Hyperparameter tuning

For the hyperparameter tuning i used the train.py internal optuna tuner. With it i had a bit troubles fixing the gamma, the arguments where always overwritten, so i hardcoded gamma inside of the python file. I let all other parameters get optimized. Below are the tables with each 50 different hyperparameter configurations. Somehow often a low update frequency lead to the best results, but when using the normal training script using these high update frequencies it didnt learn at all. Therefore i decided to use the update frequency as a changing hyperparameter for task 2.

DDPG:

#	value	batch	buffer	lr	net	noise_std	tau	freq
0	-149.07	100	10000	0.00938	medium	0.84308	0.02	8
1	-991.75	100	1000000	0.04090	small	0.62751	0.001	16
2	-1048.55	128	1000000	1.59187e-05	medium	0.93268	0.005	1
3	-147.42	32	1000000	0.00113	medium	0.54894	0.005	1
4	-1388.46	32	10000	0.01382	medium	0.72439	0.08	16
5	-152.09	100	100000	0.01243	medium	0.89335	0.005	4
6	-152.34	128	1000000	5.35279e-05	big	0.98053	0.01	32
7	-145.94	2048	100000	0.00131	medium	0.11837	0.01	16
8	-145.55	64	1000000	0.00020	medium	0.85120	0.005	4
9	-155.57	2048	10000	0.00013	big	0.08706	0.001	8
10	-153.47	64	1000000	0.00060	medium	0.86259	0.001	16
11	-148.31	256	100000	0.00625	medium	0.15197	0.01	64
12	-1388.46	2048	100000	0.01669	medium	0.02364	0.08	16
13	-178.62	64	1000000	0.01916	small	0.96590	0.005	4
14	-144.97	2048	1000000	6.64202e-05	medium	0.25988	0.01	16
15	-149.52	256	1000000	4.80522e-05	medium	0.10672	0.01	32
16	-1103.60	2048	1000000	1.06601e-05	big	0.56606	0.01	1
17	-145.98	2048	1000000	3.54569e-05	medium	0.54488	0.005	16
18	-1044.39	64	1000000	1.96854e-05	medium	0.93222	0.02	4
19	-161.33	2048	1000000	0.00063	small	0.05293	0.001	16
20	-147.23	64	1000000	0.00017	big	0.56401	0.05	64
21	-885.30	2048	1000000	1.50524e-05	medium	0.16146	0.01	512
22	-145.56	16	100000	0.00030	medium	0.11932	0.01	16
23	-149.01	16	100000	0.00250	medium	0.28022	0.01	128
24	-148.17	512	100000	4.62270e-05	big	0.10201	0.01	512
25	-1663.22	64	100000	2.26828e-05	small	0.98099	0.005	4
26	-708.34	2048	100000	2.33080e-05	medium	0.10679	0.02	8
27	-147.00	32	10000	0.00030	medium	0.20164	0.01	16
28	-145.91	64	1000000	0.00035	medium	0.16233	0.02	16
29	-283.88	16	100000	0.00236	medium	0.07124	0.001	16
30	-149.98	128	100000	5.11322e-05	medium	0.14509	0.01	16
31	-149.45	100	1000000	0.00153	medium	0.13841	0.02	16
32	-144.32	64	1000000	0.00023	medium	0.21531	0.02	256
33	-148.67	128	1000000	6.08258e-05	medium	0.04498	0.02	256
34	-148.10	16	1000000	0.00122	medium	0.83860	0.005	4
35	-145.21	64	1000000	0.00042	medium	0.39074	0.005	256
36	-145.25	64	1000000	0.00020	big	0.38317	0.02	128
37	-832.40	64	10000	4.81671e-05	medium	0.42022	0.005	256
38	-145.75	64	1000000	0.00086	big	0.48914	0.02	128
39	-146.81	1024	1000000	0.00029	medium	0.42362	0.005	32
40	-904.72	64	1000000	2.81551e-05	medium	0.27343	0.08	256
41	-154.61	64	1000000	0.00253	small	0.08582	0.005	256
42	-764.91	32	10000	4.52014e-05	medium	0.81281	0.005	128
43	-397.65	100	100000	5.61820e-05	big	0.40044	0.02	256

