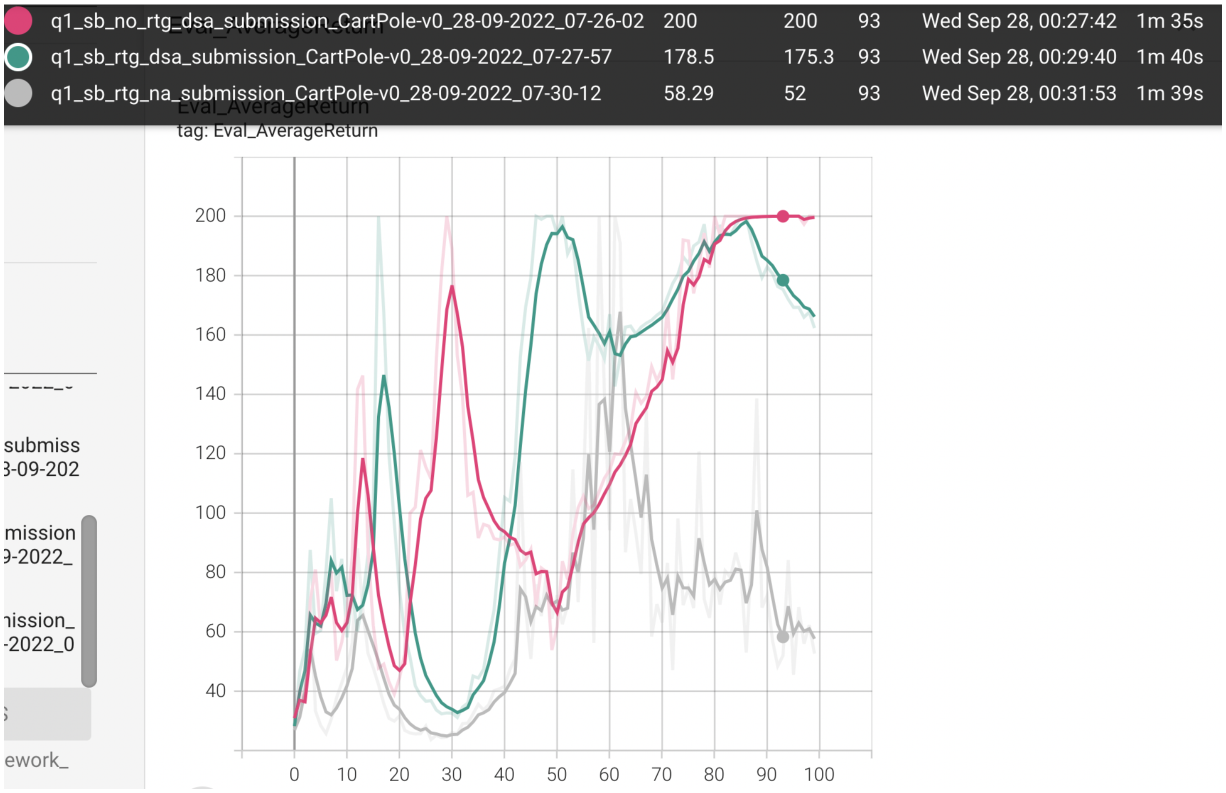
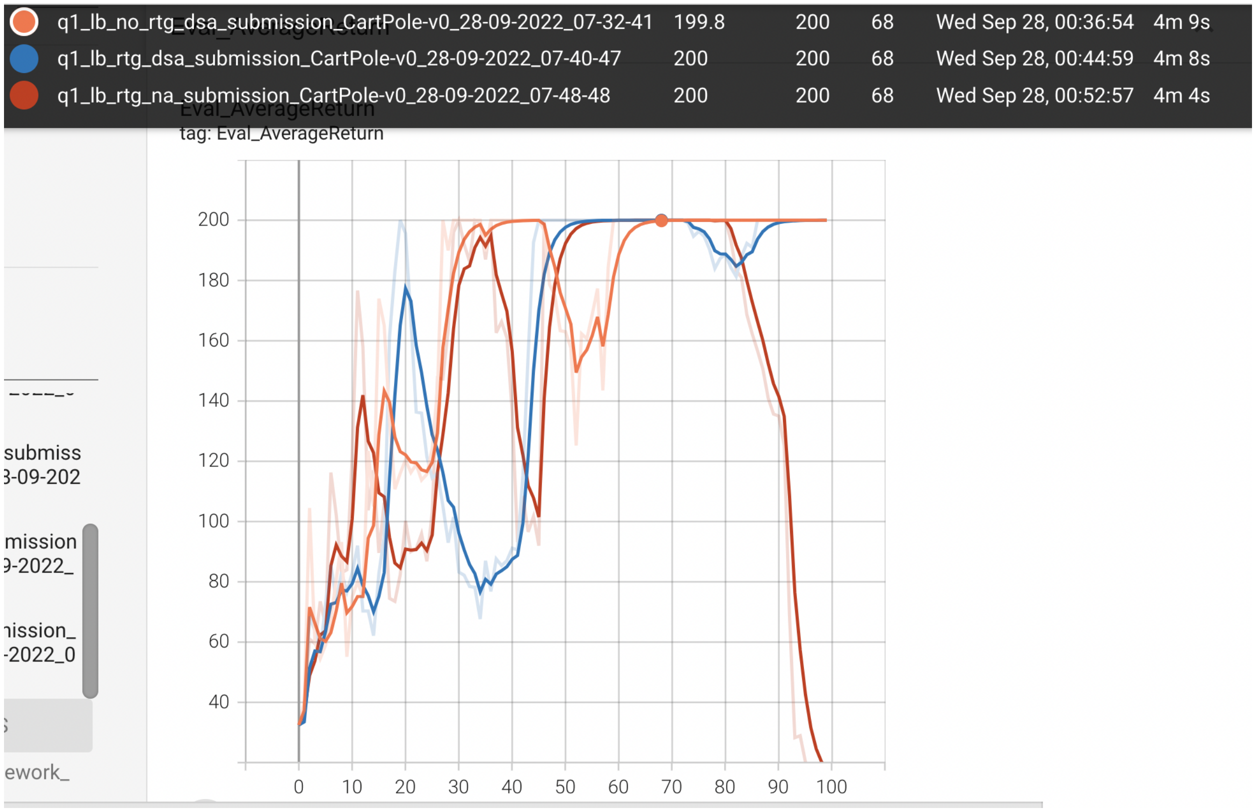
Experiment 1



