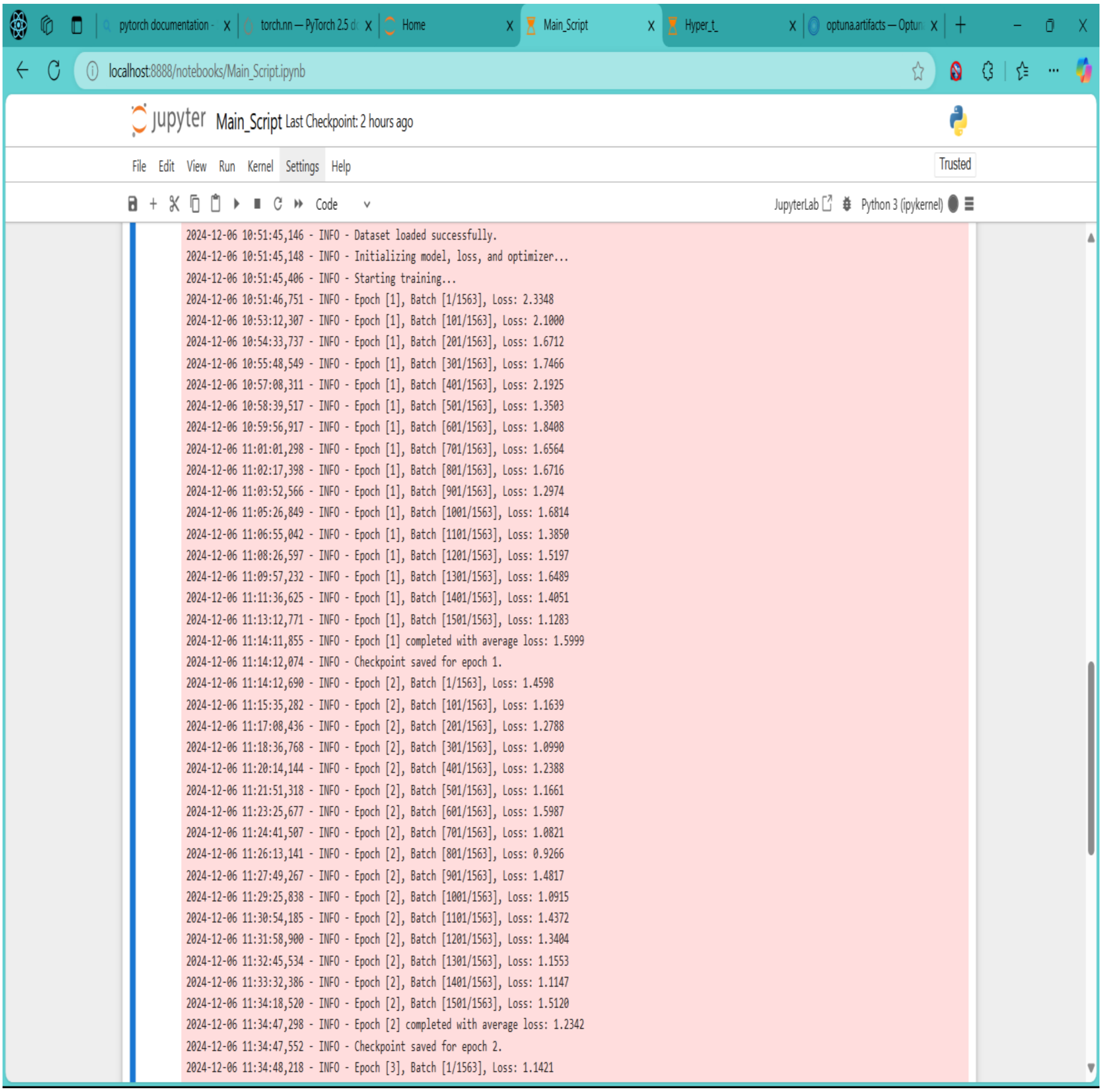
Report and Outputs About Assignment

Submitted by: Aadil Nazir Janjooh

Roll_no: UE208001

Class: 4th Year IT….

