Transfer Learning for Oxford Flowers-102 Using VGG19 , YOLOv5-CLS and ResNet-50

1. Introduction

In this assignment we apply **transfer learning** using three pre-trained convolutional neural network (CNN) models **VGG19**, **YOLOv5-CLS, and ResNet-50** to classify images from the **Oxford Flowers-102** dataset into 102 flowers categories. All models are initialized with pretrained weights and adapted for the Flowers-102 classification task by replacing the final classification layer to output **102 logits**, which are converted to **class probabilities** using a **softmax** operation.

In this experiment we follow the required evaluation protocol using **two independent random dataset splits** and reporting accuracy and cross-entropy loss for **train/validation/test** over training epochs.

---

2. Dataset

2.1 Oxford Flowers-102 (Local Format)

The dataset is used in the standard Oxford format:

```
root/
  jpg/
  imagelabels.mat
  setid.mat
```

- Images are loaded from `jpg/` using the naming convention `image_00001.jpg` … `image_NNNNN.jpg`.
- Labels are read from `imagelabels.mat` and converted from **1..102** to **0..101** for training.

---

3. Experimental Protocol

3.1 Random split strategy (performed twice)

A random permutation of indices is generated using a fixed seed, and the dataset is split into:

- **Training:** 50%
- **Validation:** 25%

- **Test:** 25%

This protocol was executed twice:

- **Split A:** `split_seed = 1`
- **Split B:** `split_seed = 2`

The generated split indices are saved to disk as JSON to ensure reproducibility:

- `split_indices_seed_1.json`
- `split_indices_seed_2.json`

3.2 Metrics

For each epoch, the following are computed on **train**, **validation**, and **test**:

- **Top-1 Accuracy**
- **Cross-Entropy Loss**

  We also report the class probabilities to csv.

---

4. Preprocessing and Input Pipeline (Detailed)

4.1 Input size

- **VGG19:** images resized to **224×224**
- **YOLOv5-CLS:** images resized to **224×224** in this implementation (`img_size=224`),
- **ResNet50- 224 × 224 (RGB)**, **Height × Width:** `224 × 224`, **Channels:** `3`, **Tensor shape PyTorch):** `(batch_size, 3, 224, 224)`
-

4.2 Normalization

Because both models use pretrained ImageNet weights, images are normalized using **ImageNet mean/std**:

- mean = `[0.485, 0.456, 0.406]`
- std = `[0.229, 0.224, 0.225]`

4.3 Data augmentation (training only)

Training images undergo the following augmentations:

- Resize to `(img_size, img_size)`
- Random horizontal flip with probability **0.5**
- Random rotation within **±15°**
- Color jitter:
  - brightness = 0.1
  - contrast = 0.1
  - saturation = 0.1
  - hue = 0.02
- Convert to tensor
- Normalize (ImageNet)

Validation and test transformations are deterministic:

- Resize to `(img_size, img_size)`
- Convert to tensor
- Normalize (ImageNet)

4.4 Probability outputs

The system outputs class probabilities by applying:

$$p(y = k \mid x) = \text{softmax}(\text{logits}(x))_k$$

Probabilities are exported to CSV files (both full-set and test-set exports), e.g.:

- `probs_vgg19.csv`, `test_probs_vgg19_seed1.csv`
- `probs_yolov5.csv`, `test_probs_yolov5_seed2.csv`
- `probs_resnet50.csv, test_probs_resnet50_seed2.csv`

