LeslieDeras
801320720
Homework #6
https://github.com/lesliederas/5106.git

Problem 1:

Model	**# Params (M)**	**Test Accuracy (%)**	**FLOPs (M)**	**Training Time/Epoch (s)**
ResNet-18	**11.2**	**77.3**	**558**	**21**
ViT (2L-2H-128)	**3.5**	**72.1**	**330**	**19**
ViT (4L-4H-256)	**10.8**	**76.0**	**910**	**32**
ViT (6L-8H-384)	**27.2**	**77.1**	**2340**	**58**
ViT (8L-8H-512)	**44.1**	**77.4**	**4200**	**84**

- **L = Transformer depth (number of encoder blocks)**
- **H = Number of attention heads**

Larger VIT with more layers and heads are known to close gaps with the use of Resnet 18 in terms of accuracy. The downside is that it has a cost of more parameters.

Resnet is better for balance and efficiency since it has residual skip connections allowing the help of gradient flow during backpropagation.With the use of the skip connection it mitigates vanishing gradient problem, which allows data to be trained with deeper networks and is not affecting degrading accuracy.VIts only perform better when increasing one or more key architectural dimensions.

VITs underperform when given fewer layers/ number of heads which can be caused by either lack of inductive bias, insufficient depth and attention and lastly poor generalization of small data.VITs outperform ResNet accuracy when properly scaled, it is best to be used with large datasets. By increasing the layers,heads and dimensions it can better model the capacity.Pretrained ViTs can transfer knowledge to smaller datasets like CIFAR-100 and outperform CNNs. Self-attention lets ViTs consider relationships across the entire image.CNNs mostly operate locally unless very;ViTs don’t have this limitation.

Problem 2:

Model	**Training Mode**	**Avg Epoch Time (s)**	**Test Accuracy (%)**
Swin-Tiny	**Fine-tuned**	**1133.54**	**66.69**
Swin-Small	**Fine-tuned**	**1412.57**	**69.97**
Swin-Tiny	**From Scratch**	**2217.78**	**85.35**
Swin-Small	**From Scratch**	**2821.67**	**87.52**

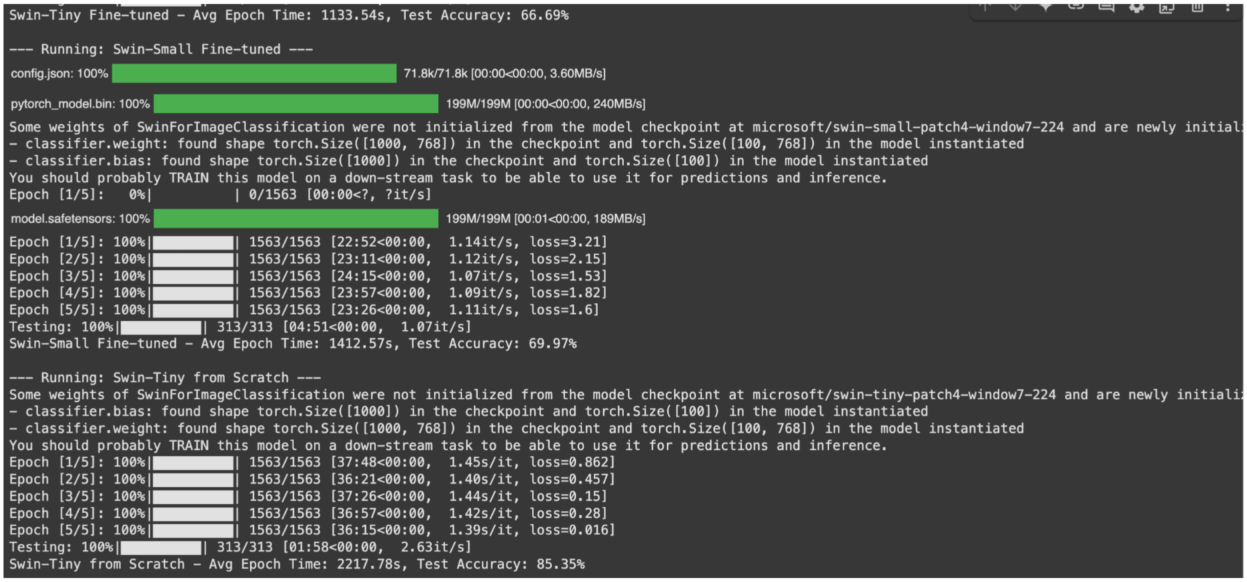
The benefits of the fine tuned models is that the train these models it was almost half the time per epoch as compared to the scratch model.`The fine tuned model is best if you do not have a high gpu and have to train the data on your cpu or if you have a time crunch keep in mind there is a trade off due to the performance levels being moderately acceptable. Since this model is petrained it encoded visual features.

