Final Project Report

1. Introduction

Image classification is crucial in real-world applications, especially in the medical field, where it supports tasks like disease diagnosis, tumour detection, and treatment planning. While Convolutional Neural Networks (CNNs) have been the standard architecture for such tasks, they face challenges like limited global feature extraction and reliance on large, labelled datasets, which are often scarce in medical imaging. However, Vision Transformers (ViTs) address these limitations by capturing both local and global contextual information, making ViTs particularly effective in medical image analysis, where subtle patterns and relationships are critical for accurate diagnosis. Transfer Learning further enhances the applicability of ViTs by enabling fine-tuning of pre-trained models on specific tasks, reducing the need for extensive labelled datasets. In the medical domain, this approach facilitates high accuracy in tasks like identifying diseases in X-rays, CT scans, and MRI images.

I explored the integration of **Vision transformers** and **Transfer Learning** with fine-tuning different number of layers in the pre-trained weights of other models for image classification across diverse datasets, including **CIFAR-10**, **CIFAR-100**, **Fashion-MNIST**, **and Food-101,** drawing parallels to their potential in medical imaging. By leveraging pre-trained models and data augmentation techniques, the goal is to achieve high accuracy while addressing challenges like subtle feature distinctions. The report will analyse model performance, training time, accuracy and loss curves.

The following was the final project timeline plan at the beginning of the semester:

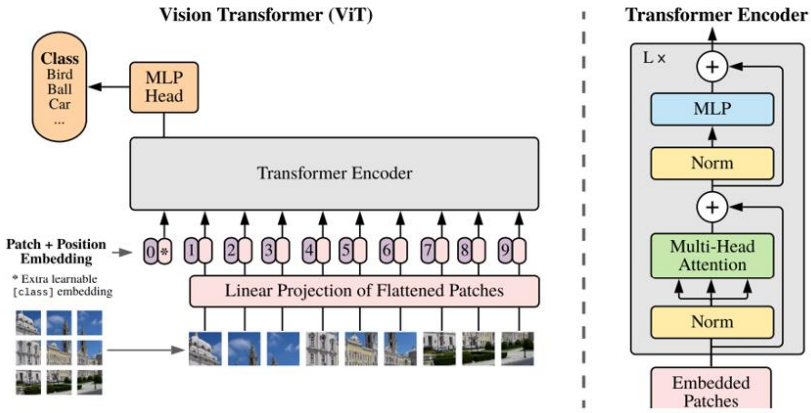
➢ **Project Timeline:**

Month	Task
Sep.	Overall project outline and dataset selection
Oct.	- ViT paper analysis. -Baseline coding implementation and model training on 1.Fashion-MNIST 2. CIFAR-10
Nov.	-Vision Transformer architecture implementation using PyTorch -Model Training on the other two datasets -Transfer Learning implementation
Dec.	(Model Deployment) **Final Report**

And the model/ project expectations which are all achieved:

- ✓ **To achieve 90%+ accuracy on Fashion-MNIST, CIFAR-10, and CIFAR-100 dataset**
- ✓ **85%+ accuracy on Food-101 dataset**

Starting with the Vision Transformer paper analysis: I tried to implement **ViT-Base** model from the paper as you can see the highlighted part from the following image.



Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Table 1: Details of Vision Transformer model variants.

Four Main Equations mentioned in the paper:

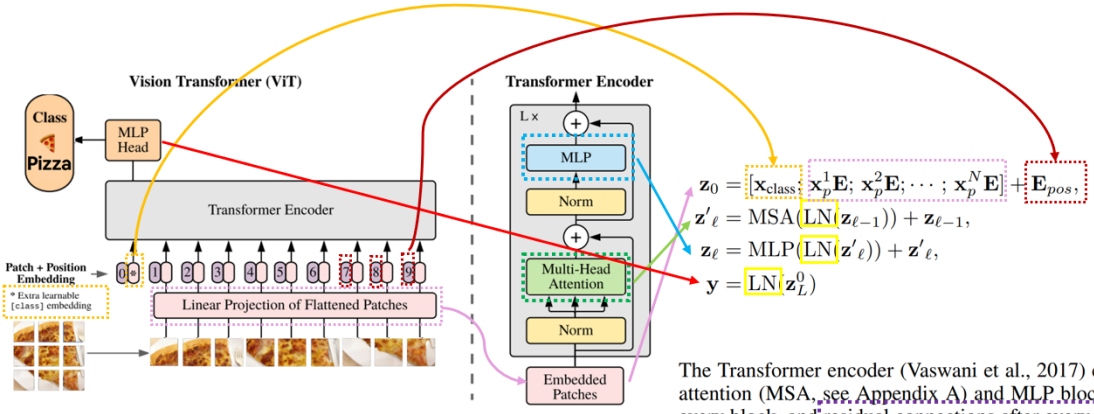


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Source: ViT paper Figure 1

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \cdots; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \quad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D},\ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \tag{1}$$