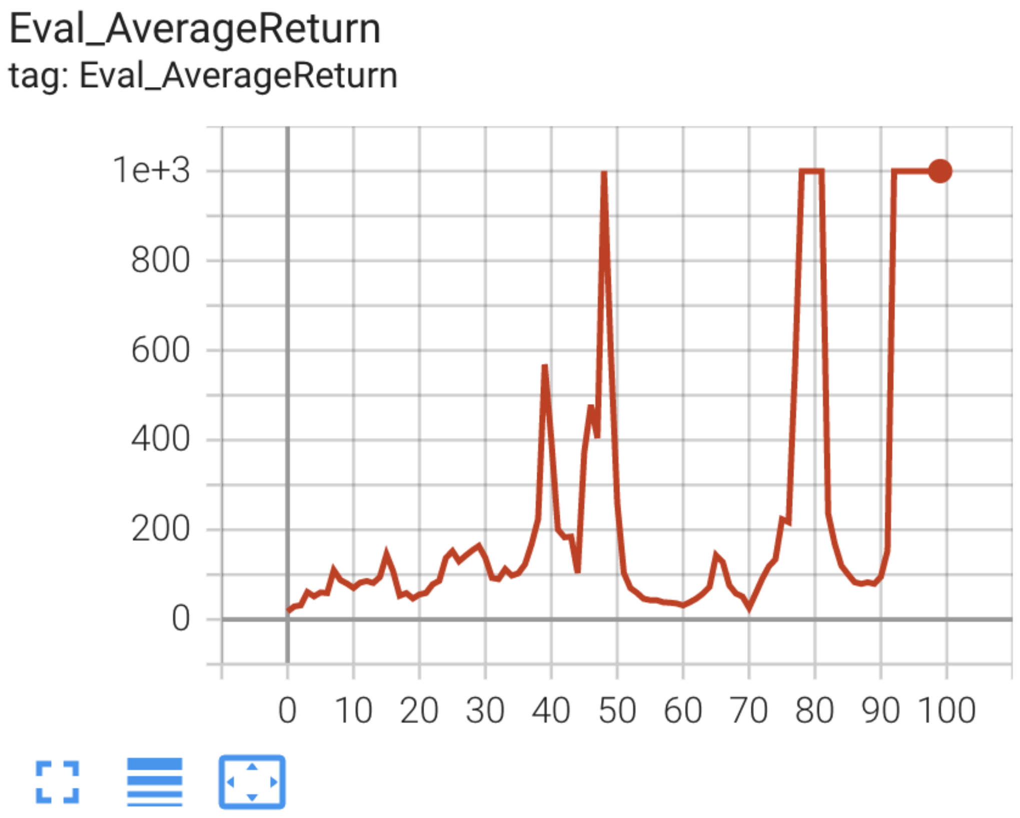
●	q1_sb_no_rtg_dsa_submission_CartPole-v0_28-09-2022_07-26-02	200	200	93	Wed Sep 28, 00:27:42	1m 35s
●	q1_sb_rtg_dsa_submission_CartPole-v0_28-09-2022_07-27-57	178.5	175.3	93	Wed Sep 28, 00:29:40	1m 40s
●	q1_sb_rtg_na_submission_CartPole-v0_28-09-2022_07-30-12	58.29	52	93	Wed Sep 28, 00:31:53	1m 39s



●	q1_lb_no_rtg_dsa_submission_CartPole-v0_28-09-2022_07-32-41	199.8	200	68	Wed Sep 28, 00:36:54	4m 9s
●	q1_lb_rtg_dsa_submission_CartPole-v0_28-09-2022_07-40-47	200	200	68	Wed Sep 28, 00:44:59	4m 8s
●	q1_lb_rtg_na_submission_CartPole-v0_28-09-2022_07-48-48	200	200	68	Wed Sep 28, 00:52:57	4m 4s

