CS 237B: Principles of Robot Autonomy II
Problem Set 3

Name: Abhyudit Singh Manhas
SUID: 06645995

Problem 1: Getting Started

(i) If we initialize all weights and biases with 0 in a deep neural network, all neurons will have the same output, leading to symmetry between neurons. Hence, each neuron will update its weights and biases in the same way during training, and so they will learn the same features in each iteration. That is, the network fails to learn different features in the data. Another downside is that it can cause the network to converge to a sub-optimal solution, or get stuck in a local optima, which leads to poor performance. One possible advantage is that this network (with initial weights and biases as 0) can act as a base-line, and be compared with models with different initialization methods. Their performances can be compared, and the best initialization method can then be chosen for the problem in hand.

(ii) "Xavier initialization" initializes the weights such that the variance of the activations are the same across each layer. This helps prevent the variance of the back-propagated gradient from exploding or vanishing. As it initializes the weights to suitable values, it leads to faster convergence, prevents the network from getting stuck in a local optima, and enables the network to learn more meaningful features (and hence avoid overfitting).

(iii) Adam optimizer combines the benefits of two other optimization algorithms, Adagrad and RMSProp. Its main advantage is that it computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. Hence learning rates are adjusted dynamically during training, which can help speed up convergence and improve accuracy.
Some of its other advantages are that it is straightforward to implement, is computationally efficient making it suitable for large-scale machine learning problems, has little memory requirements, and has been shown to work well for a wide range of applications. Also, the hyper-parameters have intuitive interpretations and typically require little tuning.
One potential drawback of using this optimizer is that it can sometimes converge to sub-optimal solutions or get stuck in a local optima, especially in high-dimensional problems. This is caused when the adaptive learning rate causes the optimizer to get stuck in a sharp, narrow "valley". Here, the learning rate can become so small, that the optimizer cannot escape from this valley.

(iv) Using the SGD optimizer, the two approaches (1) and (2) would not necessarily result in the same network. It is quite possible for the optimization to converge to a different local minimum during the first 250 epochs in (2), than what would have been obtained if trained for all 500 epochs (approach (1)). When the optimization is restarted in (2), it may converge to a different local minimum than what would have been obtained if trained for the remaining 250 epochs starting from the initial network. This can potentially result in different networks being obtained from (1) and (2).
The same holds true for the Adam optimizer as well, that is, due to the stochastic nature of the optimization we can possibly get different networks from (1) and (2). As discussed earlier, Adam uses adaptive learning rates that usually lead to faster convergence and good generalization. So the network obtained from Adam can potentially outperform a network obtained from SGD on a test set.

Problem 2: Behavior Cloning

(i) The optimization problem is to find θ that minimizes

$$\mathbb{E}_{(o,a^*)\sim P^*} L(a^*, h_\theta(o, g)) \tag{1}$$

where $P^* = P(o \mid \pi^*)$ is the distribution of observations visited by the expert, and a^* is the action taken by the expert.

(ii) Filled in the missing parts of `train_il.py`.

(iii) I used the same loss function for each goal. My training procedure was as follows:

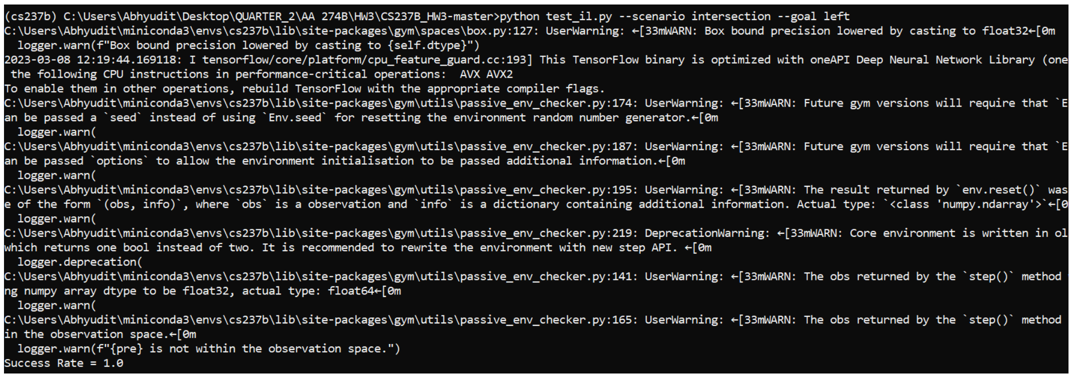
(a) For the ”left” goal: I trained the network for 1500 epochs with a learning rate of 0.001.