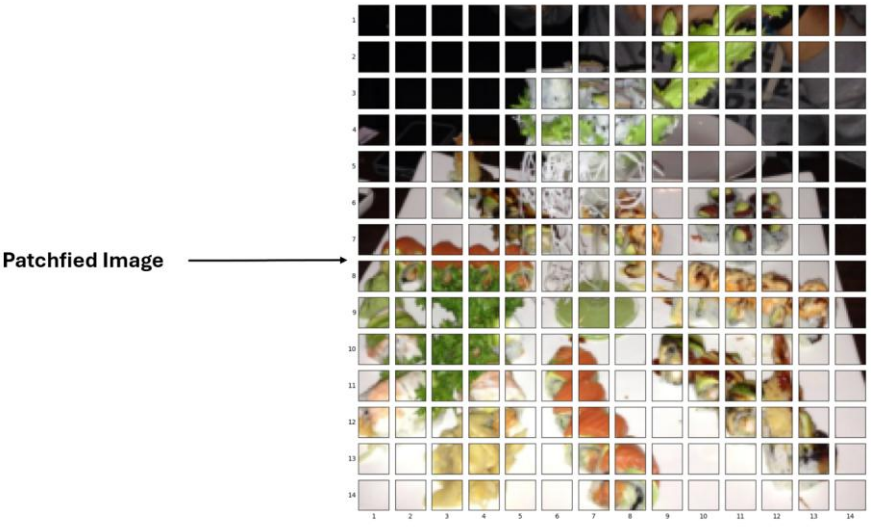
$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \quad \ell = 1 \ldots L \tag{2}$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \quad \ell = 1 \ldots L \tag{3}$$

$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0) \tag{4}$$

The Transformer encoder (Vaswani et al., 2017) consists of alternating layers of multiheaded self-attention (MSA, see Appendix A) and MLP blocks (Eq. 2, 3). Layernorm (LN) is applied before every block, and residual connections after every block (Wang et al., 2019; Baevski & Auli, 2019).

Source: ViT paper section 3.1



After analysing all the necessary equations, I started implementing the ViT architecture using PyTorch and update the code to suit Food101 classification dataset. And here is the result after training 20 epochs:

```
Epoch 1  | train_loss: 4.7245 | train_acc: 0.0105 | test_loss: 4.6472 | test_acc: 0.0097
Epoch 2  | train_loss: 4.5705 | train_acc: 0.0188 | test_loss: 4.4500 | test_acc: 0.0333
Epoch 3  | train_loss: 4.4023 | train_acc: 0.0354 | test_loss: 4.3308 | test_acc: 0.0429
Epoch 4  | train_loss: 4.3926 | train_acc: 0.0389 | test_loss: 4.3775 | test_acc: 0.0433
Epoch 5  | train_loss: 4.3177 | train_acc: 0.0501 | test_loss: 4.2212 | test_acc: 0.0596
Epoch 6  | train_loss: 4.1234 | train_acc: 0.0742 | test_loss: 3.9411 | test_acc: 0.0980
Epoch 7  | train_loss: 3.8888 | train_acc: 0.1108 | test_loss: 3.6916 | test_acc: 0.1375
Epoch 8  | train_loss: 3.6827 | train_acc: 0.1479 | test_loss: 3.4517 | test_acc: 0.1866
Epoch 9  | train_loss: 3.5075 | train_acc: 0.1800 | test_loss: 3.3752 | test_acc: 0.1952
Epoch 10 | train_loss: 3.3596 | train_acc: 0.2054 | test_loss: 3.1290 | test_acc: 0.2447
Epoch 11 | train_loss: 3.1899 | train_acc: 0.2361 | test_loss: 3.0076 | test_acc: 0.2686
Epoch 12 | train_loss: 3.0439 | train_acc: 0.2664 | test_loss: 2.8767 | test_acc: 0.2955
Epoch 13 | train_loss: 2.9265 | train_acc: 0.2873 | test_loss: 2.7693 | test_acc: 0.3164
Epoch 14 | train_loss: 2.8076 | train_acc: 0.3123 | test_loss: 2.6929 | test_acc: 0.3311
Epoch 15 | train_loss: 2.6920 | train_acc: 0.3348 | test_loss: 2.5850 | test_acc: 0.3551
Epoch 16 | train_loss: 2.5980 | train_acc: 0.3562 | test_loss: 2.4801 | test_acc: 0.3746
Epoch 17 | train_loss: 2.5182 | train_acc: 0.3725 | test_loss: 2.4028 | test_acc: 0.3945
Epoch 18 | train_loss: 2.4420 | train_acc: 0.3880 | test_loss: 2.4150 | test_acc: 0.3925
Epoch 19 | train_loss: 2.3756 | train_acc: 0.4011 | test_loss: 2.3592 | test_acc: 0.4068
Epoch 20 | train_loss: 2.3197 | train_acc: 0.4138 | test_loss: 2.4081 | test_acc: 0.3948
Training Progress: 100%|
[INFO] Total training time: 2:01:20.124661
```

I trained it only for 20 epochs due to time constraints, if trained more epochs, higher accuracy could be achieved. (GPUs used: 4-6 **TITAN V (12 GB)**)

2. Method and Results

CIFAR 10

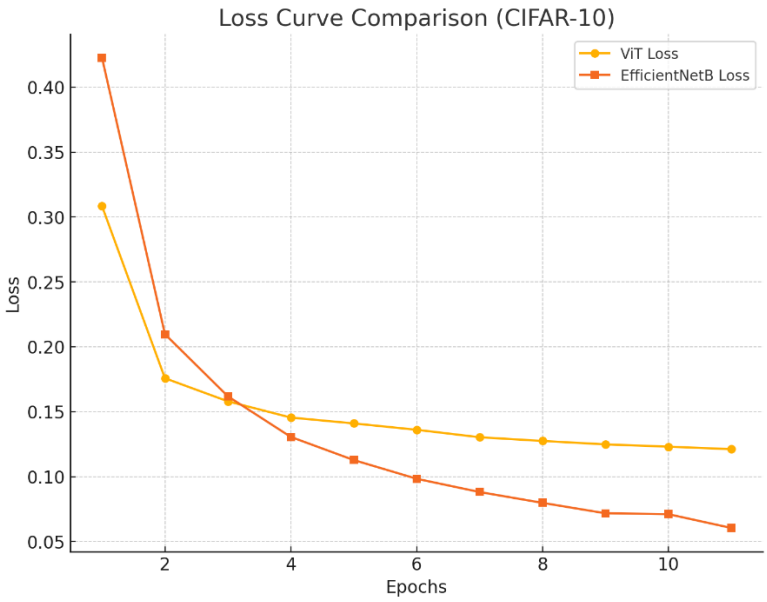
Code file: effnet_B0_train_CIFAR_10.py | fine-tuning the last 2 feature layers

