Отчет по ДЗ №1
Исполнительница: Смолкина Ю.А
Предмет: RL

_____
Практическая часть. Ссылка на гитхаб. Теоретическая часть выполнена отдельно.

Run Behavior Cloning (Problem 1)

```
################
# RL TRAINER
################

self.rl_trainer = RL_Trainer(self.params) # TODO: look in here and implement this

######################
# LOAD EXPERT POLICY
######################

print('Loading expert policy from...', self.params['expert_policy_file'])
self.loaded_expert_policy = LoadedGaussianPolicy(self.params['expert_policy_file'])
print('Done restoring expert policy...')
```

BC.1

Получение результатов в двух средах

Ant-v2

```
********** Iteration 0 ************

Training agent using sampled data from replay buffer...

Beginning logging procedure...

Collecting data for eval...
Eval_AverageReturn : 4653.3837890625
Eval_StdReturn : 39.93415451049805
Eval_MaxReturn : 4717.9111328125
Eval_MinReturn : 4594.71533203125
Eval_AverageEpLen : 1000.0
Train_AverageReturn : 4713.6533203125
Train_StdReturn : 12.196533203125
Train_MaxReturn : 4725.849609375
Train_MinReturn : 4701.45654296875
Train_AverageEpLen : 1000.0
Train_EnvstepsSoFar : 0
TimeSinceStart : 12.90553092956543
Training Loss : 0.0019285776652395725
Initial_DataCollection_AverageReturn : 4713.6533203125
Done logging...
```

HalfCheetah

```
********** Iteration 0 ************

Training agent using sampled data from replay buffer...

Beginning logging procedure...

Collecting data for eval...
Eval_AverageReturn : 3952.64990234375
Eval_StdReturn : 84.59920501708984
Eval_MaxReturn : 4082.892333984375
Eval_MinReturn : 3832.34326171875
Eval_AverageEpLen : 1000.0
Train_AverageReturn : 4205.7783203125
Train_StdReturn : 83.038818359375
Train_MaxReturn : 4288.81689453125
Train_MinReturn : 4122.7392578125
Train_AverageEpLen : 1000.0
Train_EnvstepsSoFar : 0
TimeSinceStart : 7.664066791534424
Training Loss : 0.0057833632454276085
Initial_DataCollection_AverageReturn : 4205.7783203125
Done logging...
```

Running DAgger (Problem 2)

Modify the settings above:

1. check the do_dagger box
2. set n_iters to 10 and then rerun the code.

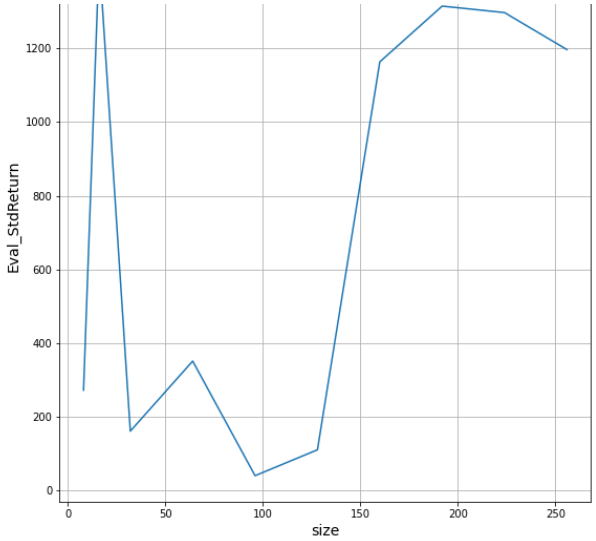
```
Streaming output truncated to the last 5000 lines.
Beginning logging procedure...

Collecting data for eval...
Eval_AverageReturn : 4968.28662109375
Eval_StdReturn : 0.0
Eval_MaxReturn : 4968.28662109375
Eval_MinReturn : 4968.28662109375
Eval_AverageEpLen : 1000.0
Train_AverageReturn : 4738.8544921875
Train_StdReturn : 0.0
Train_MaxReturn : 4738.8544921875
Train_MinReturn : 4738.8544921875
Train_AverageEpLen : 1000.0
Train_EnvstepsSoFar : 838402
TimeSinceStart : 13193.768167495728
Training Loss : 0.00014113822544459254
Initial_DataCollection_AverageReturn : 4713.6533203125
Done logging...


********** Iteration 999 ************

Collecting data to be used for training...

Relabelling collected observations with labels from an expert policy...

Training agent using sampled data from replay buffer...

Beginning logging procedure...

Collecting data for eval...
Eval_AverageReturn : 4080.0009765625
Eval_StdReturn : 0.0
Eval_MaxReturn : 4080.0009765625
Eval_MinReturn : 4080.0009765625
Eval_AverageEpLen : 1000.0
Train_AverageReturn : 4170.3046875
Train_StdReturn : 0.0
Train_MaxReturn : 4170.3046875
Train_MinReturn : 4170.3046875
Train_AverageEpLen : 1000.0
Train_EnvstepsSoFar : 999000
TimeSinceStart : 16233.86015510559
Training Loss : 0.0005803754320368171
Initial_DataCollection_AverageReturn : 4205.7783203125
Done logging...
```