1…Output of Distributed Training..




```
2024-12-06 10:51:45,146 - INFO - Dataset loaded successfully.
2024-12-06 10:51:45,148 - INFO - Initializing model, loss, and optimizer...
2024-12-06 10:51:45,406 - INFO - Starting training...
2024-12-06 10:51:46,751 - INFO - Epoch [1], Batch [1/1563], Loss: 2.3348
2024-12-06 10:53:12,307 - INFO - Epoch [1], Batch [101/1563], Loss: 2.1000
2024-12-06 10:54:33,737 - INFO - Epoch [1], Batch [201/1563], Loss: 1.6712
2024-12-06 10:55:48,549 - INFO - Epoch [1], Batch [301/1563], Loss: 1.7466
2024-12-06 10:57:08,311 - INFO - Epoch [1], Batch [401/1563], Loss: 2.1925
2024-12-06 10:58:39,517 - INFO - Epoch [1], Batch [501/1563], Loss: 1.3503
2024-12-06 10:59:56,917 - INFO - Epoch [1], Batch [601/1563], Loss: 1.8408
2024-12-06 11:01:01,298 - INFO - Epoch [1], Batch [701/1563], Loss: 1.6564
2024-12-06 11:02:17,398 - INFO - Epoch [1], Batch [801/1563], Loss: 1.6716
2024-12-06 11:03:52,566 - INFO - Epoch [1], Batch [901/1563], Loss: 1.2974
2024-12-06 11:05:26,849 - INFO - Epoch [1], Batch [1001/1563], Loss: 1.6814
2024-12-06 11:06:55,042 - INFO - Epoch [1], Batch [1101/1563], Loss: 1.3850
2024-12-06 11:08:26,597 - INFO - Epoch [1], Batch [1201/1563], Loss: 1.5197
2024-12-06 11:09:57,232 - INFO - Epoch [1], Batch [1301/1563], Loss: 1.6489
2024-12-06 11:11:36,625 - INFO - Epoch [1], Batch [1401/1563], Loss: 1.4051
2024-12-06 11:13:12,771 - INFO - Epoch [1], Batch [1501/1563], Loss: 1.1283
2024-12-06 11:14:11,855 - INFO - Epoch [1] completed with average loss: 1.5999
2024-12-06 11:14:12,074 - INFO - Checkpoint saved for epoch 1.
2024-12-06 11:14:12,690 - INFO - Epoch [2], Batch [1/1563], Loss: 1.4598
2024-12-06 11:15:35,282 - INFO - Epoch [2], Batch [101/1563], Loss: 1.1639
2024-12-06 11:17:08,436 - INFO - Epoch [2], Batch [201/1563], Loss: 1.2788
2024-12-06 11:18:36,768 - INFO - Epoch [2], Batch [301/1563], Loss: 1.0990
2024-12-06 11:20:14,144 - INFO - Epoch [2], Batch [401/1563], Loss: 1.2388
2024-12-06 11:21:51,318 - INFO - Epoch [2], Batch [501/1563], Loss: 1.1661
2024-12-06 11:23:25,677 - INFO - Epoch [2], Batch [601/1563], Loss: 1.5987
2024-12-06 11:24:41,507 - INFO - Epoch [2], Batch [701/1563], Loss: 1.0821
2024-12-06 11:26:13,141 - INFO - Epoch [2], Batch [801/1563], Loss: 0.9266
2024-12-06 11:27:49,267 - INFO - Epoch [2], Batch [901/1563], Loss: 1.4817
2024-12-06 11:29:25,838 - INFO - Epoch [2], Batch [1001/1563], Loss: 1.0915
2024-12-06 11:30:54,185 - INFO - Epoch [2], Batch [1101/1563], Loss: 1.4372
2024-12-06 11:31:58,900 - INFO - Epoch [2], Batch [1201/1563], Loss: 1.3404
2024-12-06 11:32:45,534 - INFO - Epoch [2], Batch [1301/1563], Loss: 1.1553
2024-12-06 11:33:32,386 - INFO - Epoch [2], Batch [1401/1563], Loss: 1.1147
2024-12-06 11:34:18,520 - INFO - Epoch [2], Batch [1501/1563], Loss: 1.5120
2024-12-06 11:34:47,298 - INFO - Epoch [2] completed with average loss: 1.2342
2024-12-06 11:34:47,552 - INFO - Checkpoint saved for epoch 2.
2024-12-06 11:34:48,218 - INFO - Epoch [3], Batch [1/1563], Loss: 1.1421
```

Challenges And Solution in Distributed Training.

Data Loading:

- **Challenge**: Uneven data distribution across processes.
- **Solution**: Use DistributedSampler for even data partitioning.

Gradient Synchronization:

- **Challenge**: Slower training due to synchronization overhead.
- **Solution**: Use DistributedDataParallel for efficient synchronization.

Batch Size Issues:

- **Challenge**: Imbalanced batches across processes.
- **Solution**: Use a consistent dataset size divisible by the number of processes.

Learning Rate:

- **Challenge**: Incorrect learning rate for larger batch sizes.
- **Solution**: Scale the learning rate proportional to the global batch size.