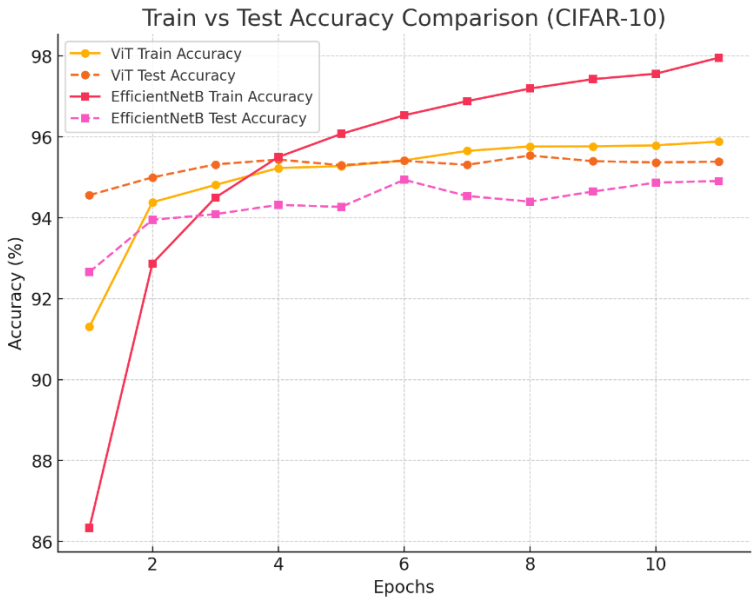
Code file: vit_train_CIFAR_10.py | fine-tuning the last 2 block layers

Data Augmentation: Resizing, random horizontal flipping, random rotation, random cropping, and normalization. *(Applied to all other model training)*

Pretrained Models: EfficientNet-B0 and Vision Transformer (ViT-B/16).

Optimization: Cross-entropy loss and Adam optimizer. *(Applied to all other model training)*





Training

```
Epoch [1/11], Loss: 0.4227, Train Acc: 86.33%, Test Acc: 92.66%
Epoch [2/11], Loss: 0.2097, Train Acc: 92.87%, Test Acc: 93.95%
Epoch [3/11], Loss: 0.1619, Train Acc: 94.50%, Test Acc: 94.09%
Epoch [4/11], Loss: 0.1306, Train Acc: 95.50%, Test Acc: 94.32%
Epoch [5/11], Loss: 0.1127, Train Acc: 96.07%, Test Acc: 94.27%
Epoch [6/11], Loss: 0.0983, Train Acc: 96.53%, Test Acc: 94.94%
Epoch [7/11], Loss: 0.0881, Train Acc: 96.88%, Test Acc: 94.54%
Epoch [8/11], Loss: 0.0798, Train Acc: 97.20%, Test Acc: 94.40%
Epoch [9/11], Loss: 0.0717, Train Acc: 97.43%, Test Acc: 94.65%
Epoch [10/11], Loss: 0.0710, Train Acc: 97.56%, Test Acc: 94.87%
Epoch [11/11], Loss: 0.0604, Train Acc: 97.96%, Test Acc: 94.91%
Training complete in 1 hours, 1 minutes, and 34.78 seconds
Best Test Accuracy: 94.94%
```

Efficient-Net B0 Weights


```
Epoch [1/11], Loss: 0.4227, Train Acc: 86.33%, Test Acc: 92.66%
Epoch [2/11], Loss: 0.2097, Train Acc: 92.87%, Test Acc: 93.95%
Epoch [3/11], Loss: 0.1619, Train Acc: 94.50%, Test Acc: 94.09%
Epoch [4/11], Loss: 0.1306, Train Acc: 95.50%, Test Acc: 94.32%
Epoch [5/11], Loss: 0.1127, Train Acc: 96.07%, Test Acc: 94.27%
Epoch [6/11], Loss: 0.0983, Train Acc: 96.53%, Test Acc: 94.94%
Epoch [7/11], Loss: 0.0881, Train Acc: 96.88%, Test Acc: 94.54%
Epoch [8/11], Loss: 0.0798, Train Acc: 97.20%, Test Acc: 94.40%
Epoch [9/11], Loss: 0.0717, Train Acc: 97.43%, Test Acc: 94.65%
Epoch [10/11], Loss: 0.0710, Train Acc: 97.56%, Test Acc: 94.87%
Epoch [11/11], Loss: 0.0604, Train Acc: 97.96%, Test Acc: 94.91%
Training complete in 1 hours, 1 minutes, and 34.78 seconds
Best Test Accuracy: 94.94%
```

