**Assignment 4: SeqTrack Inference**

**Team 16**

**GitHub Link :** **https://github.com/Amr20004/assignment\_4**

**Table 1: Inference rate results (FPS)**

In each testing cycle we have 8 sequences from the testing split. 4 sequences from each dataset class. we have 20 checkpoint to test 10 from phase1 and 10 from phase2

For checkpoint 1 Phase 1:

**FPS Values:**

1. 6.283110028661813
2. 6.405272580972249
3. 6.29853459399873
4. 6.162784306995751
5. 6.007053438951351
6. 6.246722994536062
7. 6.5789034373434285
8. 10.932376820062682

**Average FPS:** (6.28 + 6.41 + 6.30 + 6.16 + 6.01 + 6.25 + 6.58 + 10.93) / 8 = **6.86 FPS**

For Checkpoint 2 Phase 1:

**FPS Values:**

1. 5.9880753871873
2. 6.179320555204834
3. 6.107732993963295
4. 5.8512330226578015
5. 5.966566748832799
6. 6.083050037112109
7. 6.189346272989581
8. 9.056326039865395

**Average FPS:** (5.99 + 6.18 + 6.11 + 5.85 + 5.97 + 6.08 + 6.19 + 9.06) / 8 = **6.43 FPS**

For checkpoint 3 Phase 1:

**FPS Values:**

1. 5.966606842625561
2. 6.14678850129254
3. 6.0986947134230185
4. 5.883498741403853
5. 6.07992163374121
6. 6.171702775855031
7. 6.283059867121929
8. 9.76374364160436

**Average FPS:** (5.97 + 6.15 + 6.10 + 5.88 + 6.08 + 6.17 + 6.28 + 9.76) / 8 = **6.55 FPS**

For Checkpoint 4 Phase 1:

**FPS Values:**

1. 5.9715607957872745
2. 6.194618012580016
3. 6.0899063913922875
4. 5.918887808851016
5. 5.991238009058119
6. 6.18712867400991
7. 6.189382435264772
8. 9.07561433627393

**Average FPS:** (5.97 + 6.19 + 6.09 + 5.92 + 5.99 + 6.19 + 6.19 + 9.08) / 8 = **6.45 FPS**

For Checkpoint 5 Phase 1:

**FPS Values:**

1. 5.965130389387295
2. 6.178904858384307
3. 6.103951351160395
4. 5.919404403741607
5. 6.039424680096658
6. 6.062902494886813
7. 6.166595614485143
8. 9.261177921092571

**Average FPS:** (5.97 + 6.18 + 6.10 + 5.92 + 6.04 + 6.06 + 6.17 + 9.26) / 8 = **6.46 FPS**

For Checkpoint 6 Phase 1:

**FPS Values:**

1. 5.996639929351139
2. 6.188459742935889
3. 6.088284370950693
4. 6.038726352745161
5. 5.936567597085767
6. 6.25746236367847
7. 6.232329859623984
8. 9.35087685583404

**Average FPS:** (6.00 + 6.19 + 6.09 + 6.04 + 5.94 + 6.26 + 6.23 + 9.35) / 8 = **6.51 FPS**

For Checkpoint 7 Phase 1:

**FPS Values:**

1. 6.0775665291546925
2. 6.172686959355194
3. 6.107969586754936
4. 5.925683986219534
5. 6.023505467223211
6. 6.139189704041358
7. 6.253025623377052
8. 9.823525123978344

**Average FPS:** (6.08 + 6.17 + 6.11 + 5.93 + 6.02 + 6.14 + 6.25 + 9.82) / 8 = **6.57 FPS**

For Checkpoint 8 Phase 1:

**FPS Values:**

1. 6.012333305526696
2. 6.186214537892006
3. 6.1082478140032395
4. 5.938572445018916
5. 6.003704290937186
6. 6.183128921681925
7. 6.22823002432726
8. 9.255178399769825

**Average FPS:** (6.01 + 6.19 + 6.11 + 5.94 + 6.00 + 6.18 + 6.23 + 9.26) / 8 = **6.49 FPS**

For Checkpoint 9 Phase 1:

**FPS Values:**

1. 5.969668979281739
2. 6.220851189964104
3. 6.091225800895498
4. 5.899306014790825
5. 5.981787476134484
6. 6.159451576883018
7. 6.264279989942877
8. 9.465244825474274