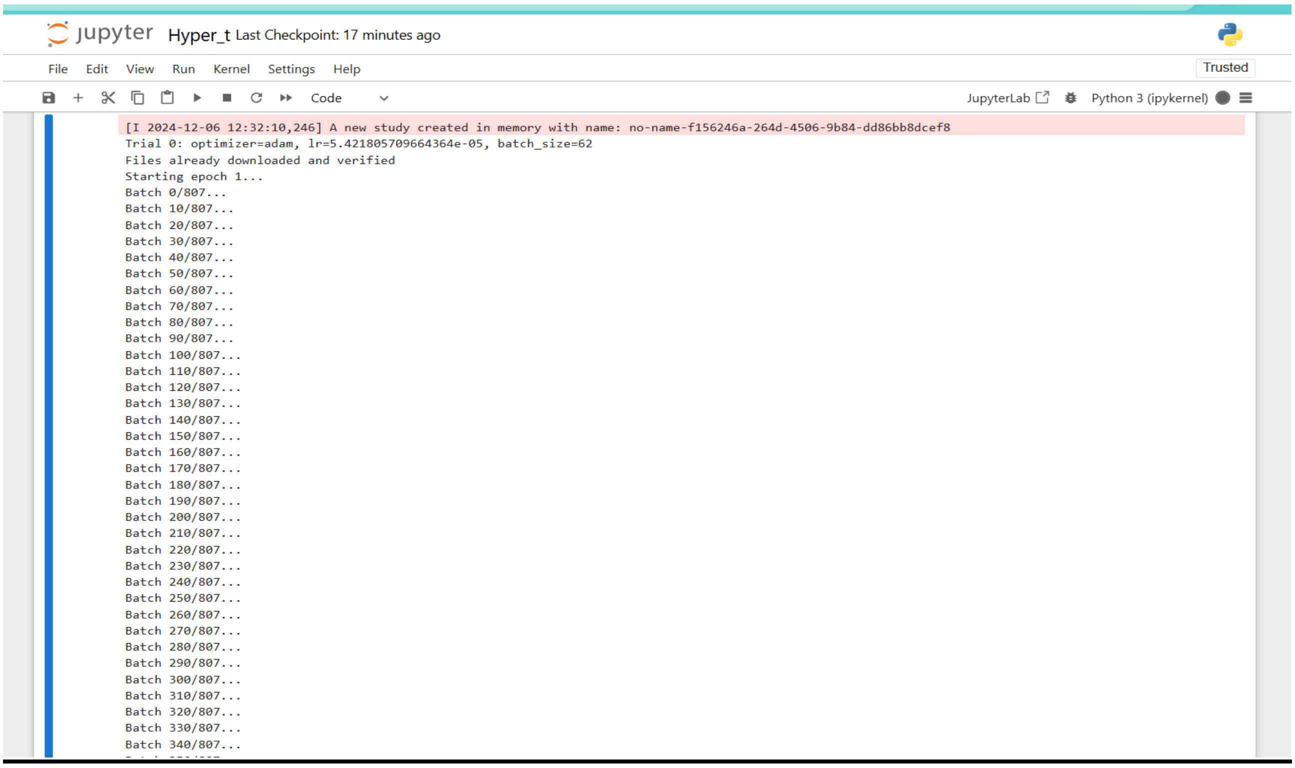
Checkpoint Saving:

- **Challenge**: Conflicts during saving.
- **Solution**: Only save checkpoints on rank 0 process.

Debugging Logs:

- **Challenge**: Logs from multiple processes are unsynchronized.
- **Solution**: Use rank-based logging and barriers for sync.