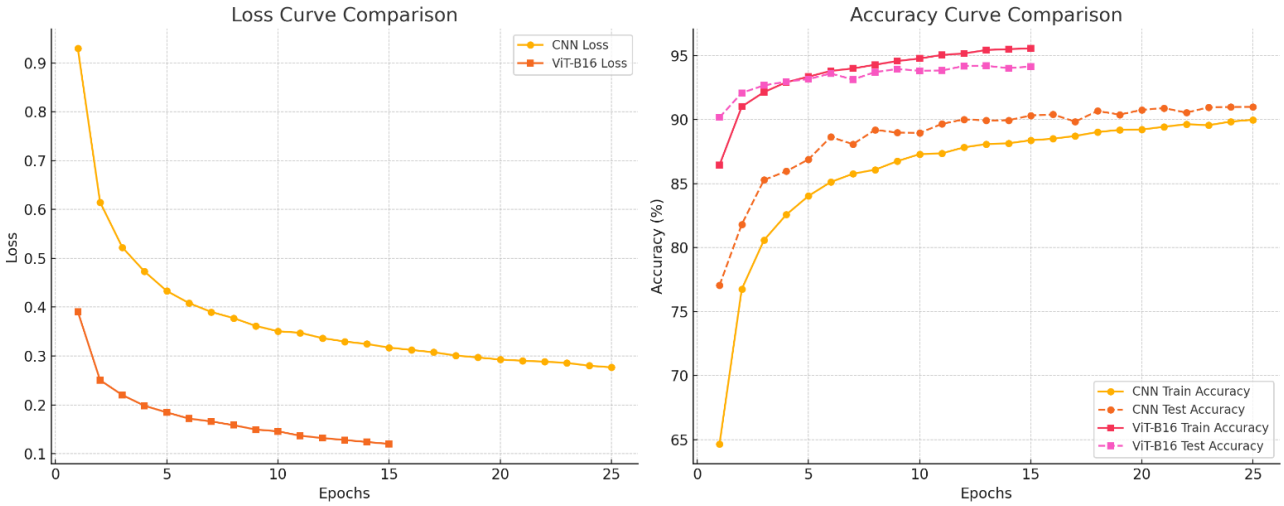
Efficient-Net B0 Weights

Fashion-MNIST

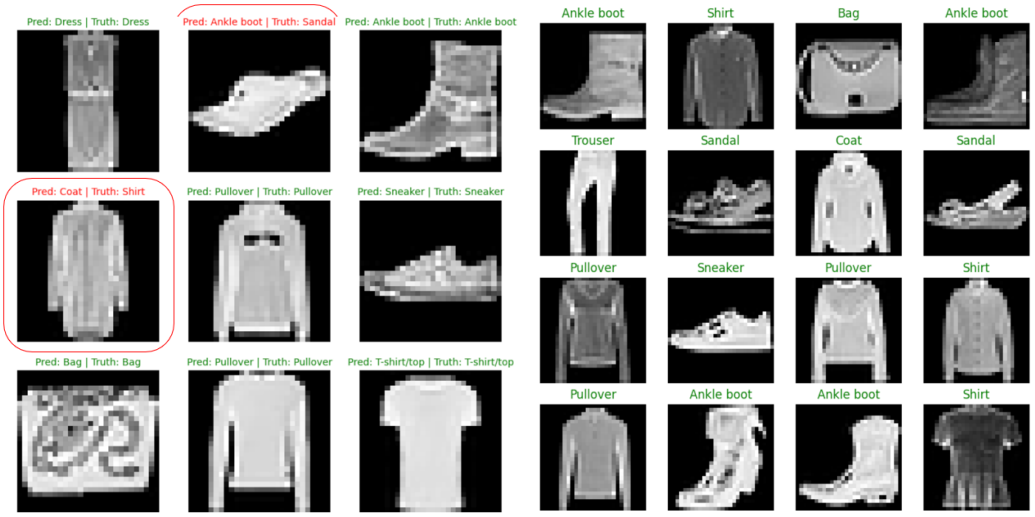
Code file: ViT_fashMNIST_ViT.py | Transfer learning without fine-tuning

Code file: CNN_fashMNIST.py | full training without transfer learning

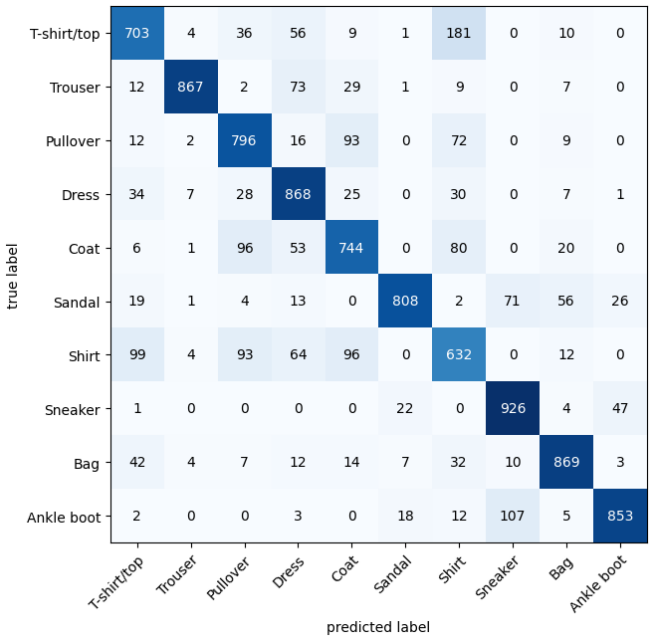
Pretrained Models: Vision Transformer (ViT-B/16).