(b) For the ”right” goal: I trained the network for 1200 epochs with a learning rate of 0.001.

(c) For the ”straight” goal: I trained the network for 900 epochs with a learning rate of 0.001, used `--restore`, and retrained the network for 600 more epochs with a learning rate of 0.0001.

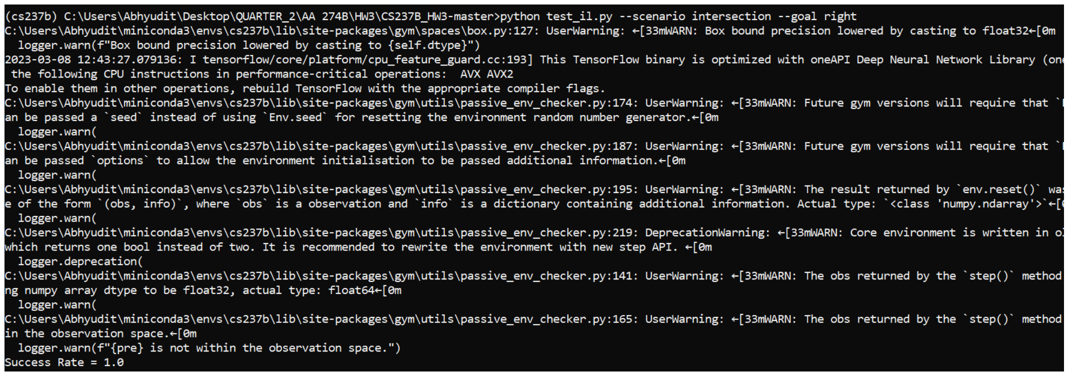
(iv) My success rates are:

(a) For the ”left” goal: Success = 1.



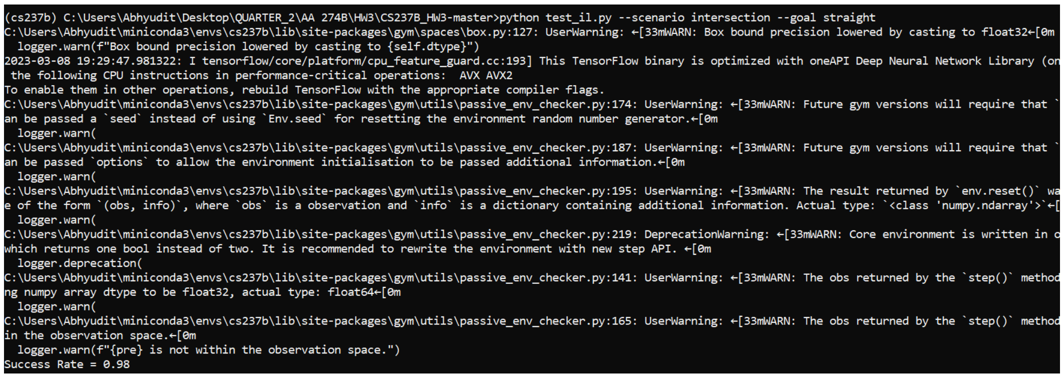
```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_il.py --scenario intersection --goal left
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 12:19:44.169118: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (one
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `E
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `E
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` was
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[0
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in ol
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 1.0
```

(b) For the ”right” goal: Success = 1.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_il.py --scenario intersection --goal right
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 12:43:27.079136: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (one
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 1.0
```

(c) For the ”straight” goal: Success = 0.98.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_il.py --scenario intersection --goal straight
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 19:29:47.981322: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 0.98
```

(v) I have used the L2 loss for both steering (0th dimension or 1st column) and throttle (1st dimension or 2nd column). That is,

$$\begin{aligned} L_{steering} &= \frac{1}{N}\sum_{i=1}^{N}(a^*_{i1} - a^{est}_{i1})^2 \\ L_{throttle} &= \frac{1}{N}\sum_{i=1}^{N}(a^*_{i2} - a^{est}_{i2})^2 \end{aligned} \tag{2}$$