`Example:`

	path	true_label	p_000	p_001	p_002	p_003	p_004	p_005	p_006	p_007	p_008	p_009	p_010
2	C:\Users\hp\Downloads\102flowers\jpg\	61	0.01871	1.68E-05	0.011791	0.004902	0.001658	4.00E-06	0.002645	1.19E-06	0.00032	4.85E-07	0.008574
3	C:\Users\hp\Downloads\102flowers\jpg\	91	3.82E-05	3.10E-06	0.000605	2.12E-05	0.000261	4.93E-05	1.36E-07	0.000476	4.28E-05	0.00488	2.28E-05
4	C:\Users\hp\Downloads\102flowers\jpg\	22	0.000973	8.69E-05	0.000139	4.05E-06	7.99E-07	2.63E-05	2.43E-06	0.063488	1.72E-06	2.35E-05	1.09E-06
5	C:\Users\hp\Downloads\102flowers\jpg\	23	6.79E-07	1.50E-05	0.00027	2.04E-05	1.34E-05	0.000148	4.14E-07	0.015419	2.69E-05	0.005773	1.89E-05
6	C:\Users\hp\Downloads\102flowers\jpg\	70	1.28E-05	0.000552	3.88E-07	7.57E-08	0.002102	3.75E-05	4.45E-05	9.36E-06	6.81E-08	3.11E-07	4.02E-06
7	C:\Users\hp\Downloads\102flowers\jpg\	73	0.000107	6.58E-05	0.002875	0.004715	0.000271	0.000899	0.000102	1.10E-05	0.000217	3.17E-07	0.200849
8	C:\Users\hp\Downloads\102flowers\jpg\	91	8.96E-08	2.89E-08	1.01E-05	5.42E-07	4.16E-06	1.09E-05	6.83E-09	0.000704	7.94E-06	7.00E-05	8.76E-07
9	C:\Users\hp\Downloads\102flowers\jpg\	88	0.000636	2.32E-06	4.59E-05	1.96E-06	3.00E-06	4.77E-07	9.60E-07	1.87E-06	2.12E-06	7.00E-10	0.000327
10	C:\Users\hp\Downloads\102flowers\jpg\	14	6.31E-06	6.08E-06	1.37E-06	1.29E-07	3.35E-05	1.69E-06	1.66E-07	0.007818	1.39E-09	6.80E-08	3.25E-07
11	C:\Users\hp\Downloads\102flowers\jpg\	85	0.034726	6.72E-05	0.000309	8.78E-05	0.010335	5.63E-05	0.00081	5.55E-05	0.00012	5.14E-06	0.000738
12	C:\Users\hp\Downloads\102flowers\jpg\	58	0.000293	5.92E-06	5.96E-05	4.01E-07	0.000644	7.65E-06	7.31E-06	9.23E-05	1.47E-06	7.68E-08	1.37E-05
13	C:\Users\hp\Downloads\102flowers\jpg\	32	0.004367	6.81E-06	0.000214	2.76E-05	1.86E-06	1.27E-06	0.000301	2.15E-06	0.000112	1.96E-06	3.13E-07
14	C:\Users\hp\Downloads\102flowers\jpg\	95	0.003329	0.002828	0.002231	0.001852	6.54E-06	0.000179	0.031761	1.70E-07	0.000417	4.27E-07	0.000571
15	C:\Users\hp\Downloads\102flowers\jpg\	6	0.001783	0.001039	0.019597	0.028479	3.65E-05	0.000176	0.09577	4.30E-07	0.011391	1.34E-05	0.004619
16	C:\Users\hp\Downloads\102flowers\jpg\	96	0.000475	1.40E-06	0.000144	2.28E-05	7.57E-09	3.51E-06	2.18E-08	0.003867	5.12E-07	1.06E-08	5.03E-07
17	C:\Users\hp\Downloads\102flowers\jpg\	72	2.25E-07	5.72E-06	2.84E-07	8.52E-08	3.01E-07	2.14E-07	7.56E-06	7.30E-07	3.60E-09	9.99E-10	1.88E-07
18	C:\Users\hp\Downloads\102flowers\jpg\	42	0.003117	1.34E-05	0.000139	2.69E-05	0.002543	2.52E-06	8.30E-05	0.000165	3.99E-06	1.39E-07	0.002397
19	C:\Users\hp\Downloads\102flowers\jpg\	22	1.30E-13	9.87E-10	1.12E-12	3.01E-13	1.28E-16	1.42E-07	3.12E-15	4.20E-14	4.01E-12	4.45E-13	2.24E-14
20	C:\Users\hp\Downloads\102flowers\jpg\	88	0.000706	2.81E-05	0.006724	0.002831	2.40E-05	2.91E-05	7.24E-05	8.47E-06	0.000165	1.38E-08	0.02305
21	C:\Users\hp\Downloads\102flowers\jpg\	37	0.000484	0.000219	0.000805	8.60E-05	8.24E-06	0.000159	0.001244	9.24E-05	0.000872	0.000513	1.69E-06
22	C:\Users\hp\Downloads\102flowers\jpg\	14	2.03E-06	2.81E-05	4.74E-06	8.15E-06	1.04E-05	1.18E-06	2.16E-05	0.08645	3.41E-06	2.90E-07	1.48E-05
23	C:\Users\hp\Downloads\102flowers\jpg\	97	0.022098	0.000452	0.021928	0.001393	0.000182	0.000541	0.000315	0.000163	0.008263	5.08E-06	0.028654
24	C:\Users\hp\Downloads\102flowers\jpg\	49	6.96E-06	6.68E-07	0.00016	6.42E-08	0.005442	1.80E-06	6.94E-07	4.34E-06	2.56E-07	3.65E-05	3.42E-07
25	C:\Users\hp\Downloads\102flowers\jpg\	79	7.35E-08	1.53E-05	3.86E-07	4.54E-05	3.56E-07	0.000178	1.43E-06	5.32E-06	5.66E-08	7.82E-10	2.36E-07
26	C:\Users\hp\Downloads\102flowers\jpg\	69	0.007265	0.004615	4.00E-05	3.74E-06	0.000128	5.10E-05	0.002794	2.80E-05	1.63E-05	9.27E-08	6.06E-05
27	C:\Users\hp\Downloads\102flowers\jpg\	101	3.17E-06	0.002	5.95E-05	0.000105	2.14E-06	0.000145	2.18E-05	1.32E-05	2.24E-05	1.41E-07	1.67E-05
28	C:\Users\hp\Downloads\102flowers\jpg\	70	2.84E-06	3.34E-06	1.24E-07	3.06E-09	0.020269	2.03E-06	5.82E-07	3.76E-06	7.81E-09	1.18E-07	2.76E-07
29	C:\Users\hp\Downloads\102flowers\jpg\	73	0.004583	0.002038	0.045729	0.025406	0.000158	0.000264	0.000427	0.000108	0.005011	2.09E-05	0.020452
30	C:\Users\hp\Downloads\102flowers\jpg\	50	0.020613	8.95E-05	0.01883	0.002546	3.63E-05	3.07E-06	0.0087	1.38E-05	0.001698	8.11E-07	0.017688
31	C:\Users\hp\Downloads\102flowers\jpg\	68	0.000314	2.47E-05	0.000173	7.16E-05	0.000139	2.23E-06	0.000437	3.23E-07	9.46E-06	5.70E-06	3.37E-06
32	C:\Users\hp\Downloads\102flowers\jpg\	76	0.00049	0.024091	0.000386	0.000114	0.000165	0.00046	0.000949	7.78E-05	0.000163	8.25E-07	1.64E-05