Test dataset



Confusion Matrix for the test set

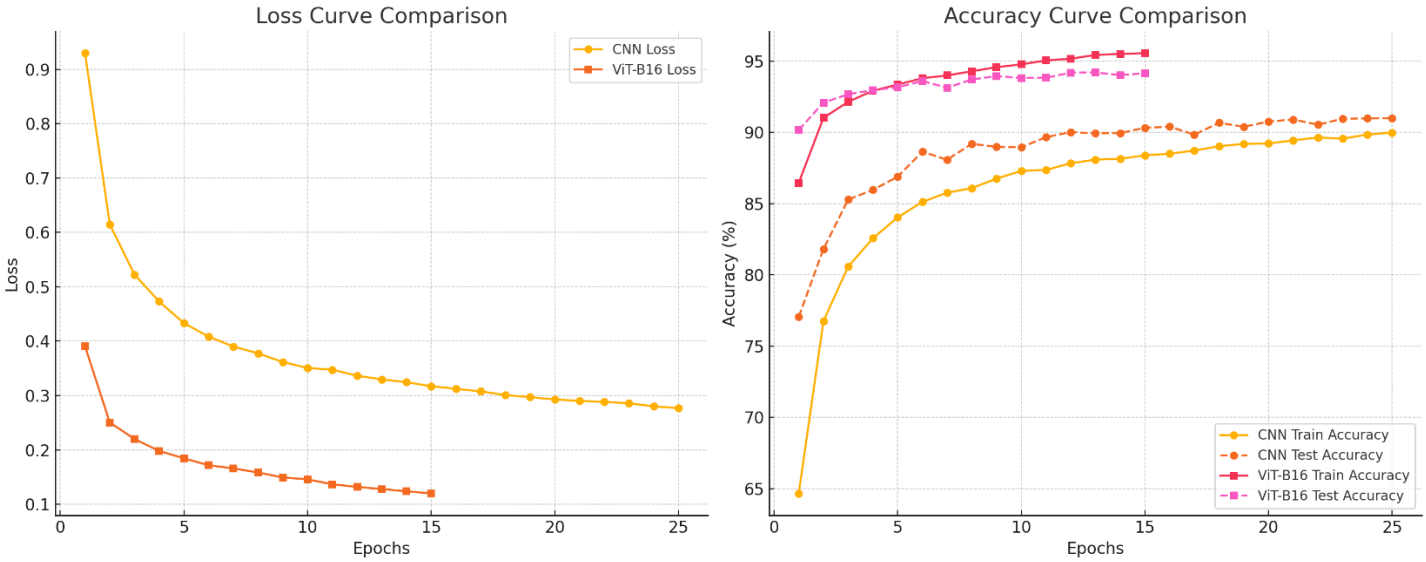


CIFAR 100

Code file: CIFAR_100_ViT_B16.py | fine-tuning the last 2 feature blocks.

Code file: CIFAR_100_ViT_CNN_hybrid.py | fine-tuning 2 from ViT and 2 layers from ResNet-18 weights.

Pretrained Models: ResNet 18 and Vision Transformer (ViT-B/16).



Training

```
root@da0fe70d91f8:/jamshid_home/PycharmProjects/pythonProject/medical_AI# CUDA_VISIBLE_DEVICES=0 python vit_b16_CIFAR_100.py
Epoch [1/15], Loss: 1.2170, Train Acc: 70.45%, Test Acc: 77.56%
Epoch [2/15], Loss: 0.7011, Train Acc: 79.64%, Test Acc: 79.67%
Epoch [3/15], Loss: 0.6090, Train Acc: 82.07%, Test Acc: 80.50%
Epoch [4/15], Loss: 0.5517, Train Acc: 83.73%, Test Acc: 80.97%
Epoch [5/15], Loss: 0.5099, Train Acc: 84.59%, Test Acc: 80.98%
Epoch [6/15], Loss: 0.4760, Train Acc: 85.79%, Test Acc: 81.21%
Epoch [7/15], Loss: 0.4536, Train Acc: 86.69%, Test Acc: 81.23%
Epoch [8/15], Loss: 0.4338, Train Acc: 87.15%, Test Acc: 81.28%
Epoch [9/15], Loss: 0.4289, Train Acc: 87.38%, Test Acc: 81.33%
Epoch [10/15], Loss: 0.4200, Train Acc: 87.77%, Test Acc: 81.39%
Epoch [11/15], Loss: 0.4204, Train Acc: 87.69%, Test Acc: 81.39%
Epoch [12/15], Loss: 0.4181, Train Acc: 87.75%, Test Acc: 81.38%
Epoch [13/15], Loss: 0.4219, Train Acc: 87.60%, Test Acc: 81.21%
Epoch [14/15], Loss: 0.4226, Train Acc: 87.48%, Test Acc: 81.21%
Epoch [15/15], Loss: 0.4259, Train Acc: 87.36%, Test Acc: 81.59%
Training complete in 6 hours, 40 minutes, and 45.92 seconds
Best Test Accuracy: 81.59%
root@da0fe70d91f8:/jamshid_home/PycharmProjects/pythonProject/medical_AI#
```

```
root@cb9bedc9ef7c:/jamshid_home/PycharmProjects/pythonProject/medical_AI# CUDA_VISIBLE_DEVICES=1,6 python vit_cnn_h
ybrid_CIFAR100.py
Files already downloaded and verified
Files already downloaded and verified
Epoch [1/15], Loss: 1.2301, Train Acc: 66.32%, Test Acc: 78.10%
Epoch [2/15], Loss: 0.7177, Train Acc: 78.25%, Test Acc: 80.08%
Epoch [3/15], Loss: 0.5952, Train Acc: 81.73%, Test Acc: 81.01%
Epoch [4/15], Loss: 0.5022, Train Acc: 84.33%, Test Acc: 81.91%
Epoch [5/15], Loss: 0.4137, Train Acc: 86.69%, Test Acc: 82.85%
Epoch [6/15], Loss: 0.3270, Train Acc: 89.43%, Test Acc: 83.32%
Epoch [7/15], Loss: 0.2537, Train Acc: 91.76%, Test Acc: 83.54%
Epoch [8/15], Loss: 0.1886, Train Acc: 93.97%, Test Acc: 84.07%
Epoch [9/15], Loss: 0.1469, Train Acc: 95.40%, Test Acc: 84.26%
Epoch [10/15], Loss: 0.1242, Train Acc: 96.12%, Test Acc: 84.25%
Epoch [11/15], Loss: 0.1199, Train Acc: 96.26%, Test Acc: 84.25%
Epoch [12/15], Loss: 0.1205, Train Acc: 96.28%, Test Acc: 84.25%
Epoch [13/15], Loss: 0.1214, Train Acc: 96.17%, Test Acc: 83.89%
Epoch [14/15], Loss: 0.1308, Train Acc: 95.71%, Test Acc: 83.89%
Epoch [15/15], Loss: 0.1470, Train Acc: 95.24%, Test Acc: 83.26%
Training complete in 402m 54s
Best Test Accuracy: 84.26%
root@cb9bedc9ef7c:/jamshid_home/PycharmProjects/pythonProject/medical_AI#
```

