Neural Networks and Deep Learning Coursework: CIFAR-10 Classification

Name: Sunantha Kannan
Student ID Number: 230841125

Task 1: Reading Dataset and Creating Data Loaders:

Initially, we defined train and test dataset and turning it into tensors and passed it to base line model. But accuracy is not good, so the advanced architecture has been built with transformed test and train dataset to improve accuracy.

Defining transformations on train and test data:

- **Train Transformations** includes random horizontal flipping, random cropping (image size of 32x32) with padding, converting the image to a tensor, and normalising the tensor with mean and standard deviation values of (0.5, 0.5, 0.5).

- **Test Transformations** includes converting the image to a tensor and normalisation with the same mean and standard deviation values.

Downloaded the CIFAR-10 dataset and applies the transformation defined above. The train_transformed_dataset is used for training the model and test_transformed_dataset is used for validation.

Finally, the Dataloader splits the datasets into batches of 64 which is used to efficiently load and process the dataset for training and testing the model.

Task 2: Creating the Model

Constructed the model based on the defined architecture with series of blocks and each will execute image parallelly with 'N' number of layers in the block.

Basic Architecture Model:

The model begins with a `**BasicIntermediateBlock**` that applies convolutions, batch normalization, and ReLU activation to start the feature extraction process. Follows the initial block with multiple `**BasicIntermediateBlock**` instances, each refining the features further. These blocks are stacked to deepen the network's capacity to analyse and interpret complex image features. Each block calculates a set of weights from the mean of the features, which are used to emphasize important features in the output through a weighted sum of the convolution outputs. The final `**BasicOutputBlock**` consolidates the deeply processed features into final predictions. It uses fully connected layers followed by a ReLU activation to map the extracted features to the output classes, finalizing the classification.

Note: As baseline model leads to overfitting and created advanced model by including Max pooling layers, changing activation function from Relu to leakyRelu, dropout rate to prevent overfitting.

Advanced Architecture Model:

Conv Block Initial: IntermediateBlock

This is the first stage in processing the input images:
Adaptive Average Pooling: Simplifies each feature map to a single average value, reducing complexity.
Convolutional Layers: A series of layers that apply filters to extract features, normalize outputs, activate non-linearly(**LeakyReLU** function that allow a small gradient when the unit is not active), reduce spatial dimensions (via pooling), and randomly disable connections (**dropout**) to enhance model and prevent overfitting by regularization.
Fully Connected Layer: Transforms pooled features to create a set of weights for prioritizing certain features over others.

Intermediate Blocks:

These blocks repetitively refine the features extracted by the initial block. Each block uses the same configuration as the initial block to further process features. This includes convolution, batch normalization, activation, pooling, and dropout. By applying multiple such blocks, the network can learn more complex features for accurate image classification.

Output Block

The final component that prepares the model's output for classification:

Linear Layers: Processes the features from the last intermediate block through several layers, maintaining feature complexity and applying non-linear activation.

Final Classification Layer: Reduces the dimensionality to the number of target classes, producing the final classification scores.

Task 3: Creating Loss Function, Optimiser:

Cross Entropy Loss Function: CrossEntropyLoss is used for our model training because it considers the probabilities assigned by the model to each class. It penalises the model more heavily for assigning low probabilities to the correct class, and less heavily for assigning high probabilities to incorrect classes.

AdamW Optimiser: We have used AdamW optimizer as it applies weight decay. AdamW optimizer applies weight decay, which is a regularisation technique that helps to prevent overfitting. In our multi-classification, overfitting can occur when the model learns the training data too well and fails to generalise to unseen data. Therefore, Weight decay has helped to prevent overfitting by forcing the model to have smaller weight.

Task 4: Training and Validation Function for Model Evaluation:

accuracy_fn:

- Compares the true labels (`y_true`) with the predicted labels (`y_pred`).
- Counts how many predictions are correct.
- Calculates the accuracy percentage based on the correct predictions.

