Vision Transformer

CIFAR-10 Image Classifier ViT

Kamana Gupta

EEA Winter Project

Contents

Abstract

This report presents the implementation and analysis of a Vision Transformer (ViT) model for image classification on the CIFAR-10 dataset. Since CIFAR-10 contains low-resolution images (32×32), strong preprocessing and regularization techniques were applied to improve generalization. The effects of hyperparameter tuning, regularization strategies, and the internal working mechanism of Vision Transformers are discussed along with experimental results.

1 Dataset and Preprocessing

The CIFAR-10 dataset consists of 60,000 RGB images of size 32×32 belonging to 10 classes. To improve robustness and generalization, the following preprocessing steps were applied:

- **Normalization**: Pixel values were normalized to stabilize training.
- **Random Cropping**: Images were randomly cropped with padding to introduce spatial variations.
- **Random Horizontal Flipping**: Improves invariance to orientation changes.

Effect: These preprocessing techniques significantly reduced overfitting and improved validation accuracy, which is crucial for Vision Transformers due to the absence of convolutional inductive bias.

2 Regularization Techniques

2.1 Dropout

Dropout was applied after patch embedding and within the MLP layers of each transformer encoder block. This prevents neuron co-adaptation and improves generalization.

2.2 Weight Decay

The AdamW optimizer was used with weight decay to penalize large weights and reduce overfitting.

2.3 Data Augmentation as Regularization

Random cropping and flipping act as implicit regularization by increasing dataset diversity.

Overall Impact:

- Without regularization: ∼55–60% accuracy
- With regularization: ∼70–78% accuracy

3 Hyperparameter Tuning

Several hyperparameters were tuned experimentally to achieve optimal performance:

- **Patch Size**: 4×4 patches
- **Embedding Dimension**: 256
- **Number of Transformer Layers**: 6

- **Number of Attention Heads**: 8
- **Learning Rate**: 3×10^{-4} with cosine annealing
- **Batch Size**: 128
- **Epochs**: 25

Effect: Proper tuning ensured stable convergence, improved representation learning, and prevented underfitting or overfitting.

4 Vision Transformer Mechanism

4.1 Patch Embedding

The input image is divided into fixed-size patches, each of which is projected into an embedding space. This converts the image into a sequence of tokens.

```python
class PatchEmbedding(nn.Module):
    def __init__(self, img_size, patch_size, in_channels=3, emb_dim=256):
        super().__init__()
        self.patch_size = patch_size
        self.n_patches = (img_size // patch_size) ** 2
        self.proj = nn.Conv2d(
            in_channels, emb_dim,
            kernel_size=patch_size,
            stride=patch_size
        )
    def forward(self, x):
        x = self.proj(x)
        x = rearrange(x, 'b c h w -> b (h w) c')
        return x
```

Figure 1: Patch embedding implementation

4.2 Positional Encoding

Since transformers are permutation invariant, learnable positional embeddings are added to preserve spatial information.

4.3 Class Token

A learnable class token ([CLS]) is appended to the patch sequence. This token aggregates global information and is used for classification.

4.4 Self-Attention Mechanism

Multi-head self-attention allows each patch to attend to every other patch, enabling global context learning.

```python
class TransformerBlock(nn.Module):
    def __init__(self, dim, heads, mlp_dim, dropout):
        super().__init__()
        self.norm1 = nn.LayerNorm(dim)
        self.attn = nn.MultiheadAttention(dim, heads, dropout=dropout, batch_first=True)
        self.norm2 = nn.LayerNorm(dim)
        self.mlp = nn.Sequential(
            nn.Linear(dim, mlp_dim),
            nn.GELU(),
            nn.Dropout(dropout),
            nn.Linear(mlp_dim, dim),
            nn.Dropout(dropout)
        )
    def forward(self, x):
        x = x + self.attn(self.norm1(x), self.norm1(x), self.norm1(x))[0]
        x = x + self.mlp(self.norm2(x))
        return x
```

Figure 2: Multi-head self-attention block

4.5 Classification Head

The output corresponding to the class token is passed through a linear layer to predict class probabilities.

5 Results

The model was trained for 25 epochs on the CIFAR-10 dataset.

- Training Accuracy: **71.18%**
- Test Accuracy: **69.95%**

The results demonstrate that Vision Transformers can achieve competitive performance on CIFAR-10 with sufficient regularization and tuning.

6 Conclusion

This work demonstrates the effectiveness of Vision Transformers for image classification on the CIFAR-10 dataset. Despite lacking convolutional inductive bias, careful preprocessing, regularization, and hyperparameter tuning enable strong performance. Future improvements may include pretraining, deeper architectures, or hybrid CNN–Transformer models.