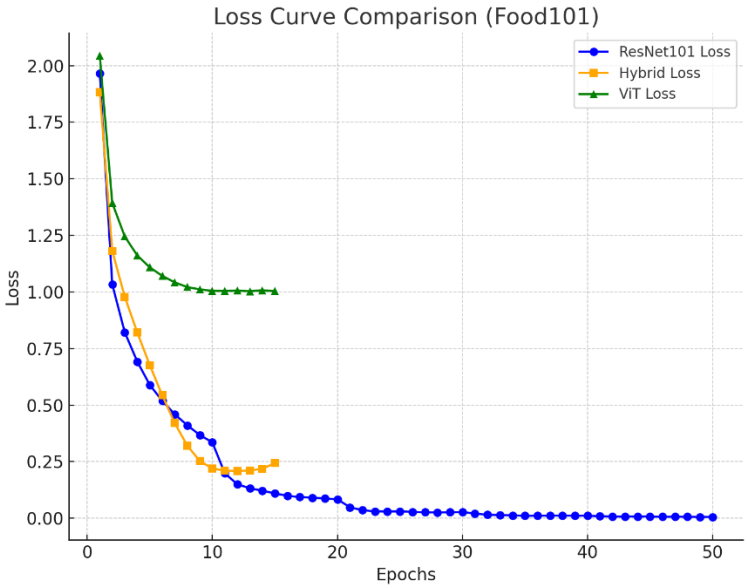
Food-101

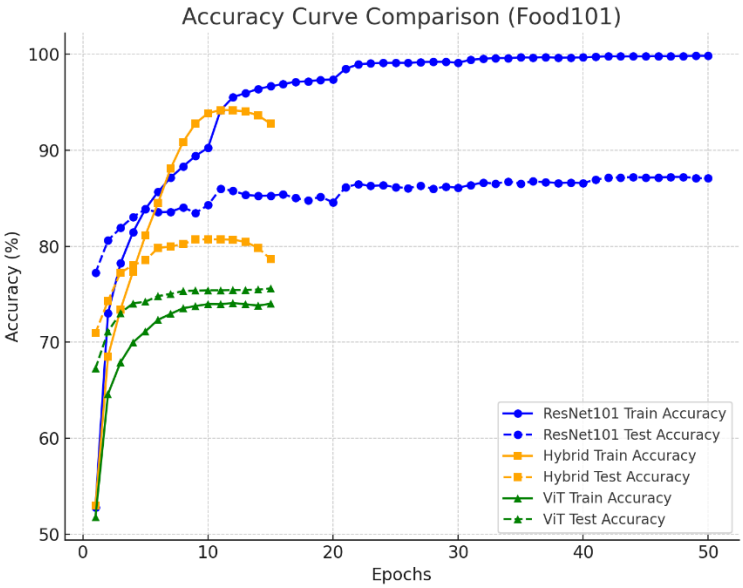
Code file: resnet101_food101.py | FULL fine-tuning

Code file: vit_b16_Food101.py | fine-tuning the last encoder block

Code file: vit_cnn_hybrid_Food101.py | fine-tuning the last 2 feature layers from ViT and Resnet 18.

Pretrained Models: ResNet-18 and Vision Transformer (ViT-B/16).