#	value	batch	buffer	lr	net	noise_std	tau	freq
44	-147.79	2048	1000000	4.20083e-05	medium	0.12080	0.01	16
45	-145.18	64	10000	0.00030	big	0.93422	0.005	4
46	-150.01	64	10000	0.00352	big	0.98097	0.005	4
47	-145.25	64	1000000	0.00077	medium	0.42888	0.02	256
48	-149.58	2048	1000000	0.00151	medium	0.60592	0.02	256
49	-146.28	32	10000	0.00017	big	0.74653	0.01	4

TD3:

#	value	batch	buffer	lr	arch	tau	freq
0	-170.120	64	100000	0.001878	med	0.02	4
1	-170.859	32	1000000	0.002625	med	0.02	128
2	-170.360	64	10000	0.002672	big	0.08	128
3	-185.345	128	10000	0.001514	med	0.08	1
4	-192.407	128	10000	0.000144	med	0.08	128
5	-171.968	2048	10000	0.001408	med	0.08	64
6	-192.055	128	10000	0.000278	med	0.05	16
7	-169.409	256	100000	0.002469	med	0.05	64
8	-169.663	512	10000	0.000678	small	0.005	128
9	-1366.389	128	10000	0.221319	med	0.001	8
10	-1154.009	1024	100000	0.000013	med	0.05	64
11	-176.221	512	10000	0.000244	small	0.005	64
12	-168.527	64	100000	0.000653	med	0.05	64
13	-378.299	64	100000	0.000079	med	0.05	64
14	-1366.389	256	100000	0.037983	med	0.08	8
15	-168.197	64	1000000	0.003914	med	0.05	512
16	-168.123	128	1000000	0.004700	med	0.05	256
17	-170.182	128	1000000	0.001578	big	0.05	256
18	-1366.389	32	1000000	0.019616	med	0.05	512
19	-1498.504	16	1000000	0.007883	med	0.01	256
20	-170.925	64	1000000	0.003275	med	0.08	512
21	-173.708	16	100000	0.000276	med	0.05	32
22	-169.252	64	1000000	0.000243	med	0.02	64
23	-168.150	64	10000	0.002066	med	0.05	256
24	-1366.389	64	10000	0.013388	big	0.05	256
25	-170.658	32	10000	0.000241	med	0.05	256
26	-1498.504	64	10000	0.013306	med	0.05	4
27	-170.845	1024	1000000	0.011211	small	0.05	256
28	-1498.504	128	100000	0.028433	med	0.005	256
29	-171.030	1024	1000000	0.000664	med	0.05	512
30	-167.129	64	1000000	0.001113	small	0.08	256
31	-170.266	64	1000000	0.001142	big	0.08	512
32	-442.073	64	1000000	0.000135	small	0.08	256
33	-188.192	128	1000000	0.000577	med	0.005	1
34	-170.077	128	1000000	0.003390	med	0.05	16
35	-1366.389	128	1000000	0.042838	med	0.05	256
36	-171.219	100	1000000	0.000448	med	0.05	8
37	-173.003	64	10000	0.001288	med	0.005	256
38	-1498.504	64	1000000	0.065485	small	0.08	32
39	-170.157	512	1000000	0.000838	med	0.08	256
40	-1366.389	64	1000000	0.010813	med	0.005	512
41	-173.975	64	1000000	0.008827	med	0.05	256
42	-169.295	64	100000	0.001614	med	0.05	512
43	-168.300	1024	100000	0.002654	med	0.08	64
44	-170.221	1024	100000	0.000738	small	0.08	64

#	value	batch	buffer	lr	arch	tau	freq
45	-170.193	256	1000000	0.003953	small	0.08	256
46	-171.354	2048	100000	0.000519	med	0.08	64
47	-167.360	1024	100000	0.002683	med	0.01	128
48	-169.632	512	100000	0.000856	med	0.01	128
49	-1498.504	1024	100000	0.082599	med	0.01	128

zac:

#	value	batch	buffer	lr	arch	tau	freq
0		128	1000000	0.742	med	0.08	4
1	-1170.968	16	10000	0.000031	med	0.005	4
2	-199.828	2048	100000	0.015	big	0.05	512
3		128	10000	0.304	small	0.01	256
4	-206.143	1024	1000000	0.001	med	0.08	512
5		64	100000	0.162	big	0.05	64
6	-205.857	128	10000	0.000342	big	0.005	4
7	-720.322	1024	10000	0.00002	big	0.08	256
8	-209.381	64	1000000	0.001	med	0.01	1
9	-210.097	32	1000000	0.022	big	0.08	32
10	-196.526	256	100000	0.002	big	0.05	512
11	-211.748	16	100000	0.001	big	0.05	512
12		2048	100000	0.917	big	0.05	32
13	-198.721	256	100000	0.021	med	0.05	512
14	-205.859	256	100000	0.040	med	0.005	512
15	-209.282	256	100000	0.00048	med	0.05	256
16	-949.106	256	10000	0.000018	big	0.05	512
17	-1351.721	256	100000	0.023	med	0.01	128
18		256	100000	0.318	med	0.05	128
19	-203.517	512	10000	0.002	med	0.05	512
20	-201.847	256	10000	0.003	big	0.05	8
21	-1471.765	256	1000000	0.027	big	0.01	512
22	-200.449	256	100000	0.005	small	0.05	512
23	-213.592	128	100000	0.037	med	0.05	512
24	-195.845	2048	100000	0.001	big	0.005	512
25	-197.281	2048	100000	0.001	big	0.005	256
26	-197.943	2048	100000	0.002	big	0.005	1
27	-203.756	128	10000	0.001	big	0.005	256
28	-203.392	256	100000	0.000164	big	0.005	32
29	-206.195	128	100000	0.000222	med	0.005	512
30	-1250.295	2048	100000	0.000052	big	0.05	256
31	-209.829	2048	10000	0.001	big	0.005	512
32	-200.101	16	100000	0.002	big	0.005	128
33	-197.487	2048	100000	0.001	big	0.08	512
34	-199.714	2048	100000	0.001	small	0.01	256
35	-226.942	64	100000	0.000142	big	0.001	512
36	-210.248	2048	100000	0.002	small	0.08	512
37	-197.948	2048	100000	0.000158	big	0.08	8
38	-203.009	32	100000	0.006	big	0.08	16
39	-210.316	64	100000	0.002	small	0.005	512
40	-205.044	256	100000	0.000278	big	0.05	16
41	-194.788	2048	100000	0.008	big	0.005	256
42	-196.498	2048	100000	0.003	big	0.08	16
43	-197.321	2048	100000	0.001	big	0.005	64
44	-203.455	64	1000000	0.003	big	0.08	16
45	-199.028	2048	100000	0.002	big	0.02	16

#	value	batch	buffer	lr	arch	tau	freq
46	-195.788	2048	100000	0.016	big	0.08	256
47	-211.911	16	100000	0.009	big	0.08	256
48	-202.569	512	100000	0.018	med	0.08	16
49		1024	100000	0.376	big	0.08	256

Task 2

I experimented with changing both the update frequency and the learning rate. For plotting this part i wrote a custom script, since i couldnt find my way with the plot_train.py script. Somehow i was not able to find detailed documentation and therefore failed to set the timestamp range.

