Atul Aravind Das

U73275808

MSDS

DS542 MIDTERM CHALLENGE

COMPREHENSIVE AI DISCLOSURE STATEMENT

ChatGPT was used to resolve errors and ask for suggestions to enhance the model and to also understand how the code of the CNNs works.I also learnt more about CosineAnnealingLR and its behavior. I also used it to get valuable suggestions for Part 3 of the midterm. Thus, AI was used as an aid for my work.

MODEL DESCRIPTIONS

Part 1 SimpleCNN(5 epochs)

The first part involved was the creation of the SimpleCNN, a Simple Neural Network that was designed to be run on the CIFAR100 dataset. This code involved the creation of a convolution neural network. This neural network is made up of four convolutional blocks, each of which has a convolutional layer, a max pooling layer to reduce spatial dimensions, batch normalization, and a ReLU activation function. The first block generates 32 feature maps from a 3-channel input, which is common for RGB images.

To preserve spatial dimensions before pooling, subsequent blocks gradually increase the number of output channels to 64, 128, and 256, respectively, while keeping the same kernel size of 3 and padding of 1.

The feature maps are flattened into a vector of size 256 × 2 × 2 (assuming the input image size is 32 × 32) following the last convolutional block. This vector is then passed through two fully connected layers.

The final layer generates 100 logits, which correspond to the 100 classes in the CIFAR-100 dataset, while the first fully connected layer contains 512 units with a ReLU activation that ensures effective performance of multi-class image classification by capturing progressively more complex features at each layer.

SimpleCNN Observations

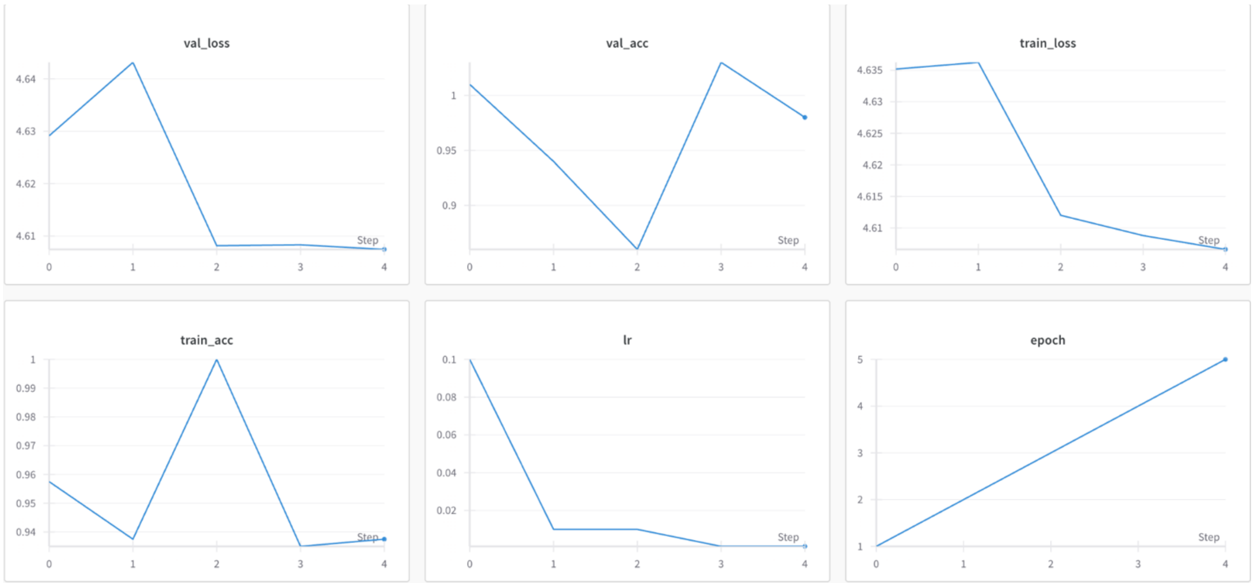

Figure 1: WandB Observations for the 5-epoch SimpleCNN

The model is learning, albeit very slowly, as evidenced by the training loss and validation loss, which both begin slightly above 4.63 and gradually decline. The fact that the drop is not significant suggests that learning is very low and may be experiencing underfitting.