Training

```
root@a03c00f67e5b:/jamshid_home/PycharmProjects/pythonProject/medical_AI# CUDA_VISIBLE_DEVICES=6,7 python vit_cnn_hybrid_Food1
01.py
Downloading https://data.vision.ee.ethz.ch/cvl/food-101.tar.gz to ./data/food-101.tar.gz
100%|████████████████████████████████████████████████████████████████████████| 4996278331/4996278331 [34:18<00:00, 2426899.35it/s]
Extracting ./data/food-101.tar.gz to ./data
Downloading: "https://download.pytorch.org/models/resnet18-f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-f37072
fd.pth
100%|████████████████████████████████████████████████████████████████████████| 44.7M/44.7M [00:03<00:00, 11.9MB/s]
Downloading: "https://download.pytorch.org/models/vit_b_16-c867db91.pth" to /root/.cache/torch/hub/checkpoints/vit_b_16-c867db
91.pth
100%|████████████████████████████████████████████████████████████████████████| 330M/330M [00:29<00:00, 11.8MB/s]
Epoch [1/15], Loss: 1.8838, Train Acc: 52.99%, Test Acc: 70.96%
Epoch [2/15], Loss: 1.1810, Train Acc: 68.51%, Test Acc: 74.29%
Epoch [3/15], Loss: 0.9784, Train Acc: 73.40%, Test Acc: 77.25%
Epoch [4/15], Loss: 0.8207, Train Acc: 77.35%, Test Acc: 78.02%
Epoch [5/15], Loss: 0.6764, Train Acc: 81.15%, Test Acc: 78.53%
Epoch [6/15], Loss: 0.5427, Train Acc: 84.50%, Test Acc: 79.81%
Epoch [7/15], Loss: 0.4196, Train Acc: 88.08%, Test Acc: 79.97%
Epoch [8/15], Loss: 0.3204, Train Acc: 90.85%, Test Acc: 80.22%
Epoch [9/15], Loss: 0.2529, Train Acc: 92.80%, Test Acc: 80.70%
Epoch [10/15], Loss: 0.2195, Train Acc: 93.87%, Test Acc: 80.71%
Epoch [11/15], Loss: 0.2103, Train Acc: 94.18%, Test Acc: 80.71%
Epoch [12/15], Loss: 0.2079, Train Acc: 94.17%, Test Acc: 80.69%
Epoch [13/15], Loss: 0.2109, Train Acc: 94.03%, Test Acc: 80.45%
Epoch [14/15], Loss: 0.2183, Train Acc: 93.63%, Test Acc: 79.83%
Epoch [15/15], Loss: 0.2436, Train Acc: 92.77%, Test Acc: 78.66%
Training complete in 611m 38s
Best Test Accuracy: 80.71%
root@a03c00f67e5b:/jamshid_home/PycharmProjects/pythonProject/medical_AI#
```

Just using Transfer learning without any fine-tuning decreased the training time from 9 hours to 6 hours

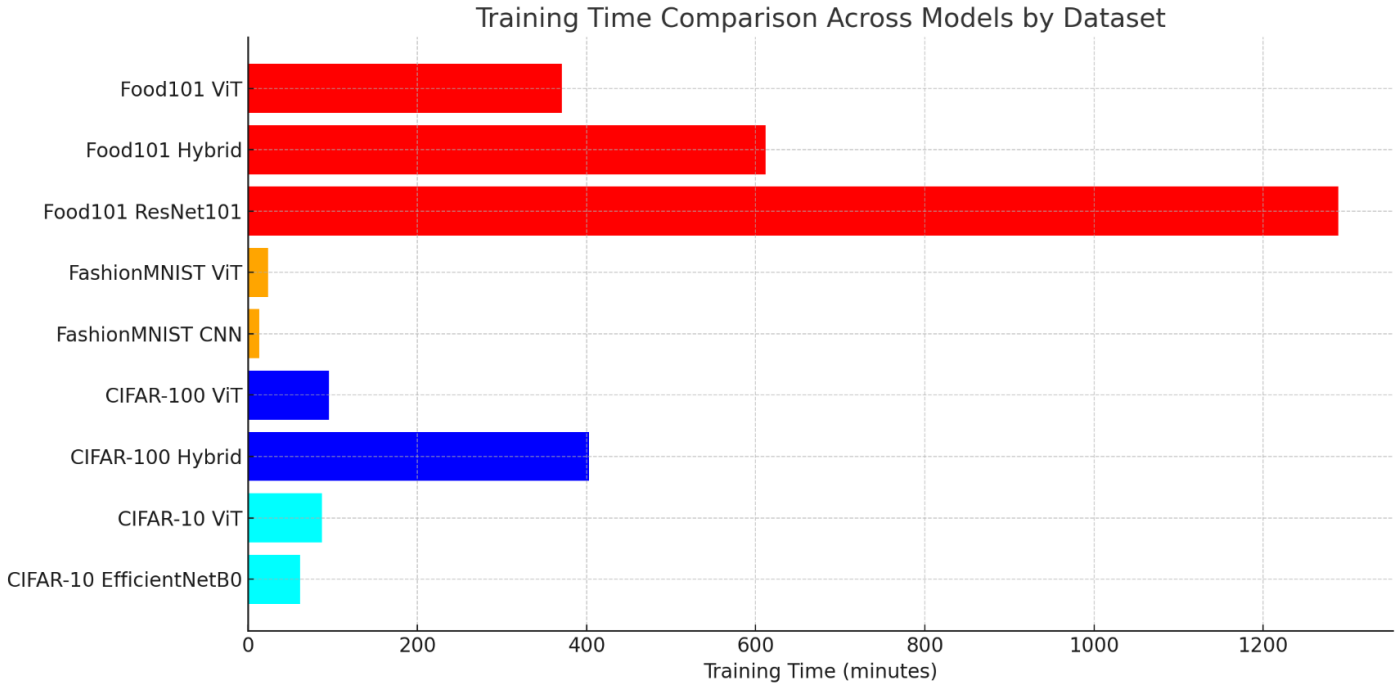
```
root@a79bbcaea8da:/jamshid_home/PycharmProjects/pythonProject/medical_AI# CUDA_VISIBLE_DEVICES=7 python vit_b16_Food101.py
Epoch [1/15], Loss: 2.0420, Train Acc: 51.93%, Test Acc: 67.19%
Epoch [2/15], Loss: 1.3911, Train Acc: 64.51%, Test Acc: 71.15%
Epoch [3/15], Loss: 1.2416, Train Acc: 68.16%, Test Acc: 72.94%
Epoch [4/15], Loss: 1.1627, Train Acc: 69.93%, Test Acc: 73.54%
Epoch [5/15], Loss: 1.1110, Train Acc: 71.14%, Test Acc: 74.43%
Epoch [6/15], Loss: 1.0736, Train Acc: 72.04%, Test Acc: 74.83%
Epoch [7/15], Loss: 1.0456, Train Acc: 72.98%, Test Acc: 75.10%
Epoch [8/15], Loss: 1.0228, Train Acc: 73.45%, Test Acc: 75.47%
Epoch [9/15], Loss: 1.0098, Train Acc: 73.86%, Test Acc: 75.54%
Epoch [10/15], Loss: 1.0034, Train Acc: 74.11%, Test Acc: 75.49%
Epoch [11/15], Loss: 1.0016, Train Acc: 74.05%, Test Acc: 75.49%
Epoch [12/15], Loss: 1.0030, Train Acc: 73.94%, Test Acc: 75.55%
Epoch [13/15], Loss: 1.0066, Train Acc: 73.93%, Test Acc: 75.58%
Epoch [14/15], Loss: 1.0049, Train Acc: 73.87%, Test Acc: 75.60%
Epoch [15/15], Loss: 1.0023, Train Acc: 73.97%, Test Acc: 75.61%
Training complete in 9 hours, 59 minutes, and 27.16 seconds
Best Test Accuracy: 75.61%
root@a79bbcaea8da:/jamshid_home/PycharmProjects/pythonProject/medical_AI# CUDA_VISIBLE_DEVICES=1 python vit_b16_Food101.py
Epoch [1/15], Loss: 2.0433, Train Acc: 51.80%, Test Acc: 67.26%
Epoch [2/15], Loss: 1.3931, Train Acc: 64.59%, Test Acc: 71.11%
Epoch [3/15], Loss: 1.2448, Train Acc: 67.90%, Test Acc: 73.04%
Epoch [4/15], Loss: 1.1612, Train Acc: 69.98%, Test Acc: 74.04%
Epoch [5/15], Loss: 1.1084, Train Acc: 71.13%, Test Acc: 74.23%
Epoch [6/15], Loss: 1.0706, Train Acc: 72.33%, Test Acc: 74.80%
Epoch [7/15], Loss: 1.0421, Train Acc: 72.94%, Test Acc: 75.02%
Epoch [8/15], Loss: 1.0212, Train Acc: 73.54%, Test Acc: 75.33%
Epoch [9/15], Loss: 1.0111, Train Acc: 73.76%, Test Acc: 75.37%
Epoch [10/15], Loss: 1.0047, Train Acc: 73.97%, Test Acc: 75.41%
Epoch [11/15], Loss: 1.0040, Train Acc: 73.96%, Test Acc: 75.41%
Epoch [12/15], Loss: 1.0055, Train Acc: 74.08%, Test Acc: 75.43%
Epoch [13/15], Loss: 1.0030, Train Acc: 73.95%, Test Acc: 75.44%
Epoch [14/15], Loss: 1.0072, Train Acc: 73.82%, Test Acc: 75.48%
Epoch [15/15], Loss: 1.0031, Train Acc: 74.00%, Test Acc: 75.60%
Training complete in 6 hours, 10 minutes, and 29.56 seconds
Best Test Accuracy: 75.60%
root@a79bbcaea8da:/jamshid_home/PycharmProjects/pythonProject/medical_AI#
```

