Deep Learning Lab 4 Report

Shusrith S

PES1UG22AM155

February 9, 2025

1 CIFAR-10 Classification

1.1 Architecture Used

The model presented is a Convolutional Neural Network with Residual Blocks, inspired by the ResNet architecture. The model leverages residual learning, where shortcut connections help gradients flow through the network, allowing for deeper architectures without suffering from vanishing gradient issues. The network consists of an initial convolutional layer, followed by three stacked Residual Blocks with increasing channel sizes, concluding with global average pooling and a fully connected layer for classification.

1.2 Description of Architecture

The first layer is a 3×3 convolution with 64 filters, batch normalization, and ReLU activation, followed by three Residual Blocks, where each block consists of two convolutional layers with batch normalization and ReLU activation. A crucial component of these blocks is the skip connection, which helps retain information from previous layers, preventing degradation in performance as the network deepens.

1.3 Design Choices

One key design choice is the use of Residual Blocks, which improve the network's ability to train deeper architectures effectively. The shortcut connection allows direct flow of gradients during backpropagation, reducing the risk of vanishing gradients. Another important choice is the use of Batch Normalization after every convolution, which helps in stabilizing training and accelerating convergence. Additionally, Adaptive Average Pooling is used before the final fully connected layer, allowing the model to handle varying input sizes efficiently while reducing overfitting compared to a traditional flattening approach.

1.4 Effectiveness with CIFAR-10

CIFAR-10 images are small (32×32), so very deep networks with large receptive fields are not needed. Instead, the model captures important patterns step by step using multiple convolutional layers. Strided convolutions help reduce image size while keeping important details, making the network more efficient. The final fully connected layer gives 10 outputs, one for each class, making it suitable for classification. By using residual connections and batch normalization, this model trains more smoothly, learns faster, and performs better than a basic CNN.