После перебора параметров

```
********** Iteration 0 ************

Training agent using sampled data from replay buffer...

Beginning logging procedure...

Collecting data for eval...
Eval_AverageReturn : 4540.22509765625
Eval_StdReturn : 658.97314453125
Eval_MaxReturn : 4839.306640625
Eval_MinReturn : 805.0270385742188
Eval_AverageEpLen : 1000.0
Train_AverageReturn : 4713.6533203125
Train_StdReturn : 12.196533203125
Train_MaxReturn : 4725.849609375
Train_MinReturn : 4701.45654296875
Train_AverageEpLen : 1000.0
Train_EnvstepsSoFar : 0
TimeSinceStart : 82.07229471206665
Training Loss : 0.001628559548407793
Initial_DataCollection_AverageReturn : 4713.6533203125
Done logging...
```

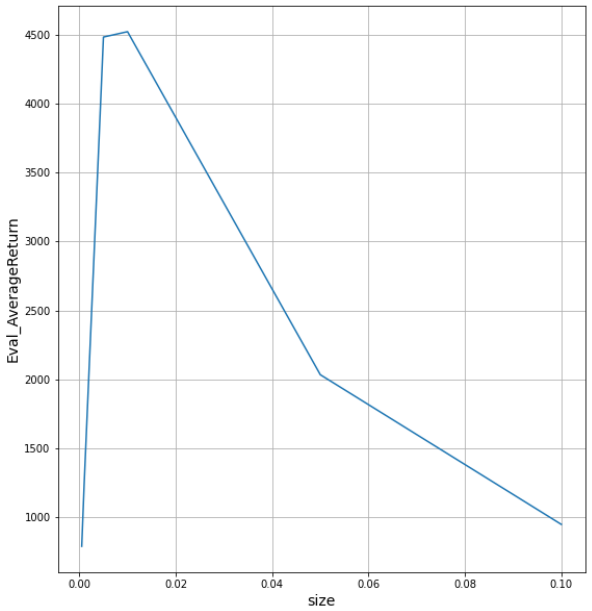
Размер сетки sizes = [8, 16, 32, 64, 96, 128, 160, 192, 224, 256]
Eval_AverageReturn = [4328.10, 3748.15, 4596.28, 4482.98, 4653.38, 4565.41, 4217.86, 3824.80, 4051.50, 3922.78]
Eval_StdReturn = [272.07, 1449.28, 160.85, 351.10, 39.93, 110.17, 1163.59, 1315.03, 1297.32, 1196.88]

Пример



Аналогично с train_batch_size

learning_rate sizes = [1e-1, 5e-2, 1e-2, 5e-3, 1e-3, 5e-4]