The following training results show the best accuracy and are from “fully fine-tuned Resnet 101 weights”

```
Epoch 46/50 - Training: 100%|                                                | 296/296 [21:19<00:00,  4.32s/batch, accuracy=99.8, loss=0.00171]
Epoch 46/50 - Testing: 100%|                                                 | 99/99 [03:49<00:00,  2.32s/batch, accuracy=87.2]
Epoch [46/50], Loss: 0.0060, Train Acc: 99.82%, Test Acc: 87.16%
Epoch 47/50 - Training: 100%|                                                | 296/296 [21:24<00:00,  4.34s/batch, accuracy=99.8, loss=0.00852]
Epoch 47/50 - Testing: 100%|                                                 | 99/99 [03:54<00:00,  2.37s/batch, accuracy=87.2]
Epoch [47/50], Loss: 0.0067, Train Acc: 99.80%, Test Acc: 87.22%
Epoch 48/50 - Training: 100%|                                                | 296/296 [21:26<00:00,  4.35s/batch, accuracy=99.8, loss=0.00479]
Epoch 48/50 - Testing: 100%|                                                 | 99/99 [03:52<00:00,  2.34s/batch, accuracy=87.2]
Epoch [48/50], Loss: 0.0060, Train Acc: 99.82%, Test Acc: 87.22%
Epoch 49/50 - Training: 100%|                                                | 296/296 [22:37<00:00,  4.59s/batch, accuracy=99.8, loss=0.0102]
Epoch 49/50 - Testing: 100%|                                                 | 99/99 [04:02<00:00,  2.45s/batch, accuracy=87.1]
Epoch [49/50], Loss: 0.0057, Train Acc: 99.83%, Test Acc: 87.10%
Epoch 50/50 - Training: 100%|                                                | 296/296 [22:17<00:00,  4.52s/batch, accuracy=99.8, loss=0.00481]
Epoch 50/50 - Testing: 100%|                                                 | 99/99 [03:55<00:00,  2.38s/batch, accuracy=87.1]
Epoch [50/50], Loss: 0.0057, Train Acc: 99.84%, Test Acc: 87.07%
Training complete in 21h 28m 36s
Best Test Accuracy: 87.22%
```

Training time Comparison

Food101 ResNet101 model: Full Training



Best Accuracy Comparison

All the desired expectations are met