1.5 Loss and Accuracy plots

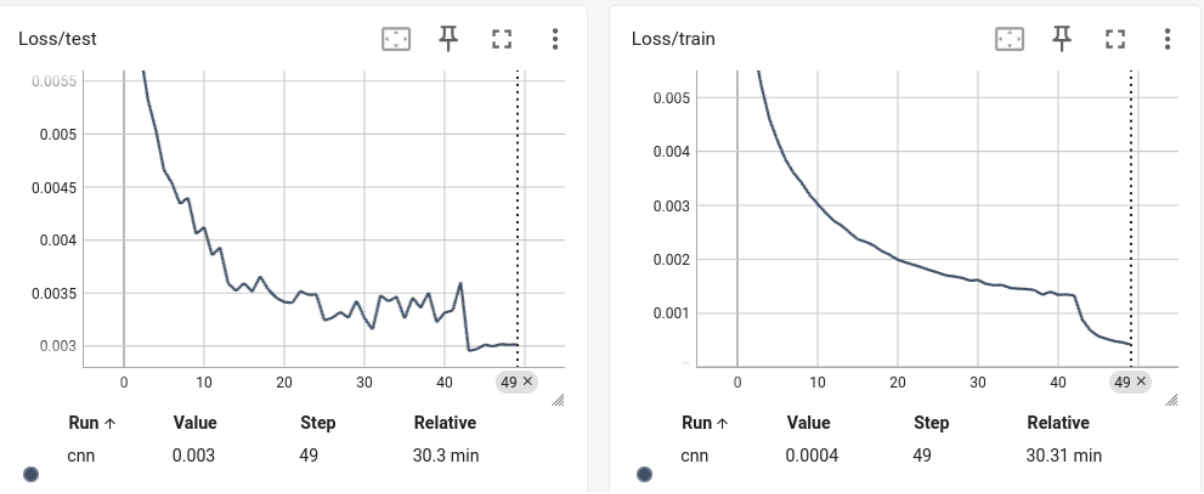

Figure 1: Train and test loss plots

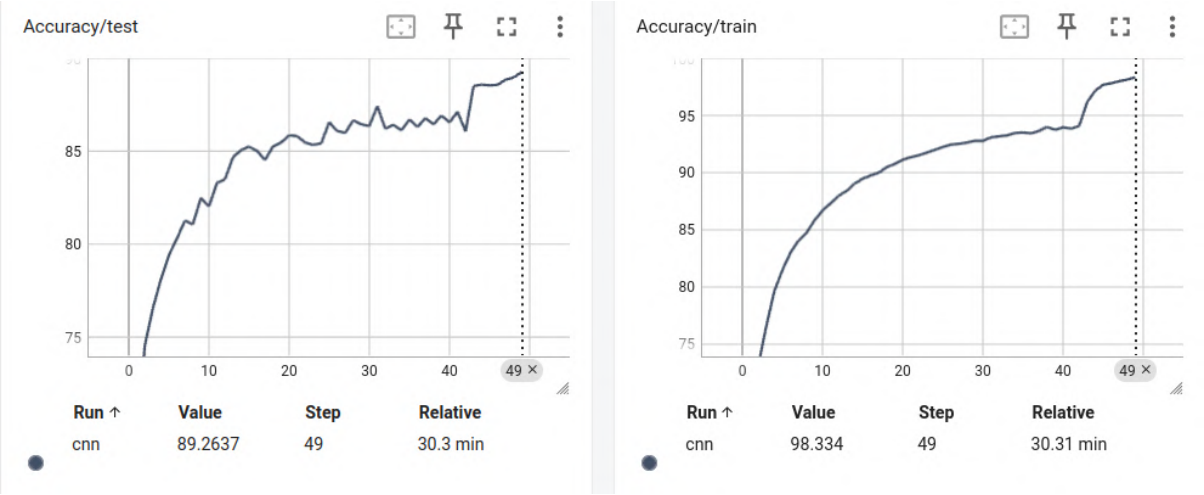

Figure 2: Train and test accuracy plot

1.6 Visualization of filters

The first five filters of the first convolutional layer have been visualised. The first convolutional layer is of shape (64 x 3 x 3) which gives us filters of shape (3 x 3). The first five filters have been selected and normalized between 0 and 1 and visualised. The filters of the first few layers capture simple patters and geometric features, which can be seen in the full black (0) and full white (1) regions of the filter, showing us that these filters select some areas while ignoring others.

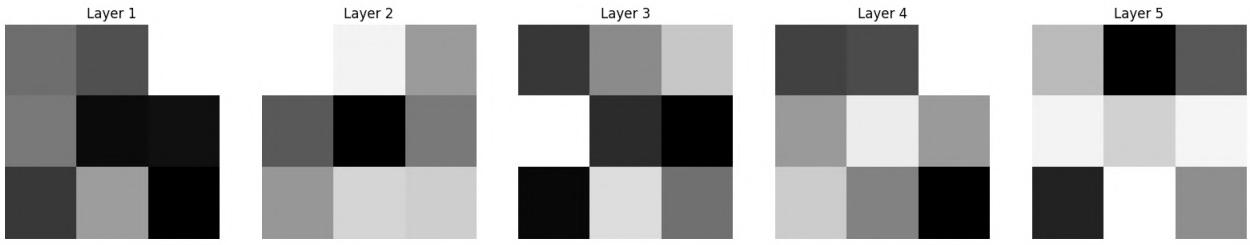

Figure 3: Visualization of the first five filters

1.7 Accuracy

The model trains with a final train accuracy of 98.33% and a a test accuracy of 89.26%. The final losses for train and test are 4×10^{-4} and 3×10^{-3}. It is visible that the model has overfit despite best attempts to include various regularization techniques like Batch Normalization, learning rate scheduling and weight decay. It is the nature of deeper Convolution Neural Networks to tend to overfit and hence it can be concluded that better skip connections and fewer parameters might be needed to reduce the overfitting.