The metrics for training and validation accuracy are abnormally high; they are close to or even higher than 1.0, which isn't feasible in practical situations. This is the result of calculating accuracy over a small sample.

There is also a presence of high variance across epochs.As a result of a StepLR scheduler, the learning rate (lr) begins at 0.1 and rapidly decreases after the first epoch. Loss is hardly decreasing after epoch 2 because the learning rate is so low by epoch 4 (near 0.001) that the updates are too subtle to make a significant difference.

PART 2(ResNet-34 for 100 Epochs)

The second part of the midterm involved implementing the ResNet-34 network, which has been run for 100 epochs. Regarding ResNet-34, it is an image classification model structured as a 34-layer convolutional neural network pre-trained on the ImageNet dataset, which contains over 100,000 images across 200 different classes. Another factor to consider is that a new function called set_seed has been defined. This function establishes a fixed random seed across all relevant libraries to ensure experiment reproducibility. It sets the seed for Python's built-in random module (random.seed), PyTorch (torch.manual_seed), and NumPy (np.random.seed) so that randomness in weight initialization, data shuffling, and other stochastic processes remains consistent throughout runs. A set_seed function was also created to enhance performance and find the optimal seed to be used for CUDA GPUs

ResNet34 Observations

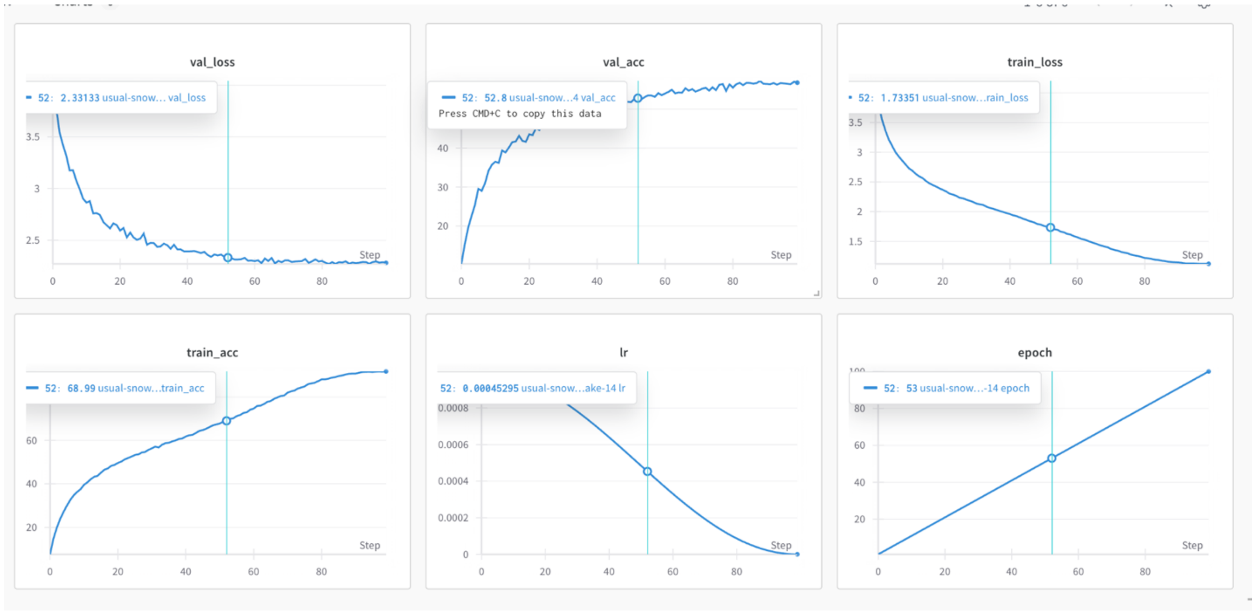

Figure 2: WandB Observations for ResNet-34

The CIFAR-100 dataset was used to train the ResNet34 model for 100 epochs, and the performance metrics show a strong baseline with encouraging outcomes. By the last epoch, the training loss had smoothly decreased from about 4 to 1.73, displaying a steady downward trend. This indicates that the model learned from the training data in an efficient manner. Strong performance on the training set was demonstrated by the training accuracy, which rose gradually to about 68.99%.