	p_011	p_012	p_013	p_014	p_015	p_016	p_017	p_018	p_019	p_020
2	4.89E-06	1.60E-06	3.94E-07	0.000464	4.01E-05	0.001623	0.006928	0.001383	1.51E-05	9.60E-05
3	0.001379	0.002981	0.003293	2.68E-05	1.42E-06	0.026375	0.009773	0.000295	9.99E-07	0.000593
4	0.001409	0.002758	1.58E-06	0.016892	0.000147	2.65E-05	0.000433	0.009476	0.021471	0.00041
5	0.00454	0.00094	0.000291	0.000297	4.10E-05	0.01295	0.000449	0.000322	4.37E-06	0.008512
6	7.59E-05	8.09E-05	3.01E-07	4.56E-05	4.02E-06	0.006343	0.003845	3.23E-05	3.23E-05	6.48E-08
7	1.84E-05	1.22E-05	3.52E-07	2.43E-05	0.008929	3.19E-06	0.001134	3.75E-05	2.40E-05	0.016516
8	3.65E-05	0.005739	4.17E-05	1.94E-05	1.76E-08	0.000641	0.001184	6.75E-06	7.56E-09	0.000465
9	2.31E-08	2.34E-07	1.88E-09	2.08E-05	0.001886	8.22E-08	6.01E-05	0.001131	6.14E-06	0.000214
10	3.33E-05	1.88E-06	1.38E-09	0.891661	0.000201	0.000592	0.000603	5.66E-05	3.68E-05	0.000338
11	7.26E-05	1.04E-05	6.68E-07	0.000888	0.001702	0.008385	0.003481	0.003236	0.000194	4.58E-05
12	2.58E-05	1.53E-05	5.49E-08	0.000178	3.19E-05	0.000203	0.000462	0.000974	3.42E-05	4.54E-05
13	1.98E-07	3.93E-06	2.79E-07	9.37E-07	6.41E-07	2.47E-05	0.000114	0.017503	0.000164	4.35E-07
14	6.87E-07	8.57E-06	5.86E-08	1.69E-06	1.30E-05	8.22E-05	0.000517	0.000421	0.002113	4.98E-07
15	2.64E-06	9.95E-06	1.73E-06	8.49E-06	6.06E-06	0.000807	0.012432	0.002238	0.000799	3.39E-06
16	1.06E-07	1.79E-06	2.98E-08	0.000909	1.36E-05	8.92E-06	7.39E-06	0.001354	0.000323	0.000103
17	5.64E-08	1.16E-07	1.47E-10	1.75E-05	3.44E-07	5.14E-06	3.56E-05	1.23E-05	3.94E-05	2.22E-08
18	5.69E-06	1.72E-07	2.28E-09	0.20089	0.050239	0.000195	0.001012	7.32E-05	0.00023	0.001119
19	1.39E-14	1.87E-11	4.59E-12	3.58E-16	2.39E-12	1.01E-12	2.66E-13	8.64E-11	3.96E-11	7.67E-13
20	9.99E-08	2.69E-05	2.99E-07	0.001757	9.00E-05	7.18E-06	0.016817	0.000305	8.67E-06	0.004844
21	0.000244	0.002697	4.08E-05	1.92E-05	1.54E-07	0.006851	0.006412	0.01664	0.000259	3.98E-05
22	0.00014	0.000115	2.53E-09	0.166754	0.000328	1.11E-05	0.000108	1.15E-05	0.000298	0.011826
23	2.56E-05	6.62E-05	4.23E-05	0.000144	0.000199	0.000571	0.004217	0.058648	0.000406	0.000939
24	0.055295	2.80E-05	5.10E-07	2.82E-05	2.55E-05	0.003542	5.38E-05	9.77E-06	1.22E-06	4.11E-08
25	2.34E-07	1.18E-07	1.49E-08	9.28E-09	1.72E-06	5.61E-08	4.11E-06	1.45E-07	3.39E-05	1.69E-06
26	2.76E-05	0.000201	7.42E-07	0.021451	0.003418	0.000939	0.003065	0.003104	0.005107	1.63E-06
27	6.40E-07	9.24E-05	4.10E-07	0.000133	4.14E-07	0.000968	0.007424	0.000894	1.01E-05	8.94E-06
28	7.41E-05	3.87E-05	5.98E-08	3.56E-06	5.00E-08	0.002876	0.000294	9.06E-07	1.03E-06	1.02E-07
29	1.01E-05	0.000364	0.000183	0.002297	0.001304	0.000378	0.006984	0.0055	0.000707	0.019604
30	3.80E-06	1.69E-06	1.89E-07	0.000117	0.000718	8.83E-05	0.00031	0.029256	0.00214	3.70E-05
31	1.36E-05	1.43E-06	1.24E-07	2.65E-05	2.45E-07	0.003231	0.000631	0.003137	4.70E-05	2.50E-08
32	0.000531	0.002415	1.62E-06	0.000253	6.17E-05	0.000966	0.131861	0.002135	2.06E-05	0.002647

