# Vision Transformer End Term Evaluation Report

**Krishna Soni (240565)**

February 8, 2026

## 1 Introduction

This report summarizes my work on implementing a Vision Transformer (ViT) model from scratch. Unlike standard convolutional models that process images through local pixel neighborhoods, the ViT architecture treats an image as a sequence of patches. I implemented this using PyTorch and conducted training on the CIFAR-10 dataset to evaluate how well a Transformer-based approach can generalize on smaller image data.

## 2 Data Preprocessing and Regularization

The CIFAR-10 dataset contains 60,000 color images ($32 \times 32$) distributed across 10 classes. Because Transformers lack the "inductive bias" of CNNs (like understanding that neighboring pixels are related), preprocessing and regularization were critical parts of my setup.

### 2.1 Preprocessing Pipeline

To improve the model's ability to learn orientation and translation invariant features, I applied several transformations:

- **Random Horizontal Flip:** Used to help the model recognize objects regardless of their direction.
- **Random Crop with Padding:** I added 4 pixels of padding before cropping back to $32 \times 32$, which helps the model handle slight shifts in object positioning.
- **Normalization:** I centered the pixel values around zero using the CIFAR-10 mean and standard deviation to stabilize the gradients.

### 2.2 Regularization Techniques

To prevent the model from simply memorizing the training set, I implemented the following:

- **Dropout:** Applied at a 0.1 rate to force the network to learn more robust, redundant features.
- **Weight Decay:** Used AdamW with a 0.05 decay factor to penalize excessively large parameters.
- **Gradient Clipping:** I enforced a maximum norm of 1.0 to maintain training stability and prevent exploding gradients.

## 3 Core Architecture and Mechanism

The defining feature of the ViT is "tokenization." Instead of looking at individual pixels, the model breaks the image into patches that it can process like words in a sentence.

### 3.1 Patch and Positional Embeddings

I divided each $32 \times 32$ image into $4 \times 4$ non-overlapping patches. This resulted in a sequence of 64 patches. Each patch was projected into a 384-dimensional vector. Since Transformers don't naturally know the order of tokens, I added learnable positional embeddings to give the model spatial context.

### 3.2 Self-Attention and Classification

The Multi-Head Self-Attention (MHSA) layer allows each patch to "talk" to every other patch. This means the model can learn global relationships across the entire image right from the first layer. I also used a special [CLS] token at the start of the sequence to represent the overall image features for the final classification.

Listing 1: Attention Implementation Snippet

```
# Scaled dot-product attention logic
attn_weights = (q_states @ k_states.transpose(-2, -1)) / math.sqrt(self
    .head_dim)
attn_weights = F.softmax(attn_weights, dim=-1)
# Dropout applied to attention weights
attn_weights = F.dropout(attn_weights, p=self.dropout, training=self.
    training)
```

## 4 Hyperparameter Optimization

I tested several configurations to find the best balance between performance and training time. Smaller patch sizes generally yielded better accuracy but significantly increased the computational cost per epoch.

| Parameter | Tested Values | Final Choice |
|---|---|---|
| Patch Size | 2, 4, 8 | 4 |
| Embedding Dim | 256, 384, 768 | 384 |
| Attention Heads | 4, 8 | 8 |
| Encoder Layers | 4, 6, 8 | 6 |
| Learning Rate | 1e-3, 3e-4, 1e-4 | 3e-4 |
| Batch Size | 64, 128 | 128 |

## 5 Final Results and Observations

After 50 epochs of training, the model reached the following metrics:

- **Training Accuracy:** 95.03%

- **Best Test Accuracy:** 79.30%

The high training accuracy shows the model's massive capacity to learn. However, the gap between training and test results suggests that Transformers really need larger datasets (like ImageNet) to shine. For a small dataset like CIFAR-10, achieving nearly 80% accuracy from scratch demonstrates that the model successfully learned meaningful spatial representations.

## 6 Conclusion

This project successfully demonstrated the implementation of a Vision Transformer from the ground up. By moving from convolutions to self-attention, the model learned to capture global image patterns. Future work could involve adding data augmentation techniques like MixUp or using a larger pre-trained backbone to further bridge the gap in test accuracy.