Computer Vision Systems CAP6411 Assignment#01

Ashmal Vayani, UCF ID: 5669011

Report: Human Action Recognition with ResNet-18 and ViT

Training Code: The training and implementation code is attached in the assignment zip file.

1 Human Action Recognition Dataset

The Human Action Recognition dataset, introduced on Kaggle by Shashank Rapolu, contains around 12,600 labeled images across 15 action classes such as *calling, dancing, eating, running, sleeping,* and *using_laptop*. The data is organized in a simple folder-per-class structure with predefined train/test splits, making it directly compatible with libraries like `torchvision`. Unlike larger video-based benchmarks (e.g., UCF101), this dataset focuses on single still images of human activities, offering a manageable yet diverse benchmark for testing deep learning models such as ResNet-18 and Vision Transformer (ViT).

To provide an overview of the dataset, Figure 1 shows one example image from each of the 15 action classes. This demonstrates the diversity of actions such as body posture (e.g., *sitting, sleeping*), object interaction (e.g., *using_laptop, drinking*), and social behaviors (e.g., *hugging, fighting*).


(a) calling


(b) clapping


(c) cycling


(d) dancing


(e) drinking


(f) eating


(g) fighting


(h) hugging


(i) laughing


(j) listening_to_music


(k) running


(l) sitting


(m) sleeping


(n) texting


(o) using_laptop

Figure 1: Example images from the 15 classes in the Human Action Recognition dataset.

2 Report and Code

2.1 Errors and Obstacles Faced

- **Dataset structure confusion:** Initially, the test directory was assumed to contain flat unlabeled images, but it was organized into 15 class subfolders. This caused my code line `len(os.listdir(test_dir))` to return 15 instead of the actual number of test images. Fixed by using ImageFolder consistently for train, val, and test splits.

- **GPU/Memory issues:** When I was running the code on Colab free tier, Vision Transformer (`vit_b_16`) often hit OOM errors with batch size 32. Then I ran my code on the CRCV cluster with an Ampere GPU and 48 GB of memory, using the same batch size (32), and it ran successfully.
- **Training time:** ResNet-18 trained reasonably fast (30 seconds per epoch on a single GPU), but ViT took 2.5- 3x longer (1.5 minutes) due to more parameters. This time was much higher when I ran it on Google Colab (5 mins for ViT, 1 minute for ResNet-18).
- **SLURM vs. Local differences:** On SLURM cluster runs, ensuring the correct conda environment and GPU visibility (`CUDA_VISIBLE_DEVICES`) was essential. Errors occurred when the environment path was mis-specified, but were fixed by explicitly activating `/home/ashmal/anaconda3/envs/cvs_ass1`.

2.2 Requirements and Dependencies

The `requirements.txt` file is attached in the zip file, but some main dependencies involve:

requirements.txt

```
torch==2.2.0
torchvision==0.17.0
timm==0.9.16
numpy
pandas
scikit-learn
tqdm
matplotlib
pillow
```

2.3 CLI Commands to Reproduce

```
# Train and evaluate ResNet-18
python train_resnet.py

# Train and evaluate ViT
python train_vit.py

# Or submit via SLURM
sbatch slurm/train_resnet.slurm
sbatch slurm/train_vit.slurm
```

This will generate:

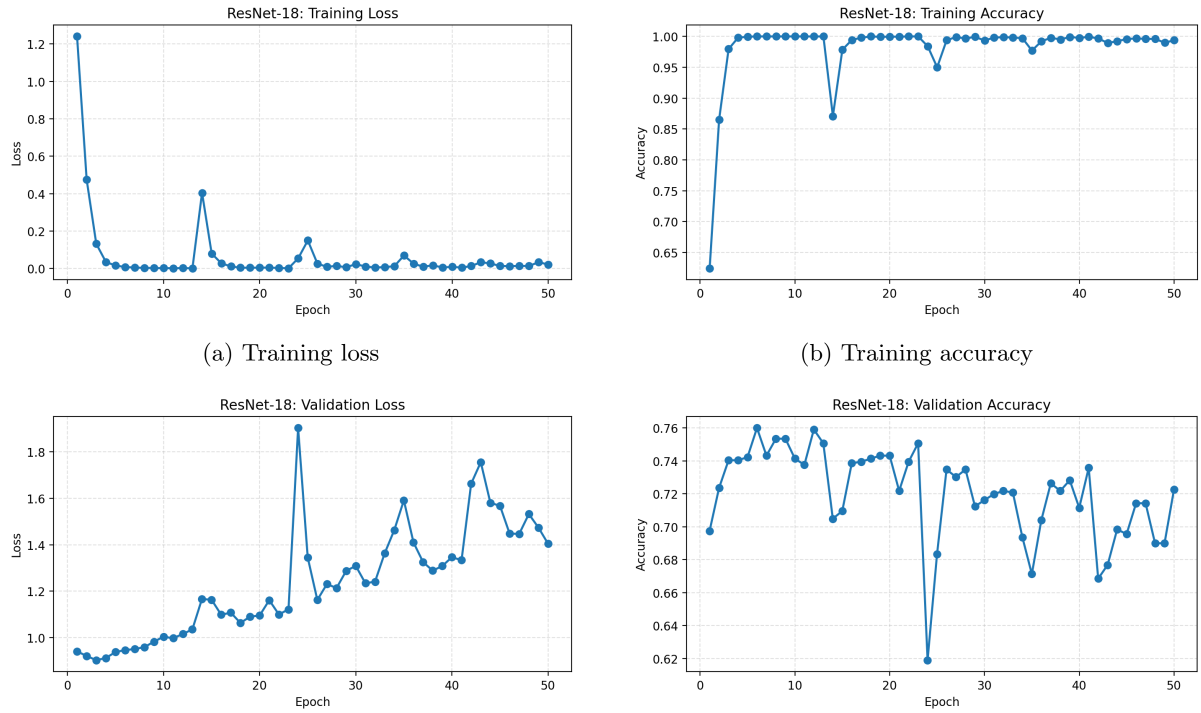
- `Output/ResNet-18/best_resnet_model.pth`
- `Output/ResNet-18/resnet_test_predictions.csv`
- `Output/ResNet-18/resnet_test_eval.txt`
- `Output/ResNet-18/resnet_confusion_matrix.png`

and the analogous files under `Output/ViT/`.

Training Logs

Both scripts (`train_resnet.py`, `train_vit.py`) stream epoch-wise logs and also write them into:

- `resnet_training_log.txt`
- `vit_training_log.txt`



(a) Training loss

(b) Training accuracy

(c) Validation loss

(d) Validation accuracy

Figure 2: ResNet–18 training curves over 50 epochs.

3 Training Plots (ResNet–18)

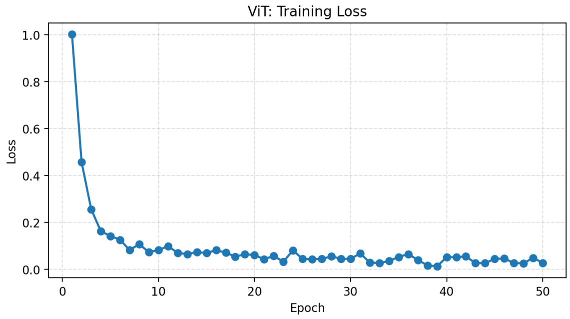
Observations: As plotted in fig. 2, following observations about ResNet-18 can be observed:

- **Rapid memorization.** Training accuracy rises from ~ 0.62 to ~ 0.98 by epoch 3 and stays near 0.99 thereafter; training loss falls close to zero, with a few transient spikes (e.g., around epochs $\sim 14, 25, 35$).
- **Validation peak early.** Validation accuracy reaches its maximum near 0.75–0.76 in the first 10–15 epochs and then oscillates, indicating the model has already extracted most generalizable signal early.
- **Overfitting trend.** While training continues improving, validation loss slowly drifts upward with occasional spikes (e.g., epochs $\sim 24, 41$), and validation accuracy fluctuates between ~ 0.68 and ~ 0.74—a classic overfitting signature.
- **Actionable fixes.** Early stopping around the first validation peak (epochs 6–13), cosine LR with warmup, stronger regularization (weight decay, label smoothing), and heavier augmentation (RandAugment/MixUp/CutMix) would likely smooth the validation curves and improve generalization.