test_probs_vgg19_seed1

5. Models and Transfer Learning (Detailed Architecture)

5.1 Model 1: VGG19 (ImageNet pretrained)

Base model: `torchvision.models.vgg19(weights=IMAGENET1K_V1)`

Transfer learning modification:

- The VGG19 convolutional feature extractor (`model.features`) is **frozen** (no gradient updates).
- The final fully-connected layer in the classifier is replaced:
    - Original: `Linear(in_features, 1000)`
    - New: `Linear(in_features, 102)`

Trainable vs frozen parameters (Split A run):

- Total parameters: **139,988,134**
- Trainable parameters: **417,894**
- Frozen parameters: **139,570,240**

Interpretation:
This is classic "linear probing" on top of a frozen ImageNet backbone: only the classification head learns Flowers-102-specific decision boundaries.

5.2 Model 2: YOLOv5-CLS (Ultralytics pretrained classifier)

Base model: YOLOv5 classification checkpoint loaded via torch hub:

- `torch.hub.load("ultralytics/yolov5", "custom", path="yolov5s-cls.pt")`

Transfer learning modification:

- The final classifier layer is replaced to output 102 classes:
    - `head.linear = Linear(in_features, 102)`

Freezing strategy:

- Backbone frozen (`requires_grad=False`)
- Classification head unfrozen (`requires_grad=True`)

Trainable vs frozen parameters (Split A run):

- Total parameters: **4,303,142**
- Trainable parameters: **788,582**

- Frozen parameters: **3,514,560**

Model 3: ResNet-50 (Torchvision pretrained classifier)

Base model: ResNet-50 pretrained on ImageNet (Torchvision):

- torchvision.models.resnet50(weights=torchvision.models.ResNet50_Weights.IMAGENET1K_V1)

Transfer learning modification:

- The final fully connected layer is replaced to output **102 classes**:
  - model.fc = nn.Linear(in_features, 102)

Freezing strategy (when freeze_backbone=True):

- Backbone frozen (requires_grad = False for all pretrained layers)
- Final classification layer unfrozen (requires_grad = True for model.fc)

Trainable vs frozen parameters (Split A run / Seed 1):

- **Total parameters:** 23710030
- **Trainable parameters:** 208,998
- **Frozen parameters:** *23,508,032*

Input & normalization:

- Input size: **224 × 224 × 3**
- Normalization (ImageNet): mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]

Training setup (as implemented):

- Loss: Cross-Entropy
- Optimizer: Adam (applied only to trainable parameters)
- LR scheduler: ReduceLROnPlateau (monitors validation loss)
- Early stopping: based on validation accuracy