On the validation side, the loss plateaued after about 50 epochs after dropping from a high starting value to about 2.33. This suggests a possible convergence as improvements on the validation set started to level off while the model was still learning on the training set. Similar trends were seen in the validation accuracy, which increased quickly in the early epochs before leveling off at about 52.8%.

With a small discrepancy of about 16% between training and validation accuracy—a common indication of mild overfitting, but nothing particularly alarming—this indicates the model was able to generalize to unseen data well. The learning rate plot, which progressively lowers the rate from roughly 0.0009 to almost zero, validates the use of a learning rate scheduler. This probably helped the model converge steadily and reduce oscillations in loss in subsequent epochs.

PART 3(ResNet-50 for 100 Epochs)

For the third part of the project, the ResNet-50 model was used. Like ResNet-34, the ResNet-50 was also trained on the ImageNet dataset. This neural network was initially pretrained in nature and initially, the default pretrained weights were used. After using the initial weights, the plan involved dividing the process into two phases. Phase 1 involved training only the final classification head layer in the 100 classes present in CIFAR100 and all the other layers were frozen. This was done for 10 epochs. After this phase, Phase 2 unfreezed the other layers and it was run for 100 iterations. This enhanced the model's performance and made it the best model to be used for the process,

ResNet50 Observations

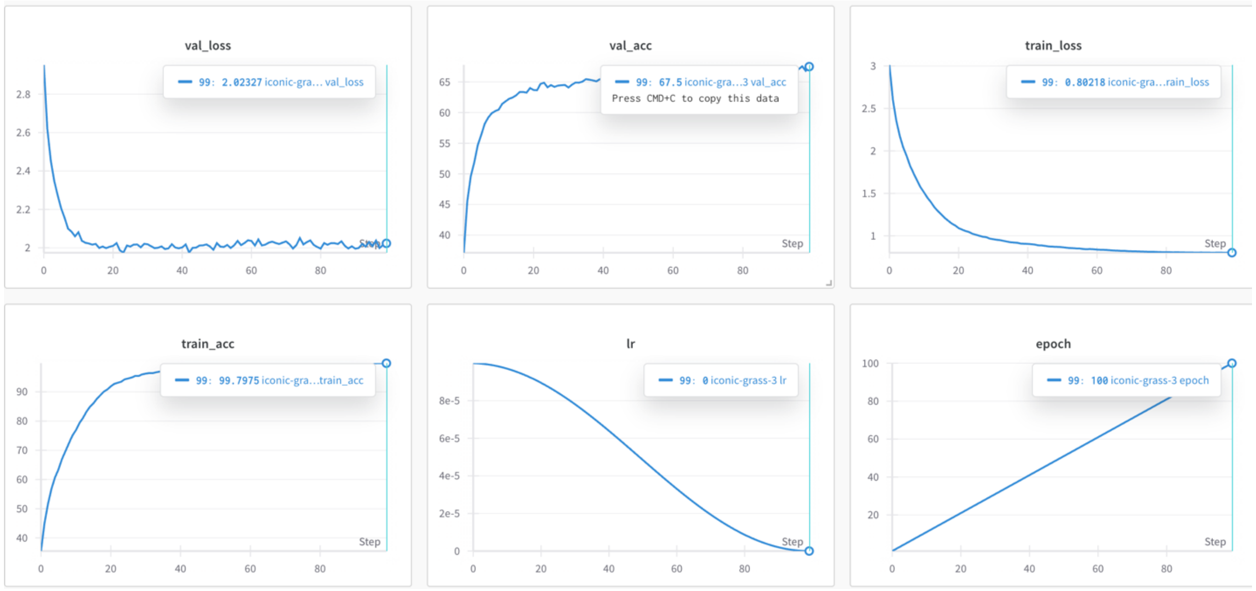

Figure 3: WandB Observations for the 100 epoch ResNet-50

The model has converged 100 epochs, according to the training curves. While the validation loss stabilizes at 2.02, the training loss gradually drops to about 0.80, indicating that the model is learning but may be overfitting a little. There is a generalization gap as the validation accuracy peaks at about 67.5% while the training accuracy approaches 99.8%. By the end of training, the learning rate will have tapered smoothly from a high value to almost zero, following a cosine decay schedule. Although the model performs exceptionally well on the training set, these metrics indicate that the validation performance could be enhanced, perhaps by regularization, data augmentation, or model architecture improvement.