Experiment 2

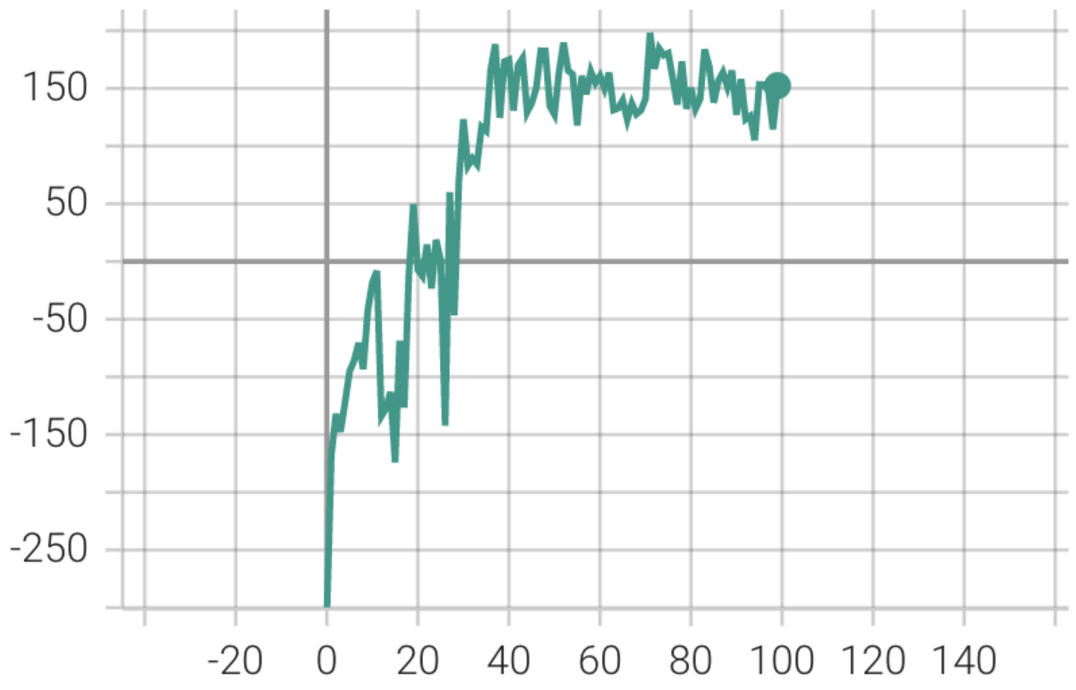
for best performance:
b = 500
r = 1e-2



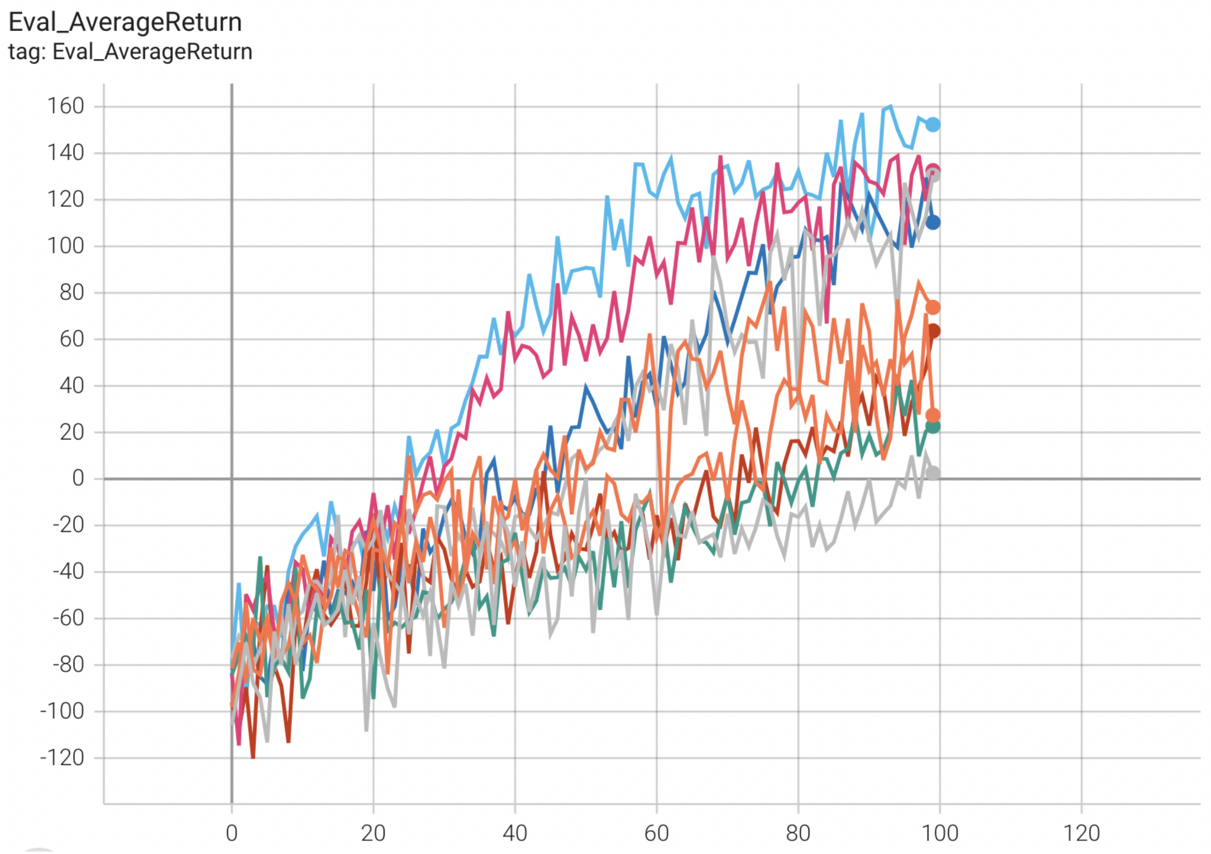
Commands see “submission_commands.txt”