2 Gradient ascent on InceptionV3

2.1 InceptionV3

InceptionV3 is a deep CNN for image classification that improves efficiency using factorized convolutions, asymmetric kernels, and batch normalization. It reduces computation while maintaining high accuracy, making it a strong choice for tasks like image classification. A pre-trained model is loaded, trained on the ImageNet1K dataset.

2.2 Class chosen

The class chosed is index 107, representing a jellyfish. This is a sample image for class 107.

Figure 4: Sample image for class 107 : jellyfish

2.3 Image generation with gradient ascent

A random image is chosen from a normal distribution of mean 0 and variance 1 of shape (3 x 299 x 299) in accordance to the input shape of the Inception model. The input image is converted to a leaf tensor and is declared to required gradients in order to perform gradient ascent over it. In each epoch, the negative of the predicted score for the class is considered as the loss and gradients are calculated. A learning rate of 0.1 is used to update the values in the input image and these values are clipped between 0 and 1 to ensure that the pixel values remain normalized for visualization in the future. This process was repeated over 10000 epochs. The final image generated is shown.

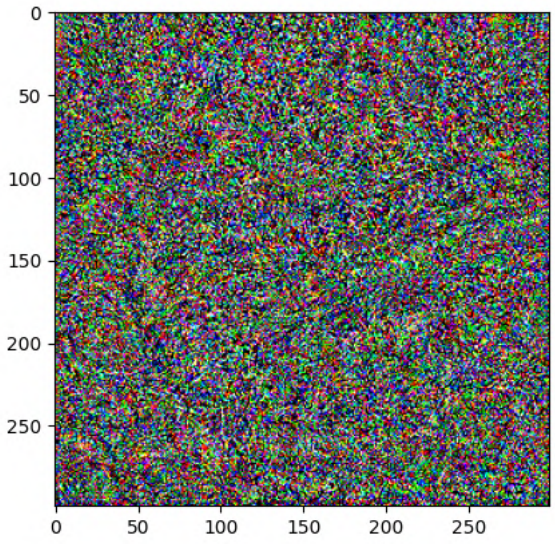

Figure 5: Image generated by gradient ascent

The final score of the class 107 on the image generated was 65.59, while the score at the first epoch was -1.33, showing effective maximization of the score over class 107.

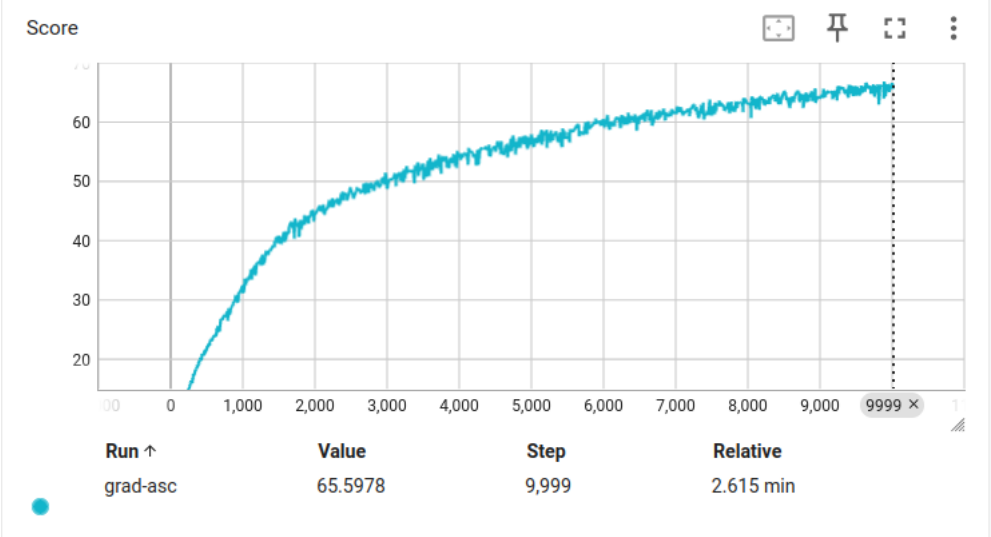

Figure 6: Score vs epoch