HYPERPARAMETER TUNING

Considering the ResNet-50 pretrained model, hyperparameter tuning was implemented. In the first phase, the learning rate was 1e-3 and we used it to make the classification head layer learn more about the CIFAR100 dataset and make it adaptable to it. For phase 2, 1e-4 was used considering the number of epochs being increased to 100 as the layers were unfreezed and more training was required. Considering the GPU memory and training speed trade-off, the Batch Size was set at 128. The Adam optimizer was chosen for faster convergence with transfer learning and the CosineAnnealingLR scheduler was used to prevent the model from overshooting the minima.

REGULARIZATION TECHNIQUES

To improve the model's generalization and avoid overfitting, a variety of regularization techniques were used. First, the Adam optimizer (with a value of 5e-4) was modified to include weight decay, which penalizes large weights and aids in lowering model complexity, thereby minimizing overfitting. Furthermore, the cross-entropy loss function (CrossEntropyLoss(label_smoothing=0.1)) employed label smoothing, which softens the target distribution and prevents the model from becoming overconfident in its predictions. Improved generalization is frequently the result, particularly in multi-class classification tasks such as CIFAR-100. Additionally, the model was manually saved only when the validation accuracy increased over epochs, implementing an early stopping logic. In addition, the model was manually saved only when the validation accuracy increased over epochs, implementing an early stopping logic. This successfully reduced the risk of over-training by guaranteeing that the model that performed the best on unseen data was kept. When combined, these regularization strategies produced a model that was more dependable and resilient.

DATA AUGMENTATION STRATEGY

RandomHorizontalFlip() makes the model invariant to horizontal flips, while RandomCrop(32, padding=4) introduces localized variations in image positioning for training. Furthermore, to improve robustness, ColorJitter was employed with moderate settings to mimic variations in lighting. The pretrained ResNet50 backbone's expectations were then met by normalizing each image using ImageNet statistics. Only cropping and normalization were used for testing and validation to preserve data integrity and guarantee consistency.

RESULTS ANALYSIS

To compare the results, the wandb experiment tracking graphs were used to determine the best model from the three models. The results are as follows

Table 1: Table of Accuracies and Losses

Model Name	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
SimpleCNN	0.9375	0.98	4.6066	4.60743
ResNet-34	91.7325	56.77	1.12046	2.2807
ResNet-50(pretrained)	99.7975	67.5	0.802175	2.02326

From the above table, we can notice that the best performing model is the ResNet-50(pretrained) model. It has the greatest training and validation accuracies among the three models and the least training and validation losses among the three models. There are many other factors to be considered as well. The SimpleCNN had very less layers and very less number of iterations as well when compared to the other two models and this explains why it has the highest losses and the least accuracies. The models have an increasing number of layers as it moves from Part 1 to Part 3 as well.

LIMITATIONS AND FUTURE SCOPE

The analysis has explained intricate details about the most effective model among the three models. However, there are some limitations. The training and validation accuracies have differences and so this is a possible indicator of overfitting. Even though there is presence of faster convergence, the model can further benefit with an increase in fine-tuning of more layers and also a possibility of longer training.

EXPERIMENT TRACKING SUMMARY

Weights and Biases(wandb) was used for experiment. Three different projects were created namely. -sp25-ds542-challengefor the SimpleCNN, sp25-ds542-challenge, for ResNet-34, and Resnet_50_Part_3 for ResNet-50. The figures(Figure 1, Figure 2 and Figure 3) are the key graphs generated by WandB during the runtime of the three neural networks. These key graphs represent the important metrics such as losses, accuracies and epochs. The table Table 1, represents a comprehensive view of the important metrics of all three models and the metric values have been used to come to the decision of the best model.

CONCLUSION

Using the CIFAR-100 dataset, this midterm challenge, in summary, offered insightful information about developing and assessing convolutional neural networks for image classification. Every stage showed gains in model performance, from the basic SimpleCNN to the sophisticated pretrained ResNet-50 model, underscoring the importance of architectural depth, transfer learning, and hyperparameter tuning. Future improvements could concentrate on addressing overfitting and investigating longer training or more reliable data augmentation techniques, even though ResNet-50 proved to be the best-performing model. All things considered, this project has greatly expanded my knowledge of deep learning processes and useful model optimization.