Experiment 3

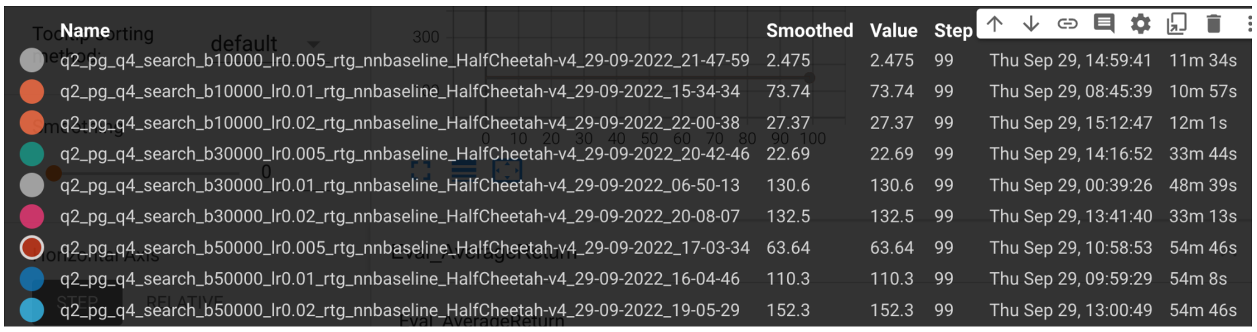
Eval_AverageReturn
tag: Eval_AverageReturn



Experiment 4

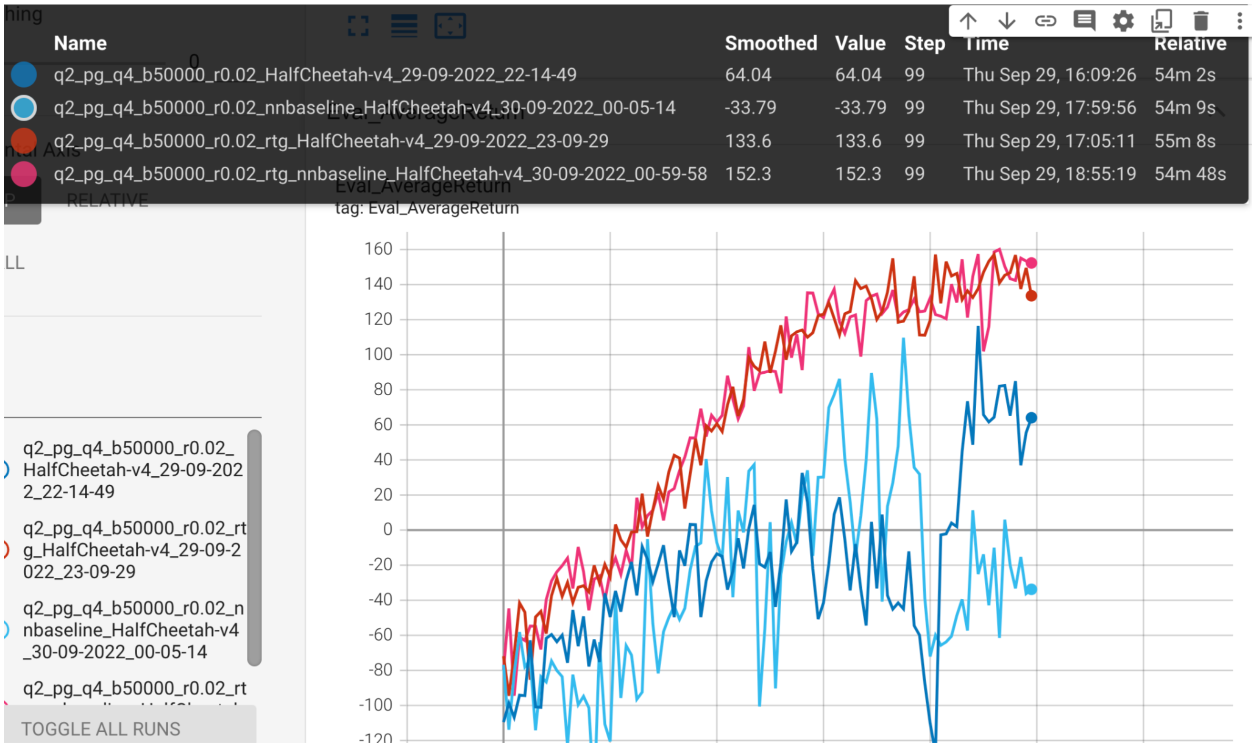


Legend:



	Name	Smoothed	Value	Step		
●	q2_pg_q4_search_b10000_lr0.005_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_21-47-59	2.475	2.475	99	Thu Sep 29, 14:59:41	11m 34s
●	q2_pg_q4_search_b10000_lr0.01_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_15-34-34	73.74	73.74	99	Thu Sep 29, 08:45:39	10m 57s
●	q2_pg_q4_search_b10000_lr0.02_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_22-00-38	27.37	27.37	99	Thu Sep 29, 15:12:47	12m 1s
●	q2_pg_q4_search_b30000_lr0.005_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_20-42-46	22.69	22.69	99	Thu Sep 29, 14:16:52	33m 44s
●	q2_pg_q4_search_b30000_lr0.01_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_06-50-13	130.6	130.6	99	Thu Sep 29, 00:39:26	48m 39s
●	q2_pg_q4_search_b30000_lr0.02_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_20-08-07	132.5	132.5	99	Thu Sep 29, 13:41:40	33m 13s
●	q2_pg_q4_search_b50000_lr0.005_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_17-03-34	63.64	63.64	99	Thu Sep 29, 10:58:53	54m 46s
●	q2_pg_q4_search_b50000_lr0.01_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_16-04-46	110.3	110.3	99	Thu Sep 29, 09:59:29	54m 8s
●	q2_pg_q4_search_b50000_lr0.02_rtg_nnbaseline_HalfCheetah-v4_29-09-2022_19-05-29	152.3	152.3	99	Thu Sep 29, 13:00:49	54m 46s

