higher training accuracy (0.9245) compared to the validation accuracy (0.6070). This suggests that the model is over-fitting to the training data, and it may not generalize well to new data. Additionally, the loss curve shows that the model's training loss decreases steadily over time, while the validation loss stops improving after a few epochs. This is another indication that the model is over-fitting to the training data.

Analysis of Correct and Incorrect Predictions

To gain insight into the model's performance, I created plots showing example test images from each class that were correctly and incorrectly labelled by the model. From the plots, I noticed that the model was particularly confident in predicting the "deer" class, often predicting it even when the correct label was a different class which shows that the model is over-fitting to certain features of the deer class.

In conclusion, the model achieved reasonable performance on the CIFAR-10 dataset. However, there is still room for improvement, as the validation accuracy was significantly lower than the training accuracy. This can be addressed by adjusting the learning rate or using a different optimizer to improve convergence.

TASK 2

I used the CIFAR dataset and trained a deep learning model for 20 epochs using three different optimizers: SGD, Adam, and RMSprop. For each optimizer, i used three different learning rates: 0.01, 0.001, and 0.0001. I used these learning rates because It's generally not recommended to use a high learning rate like 0.1 for training convolutional neural networks on image datasets like CIFAR because it can cause the optimization process to be unstable and the model may fail to converge to a good solution. I then recorded the accuracy and loss on both the validation and test sets. I also plotted the loss and accuracy for each optimizer and learning rate.

Test Result

The results of the experiment suggest that the choice of optimizer and learning rate has a significant impact on the performance of the model. According to the plots, Adam with a learning rate of 0.001 gave the best accuracy on the test set, followed closely by RMSprop with the same learning rate. SGD, on the other hand, gave the worst performance for all learning rates. Interestingly, for Adam, a learning rate of 0.01 led to the worst performance, while for RMSprop, it was a learning rate of 0.0001.

Furthermore, we can observe instances of over-fitting and under-fitting in the results. For example, the model trained with SGD and lr=0.01 achieved a high training accuracy of 0.6315, but a lower validation accuracy of 0.5892, indicating some degree of over-fitting. Similarly, the model trained with RMSprop and lr=0.0001 achieved a low training accuracy of 0.1307, which suggests under-fitting.

In summary, these results suggest that the choice of optimizer and learning rate can have a significant impact on the performance of a model on a given dataset, and it is important to carefully choose these hyperparameters to optimize model performance.

TASK 3

I trained the Cifar model with five different batch sizes: 16, 32, 64, 128 and 256. The model was trained using the sparse categorical cross-entropy loss function and the Adam optimizer with a learning rate of 0.001. The Adam optimizer and learning rate was used because it gave the best result for my previous task. The training was stopped after 50 epochs.

After training the model, the results showed from the plots that the time per epoch decreased with increasing batch size, also the total training time decreases as the batch size increases, with a total training time of 141.2 seconds for batch size 16, compared to 17.7 seconds for batch size 256. This is because larger batch sizes can process more data per step, leading to a faster convergence per epoch as evidenced by the decreasing test loss and increasing test accuracy with larger batch sizes in our result .

Overall, the choice of batch size depends on the trade-off between faster convergence and longer training time per epoch. In this case, a batch size of 128 seems to strike a good balance between the two, as it achieved a relatively high test accuracy with a reasonable training time per epoch and total training time.

TASK 4

Comparison of Results:

The second model performed better than the first model in terms of both training and validation accuracy. The train accuracy increased from 0.9245 to 0.9313, while the validation accuracy increased from 0.6070 to 0.7701. This improvement can be attributed to the use of the Adam optimizer, which has been shown to converge faster than SGD. The learning rate of 0.001 was also found to be optimal for this dataset.

The second model also had a lower validation loss than the first model (1.2397 compared to 1.9163), indicating that the model was better able to generalize to new data. This is supported by the confusion matrix, which showed that the model made more accurate predictions across all classes, compared to the first model.
The model had higher predicted numbers for all classes, indicating that it was able to better distinguish between different types of images. Additionally, the second model's use of dropout and L2 regularization helped to prevent over-fitting and improve the model's ability to generalize to new data.

Regarding the differences in the predicted classifications, the over-fitting to the deer class in the first model may have been due to the model picking up on certain features of the deer class that were not present in other classes. The use of dropout and L2 regularization in the second model helped to prevent the model from relying too heavily on certain features of the deer class, allowing it to make more accurate predictions across all classes.
Overall, the second model's use of the Adam optimizer, optimal learning rate, and regularization techniques led to improved accuracy, generalization, and ability to distinguish between different types of images.

Notebook Link: https://colab.research.google.com/drive/1b-9ArkzMonUXyrhmicGEubTD0ktaN1B9?usp=sharing