Проделаем тоже самое с HalfCheetah-v2

Отчет по ДЗ №2
Исполнительница: Смолкина Ю.А
Предмет: RL

_____
Практическая часть. Ссылка на гитхаб. Картинки можно найти отдельно тут.Теоретическая часть выполнена отдельно.

Run Policy Gradients

```
####################
## SET AGENT PARAMS
####################

computation_graph_args = {
    'n_layers': params['n_layers'],
    'size': params['size'],
    'learning_rate': params['learning_rate'],
    }

estimate_advantage_args = {
    'gamma': params['discount'],
    'standardize_advantages': not(params['dont_standardize_advantages']),
    'reward_to_go': params['reward_to_go'],
    'nn_baseline': params['nn_baseline'],
    'gae_lambda': params['gae_lambda'],
}

train_args = {
    'num_agent_train_steps_per_iter': params['num_agent_train_steps_per_iter'],
}

agent_params = {**computation_graph_args, **estimate_advantage_args, **train_args}

self.params = params
self.params['agent_class'] = PGAgent
self.params['agent_params'] = agent_params
self.params['batch_size_initial'] = self.params['batch_size']

################
## RL TRAINER
################

self.rl_trainer = RL_Trainer(self.params)
```

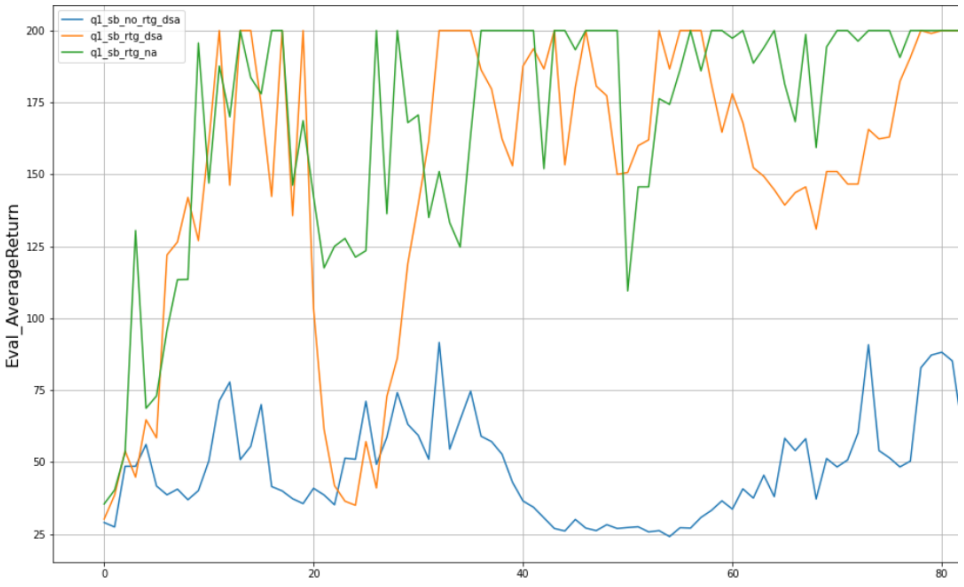
Experiment 1: CartPole
Сначала проводим обучение

```
********** Iteration 99 ************

Collecting data to be used for training...

Training agent using sampled data from replay buffer...

Beginning logging procedure...

Collecting data for eval...
Eval_AverageReturn : 135.6666717529297
Eval_StdReturn : 6.236095905303955
Eval_MaxReturn : 144.0
Eval_MinReturn : 129.0
Eval_AverageEpLen : 135.66666666666666
Train_AverageReturn : 131.0
Train_StdReturn : 10.794110298156738
Train_MaxReturn : 155.0
Train_MinReturn : 107.0
Train_AverageEpLen : 131.0
Train_EnvstepsSoFar : 504317
TimeSinceStart : 627.4800820350647
Training Loss : 36445.78125
Initial_DataCollection_AverageReturn : 25.65816307067871
Done logging...
```

```
********** Iteration 99 ************

Collecting data to be used for training...

Training agent using sampled data from replay buffer...

Beginning logging procedure...

Collecting data for eval...
Eval_AverageReturn : 108.5
Eval_StdReturn : 3.2015621662139893
Eval_MaxReturn : 113.0
Eval_MinReturn : 104.0
Eval_AverageEpLen : 108.5
Train_AverageReturn : 101.80000305175781
Train_StdReturn : 12.180312156677246
Train_MaxReturn : 115.0
Train_MinReturn : 25.0
Train_AverageEpLen : 101.8
Train_EnvstepsSoFar : 504630
TimeSinceStart : 710.610454082489
Training Loss : -158.98486328125
Initial_DataCollection_AverageReturn : 25.65816307067871
Done logging...
```