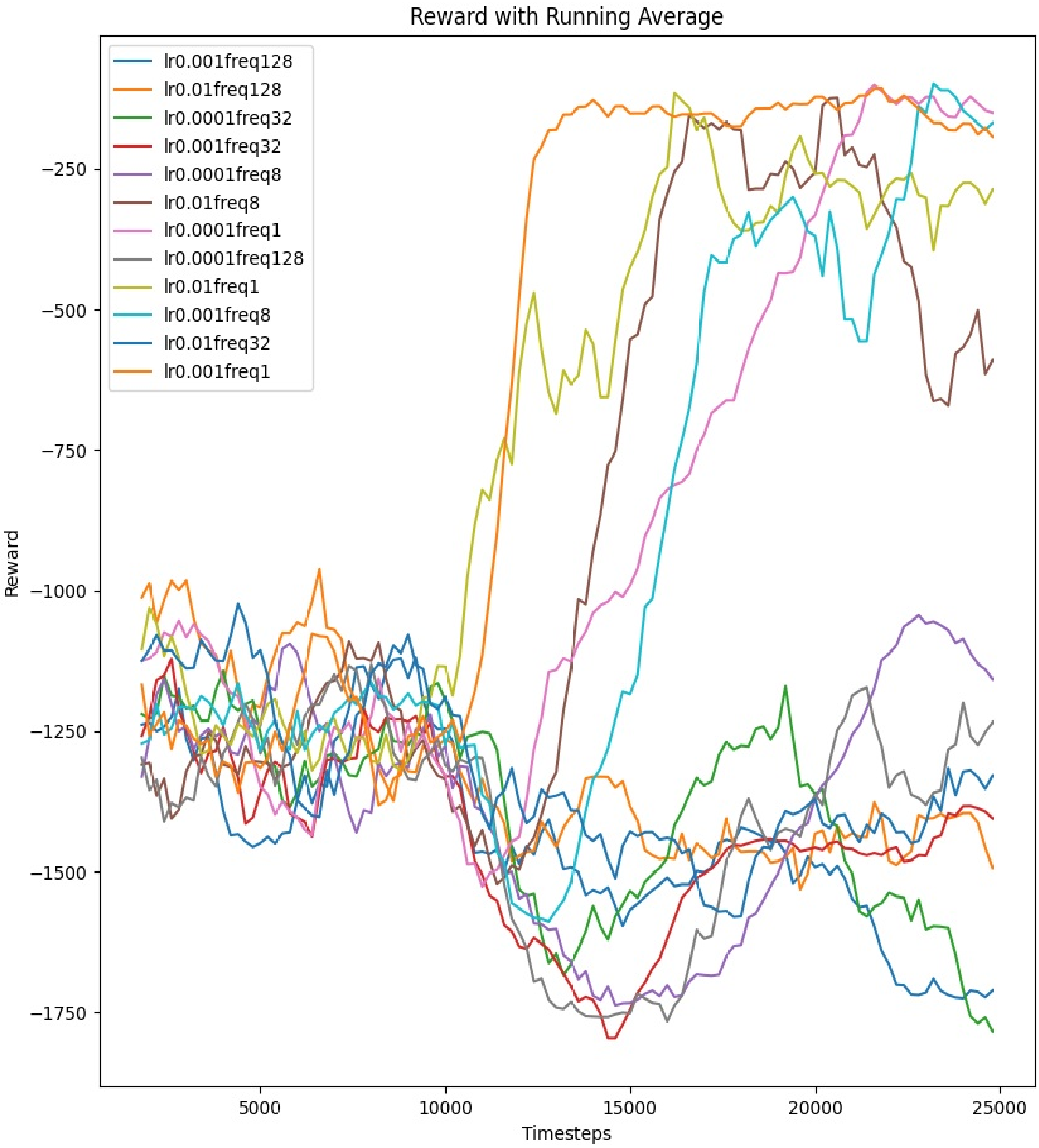
I let the n_timestamps on all three experiments at 25000, even though td3 and sac with good parameters learned way faster. I did that to also catch the slower learning settings. This shows, that in sac the good configurations learn super fast, but the worse configurations are about as slow as they are in td3. While in td3 the difference between good and bad configurations is lower, but overall slower.

Since higher learning rates looked generally better did another run with 0.1 lr, but there already no one learned anything.