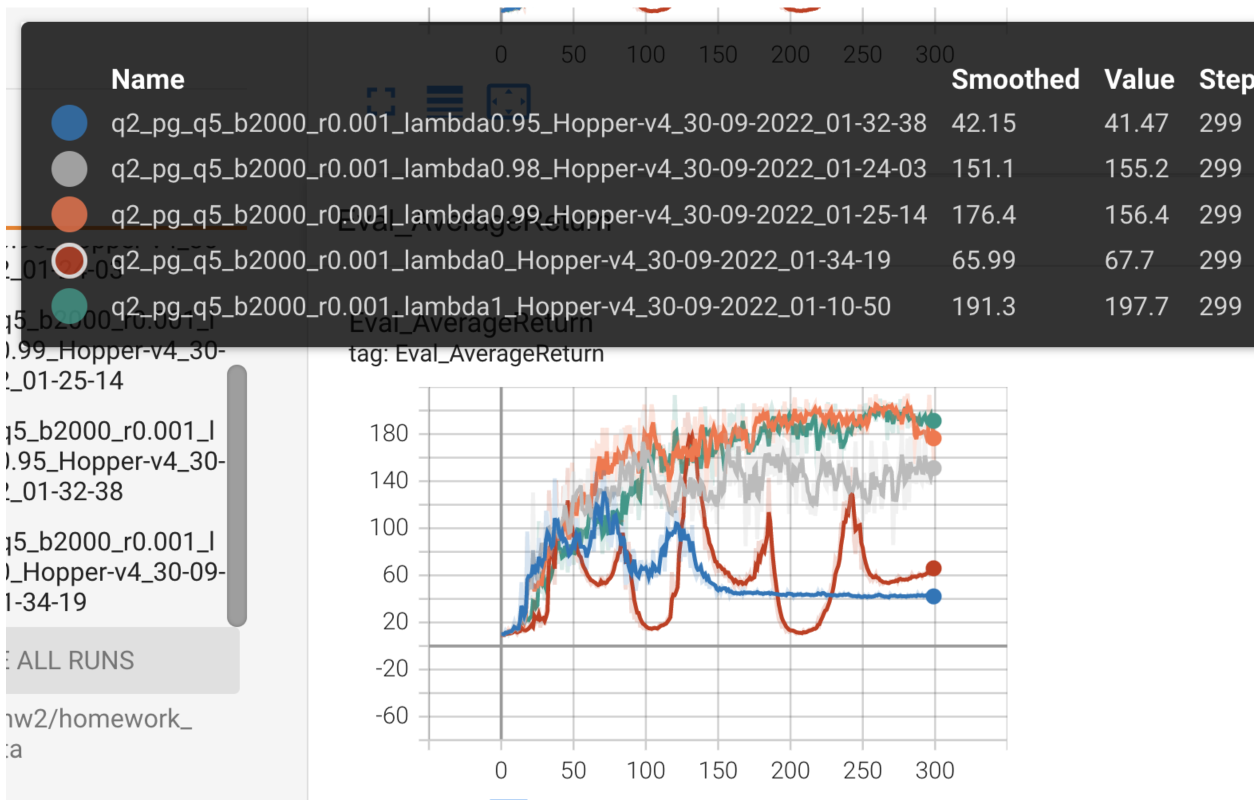
The results shows that among {0.005, 0.01, 0.02}, the larger the learning rate the better the result. Larger batch size seems to also be better but the effect in this experiment is limited comparing to the learning rate.



	Name	Smoothed	Value	Step	Time	Relative
●	q2_pg_q4_b50000_r0.02_HalfCheetah-v4_29-09-2022_22-14-49	64.04	64.04	99	Thu Sep 29, 16:09:26	54m 2s
●	q2_pg_q4_b50000_r0.02_nnbaseline_HalfCheetah-v4_30-09-2022_00-05-14	-33.79	-33.79	99	Thu Sep 29, 17:59:56	54m 9s
●	q2_pg_q4_b50000_r0.02_rtg_HalfCheetah-v4_29-09-2022_23-09-29	133.6	133.6	99	Thu Sep 29, 17:05:11	55m 8s
●	q2_pg_q4_b50000_r0.02_rtg_nnbaseline_HalfCheetah-v4_30-09-2022_00-59-58	152.3	152.3	99	Thu Sep 29, 18:55:19	54m 48s

Eval_AverageReturn
tag: Eval_AverageReturn
160
140
120
100
80
60
40
20
0
-20
-40
-60
-80
-100
-120
q2_pg_q4_b50000_r0.02_HalfCheetah-v4_29-09-2022_22-14-49
q2_pg_q4_b50000_r0.02_rtg_HalfCheetah-v4_29-09-2022_23-09-29
q2_pg_q4_b50000_r0.02_nnbaseline_HalfCheetah-v4_30-09-2022_00-05-14
q2_pg_q4_b50000_r0.02_rt
RELATIVE
TOGGLE ALL RUNS