**Average FPS:** (5.97 + 6.22 + 6.09 + 5.90 + 5.98 + 6.16 + 6.26 + 9.47) / 8 = **6.51 FPS**

For Checkpoint 10 Phase 1:

**FPS Values:**

1. 5.97970345678594
2. 6.186509618797683
3. 6.106568298095561
4. 5.951658861606881
5. 6.021032901721535
6. 6.206793774284645
7. 6.208731884606671
8. 9.085629028780208

**Average FPS:** (5.98 + 6.19 + 6.11 + 5.95 + 6.02 + 6.21 + 6.21 + 9.09) / 8 = **6.22 FPS**

For checkpoint 1 Phase 2:

**FPS Values:**

1. 6.283110028661813
2. 6.405272580972249
3. 6.29853459399873
4. 6.162784306995751
5. 6.007053438951351
6. 6.246722994536062
7. 6.5789034373434285
8. 10.932376820062682

**Average FPS:** (6.28 + 6.41 + 6.30 + 6.16 + 6.01 + 6.25 + 6.58 + 10.93) / 8 = **6.86 FPS**

For Checkpoint 2 Phase 2:

**FPS Values:**

1. 5.9880753871873
2. 6.179320555204834
3. 6.107732993963295
4. 5.8512330226578015
5. 5.966566748832799
6. 6.083050037112109
7. 6.189346272989581
8. 9.056326039865395

**Average FPS:** (5.99 + 6.18 + 6.11 + 5.85 + 5.97 + 6.08 + 6.19 + 9.06) / 8 = **6.43 FPS**

For checkpoint 3 Phase 2:

**FPS Values:**

1. 5.966606842625561
2. 6.14678850129254
3. 6.0986947134230185
4. 5.883498741403853
5. 6.07992163374121
6. 6.171702775855031
7. 6.283059867121929
8. 9.76374364160436

**Average FPS:** (5.97 + 6.15 + 6.10 + 5.88 + 6.08 + 6.17 + 6.28 + 9.76) / 8 = **6.55 FPS**

For Checkpoint 4 Phase 2:

**FPS Values:**

1. 5.996480405152977
2. 6.194470731323487
3. 6.105990102319757
4. 5.8840955110367155
5. 5.989377569673795
6. 6.12488456276789
7. 6.224780377810366
8. 9.278649716796288

**Average FPS:** (6.00 + 6.19 + 6.11 + 5.88 + 5.99 + 6.12 + 6.22 + 9.28) / 8 = **6.47 FPS**

For Checkpoint 5 Phase 2:

**FPS Values:**

1. 6.094878680229456
2. 6.214960499582734
3. 6.175816629808346
4. 5.846672500592439
5. 6.041858652486824
6. 6.195731839854018
7. 6.247064177726094
8. 9.542927528417296

**Average FPS:** (6.09 + 6.21 + 6.18 + 5.85 + 6.04 + 6.20 + 6.25 + 9.54) / 8 = **6.55 FPS**

For Checkpoint 6 Phase 2:

**FPS Values:**

1. 5.999754326384149
2. 6.23752703617003
3. 6.130338224669655
4. 5.748398618950896
5. 5.965524355040067
6. 5.896763142447584
7. 6.138152489657123
8. 10.075455514844084

**Average FPS:** (6.00 + 6.24 + 6.13 + 5.75 + 5.97 + 5.90 + 6.14 + 10.08) / 8 = **6.53 FPS**

For Checkpoint 7 Phase 2:

**FPS Values:**

1. 5.8315659694043
2. 5.950290997591378
3. 5.8920097201503845
4. 5.633091454284762
5. 5.739079693656964
6. 5.913120051024754
7. 5.984764203736743
8. 9.327248669228638

**Average FPS:** (5.83 + 5.95 + 5.89 + 5.63 + 5.74 + 5.91 + 5.98 + 9.33) / 8 = **6.28 FPS**

For Checkpoint 8 Phase 2:

**FPS Values:**

1. 5.616524820949969
2. 5.853079888051752
3. 5.813472249304642
4. 5.600297114880359
5. 5.647382633388747
6. 5.848266549708234
7. 6.014528341070409
8. 8.449994622929852

**Average FPS:** (5.62 + 5.85 + 5.81 + 5.60 + 5.65 + 5.85 + 6.01 + 8.45) / 8 = **6.11 FPS**

for Checkpoint 9 Phase 2:

**FPS Values:**

1. 5.702869089392705
2. 5.942213459325294
3. 5.875527839759283
4. 5.635915392620756
5. 5.805883544351087
6. 5.92674756661424
7. 5.987661480941642
8. 9.098157293352333

**Average FPS:** (5.70 + 5.94 + 5.88 + 5.64 + 5.81 + 5.93 + 5.99 + 9.10) / 8 = **6.25 FPS**

For Checkpoint 10 Phase 2:

**FPS Values:**

1. 5.702646442577818
2. 5.947390420951641
3. 5.8974352623444375
4. 5.612384192733902
5. 5.825321938585874
6. 5.998152285580758
7. 6.308996765375555
8. 10.64050635633497

**Average FPS:** (5.70 + 5.95 + 5.90 + 5.61 + 5.83 + 6.00 + 6.31 + 10.64) / 8 = **6.49 FPS**

**Table 2: (IoU, Precision, AUC)**

**Phase 1**

lasot | AUC | OP50 | OP75 | Precision | Norm Precision |

my\_lasot\_2class\_config | 11.40 | 3.19 | 0.13 | 7.16 | 4.77 |

my\_lasot\_2class\_config | 37.65 | 43.94 | 7.49 | 40.11 | 40.26 |

my\_lasot\_2class\_config | 42.16 | 49.81 | 10.89 | 50.82 | 51.02 |

my\_lasot\_2class\_config | 35.55 | 44.15 | 11.75 | 45.81 | 45.67 |

my\_lasot\_2class\_config | 34.90 | 44.06 | 16.41 | 47.37 | 44.27 |

my\_lasot\_2class\_config | 31.36 | 39.68 | 13.82 | 38.68 | 38.75 |

my\_lasot\_2class\_config | 33.85 | 41.07 | 15.05 | 39.00 | 41.23 |

my\_lasot\_2class\_config | 30.79 | 39.05 | 17.67 | 33.80 | 35.50 |

my\_lasot\_2class\_config | 33.34 | 41.41 | 17.92 | 36.47 | 39.68 |

my\_lasot\_2class\_config | 22.19 | 27.27 | 10.36 | 25.96 | 27.68 |

**Phase 2**

my\_lasot\_2class\_config | 11.40 | 3.19 | 0.13 | 7.16 | 4.77 |

my\_lasot\_2class\_config | 37.65 | 43.94 | 7.49 | 40.11 | 40.26 |

my\_lasot\_2class\_config | 42.16 | 49.81 | 10.89 | 50.82 | 51.02 |

my\_lasot\_2class\_config | 35.55 | 44.15 | 11.75 | 45.81 | 45.67 |

my\_lasot\_2class\_config | 34.90 | 44.06 | 16.41 | 47.37 | 44.27 |

my\_lasot\_2class\_config | 31.36 | 39.68 | 13.82 | 38.68 | 38.75 |

my\_lasot\_2class\_config | 33.85 | 41.07 | 15.05 | 39.00 | 41.23 |

my\_lasot\_2class\_config | 30.79 | 39.05 | 17.67 | 33.80 | 35.50 |

my\_lasot\_2class\_config | 33.34 | 41.41 | 17.92 | 36.47 | 39.68 |

my\_lasot\_2class\_config | 22.19 | 27.27 | 10.36 | 25.96 | 27.68 |

Here we can see that Phase 1 and Phase 2 are identically the same .

A screen shot of a computer

AI-generated content may be incorrect.

this figure shows table 2 result from the terminal.

**Graphs**

This Figure shows Phase 1 graph.

This Figure shows Phase 2 graph.

Here we see that the two graphs from phase 1 and phase 2 are identically the same

**List of The Filenames and Lines**

**File: lib/test/evaluation/tracker.py**

**Modification 1 (Around line 76):** Passed self.run\_id to the get\_parameters function call.

* **Original:** params = self.get\_parameters()
* **Modified:** params = self.get\_parameters(self.run\_id)
* **Reason:** To forward the runid (epoch number) from the test.py command line to the parameter loading mechanism.

**Modification 2 (Around line 276):** Modified the get\_parameters function definition to accept the run\_id.

* **Original:** def get\_parameters(self):
* **Modified:** def get\_parameters(self, run\_id):
* **Reason:** To allow the function to receive the runid.