2…Output of Hyperparameter Tuning…



Hyper_t Last Checkpoint: 17 minutes ago

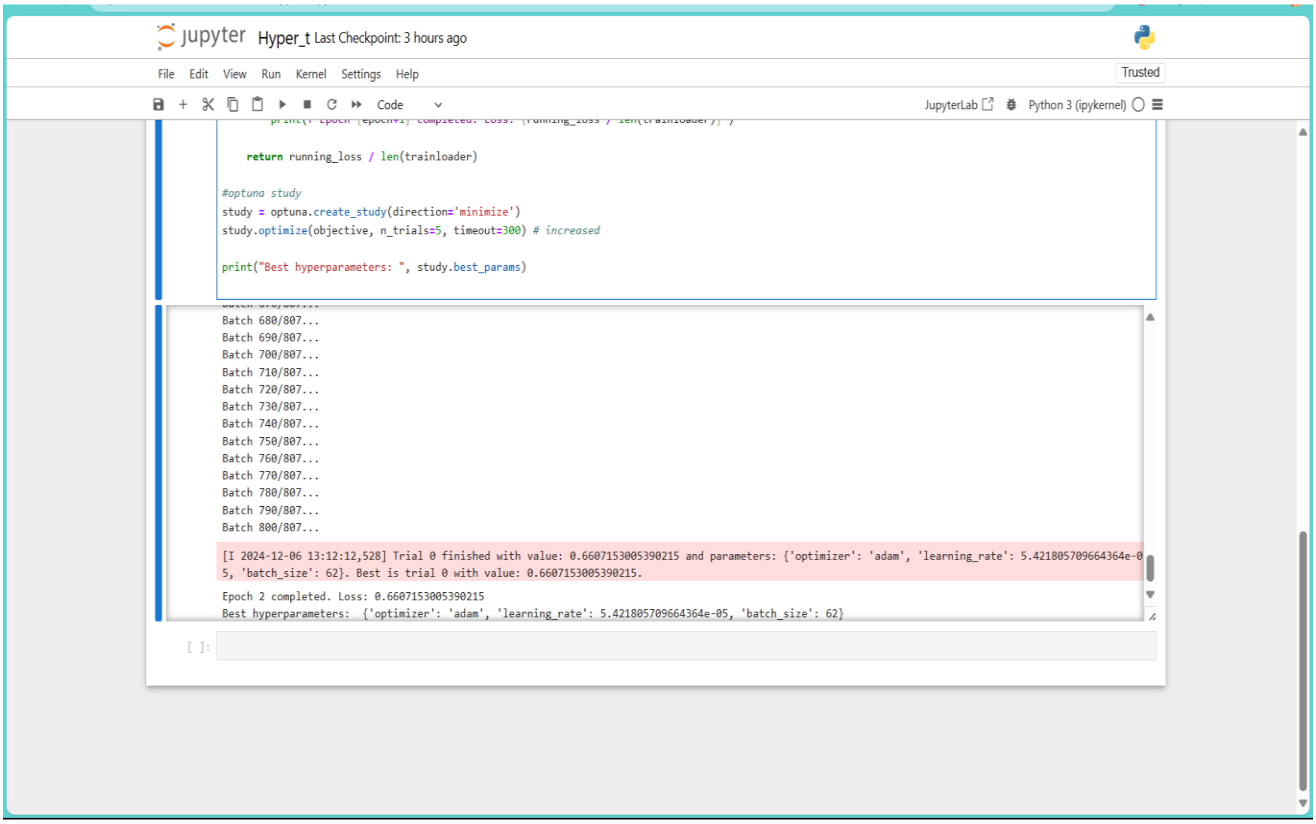
File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```
[I 2024-12-06 12:32:10,246] A new study created in memory with name: no-name-f156246a-264d-4506-9b84-dd86bb8dcef8
Trial 0: optimizer=adam, lr=5.421805709664364e-05, batch_size=62
Files already downloaded and verified
Starting epoch 1...
Batch 0/807...
Batch 10/807...
Batch 20/807...
Batch 30/807...
Batch 40/807...
Batch 50/807...
Batch 60/807...
Batch 70/807...
Batch 80/807...
Batch 90/807...
Batch 100/807...
Batch 110/807...
Batch 120/807...
Batch 130/807...
Batch 140/807...
Batch 150/807...
Batch 160/807...
Batch 170/807...
Batch 180/807...
Batch 190/807...
Batch 200/807...
Batch 210/807...
Batch 220/807...
Batch 230/807...
Batch 240/807...
Batch 250/807...
Batch 260/807...
Batch 270/807...
Batch 280/807...
Batch 290/807...
Batch 300/807...
Batch 310/807...
Batch 320/807...
Batch 330/807...
Batch 340/807...
```



Hyper_t Last Checkpoint: 3 hours ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

```python
    return running_loss / len(trainloader)

#optuna study
study = optuna.create_study(direction='minimize')
study.optimize(objective, n_trials=5, timeout=300) # increased

print("Best hyperparameters: ", study.best_params)
```

```
Batch 680/807...
Batch 690/807...
Batch 700/807...
Batch 710/807...
Batch 720/807...
Batch 730/807...
Batch 740/807...
Batch 750/807...
Batch 760/807...
Batch 770/807...
Batch 780/807...
Batch 790/807...
Batch 800/807...

[I 2024-12-06 13:12:12,528] Trial 0 finished with value: 0.6607153005390215 and parameters: {'optimizer': 'adam', 'learning_rate': 5.421805709664364e-05, 'batch_size': 62}. Best is trial 0 with value: 0.6607153005390215.