Experiment 5



	Name	Smoothed	Value	Step
●	q2_pg_q5_b2000_r0.001_lambda0.95_Hopper-v4_30-09-2022_01-32-38	42.15	41.47	299
●	q2_pg_q5_b2000_r0.001_lambda0.98_Hopper-v4_30-09-2022_01-24-03	151.1	155.2	299
●	q2_pg_q5_b2000_r0.001_lambda0.99_Hopper-v4_30-09-2022_01-25-14	176.4	156.4	299
●	q2_pg_q5_b2000_r0.001_lambda0_Hopper-v4_30-09-2022_01-34-19	65.99	67.7	299
●	q2_pg_q5_b2000_r0.001_lambda1_Hopper-v4_30-09-2022_01-10-50	191.3	197.7	299

Higher lambda values stabilize the training process.

```
----------------------------------------------------------------------
--------------------------
#@title runtime arguments

class Args:

  def __getitem__(self, key):
    return getattr(self, key)

  def __setitem__(self, key, val):
    setattr(self, key, val)

  def __contains__(self, key):
    return hasattr(self, key)

  env_name = 'CartPole-v0' #@param
  exp_name = 'q1_sb_no_rtg_dsa_submission' #@param

  #@markdown main parameters of interest
  n_iter = 100 #@param {type: "integer"}

  ## PDF will tell you how to set ep_len
  ## and discount for each environment
  ep_len = 200 #@param {type: "integer"}
  discount = 1 #@param {type: "number"}

  reward_to_go = False #@param {type: "boolean"}
  nn_baseline = False #@param {type: "boolean"}
  gae_lambda = None #@param {type: "float"}
  dont_standardize_advantages = False #@param {type: "boolean"}

  #@markdown batches and steps
  batch_size = 1000 #@param {type: "integer"}
  eval_batch_size = 400 #@param {type: "integer"}

  num_agent_train_steps_per_iter = 1 #@param {type: "integer"}
  learning_rate =  5e-3 #@param {type: "number"}

  #@markdown MLP parameters
  n_layers = 2 #@param {type: "integer"}
  size = 64 #@param {type: "integer"}

  #@markdown system
  save_params = False #@param {type: "boolean"}
  no_gpu = False #@param {type: "boolean"}
  which_gpu = 0 #@param {type: "integer"}
  seed = 1 #@param {type: "integer"}

  action_noise_std = 0 #@param {type: "float"}
  #@markdown logging
  ## default is to not log video so
  ## that logs are small enough to be
  ## uploaded to gradscope
  video_log_freq =  -1#@param {type: "integer"}
  scalar_log_freq =  1#@param {type: "integer"}


args = Args()

## ensure compatibility with hw1 code
args['train_batch_size'] = args['batch_size']

if args['video_log_freq'] > 0:
  import warnings
  warnings.warn(
      '''\nLogging videos will make eventfiles too'''
      '''\nlarge for the autograder. Set video_log_freq = -1'''
      '''\nfor the runs you intend to submit.''')
----------------------------------------------------------------------
--------------------------
#@title runtime arguments

class Args:

  def __getitem__(self, key):
    return getattr(self, key)

  def __setitem__(self, key, val):
    setattr(self, key, val)

  def __contains__(self, key):
    return hasattr(self, key)

  env_name = 'CartPole-v0' #@param
  exp_name = 'q1_sb_rtg_dsa_submission' #@param

  #@markdown main parameters of interest
  n_iter = 100 #@param {type: "integer"}

  ## PDF will tell you how to set ep_len
  ## and discount for each environment
  ep_len = 200 #@param {type: "integer"}
  discount = 1 #@param {type: "number"}

  reward_to_go = True #@param {type: "boolean"}
  nn_baseline = False #@param {type: "boolean"}
  gae_lambda = None #@param {type: "float"}
  dont_standardize_advantages = False #@param {type: "boolean"}
  #@markdown batches and steps
  batch_size = 1000 #@param {type: "integer"}
  eval_batch_size = 400 #@param {type: "integer"}

  num_agent_train_steps_per_iter = 1 #@param {type: "integer"}
  learning_rate =  5e-3 #@param {type: "number"}

  #@markdown MLP parameters
  n_layers = 2 #@param {type: "integer"}
  size = 64 #@param {type: "integer"}

  #@markdown system
  save_params = False #@param {type: "boolean"}
  no_gpu = False #@param {type: "boolean"}
  which_gpu = 0 #@param {type: "integer"}
  seed = 1 #@param {type: "integer"}

  action_noise_std = 0 #@param {type: "float"}

  #@markdown logging
  ## default is to not log video so
  ## that logs are small enough to be
  ## uploaded to gradscope
  video_log_freq =  -1#@param {type: "integer"}
  scalar_log_freq =  1#@param {type: "integer"}


args = Args()

## ensure compatibility with hw1 code
args['train_batch_size'] = args['batch_size']

if args['video_log_freq'] > 0:
  import warnings
  warnings.warn(
      '''\nLogging videos will make eventfiles too'''
      '''\nlarge for the autograder. Set video_log_freq = -1'''
      '''\nfor the runs you intend to submit.''')
----------------------------------------------------------------------
--------------------------
#@title runtime arguments

class Args:

  def __getitem__(self, key):
    return getattr(self, key)

  def __setitem__(self, key, val):
    setattr(self, key, val)
  def __contains__(self, key):
    return hasattr(self, key)