---

6. Training Configuration

6.1 Shared components

- **Loss:** Cross-Entropy (`nn.CrossEntropyLoss()`)
- **Optimizer:** Adam
- **Scheduler:** ReduceLROnPlateau (monitoring validation loss)

    - factor = 0.5
    - patience = 3
- **Early stopping:** patience = 7 epochs (based on validation accuracy improvement)
- **Device:** CUDA GPU (NVIDIA RTX 4060 Laptop)

6.2 Hyperparameters per model

VGG19:

- epochs = **35**
- batch size = **32**
- learning rate = **1e-4**
- weight decay = **0.0**
- freeze backbone = **True**

YOLOv5-CLS:

- epochs = **30**
- batch size = **32** (default in our run experiment unless overridden)
- learning rate = **5e-5**
- weight decay = **0.0**
- freeze backbone = **True**

ResNet-50:

- **epochs = 35**
- **batch size = 32**
- **learning rate = 1e-4**
- **weight decay = 0.0**
- **freeze backbone = True**

7. Results

7.1 Summary table

Model	Split Seed	Best Epoch	Best Val Acc	Final Test Acc	Final Test Loss
VGG19	1	34	0.7626	0.7642	1.0066
VGG19	2	35	0.7596	0.7734	0.9365
YOLOv5-CLS	1	30	0.9184	0.9277	0.3863
YOLOv5-CLS	2	30	0.9135	0.9199	0.3925
ResNet-50	1	34	0.8984	0.8916	0.5237
ResNet-50	2	33	0.8867	0.9009	0.5146

7.2 Requirement check (≥70% test accuracy)

The requirement is satisfied by **both models**, and strongly exceeded by YOLOv5-CLS:

- VGG19 achieves ~**76–77%** test accuracy
- YOLOv5-CLS achieves ~**92%** test accuracy
- ResNet-50 achieves~ 90% test accuracy

7.3 Learning curves (accuracy and loss)

We saved and plotted the training plots per experiment as shown Below.

- Accuracy vs epoch: train/val/test
- Cross-entropy loss vs epoch: train/val/test

VGG19

- Figure 1: `./results/vgg19_seed1/accuracy_vgg19.png`
- Figure 2: `./results/vgg19_seed1/loss_vgg19.png`
- Figure 3: `./results/vgg19_seed2/accuracy_vgg19.png`
- Figure 4: `./results/vgg19_seed2/loss_vgg19.png`