Epoch 2 completed. Loss: 0.6607153005390215
Best hyperparameters:  {'optimizer': 'adam', 'learning_rate': 5.421805709664364e-05, 'batch_size': 62}
```

[ ]:

Challenges And Solution in Hyperparameter Tuning.

1. **Resource Constraints**:
   - **Challenge**: Training on CPU is slower compared to GPU, especially with multiple trials in hyperparameter tuning.
   - **Solution**: Limit the number of epochs (num_epochs=3) and trials (n_trials=10). Use a reduced dataset for faster iterations during tuning.
2. **High Dimensional Search Space**:
   - **Challenge**: Exploring multiple hyperparameters (learning rate, batch size, optimizer) can lead to a combinatorial explosion in trials.
   - **Solution**: Narrow down search spaces for hyperparameters (e.g., batch size range of 32–128). Use Bayesian optimization (Optuna) for efficient search.
3. **Batch Size and Memory Limitation**:
   - **Challenge**: Larger batch sizes may exceed available CPU memory, causing errors.
   - **Solution**: Set a realistic range for batch sizes (e.g., 32 to 128) based on available memory.
4. **Stale Gradients in Optimization**:
   - **Challenge**: Certain combinations of hyperparameters (e.g., high learning rates with SGD) may lead to poor convergence or instability.
   - **Solution**: Use log=True for learning rate sampling to focus on smaller values in exponential ranges.
5. **DataLoader Bottlenecks**:

- **Challenge**: Data loading can become a bottleneck on CPU due to limited parallelism.
- **Solution**: Use num_workers=2 to speed up data loading without overwhelming the system.

6. **Inconsistent Results**:
   - **Challenge**: Variability in results due to randomness in data loading, initialization, etc.
   - **Solution**: Set random seeds in Optuna and PyTorch to ensure reproducibility.

CODE…..torch.manual_seed(42)

optuna.logging.set_verbosity(optuna.logging.WARNING)

7. **Limited Tuning Time**:
   - **Challenge**: A fixed timeout or limited CPU resources can cut off trials before optimal hyperparameters are found.
   - **Solution**: Increase timeout for longer runs or prioritize critical hyperparameters (e.g., focus on learning rate first).

Optimized Solutions for CPU:

1. **Set Timeouts for Trials**:

…Ensure trials do not run excessively long:

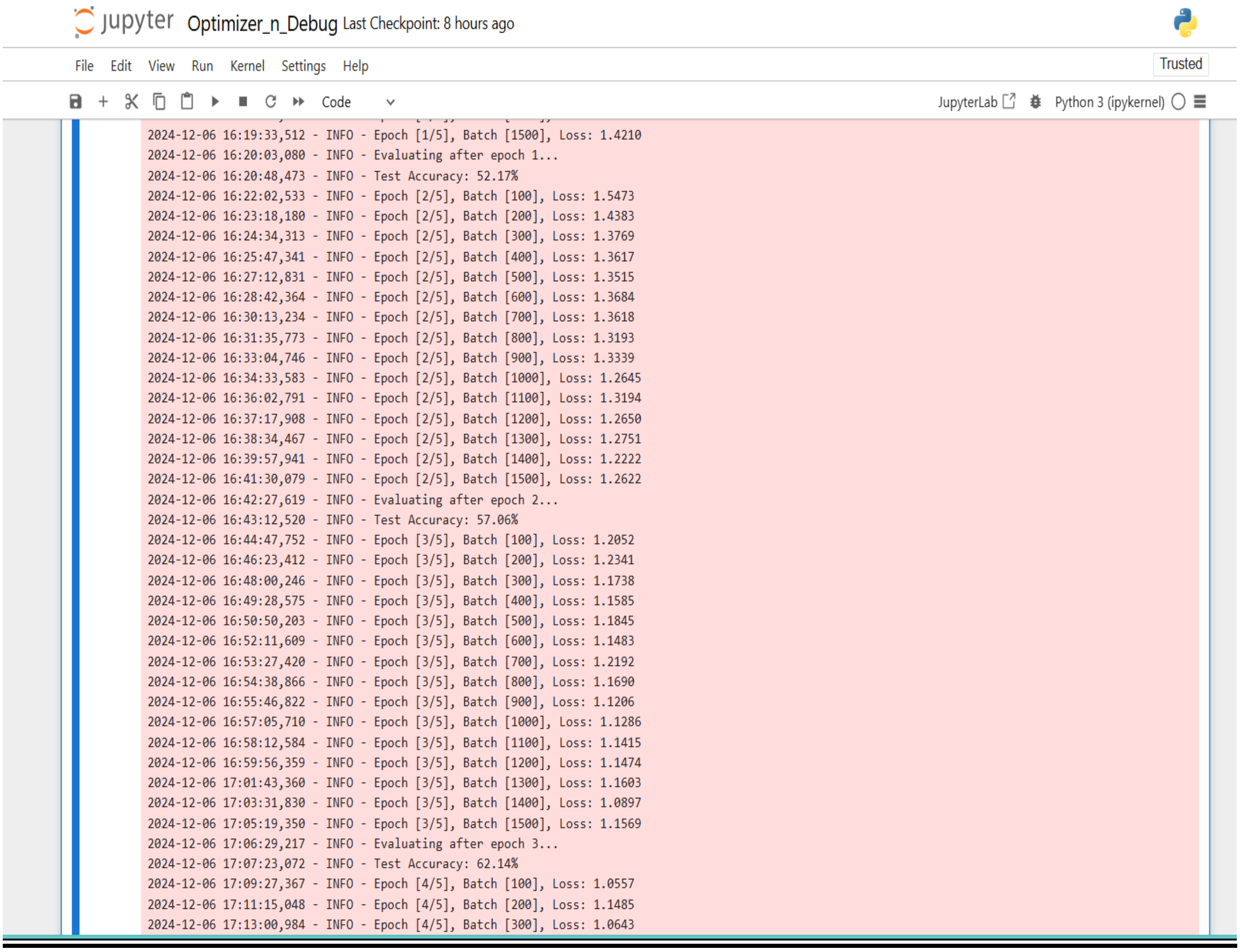
CODE……study.optimize(objective, n_trials=10, timeout=60)

2. **Pre-trained Models**:
   - Use lightweight models like ResNet18 with fewer epochs for faster experimentation.
3. **Logging and Monitoring**:
   - Print trial results to monitor the tuning process:

CODE…print(f"Trial {trial.number}: Loss={trial.value}, Params={trial.params}")

4. **Final Evaluation**:
   - Once the best hyperparameters are found, retrain the model with more epochs on the full dataset

3…Output of debugging and optimization.