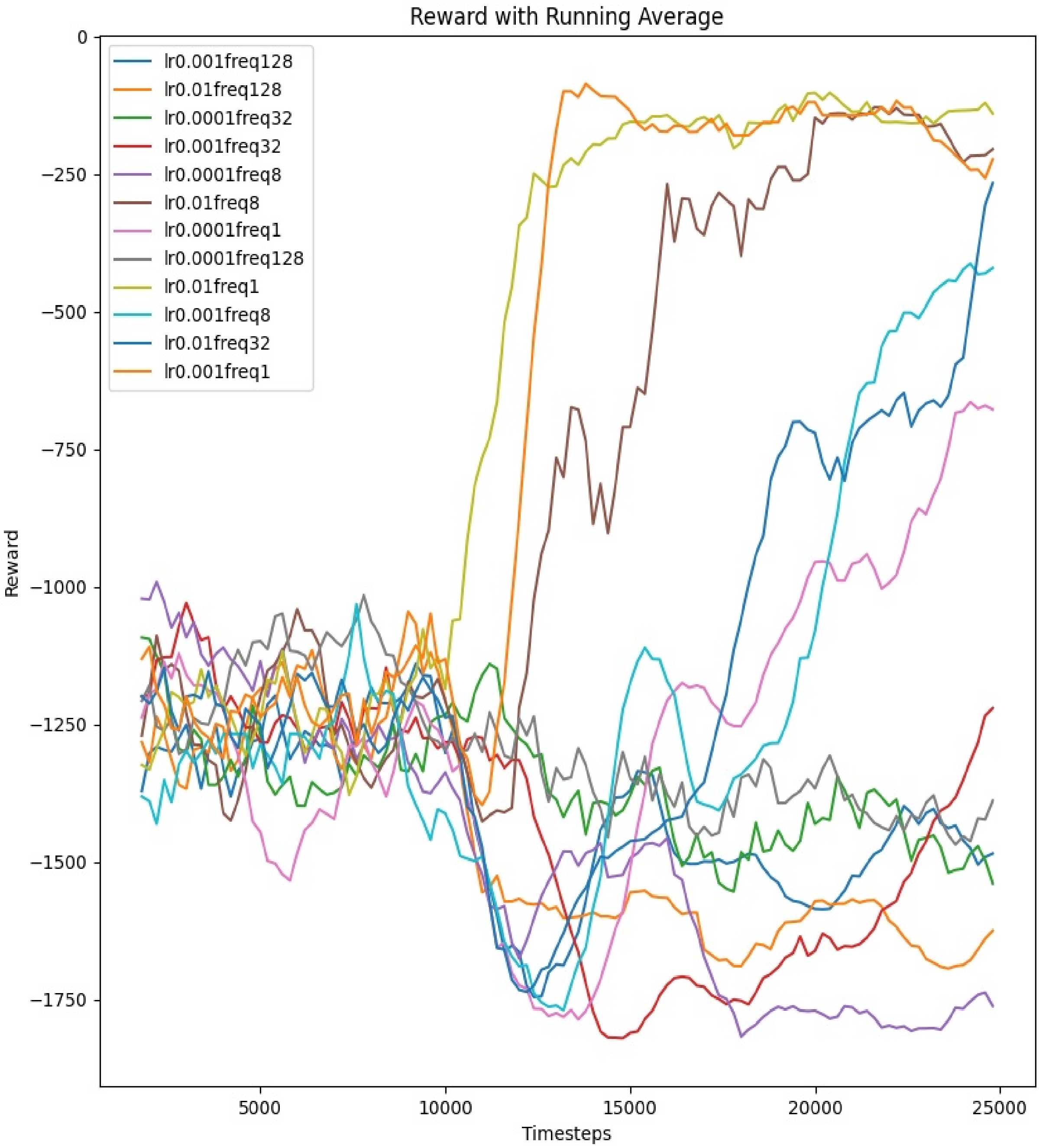
LR from 0.01 to 0.0001

The plots also have a running average running, to make the results a bit more readable.