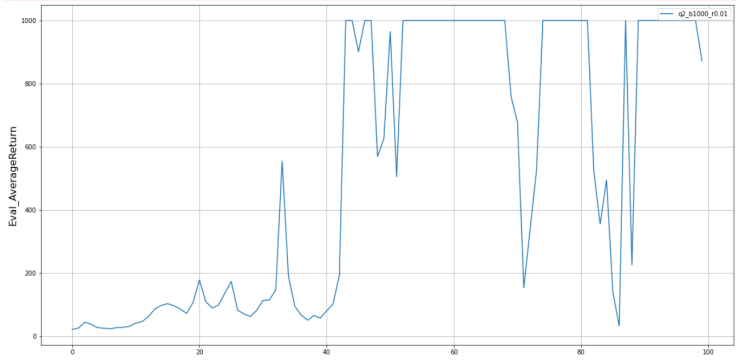
Результаты

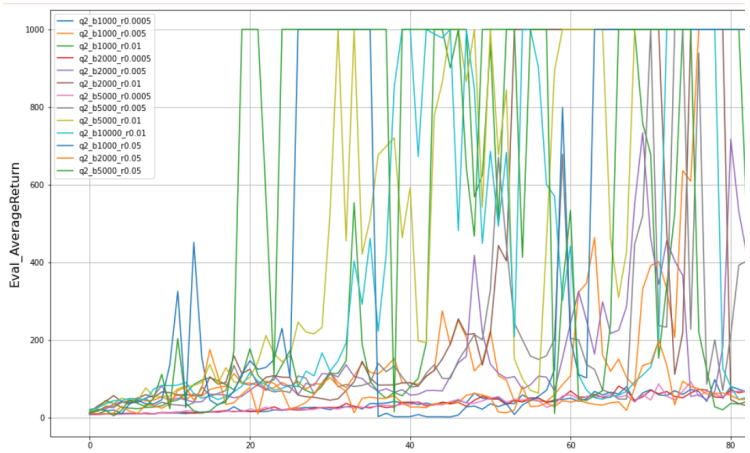
Ниже будет приведен пример кривых обучения средней отдачи на каждой итерации для экспериментов с маленьким размером пакета (batch)



- Какой из вариантов оценки отдачи имеет лучшие результаты без нормализации значения преимущества? Примерно (-b 5000 -rtg -dsa)
- Помогает ли нормализация значения преимущества? Невсегда тк разница заметна Только при очень большом пакете.
- Как влияет размер пакета на качество обучения? Изменение размера пакета сонаправлено с изменением скорости сходимости

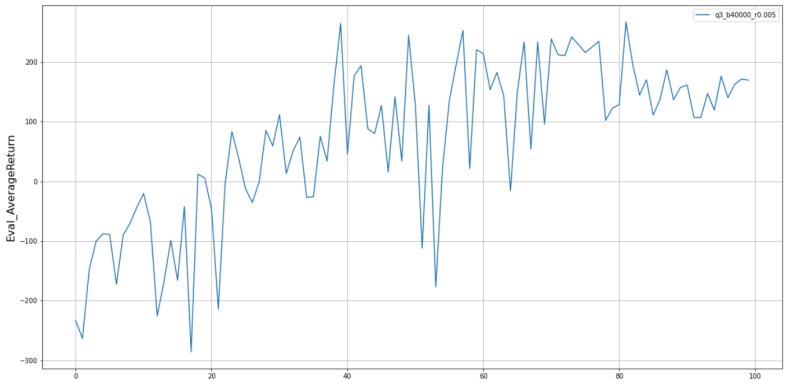
Experiment 2: InvertedPendulum.





лучший результат получился у b = 1000, r = 0.01
Большинство картинок содержится в отдельной папке.

Эксперимент 3: LunarLander



Также было проведено Эксперимент 4: HalfCheetah и Обучение с гипер параметрами, Эксперимент 5: Hopper