In this notation, a^* is the expert action, a^{est} is the action output by the neural network, and both are of size $N \times 2$. The overall loss is then computed by linearly combining $L_{steering}$ and $L_{throttle}$ through a penalizing factor α. It is given by:

$$L_{overall} = \alpha L_{steering} + (1-\alpha)L_{throttle} \tag{3}$$

I have used $\alpha = 0.99$ and so penalized the differences in the steering dimension more heavily.
If we perfectly overfit the data, the minimum loss we would see is 0. It is bounded below by 0 since the L2 loss is greater than or equal to 0; it can never be negative.

(vi) Let the action space be n-dimensional (in this case, $n = 2$). The output from the neural network is of size $n^2 + n$. The first n elements are elements of the mean vector. The remaining n^2 elements are reshaped into a $n \times n$ matrix A. The covariance matrix Σ is then computed by:

$$\Sigma = AA^T + \epsilon I_n \tag{4}$$

where ϵ is a small number ($= 0.001$) and I_n is the $n \times n$ identity matrix. This ensures that the covariance matrix Σ is always positive semi-definite or PSD.

(vii) If we perfectly overfit the data, the minimum loss we would see is $-\infty$. During training, I observe negative losses and it is not bounded below.

(viii) Filled in the missing parts of `train_ildist.py`.

(ix) I used the same loss function for each goal. My training procedure was as follows:

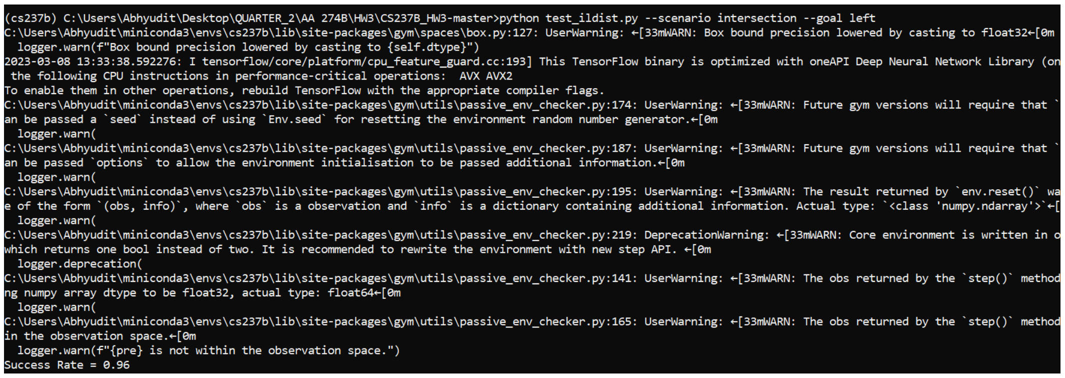
(a) For the ”left” goal: I trained the network for 1500 epochs with a learning rate of 0.001.

(b) For the ”right” goal: I trained the network for 750 epochs with a learning rate of 0.001, used `--restore`, retrained for 750 epochs with a learning rate of 0.0001, again used `--restore`, retrained for 750 epochs with a learning rate of 0.00001, again used `--restore` and finally retrained for 750 more epochs with a learning rate of 0.00001.

(c) For the ”straight” goal: I trained the network for 100 epochs with a learning rate of 0.001, used `--restore`, retrained for 750 epochs with a learning rate of 0.0001, again used `--restore`, retrained for 750 epochs with a learning rate of 0.00001, again used `--restore` and finally retrained for 750 more epochs with a learning rate of 0.00001.