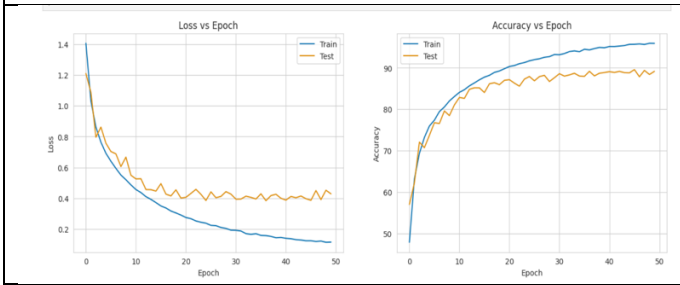
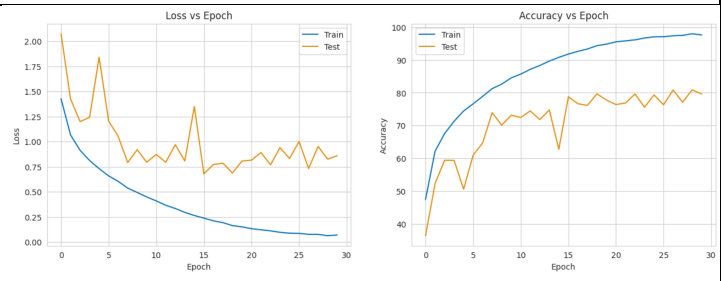
Train Step (Training the model)	Test step (Evaluating the model)
Set the model to train mode to stop certain features during training.	Set the model to evaluation mode to stop certain features to ensure model consistent
Batch Processing: Goes through each batch of training data, where it: • Predicts outcomes using the model's **forward** pass. • Calculates how incorrect these predictions are compared to the true data (**loss**). • Resets stored changes from the previous batch to ensure they don't affect the current batch's learning. • Updates the model to decrease future prediction errors (**backpropagation**). • Adjusts the model's internal settings based on the calculated loss (**optimizer step**).	**Batch Processing**: Similarly processes each batch of testing data, where it: • Uses the model to predict outcomes without making any changes to the model itself. • Calculates the **loss** to evaluate how the model performs on data it hasn't learned from. • Unlike training, it doesn't adjust the model after each batch since the goal is to evaluate, not to improve.
After training on all batches, reports the average training loss and accuracy describing how well the model is learning	Provides the average test loss and accuracy, indicating the model's generalization capability on new data

HYPERPARAMETERS:

The CIFAR-10 dataset contains colour images, so we use 3 input channels corresponding to the RGB values. We chose 512 hidden units to allow the network to capture complex features and patterns. With 10 different image categories in CIFAR-10, our output shape is 10, and we use 4 blocks of 4 convolutional layers each to build a deep enough model for accurate classification. We've set the ADAMW learning rate at 0.001 for steady convergence and weight decay at 0.01 to balance learning speed with the model's ability to generalize well.

As we see the Epochs Running on Basic CNN, the model accuracy difference is more between train and test data. There is a gap between training and testing performance, which indicates overfitting. The basic CNN model may be memorizing the training data rather than learning general patterns that apply to the test data.

(Accuracy Vs Loss)

ADVANCED CNN ARCHITECTURE :	BASIC CNN ARCHITECTURE:
	
The loss and accuracies got improved when compared with Basic CNN, as we have included max pooling layers, changed loss function from Adam to AdamW and included dropout rate to prevent overfitting and the model is good. The model quickly learns to classify images correctly, as shown by the steep initial drop in loss and surge in accuracy. After the initial learning, both loss and accuracy stabilize, indicating consistent performance over time. The small gap between training and test results suggests the model is almost as good at classifying new images as it is with those it was trained on. The model shows effective learning with minimal overfitting, maintaining a steady increase in accuracy and a stable decrease in loss as training progresses.	Test accuracy fluctuates more than training accuracy, initially dropping but then increasing to 78.75% by epoch 16. Test loss also fluctuates but generally decreases, indicating variations in how the model performs on unseen data. The gap between training and test results suggests potential overfitting, where the model is learning the training data very well but not generalizing equally well to new data. While the training performance is overfitting at the end, there may be a need to improve the model's ability to generalize. This could involve techniques like data augmentation, dropout, regularization, or cross-validation

TASK 5: MODEL ACCURACY ON CIFAR-10 DATASET:

The **'AdvancedArchModel'** was trained over 50 epochs—iterations over the entire dataset.	The **'BasicArchModel'** was trained over 30 epochs—iterations over the entire dataset.

`{'model_name': 'AdvancedArchModel',`
`'model_loss': 0.39756661653518677,`
`'model_acc': 89.64968152866243}`
	`{'model_name': 'BasicArchModel',`
	`'model_loss': 0.8601754903793335,`
	`'model_acc': 79.60788216560509}`

Model Accuracy: The model is correctly identifying 89.64% of the images in the test dataset. This high percentage suggests that the model is well-trained and effective at recognizing and classifying the images it was tested on. **Model Loss**: The loss value is approximately 0.398, which is relatively low. This means that, on average, the model's predictions are quite close to the actual values. A low loss is an indicator of good performance, especially when coupled with high accuracy.	The 'BasicArchModel' performs competently, with a decent accuracy rate indicating reliable predictions in the majority of cases. However, the loss value suggests that there are still discrepancies between the model's predictions and the actual outcomes, which leaves some scope for further refining the model to reduce errors.

`Epoch: 50`
`---------`
`Train loss: 0.13784 | Train accuracy: 95.21%`
`Epochs: 100%| | 50/50 [2:24:56<00:00, 173.94s/it]Test loss: 0.39757 | Test accuracy: 89.65%`
	`Epoch: 30`
	`---------`
	`Train loss: 0.06993 | Train accuracy: 97.61%`
	`Epochs: 100%| | 30/30 [2:39:42<00:00, 319.43s/it]Test loss: 0.86018 | Test accuracy: 79.61%`

.

[Note]: Due to page limitations, references are attached in the notebook.