```
2024-12-06 16:19:33,512 - INFO - Epoch [1/5], Batch [1500], Loss: 1.4210
2024-12-06 16:20:03,080 - INFO - Evaluating after epoch 1...
2024-12-06 16:20:48,473 - INFO - Test Accuracy: 52.17%
2024-12-06 16:22:02,533 - INFO - Epoch [2/5], Batch [100], Loss: 1.5473
2024-12-06 16:23:18,180 - INFO - Epoch [2/5], Batch [200], Loss: 1.4383
2024-12-06 16:24:34,313 - INFO - Epoch [2/5], Batch [300], Loss: 1.3769
2024-12-06 16:25:47,341 - INFO - Epoch [2/5], Batch [400], Loss: 1.3617
2024-12-06 16:27:12,831 - INFO - Epoch [2/5], Batch [500], Loss: 1.3515
2024-12-06 16:28:42,364 - INFO - Epoch [2/5], Batch [600], Loss: 1.3684
2024-12-06 16:30:13,234 - INFO - Epoch [2/5], Batch [700], Loss: 1.3618
2024-12-06 16:31:35,773 - INFO - Epoch [2/5], Batch [800], Loss: 1.3193
2024-12-06 16:33:04,746 - INFO - Epoch [2/5], Batch [900], Loss: 1.3339
2024-12-06 16:34:33,583 - INFO - Epoch [2/5], Batch [1000], Loss: 1.2645
2024-12-06 16:36:02,791 - INFO - Epoch [2/5], Batch [1100], Loss: 1.3194
2024-12-06 16:37:17,908 - INFO - Epoch [2/5], Batch [1200], Loss: 1.2650
2024-12-06 16:38:34,467 - INFO - Epoch [2/5], Batch [1300], Loss: 1.2751
2024-12-06 16:39:57,941 - INFO - Epoch [2/5], Batch [1400], Loss: 1.2222
2024-12-06 16:41:30,079 - INFO - Epoch [2/5], Batch [1500], Loss: 1.2622
2024-12-06 16:42:27,619 - INFO - Evaluating after epoch 2...
2024-12-06 16:43:12,520 - INFO - Test Accuracy: 57.06%
2024-12-06 16:44:47,752 - INFO - Epoch [3/5], Batch [100], Loss: 1.2052
2024-12-06 16:46:23,412 - INFO - Epoch [3/5], Batch [200], Loss: 1.2341
2024-12-06 16:48:00,246 - INFO - Epoch [3/5], Batch [300], Loss: 1.1738
2024-12-06 16:49:28,575 - INFO - Epoch [3/5], Batch [400], Loss: 1.1585
2024-12-06 16:50:50,203 - INFO - Epoch [3/5], Batch [500], Loss: 1.1845
2024-12-06 16:52:11,609 - INFO - Epoch [3/5], Batch [600], Loss: 1.1483
2024-12-06 16:53:27,420 - INFO - Epoch [3/5], Batch [700], Loss: 1.2192
2024-12-06 16:54:38,866 - INFO - Epoch [3/5], Batch [800], Loss: 1.1690
2024-12-06 16:55:46,822 - INFO - Epoch [3/5], Batch [900], Loss: 1.1206