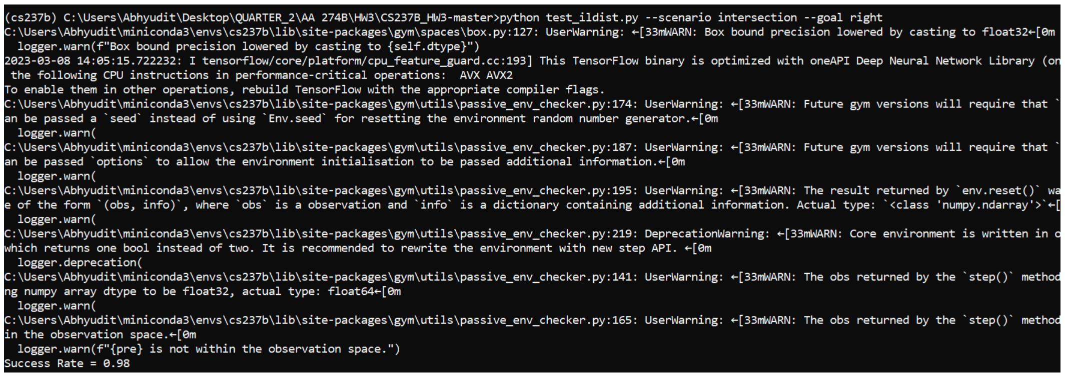
(x) My success rates are:

(a) For the "left" goal: Success = 0.96.



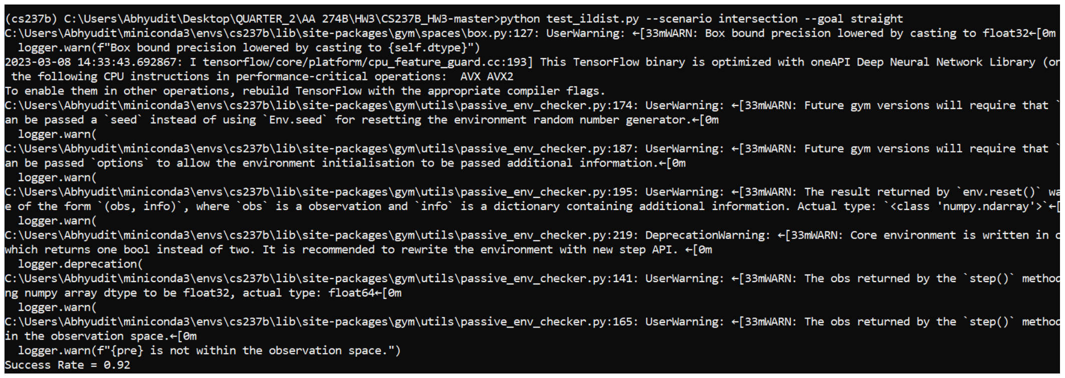
```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_ildist.py --scenario intersection --goal left
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 13:33:38.592276: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 0.96
```

(b) For the "right" goal: Success = 0.98.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_ildist.py --scenario intersection --goal right
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 14:05:15.722232: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 0.98
```

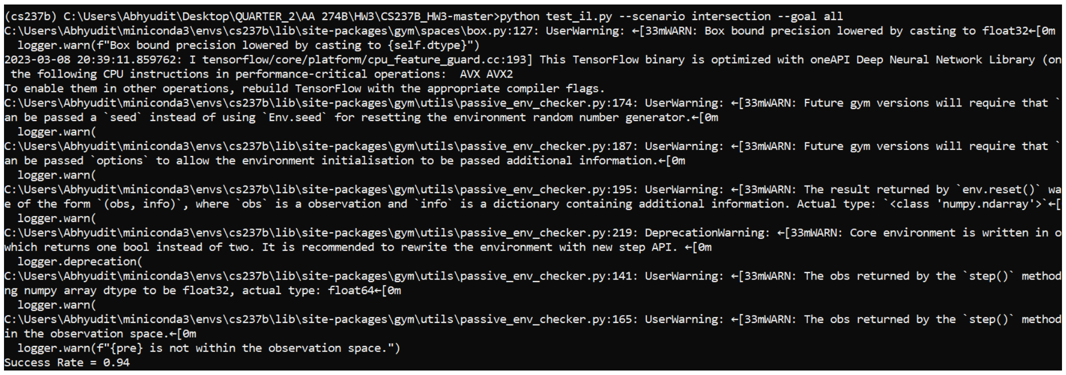
(c) For the "straight" goal: Success = 0.92.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_ildist.py --scenario intersection --goal straight
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 14:33:43.692867: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 0.92
```

Problem 3: Conditional Imitation Learning

(i) I have trained the policy for 1500 epochs with a learning rate of 0.001. I get success = 0.94.