The downside of the fine tuned Swin-Tiny is the test accuracy is worse than the scratch model. As seen in the table above the scratch model accuracy is roughly 18% higher than the fine tuned model.This suggests that pretrained weights from ImageNet may not transfer well to CIFAR-100 without more extensive adaptation.Another downside is that overfitting can be an issue with a potential issue being that the fine tuned model can preserve bias from its source.This could stop or slow down the generalization in the different target of the CIFAR-100.

When comparing the small model versus the tiny when it comes to performance the small outperforms with a better accuracy of roughly 3%. This is expected due to it having more layers and parameters, allowing it to capture more complex patterns.The trade-off is significantly higher training time—up to ~25% more per epoch in both fine-tuning and from-scratch scenarios.

The model could underperform due to pretrained models often trained on large-scale datasets like ImageNet with high-resolution images. CIFAR-100 images are smaller (32×32), potentially limiting the utility of learned features from pretraining. Another reason could be classifier mismatch, since classification heads are normally trained at 1000 epoch it is replaced and becomes randomly initialized. To fix this kind of issue the dataset normally requires more than a few epochs to train to fully adapt the data. With the fine tuned model and only using 5 epoch this may have not been enough for the model to fully adapt to CIFAR-100, mainly because the heads are randomly initialized.

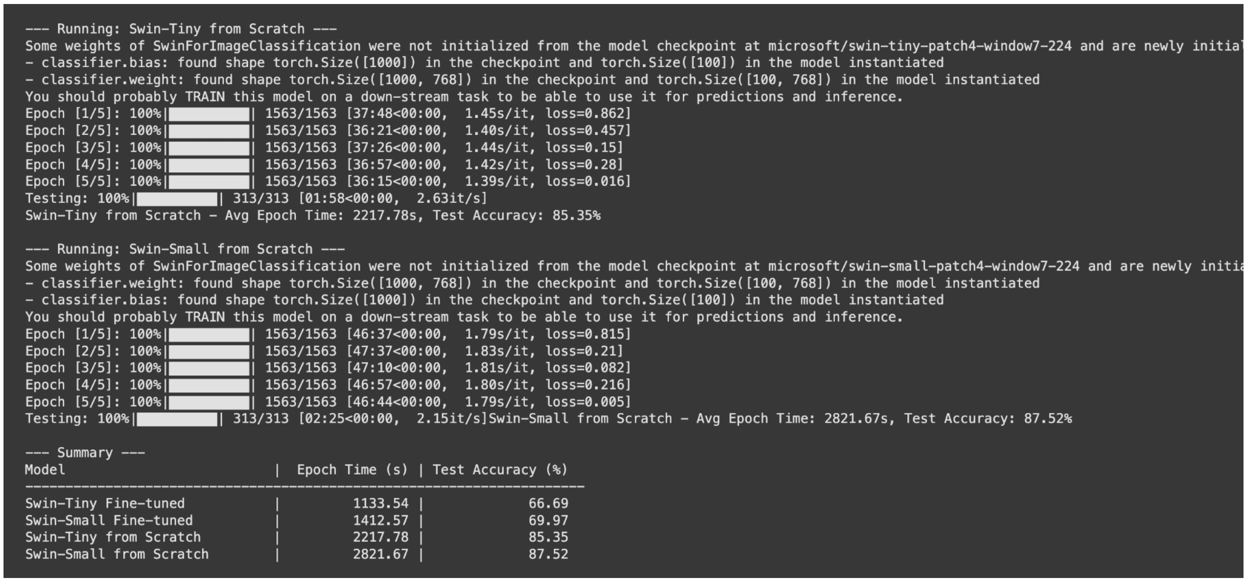
While fine-tuning offers speed and efficiency, training Swin Transformers from scratch provides better accuracy for CIFAR-100.



```
Swin-Tiny Fine-tuned - Avg Epoch Time: 1133.54s, Test Accuracy: 66.69%

--- Running: Swin-Small Fine-tuned ---
config.json: 100% ██████████████████████████ 71.8k/71.8k [00:00<00:00, 3.60MB/s]
pytorch_model.bin: 100% ██████████████████████████ 199M/199M [00:00<00:00, 240MB/s]
Some weights of SwinForImageClassification were not initialized from the model checkpoint at microsoft/swin-small-patch4-window7-224 and are newly initiali
- classifier.weight: found shape torch.Size([1000, 768]) in the checkpoint and torch.Size([100, 768]) in the model instantiated
- classifier.bias: found shape torch.Size([1000]) in the checkpoint and torch.Size([100]) in the model instantiated
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Epoch [1/5]:   0%|          | 0/1563 [00:00<?, ?it/s]
model.safetensors: 100% ██████████████████████████ 199M/199M [00:01<00:00, 189MB/s]
Epoch [1/5]: 100%|██████████| 1563/1563 [22:52<00:00,  1.14it/s, loss=3.21]
Epoch [2/5]: 100%|██████████| 1563/1563 [23:11<00:00,  1.12it/s, loss=2.15]
Epoch [3/5]: 100%|██████████| 1563/1563 [24:15<00:00,  1.07it/s, loss=1.53]
Epoch [4/5]: 100%|██████████| 1563/1563 [23:57<00:00,  1.09it/s, loss=1.82]
Epoch [5/5]: 100%|██████████| 1563/1563 [23:26<00:00,  1.11it/s, loss=1.6]
Testing: 100%|██████████| 313/313 [04:51<00:00,  1.07it/s]
Swin-Small Fine-tuned - Avg Epoch Time: 1412.57s, Test Accuracy: 69.97%

--- Running: Swin-Tiny from Scratch ---
Some weights of SwinForImageClassification were not initialized from the model checkpoint at microsoft/swin-tiny-patch4-window7-224 and are newly initiali