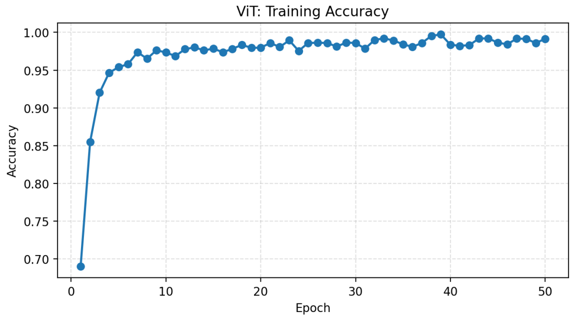
4 Training Plots (ViT-B/16)

Observations: As plotted in fig. 3, the following observations about ViT can be observed:

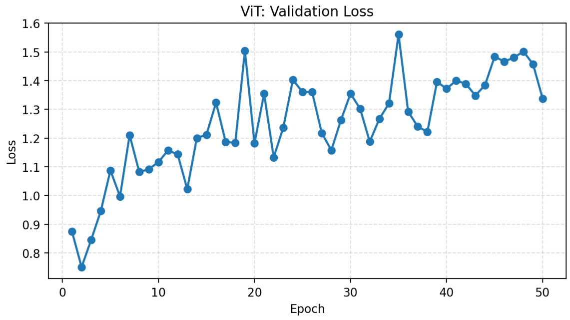
- **Rapid fit (epochs 1–3).** Training loss drops sharply ($1.00 \rightarrow 0.26$) while validation accuracy rises to $\sim$0.77 by epoch 3.
- **Overfitting onset.** From $\sim$epoch 5 onward, training accuracy keeps increasing (> 0.98), but validation loss oscillates and trends upward, a classic sign of overfitting.
- **Best validation.** Peak validation accuracy occurs around epoch 13 (0.7796), after which validation metrics generally degrade despite near-perfect training accuracy.



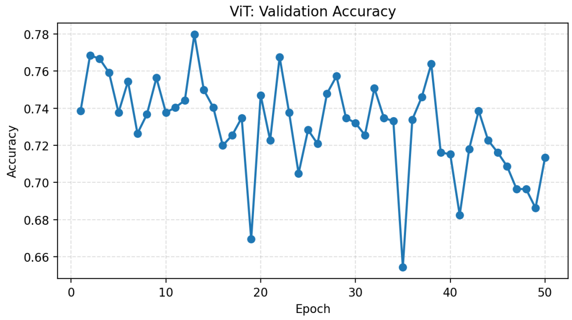
(a) Training loss



(b) Training accuracy



(c) Validation loss



(d) Validation accuracy

Figure 3: ViT training curves over 50 epochs.

- **Regularization need.** A cosine schedule with warmup, stronger augmentation (RandAugment, MixUp/CutMix), label smoothing, and early stopping would likely stabilize validation curves and push the generalization peak earlier.

5 Insights: ResNet–18 vs. ViT

5.1 Overall Accuracy and Classwise Behavior

We trained both models for 50 epochs with identical data splits and augmentations. On the held-out test set (N=1890 images), the Vision Transformer (ViT-B/16) achieved a slightly higher overall accuracy than ResNet-18:

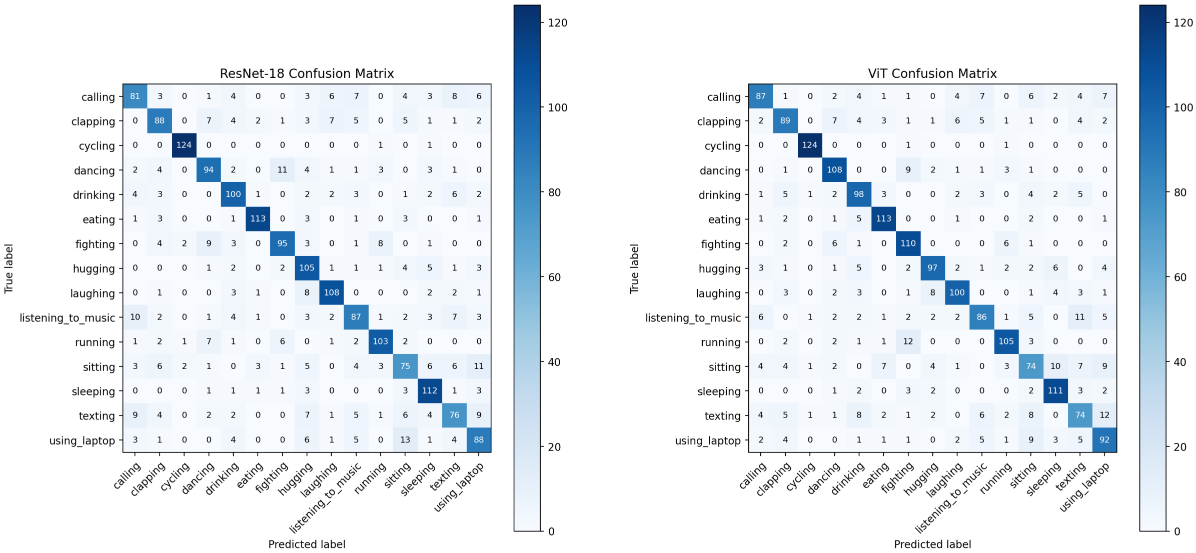
$$\text{ResNet-18: } 76.67\% \qquad \text{ViT: } 77.67\%.$$