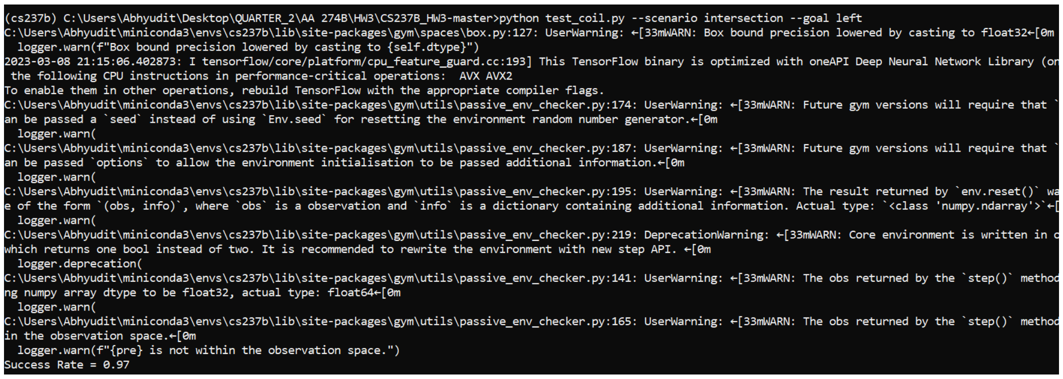
```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_il.py --scenario intersection --goal all
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 20:39:11.859762: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 0.94
```

The problem with this policy is that it almost always takes the right direction/turn. I feel I have control over which direction the car is going only when I want to turn right, but in reality I do not think I have control as it just turns right most of the times, irrespective of where I want to go.

(ii) Filled in the missing parts of `train_coil.py`.

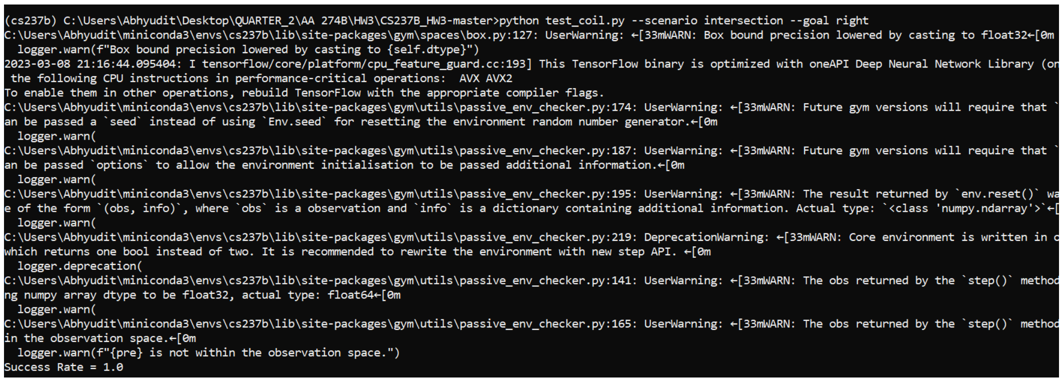
(iii) My success rates are:

(a) For the "left" goal: Success = 0.97.



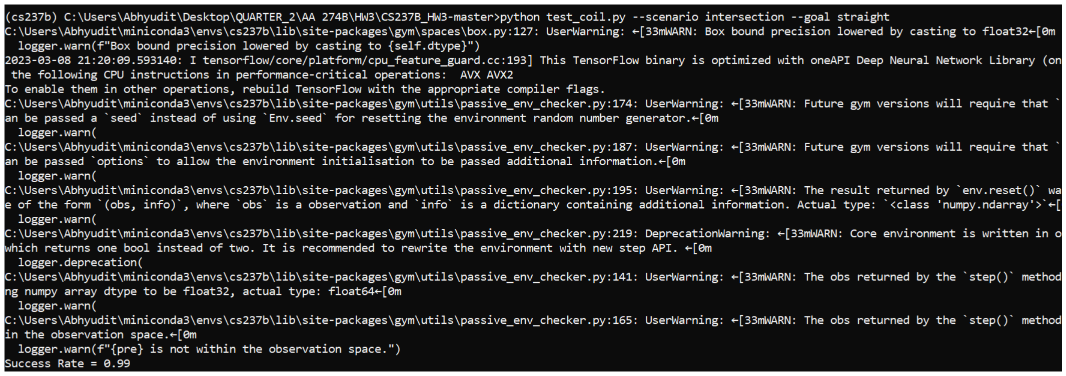
```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_coil.py --scenario intersection --goal left
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 21:15:06.402873: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 0.97
```