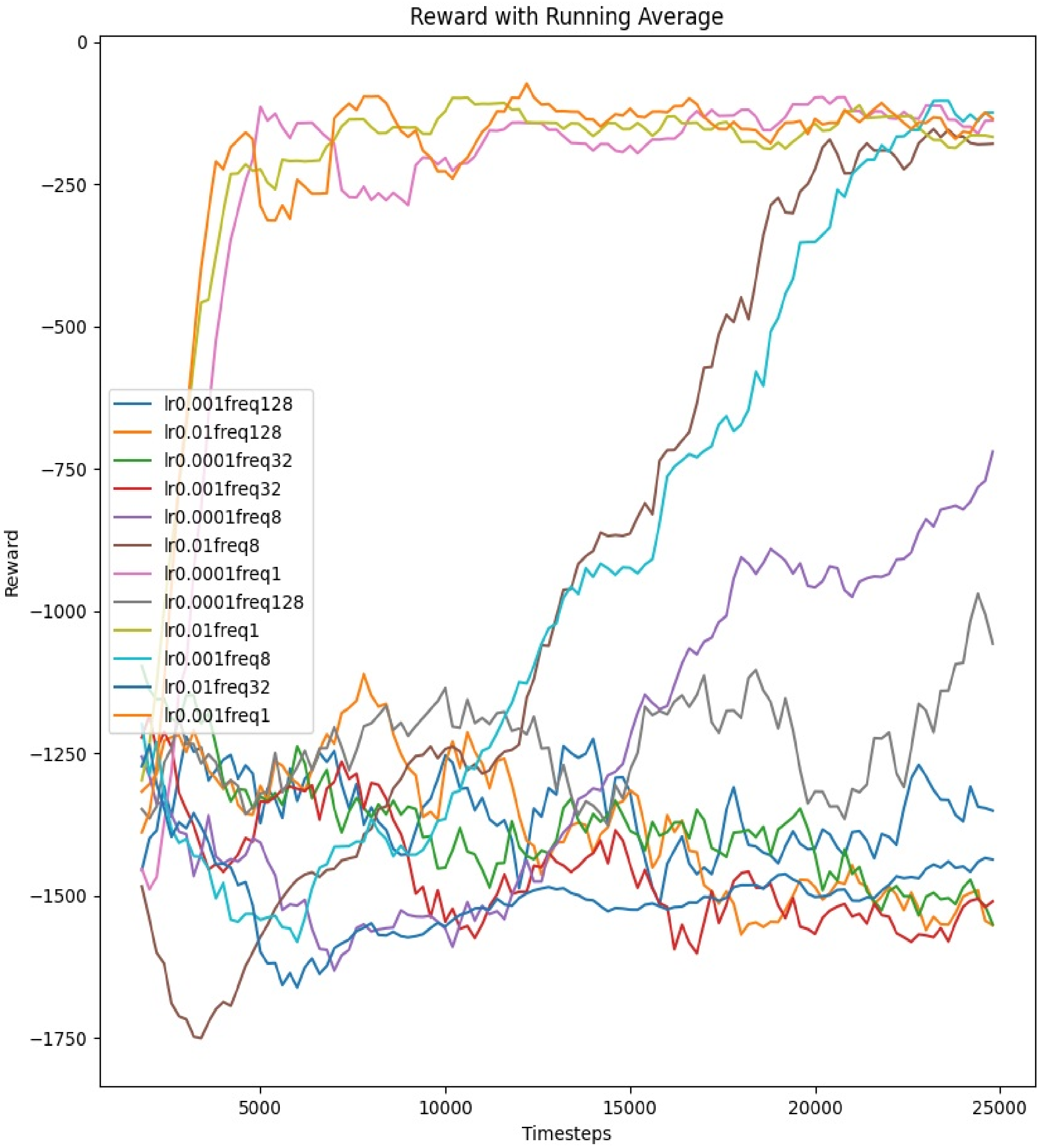
DDPG


Reward with Running Average
lr0.001freq128
lr0.01freq128
lr0.0001freq32
lr0.001freq32
lr0.0001freq8
lr0.01freq8
lr0.0001freq1
lr0.0001freq128
lr0.01freq1
lr0.001freq8
lr0.01freq32
lr0.001freq1
−250
−500
−750
−1000
−1250
−1500
−1750
Reward
5000
10000
15000
20000
25000
Timesteps