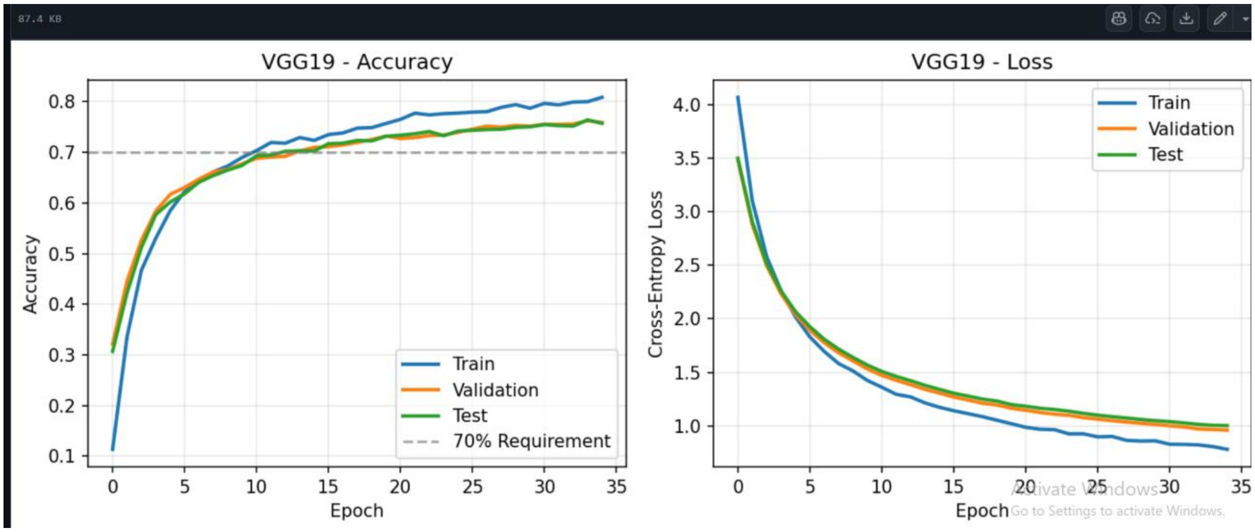

Figure 1:Vgg19_seed1 results

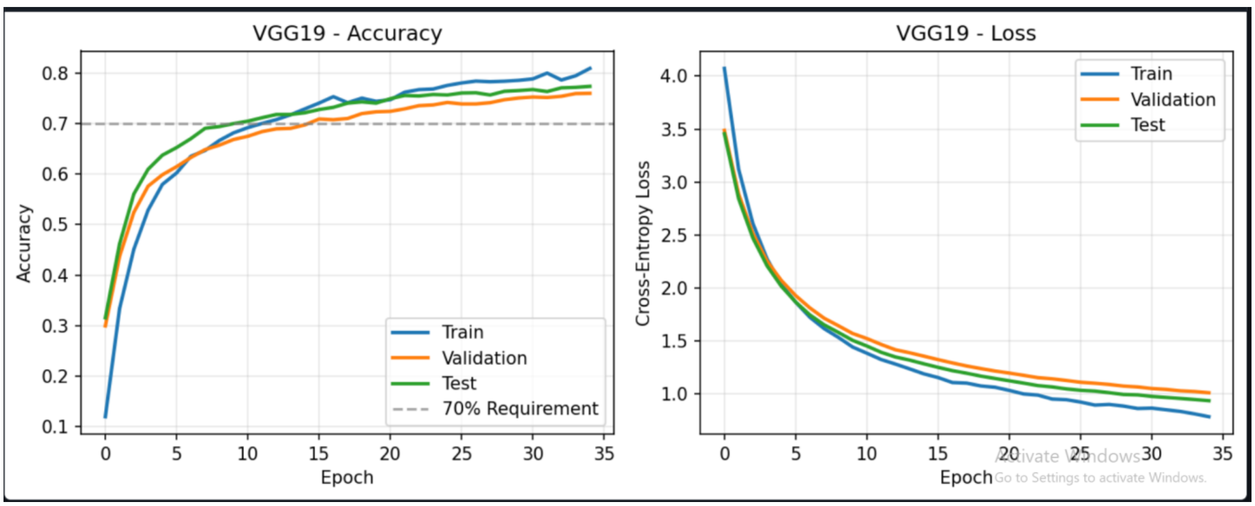

Figure 2:Vgg19_Seed2 results

YOLOv5-CLS

- Figure 5: `./results/yolov5_cls_seed1/accuracy_yolov5.png`
- Figure 6: `./results/yolov5_cls_seed1/loss_yolov5.png`
- Figure 7: `./results/yolov5_cls_seed2/accuracy_yolov5.png`
- Figure 8: `./results/yolov5_cls_seed2/loss_yolov5.png`