- classifier.bias: found shape torch.Size([1000]) in the checkpoint and torch.Size([100]) in the model instantiated
- classifier.weight: found shape torch.Size([1000, 768]) in the checkpoint and torch.Size([100, 768]) in the model instantiated
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Epoch [1/5]: 100%|██████████| 1563/1563 [37:48<00:00,  1.45s/it, loss=0.862]
Epoch [2/5]: 100%|██████████| 1563/1563 [36:21<00:00,  1.40s/it, loss=0.457]
Epoch [3/5]: 100%|██████████| 1563/1563 [37:26<00:00,  1.44s/it, loss=0.15]
Epoch [4/5]: 100%|██████████| 1563/1563 [36:57<00:00,  1.42s/it, loss=0.28]
Epoch [5/5]: 100%|██████████| 1563/1563 [36:15<00:00,  1.39s/it, loss=0.016]
Testing: 100%|██████████| 313/313 [01:58<00:00,  2.63it/s]
Swin-Tiny from Scratch - Avg Epoch Time: 2217.78s, Test Accuracy: 85.35%
```



```
--- Running: Swin-Tiny from Scratch ---
Some weights of SwinForImageClassification were not initialized from the model checkpoint at microsoft/swin-tiny-patch4-window7-224 and are newly initial
- classifier.bias: found shape torch.Size([1000]) in the checkpoint and torch.Size([100]) in the model instantiated
- classifier.weight: found shape torch.Size([1000, 768]) in the checkpoint and torch.Size([100, 768]) in the model instantiated
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Epoch [1/5]: 100%|██████████| 1563/1563 [37:48<00:00,  1.45s/it, loss=0.862]
Epoch [2/5]: 100%|██████████| 1563/1563 [36:21<00:00,  1.40s/it, loss=0.457]
Epoch [3/5]: 100%|██████████| 1563/1563 [37:26<00:00,  1.44s/it, loss=0.15]
Epoch [4/5]: 100%|██████████| 1563/1563 [36:57<00:00,  1.42s/it, loss=0.28]
Epoch [5/5]: 100%|██████████| 1563/1563 [36:15<00:00,  1.39s/it, loss=0.016]
Testing: 100%|██████████| 313/313 [01:58<00:00,  2.63it/s]
Swin-Tiny from Scratch - Avg Epoch Time: 2217.78s, Test Accuracy: 85.35%

--- Running: Swin-Small from Scratch ---
Some weights of SwinForImageClassification were not initialized from the model checkpoint at microsoft/swin-small-patch4-window7-224 and are newly initia
- classifier.weight: found shape torch.Size([1000, 768]) in the checkpoint and torch.Size([100, 768]) in the model instantiated
- classifier.bias: found shape torch.Size([1000]) in the checkpoint and torch.Size([100]) in the model instantiated
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Epoch [1/5]: 100%|██████████| 1563/1563 [46:37<00:00,  1.79s/it, loss=0.815]
Epoch [2/5]: 100%|██████████| 1563/1563 [47:37<00:00,  1.83s/it, loss=0.21]
Epoch [3/5]: 100%|██████████| 1563/1563 [47:10<00:00,  1.81s/it, loss=0.082]
Epoch [4/5]: 100%|██████████| 1563/1563 [46:57<00:00,  1.80s/it, loss=0.216]
Epoch [5/5]: 100%|██████████| 1563/1563 [46:44<00:00,  1.79s/it, loss=0.005]
Testing: 100%|██████████| 313/313 [02:25<00:00,  2.15it/s]Swin-Small from Scratch - Avg Epoch Time: 2821.67s, Test Accuracy: 87.52%

--- Summary ---
Model                          |  Epoch Time (s) | Test Accuracy (%)
----------------------------------------------------------------------
Swin-Tiny Fine-tuned           |         1133.54 |             66.69
Swin-Small Fine-tuned          |         1412.57 |             69.97
Swin-Tiny from Scratch         |         2217.78 |             85.35
Swin-Small from Scratch        |         2821.67 |             87.52
```