Figure 4 shows the confusion matrices. Several consistent patterns emerge:

- **Easy classes (both models).** *cycling, running, sleeping, laughing* have strong diagonals; e.g., **cycling** reaches ≥ 0.97 F1 for both.
- **ViT advantages.** *calling* (P/R: 0.79/0.69 vs. 0.71/0.64), *dancing* (0.79/0.86 vs. 0.76/0.75), and *fighting* (0.76/0.87 vs. 0.81/0.75) show clearer diagonals and fewer confusions in ViT, indicating ViT captures more global pose/context cues for interpersonal or dynamic activities.
- **ResNet advantages.** *eating* benefits from higher precision with ResNet (0.93 vs. 0.86), suggesting CNN locality priors help disambiguate object–mouth interactions and near-field cues.
- **Hard classes (both models).** *sitting, texting, listening_to_music* remain challenging. Errors often spread among *sitting/using_laptop/texting/listening_to_music*, which share visual layouts (seated, handheld device, ear accessories) and subtle fine-grained differences.

5.2 Computational Comparison

We observed the following practical trade-offs (same GPU, same dataloaders):



(a) ResNet-18

(b) ViT-B/16

Figure 4: Confusion matrices on the test set (15 classes). ViT slightly improves overall accuracy, notably on *calling, dancing, fighting*, while ResNet is more precise on *eating*.

Aspect	**ResNet-18**	**ViT-B/16**
Parameters (approx.)	∼11M	∼86M
Best Validation Acc.	0.760	0.780
Test Accuracy	0.767	0.777
Per-epoch time	*30 Seconds*	*80–120 Seconds*
GPU memory (bs=32)	fits comfortably	may require smaller batch
Convergence speed	quicker early gains	needs more epochs/tuning
Overall Insight	lightweight, efficient	computationally heavy, more accurate

Table 1: Comparison between ResNet-18 and ViT-B/16 on Human Action Recognition dataset.

Takeaways.

- **Throughput/latency.** ResNet-18 is lighter, trains/infer faster, and is friendlier to limited-GPU environments.
- **Capacity.** ViT's global self-attention can better exploit holistic context (*calling, dancing, fighting*) but demands more compute and regularization.
- **Generalization with limited data.** With the current augmentations, both models mildly overfit over long training (validation curves plateau and occasionally regress). ResNet's inductive bias helps maintain stable precision on object-centric cues (*eating*), whereas ViT gains on motion/interaction-heavy classes.

5.3 Why One Can Be Better (When)

- **Choose ResNet-18** when compute is constrained, real-time inference matters, or cues are local/object-centric. Its convolutional priors yield strong performance at low cost.
- **Choose ViT** when you can afford more compute and aim to leverage global spatial relations and long-range context (multi-person scenes, complex poses). With stronger data augmentation (e.g., RandAugment, MixUp/CutMix) and longer fine-tuning, ViT's headroom is higher.

5.4 Possible Actionable Improvements

1. **Data-side:** heavier augmentation; class-balanced sampling for *sitting/texting/listening_to_music*; modest resolution increase (256→224 crop) for ViT.

2. **Optimization:** cosine LR schedule with warmup, weight decay tuning; label smoothing (α=0.1); early stopping based on validation F1.

3. **Architectural:** try DeiT-S or ViT-S (smaller ViTs), or ResNet-50; add a lightweight attention head atop ResNet features for hybrid gains.

6 Model Output Visualizations

Figure 5: Sample qualitative results from the Human Action Recognition test set. Each image shows the ground truth label (GT) and the predictions from both ResNet-18 and ViT-B/16. Correct predictions are highlighted in green, while incorrect ones are highlighted in red.