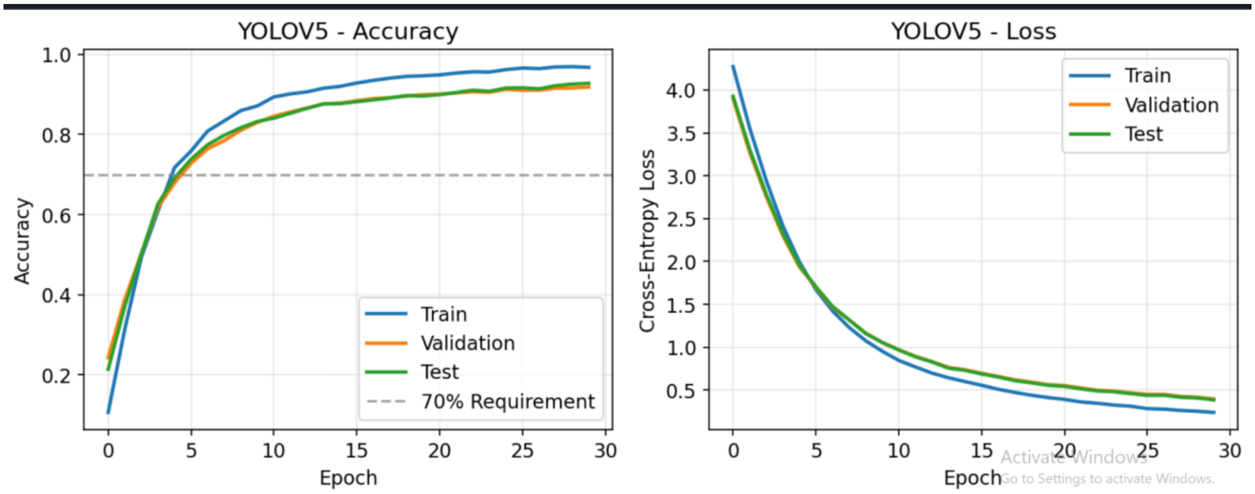

Figure 3: YOLOv5_CLS seed 1 results

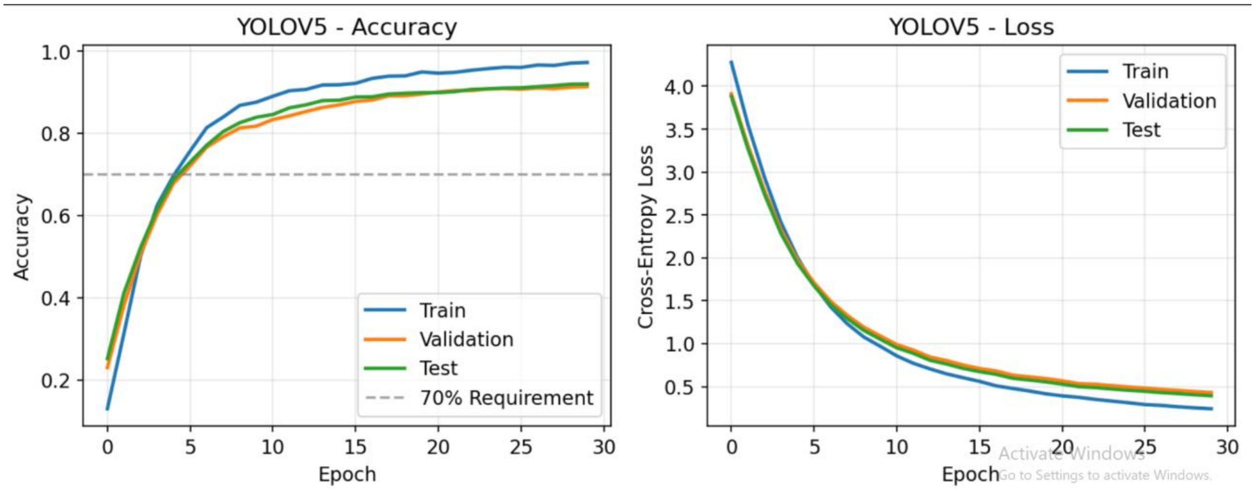

Figure 4: YOLOV5_seed2 results

ResNet-50 Outputs

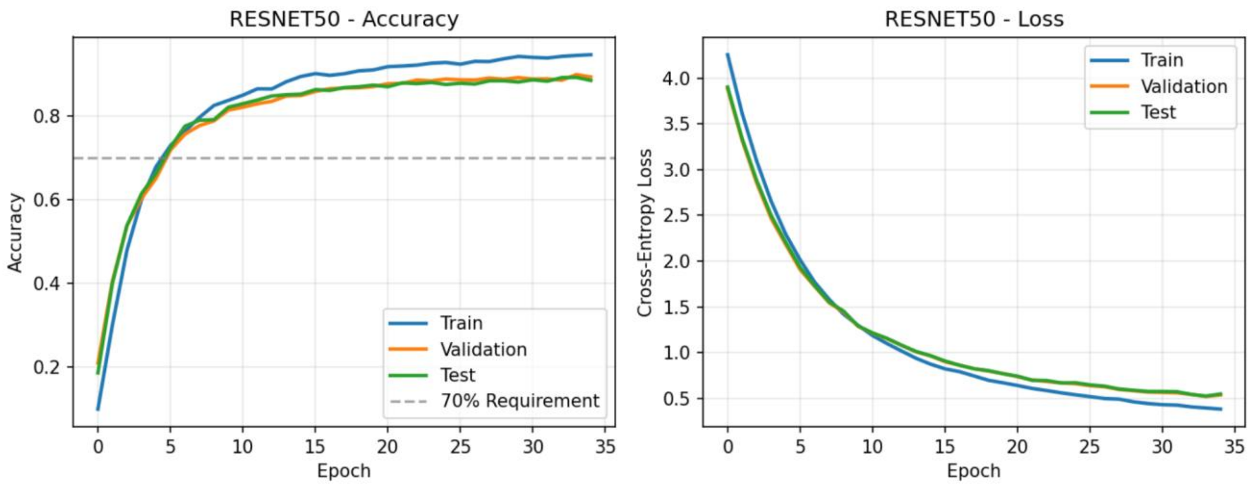

Figure 5:ResNet-50 Seed 1 results

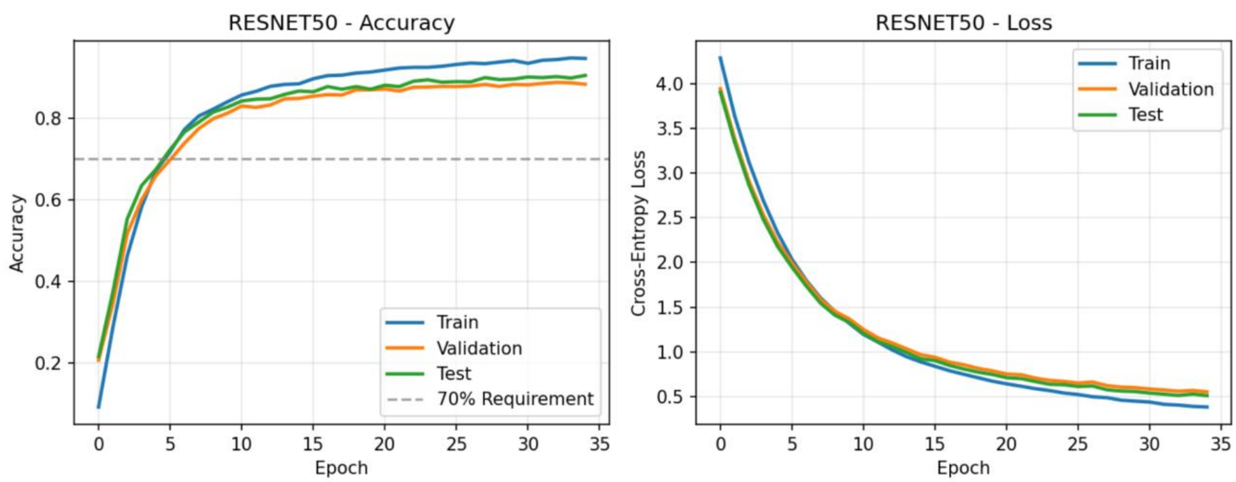

Figure 6: ResNet-50 Seed 2 results

8. Discussion

8.1 Performance comparison

YOLOv5-CLS achieves the highest overall performance on the Flowers-102 dataset, but ResNet-50 performs competitively and substantially outperforms VGG19 across both random splits.

- YOLOv5-CLS: ≈92–93% test accuracy

- ResNet-50: ≈89–90% test accuracy
- VGG19: ≈76–77% test accuracy

Compared to VGG19, ResNet-50 improves test accuracy by approximately 12–14 percentage points, while YOLOv5-CLS improves accuracy by approximately 15–16 percentage points.

Explanation: deeper residual architectures (ResNet-50) and YOLOv5-CLS's modern classification head provide stronger feature reuse and gradient flow than VGG19, whose frozen convolutional backbone limits representational adaptation. YOLOv5-CLS further benefits from a backbone and head optimized explicitly for classification rather than repurposed from a legacy architecture.

8.2 Generalization and split stability

All three models demonstrate **stable performance across different random splits**, indicating that the results are not overly sensitive to the partitioning of the dataset.

- **VGG19:** ~0.9% absolute variation in test accuracy (0.7642 → 0.7734)
- **YOLOv5-CLS:** ~0.8% absolute variation (0.9277 → 0.9199)
- **ResNet-50:** ~0.9% absolute variation (0.8916 → 0.9009)

This consistency across seeds suggests good generalization and robustness of the training pipeline.

8.3 Training dynamics and calibration

Clear differences emerge in training behavior and loss values:

- **VGG19** exhibits the highest test loss (≈0.94–1.01), indicating weaker confidence and less effective probability calibration.
- **ResNet-50** achieves substantially lower test loss (≈0.51–0.52), reflecting more confident and better-calibrated predictions.
- **YOLOv5-CLS** achieves the lowest test loss (≈0.39) alongside the highest accuracy, suggesting both strong correctness and strong probability concentration on correct classes.

These trends align with architectural differences and the degree of effective feature adaptation enabled during transfer learning.

8.2 Generalization and split stability

All three models demonstrate **stable performance across different random splits**, indicating that the results are not overly sensitive to the partitioning of the dataset.

- **VGG19:** ~0.9% absolute variation in test accuracy (0.7642 → 0.7734)
- **YOLOv5-CLS:** ~0.8% absolute variation (0.9277 → 0.9199)
- **ResNet-50:** ~0.9% absolute variation (0.8916 → 0.9009)

This consistency across seeds suggests good generalization and robustness of the training pipeline.

8.3 Training dynamics and calibration

Clear differences emerge in training behavior and loss values:

- **VGG19** exhibits the highest test loss (≈0.94–1.01), indicating weaker confidence and less effective probability calibration.
- **ResNet-50** achieves substantially lower test loss (≈0.51–0.52), reflecting more confident and better-calibrated predictions.
- **YOLOv5-CLS** achieves the lowest test loss (≈0.39) alongside the highest accuracy, suggesting both strong correctness and strong probability concentration on correct classes.

These trends align with architectural differences and the degree of effective feature adaptation enabled during transfer learning.