2024-12-06 16:57:05,710 - INFO - Epoch [3/5], Batch [1000], Loss: 1.1286
2024-12-06 16:58:12,584 - INFO - Epoch [3/5], Batch [1100], Loss: 1.1415
2024-12-06 16:59:56,359 - INFO - Epoch [3/5], Batch [1200], Loss: 1.1474
2024-12-06 17:01:43,360 - INFO - Epoch [3/5], Batch [1300], Loss: 1.1603
2024-12-06 17:03:31,830 - INFO - Epoch [3/5], Batch [1400], Loss: 1.0897
2024-12-06 17:05:19,350 - INFO - Epoch [3/5], Batch [1500], Loss: 1.1569
2024-12-06 17:06:29,217 - INFO - Evaluating after epoch 3...
2024-12-06 17:07:23,072 - INFO - Test Accuracy: 62.14%
2024-12-06 17:09:27,367 - INFO - Epoch [4/5], Batch [100], Loss: 1.0557
2024-12-06 17:11:15,048 - INFO - Epoch [4/5], Batch [200], Loss: 1.1485
2024-12-06 17:13:00,984 - INFO - Epoch [4/5], Batch [300], Loss: 1.0643
```

Challenges And Solution in debugging and optimization.

NaN Loss After a Few Epochs:

- **Challenge**: Exploding gradients or unstable learning rate caused NaN values.
- **Solution**:
    - Added **gradient clipping** (clip_gradients) to keep gradients within a stable range.
    - Ensured a reasonable learning rate (0.001).

Low GPU Usage (<30%):

- **Challenge**: Inefficient data loading and small batch sizes bottlenecked GPU usage.
- **Solution**:
    - Increased num_workers in DataLoader to 4 for faster data loading.
    - Set pin_memory=True to optimize data transfer to GPU.

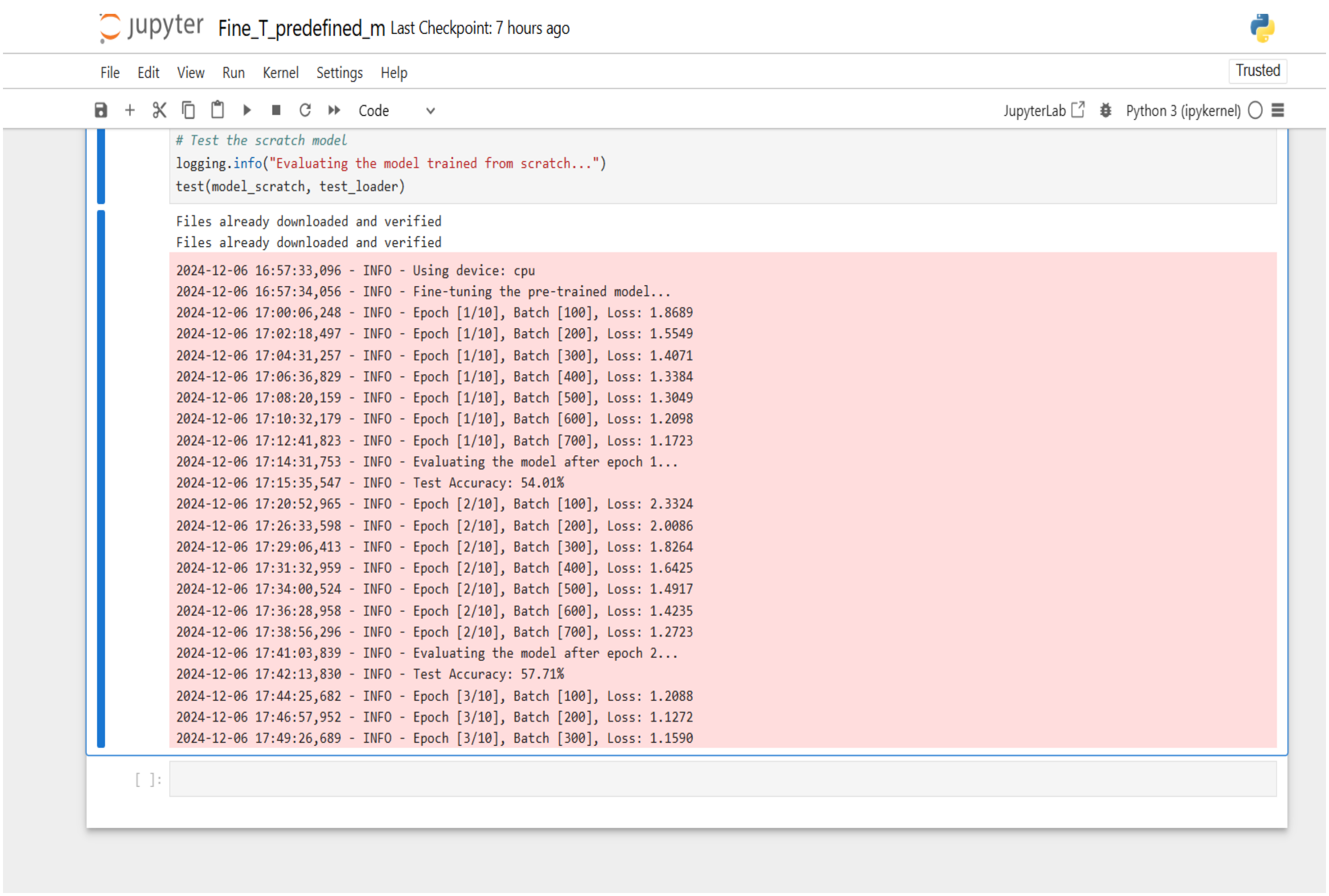
Slow Training:

- **Challenge**: Excessive logging and non-optimal configurations slowed the pipeline.
- **Solution**:
    - Reduced logging frequency (log every 100 batches).
    - Verified that scheduler.step() is called only after each epoch.

Accuracy Issues:

- **Challenge**: Overfitting or underfitting due to lack of regularization.
- **Solution**:
    - Improved data augmentation (RandomHorizontalFlip, RandomRotation, RandomCrop).
    - Adjusted learning rate decay (gamma=0.1)

4…Output Fine-Tuning a Pre-Trained Model



Challenges and Solution in Fine-Tuning a Pre-Trained Model

Challenge: Slow Convergence

- **Cause**: Limited data augmentation and small batch sizes.