  env_name = 'CartPole-v0' #@param
  exp_name = 'q1_sb_rtg_na_submission' #@param

  #@markdown main parameters of interest
  n_iter = 100 #@param {type: "integer"}

  ## PDF will tell you how to set ep_len
  ## and discount for each environment
  ep_len = 200 #@param {type: "integer"}
  discount = 1 #@param {type: "number"}

  reward_to_go = True #@param {type: "boolean"}
  nn_baseline = False #@param {type: "boolean"}
  gae_lambda = None #@param {type: "float"}
  dont_standardize_advantages = True #@param {type: "boolean"}

  #@markdown batches and steps
  batch_size = 1000 #@param {type: "integer"}
  eval_batch_size = 400 #@param {type: "integer"}

  num_agent_train_steps_per_iter = 1 #@param {type: "integer"}
  learning_rate =  5e-3 #@param {type: "number"}

  #@markdown MLP parameters
  n_layers = 2 #@param {type: "integer"}
  size = 64 #@param {type: "integer"}

  #@markdown system
  save_params = False #@param {type: "boolean"}
  no_gpu = False #@param {type: "boolean"}
  which_gpu = 0 #@param {type: "integer"}
  seed = 1 #@param {type: "integer"}

  action_noise_std = 0 #@param {type: "float"}

  #@markdown logging
  ## default is to not log video so
  ## that logs are small enough to be
  ## uploaded to gradscope
  video_log_freq =  -1#@param {type: "integer"}
  scalar_log_freq =  1#@param {type: "integer"}


args = Args()

## ensure compatibility with hw1 code
args['train_batch_size'] = args['batch_size']
```

```
if args['video_log_freq'] > 0:
  import warnings
  warnings.warn(
      '''\nLogging videos will make eventfiles too'''
      '''\nlarge for the autograder. Set video_log_freq = -1'''
      '''\nfor the runs you intend to submit.''')
-----------------------------------------------------------------------
--------------------------
#@title runtime arguments

class Args:

  def __getitem__(self, key):
    return getattr(self, key)

  def __setitem__(self, key, val):
    setattr(self, key, val)

  def __contains__(self, key):
    return hasattr(self, key)

  env_name = 'CartPole-v0' #@param
  exp_name = 'q1_lb_no_rtg_dsa_submission' #@param

  #@markdown main parameters of interest
  n_iter = 100 #@param {type: "integer"}

  ## PDF will tell you how to set ep_len
  ## and discount for each environment
  ep_len = 200 #@param {type: "integer"}
  discount = 1 #@param {type: "number"}

  reward_to_go = False #@param {type: "boolean"}
  nn_baseline = False #@param {type: "boolean"}
  gae_lambda = None #@param {type: "float"}
  dont_standardize_advantages = False #@param {type: "boolean"}

  #@markdown batches and steps
  batch_size = 5000 #@param {type: "integer"}
  eval_batch_size = 400 #@param {type: "integer"}

  num_agent_train_steps_per_iter = 1 #@param {type: "integer"}
  learning_rate =  5e-3 #@param {type: "number"}

  #@markdown MLP parameters
  n_layers = 2 #@param {type: "integer"}
  size = 64 #@param {type: "integer"}

  #@markdown system
  save_params = False #@param {type: "boolean"}
  no_gpu = False #@param {type: "boolean"}
  which_gpu = 0 #@param {type: "integer"}
  seed = 1 #@param {type: "integer"}

  action_noise_std = 0 #@param {type: "float"}

  #@markdown logging
  ## default is to not log video so
  ## that logs are small enough to be
  ## uploaded to gradscope
  video_log_freq =  -1#@param {type: "integer"}
  scalar_log_freq =  1#@param {type: "integer"}


args = Args()

## ensure compatibility with hw1 code
args['train_batch_size'] = args['batch_size']

if args['video_log_freq'] > 0:
  import warnings
  warnings.warn(
      '''\nLogging videos will make eventfiles too'''
      '''\nlarge for the autograder. Set video_log_freq = -1'''
      '''\nfor the runs you intend to submit.''')
----------------------------------------------------------------------
--------------------------
#@title runtime arguments

class Args:

  def __getitem__(self, key):
    return getattr(self, key)

  def __setitem__(self, key, val):
    setattr(self, key, val)

  def __contains__(self, key):
    return hasattr(self, key)

  env_name = 'CartPole-v0' #@param
  exp_name = 'q1_lb_rtg_dsa_submission' #@param

  #@markdown main parameters of interest
  n_iter = 100 #@param {type: "integer"}

  ## PDF will tell you how to set ep_len
  ## and discount for each environment
  ep_len = 200 #@param {type: "integer"}
  discount = 1 #@param {type: "number"}

  reward_to_go = True #@param {type: "boolean"}
  nn_baseline = False #@param {type: "boolean"}
  gae_lambda = None #@param {type: "float"}
  dont_standardize_advantages = False #@param {type: "boolean"}

  #@markdown batches and steps
  batch_size = 5000 #@param {type: "integer"}
  eval_batch_size = 400 #@param {type: "integer"}

  num_agent_train_steps_per_iter = 1 #@param {type: "integer"}
  learning_rate =  5e-3 #@param {type: "number"}

  #@markdown MLP parameters
  n_layers = 2 #@param {type: "integer"}