TD3


Reward with Running Average
lr0.001freq128
lr0.01freq128
lr0.0001freq32
lr0.001freq32
lr0.0001freq8
lr0.01freq8
lr0.0001freq1
lr0.0001freq128
lr0.01freq1
lr0.001freq8
lr0.01freq32
lr0.001freq1
0
−250
−500
−750
−1000
−1250
−1500
−1750
Reward
5000
10000
15000
20000
25000
Timesteps

SAC



Task 3

DDPG

It seems to generally learn slower, but there are 5 variations that do almost as well as all the others. Therefore it seems to handle worse hyperparameters quite well.

TD3

There are more configurations that learn, with a total of 6, but the worse configurations where noticable slower than good ones.

Zac

The good configurations learned really fast, but compared to the fast learning speed, the worse configurations learned significantly slower. Still there where a total of 6 configurations that learned.

Comparison:

Sac seems to learn the fastest with good parameters, with worse parameters it does similar to TD3. DDPG is the slowest, but with relatively similar speed between good and bad configurations. Therefore my ranking would be:

1. SAC
2. TD3
3. DDPG

Appendix

Hyperparameter tuning commands:

```
python train.py --algo ddpg --env Pendulum-v1 -n 25000 -optimize --optimization-log-path logs-opt
python train.py --algo td3 --env Pendulum-v1 -n 25000 -optimize --optimization-log-path logs-opt-
python train.py --algo sac --env Pendulum-v1 -n 25000 -optimize --optimization-log-path logs-opt-
```

Training with variations script:

```
#!/bin/bash

learning_rates=(0.1 1 10)
train_freq=(1 8 32 128)

for lr in "${learning_rates[@]}"; do
  for freq in "${train_freq[@]}"; do
    python train.py --algo ddpg --env Pendulum-v1 -n 25000 --eval-freq 1000 --save-freq 50 --log-
    mv ./ddpg_train/ddpg/Pendulum-v1_1 ./ddpg_train/ddpg/lr${lr}freq${freq}
    python train.py --algo td3 --env Pendulum-v1 -n 25000 --eval-freq 1000 --save-freq 50 --log-f
    mv ./td3_train/td3/Pendulum-v1_1 ./td3_train/td3/lr${lr}freq${freq}
    python train.py --algo sac --env Pendulum-v1 -n 25000 --eval-freq 1000 --save-freq 50 --log-f
    mv ./sac_train/sac/Pendulum-v1_1 ./sac_train/sac/lr${lr}freq${freq}
  done
done
```

Plot script

```
import pandas as pd
import matplotlib.pyplot as plt
import glob
import os


def plot_monitor_files(directory, window_size=10):
    # Find all 0.monitor.csv files in the directory
    file_paths = glob.glob(os.path.join(directory, "**", "0.monitor.csv"), recursive=True)

    plt.figure(figsize=(10, 6))

    for file_path in file_paths:
        df = pd.read_csv(file_path, skiprows=1)
        df["r"] = pd.to_numeric(df["r"], errors="coerce")  # Convert 'r' column to numeric
        df["r_avg"] = df["r"].rolling(window=window_size).mean()
        plt.plot(df.index * 200, df["r_avg"], label=os.path.basename(os.path.dirname(file_path)))

    plt.xlabel("Timesteps")
    plt.ylabel("Reward")
    plt.title("Reward with Running Average")
    plt.legend()
```

```
    plt.show()


# Example usage
directory = "/home/fabian/github/advancedML/Exercise2/rl-baselines3-zoo/ddpg_train"
plot_monitor_files(directory)
directory = "/home/fabian/github/advancedML/Exercise2/rl-baselines3-zoo/td3_train"
plot_monitor_files(directory)
directory = "/home/fabian/github/advancedML/Exercise2/rl-baselines3-zoo/sac_train"
plot_monitor_files(directory)
```