**Modification 3 (Around line 280):** Passed the received run\_id to the param\_module.parameters function call.

* **Original:** params = param\_module.parameters(self.parameter\_name)
* **Modified:** params = param\_module.parameters(self.parameter\_name, run\_id)
* **Reason:** To pass the runid further down to the specific parameter file (seqtrack.py) where the checkpoint path is determined.

**File: lib/test/parameter/seqtrack.py**

**Modification 1 (Around line 8):** Modified the parameters function definition to accept the run\_id.

* **Original:** def parameters(yaml\_name: str):
* **Modified:** def parameters(yaml\_name: str, run\_id: int):
* **Reason:** To allow the function to receive the runid passed from tracker.py.

**Modification 2 (around lines ~33-46):** Implemented logic to dynamically construct the checkpoint path using the run\_id instead of the hardcoded cfg.TEST.EPOCH.

* **Original Block:**

# Network checkpoint path

params.checkpoint = os.path.join(save\_dir, "checkpoints/train/seqtrack/%s/SEQTRACK\_ep%04d.pth.tar" % (yaml\_name, cfg.TEST.EPOCH))

* Modified Block:

if run\_id is not None:

epoch\_to\_load = run\_id

else:

epoch\_to\_load = cfg.TEST.EPOCH

output\_save\_dir = os.path.join(save\_dir, "output")

params.checkpoint = os.path.join(output\_save\_dir, "checkpoints/train/seqtrack/%s/SEQTRACK\_ep%04d.pth.tar" % (yaml\_name, epoch\_to\_load))

**Reason:** This is the core change that enables loading the specific epoch checkpoint (1 through 20) based on the --runid argument provided to test.py. It also corrects the path to include the output directory.

**File: tracking/analysis\_results.py**

**Modification 1 (Around line 13):** Changed the parameter\_name to match your config and set run\_ids to evaluate all 20 checkpoints.

* Original:

trackers.extend(trackerlist(name='seqtrack', parameter\_name='seqtrack\_b256', dataset\_name=dataset\_name, run\_ids=None, display\_name='seqtrack\_b256'))

* Modified:

trackers.extend(trackerlist(name='seqtrack', parameter\_name='my\_lasot\_2class\_config', dataset\_name=dataset\_name,

run\_ids=list(range(1, 21)), display\_name='my\_lasot\_2class\_config'))

**Reason:** This was necessary to instruct the script to find the 20 specific result folders associated with your my\_lasot\_2class\_config runs (e.g., ...\_001, ...\_002, etc.) and evaluate all of them.

**Modification 2 (Around line 20):** Changed the merge\_results argument from True to False.

* **Original:**

print\_results(trackers, dataset, dataset\_name, merge\_results=True, plot\_types=('success', 'prec', 'norm\_prec'),

force\_evaluation=True)

* Modified:

print\_results(trackers, dataset, dataset\_name, merge\_results=False, plot\_types=('success', 'prec', 'norm\_prec'),

force\_evaluation=True)

**Reason:** Setting merge\_results=False was the key to getting the final table. It tells the script to print a separate performance line for each individual checkpoint (epoch) rather than combining all 20 runs into a single average score. This provided the data needed for Table 2.

**We couldn’t download checkpoints from Hugging Face because some problem with Hugging Face so we had to test them all locally.**

**Our Reflection**

In this assignment, I learned that running inference and evaluation on a model like SeqTrack involves more than just executing a script. I immediately faced a ModuleNotFoundError for the 'lib' directory. This taught me the importance of properly setting the PYTHONPATH variable to ensure the project can find its own files, which is a key part of managing complex project environments. My next challenge was a 'not enough memory' error. At first, I was confused, but I realized it was a CPU RAM issue, not GPU, caused by setting --threads 8. By reducing the threads to --threads 2, I learned the practical lesson of balancing processing speed with hardware limitations, as the script was trying to load too many models into memory at once. The biggest lesson was in the code itself. To test all 20 epochs, I couldn't just use the default scripts. I had to modify lib/test/evaluation/tracker.py and lib/test/parameter/seqtrack.py to pass the runid (my epoch number) and dynamically build the correct checkpoint path. I also had to change analysis\_results.py to get the per-epoch table. This process showed me that a critical part is not just training the model, but having the ability to modify and adapt the surrounding code to properly run experiments and evaluate the results.