  size = 64 #@param {type: "integer"}

  #@markdown system
  save_params = False #@param {type: "boolean"}
  no_gpu = False #@param {type: "boolean"}
  which_gpu = 0 #@param {type: "integer"}
  seed = 1 #@param {type: "integer"}

  action_noise_std = 0 #@param {type: "float"}

  #@markdown logging
  ## default is to not log video so
  ## that logs are small enough to be
  ## uploaded to gradscope
  video_log_freq =  -1#@param {type: "integer"}
  scalar_log_freq =  1#@param {type: "integer"}


args = Args()

## ensure compatibility with hw1 code
args['train_batch_size'] = args['batch_size']

if args['video_log_freq'] > 0:
  import warnings
  warnings.warn(
      '''\nLogging videos will make eventfiles too'''
      '''\nlarge for the autograder. Set video_log_freq = -1'''
      '''\nfor the runs you intend to submit.''')
----------------------------------------------------------------------
--------------------------
#@title runtime arguments

class Args:
  def __getitem__(self, key):
    return getattr(self, key)

  def __setitem__(self, key, val):
    setattr(self, key, val)

  def __contains__(self, key):
    return hasattr(self, key)

  env_name = 'CartPole-v0' #@param
  exp_name = 'q1_lb_rtg_na_submission' #@param

  #@markdown main parameters of interest
  n_iter = 100 #@param {type: "integer"}

  ## PDF will tell you how to set ep_len
  ## and discount for each environment
  ep_len = 200 #@param {type: "integer"}
  discount = 1 #@param {type: "number"}

  reward_to_go = True #@param {type: "boolean"}
  nn_baseline = False #@param {type: "boolean"}
  gae_lambda = None #@param {type: "float"}
  dont_standardize_advantages = True #@param {type: "boolean"}

  #@markdown batches and steps
  batch_size = 5000 #@param {type: "integer"}
  eval_batch_size = 400 #@param {type: "integer"}

  num_agent_train_steps_per_iter = 1 #@param {type: "integer"}
  learning_rate =  5e-3 #@param {type: "number"}

  #@markdown MLP parameters
  n_layers = 2 #@param {type: "integer"}
  size = 64 #@param {type: "integer"}

  #@markdown system
  save_params = False #@param {type: "boolean"}
  no_gpu = False #@param {type: "boolean"}
  which_gpu = 0 #@param {type: "integer"}
  seed = 1 #@param {type: "integer"}

  action_noise_std = 0 #@param {type: "float"}

  #@markdown logging
  ## default is to not log video so
  ## that logs are small enough to be
  ## uploaded to gradscope
  video_log_freq =  -1#@param {type: "integer"}
  scalar_log_freq =  1#@param {type: "integer"}


args = Args()

## ensure compatibility with hw1 code
args['train_batch_size'] = args['batch_size']

if args['video_log_freq'] > 0:
  import warnings
  warnings.warn(
      '''\nLogging videos will make eventfiles too'''
      '''\nlarge for the autograder. Set video_log_freq = -1'''
      '''\nfor the runs you intend to submit.''')
----------------------------------------------------------------------
--------------------------
#@title runtime arguments

class Args:

  def __getitem__(self, key):
    return getattr(self, key)

  def __setitem__(self, key, val):
    setattr(self, key, val)

  def __contains__(self, key):
    return hasattr(self, key)

  env_name = 'InvertedPendulum-v4' #@param
  exp_name = 'q2_b500_r1e-2' #@param

  #@markdown main parameters of interest
  n_iter = 100 #@param {type: "integer"}

  ## PDF will tell you how to set ep_len
  ## and discount for each environment
  ep_len = 1000 #@param {type: "integer"}
  discount = 0.9 #@param {type: "number"}

  reward_to_go = True #@param {type: "boolean"}
  nn_baseline = False #@param {type: "boolean"}
  gae_lambda = None #@param {type: "float"}
  dont_standardize_advantages = False #@param {type: "boolean"}

  #@markdown batches and steps
  batch_size = 500 #@param {type: "integer"}
  eval_batch_size = 400 #@param {type: "integer"}

  num_agent_train_steps_per_iter = 1 #@param {type: "integer"}
  learning_rate =  1e-2 #@param {type: "number"}

  #@markdown MLP parameters
  n_layers = 2 #@param {type: "integer"}
  size = 64 #@param {type: "integer"}

  #@markdown system
  save_params = False #@param {type: "boolean"}
  no_gpu = True #@param {type: "boolean"}
  which_gpu = 0 #@param {type: "integer"}
  seed = 0 #@param {type: "integer"}

  action_noise_std = 0 #@param {type: "float"}

  #@markdown logging
  ## default is to not log video so
  ## that logs are small enough to be
  ## uploaded to gradscope
  video_log_freq =  -1#@param {type: "integer"}
  scalar_log_freq =  1#@param {type: "integer"}


args = Args()

## ensure compatibility with hw1 code
args['train_batch_size'] = args['batch_size']

if args['video_log_freq'] > 0:
  import warnings
  warnings.warn(
      '''\nLogging videos will make eventfiles too'''
      '''\nlarge for the autograder. Set video_log_freq = -1'''
      '''\nfor the runs you intend to submit.''')
```