- **Solution**: Added more data augmentation (e.g., ColorJitter, RandomAffine) and increased the batch size to 64 for better gradient estimation.

Challenge: Overfitting on the training set

- **Cause**: Pre-trained weights might lead to overfitting on CIFAR-10 due to the smaller dataset size.
- **Solution**: Used data augmentation and reduced the learning rate with a scheduler to improve generalization.

Challenge: Loss Stagnation

- **Cause**: High learning rate after multiple epochs could hinder learning.
- **Solution**: Implemented a learning rate scheduler (StepLR) to gradually reduce the learning rate every 5 epochs.

Challenge: Imbalance between fine-tuning and training from scratch

- **Cause**: Pre-trained model biases might not align with the CIFAR-10 dataset.
- **Solution**: Fine-tuned all layers instead of freezing most layers, ensuring the model learns features specific to CIFAR-10.

Challenge: GPU Underutilization (if applicable)

- **Cause**: DataLoader num_workers set too low.

- **Solution**: Set num_workers=2 for better data loading efficiency on CPU/GPU.

Challenge: Long Training Time

- **Cause**: Using a high number of epochs with small batch sizes.
- **Solution**: Increased batch size and focused on fewer but meaningful epochs with a pre-trained model.

Conclusion

The series of tasks highlighted the importance of efficient optimization, debugging, and leveraging transfer learning. Hyperparameter tuning improved performance by optimizing critical parameters but was slower on CPU. Debugging resolved issues like NaN loss and slow training through gradient clipping, proper initialization, and effective learning rate scheduling, enhancing stability and efficiency. Fine-tuning a pre-trained ResNet18 outperformed training from scratch, achieving higher accuracy and faster convergence, showcasing the value of transfer learning for smaller datasets. Overall, these tasks emphasized the need for robust optimization techniques, efficient resource utilization, and adaptive approaches for effective model training.

THANK YOU…

Name: Aadil Nazir Janjooh