(b) For the "right" goal: Success = 1.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_coil.py --scenario intersection --goal right
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 21:16:44.095404: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 1.0
```

(c) For the "straight" goal: Success = 0.99.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_coil.py --scenario intersection --goal straight
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 21:20:09.593140: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 0.99
```

(iv) I trained the network for 150 epochs with a learning rate of 0.01, used `--restore`, retrained for 150 epochs with a learning rate of 0.001, again used `--restore`, and finally retrained for 150 more epochs with a learning rate of 0.0001.

Problem 4: Intent Inference & Shared Autonomy

(i) For a particular goal $g \in \mathcal{G}$, we have:

$$P(g \mid o, a) = \frac{P(a, o, g)}{P(a, o)} = \frac{P(a, o, g)}{\sum\limits_{g \in \mathcal{G}} P(a, o, g)} \tag{5}$$

where we have used the total probability theorem. Since $P(a, o, g) = P(a \mid o, g)P(g \mid o)P(o)$, we get:

$$P(g \mid o, a) = \frac{P(a \mid o, g)P(g \mid o)P(o)}{\sum\limits_{g \in \mathcal{G}} P(a \mid o, g)P(g \mid o)P(o)} = \frac{P(a \mid o, g)P(g \mid o)}{\sum\limits_{g \in \mathcal{G}} P(a \mid o, g)P(g \mid o)} \tag{6}$$

Since we assume a uniform prior over the goals, the term $P(g \mid o)$ cancels out in both the numerator and denominator, giving us the final expression:

$$P(g \mid o, a) = \frac{P(a \mid o, g)}{\sum\limits_{g \in \mathcal{G}} P(a \mid o, g)} \tag{7}$$

We can obtain $P(a \mid o, g)$ from the mixture density network. Then $P(g \mid o, a)$ is just the normalized value of $P(a \mid o, g)$.

(ii) Filled in the missing parts of `intent_inference.py`.

(iii) For most of the episodes, it ends up predicting the correct intent. It always predicts my intent to go "straight" correctly. It also predicts my intent to go "left" or "right" mostly correctly. It only fails when I take a left or right turn very late. It predicts those late turns as "straight". Before making a late turn, I just go straight for most of my route which leads it to believe that my intent is "straight". Making a late turn thereafter does not update its confidence by a lot, leading it to incorrectly predict my intent as "straight".

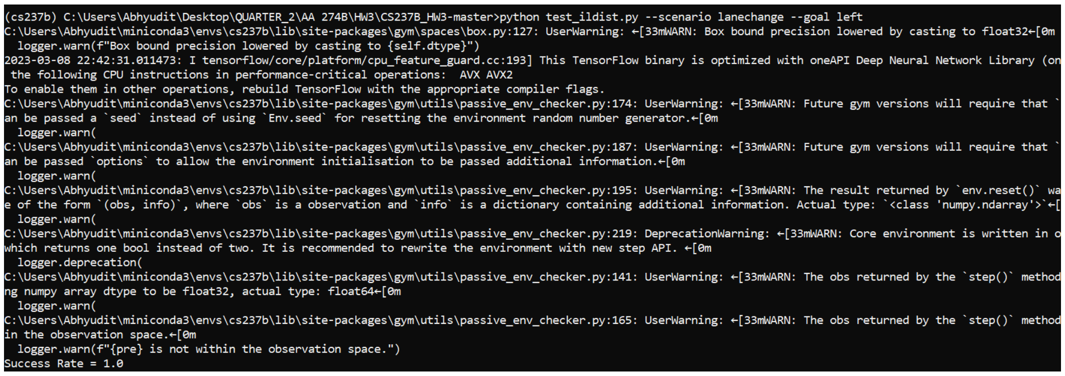
(iv) My training procedure was as follows:

(a) For the "left" goal: I trained the network for 1200 epochs with a learning rate of 0.001.

(b) For the "right" goal: I trained the network for 1200 epochs with a learning rate of 0.001.

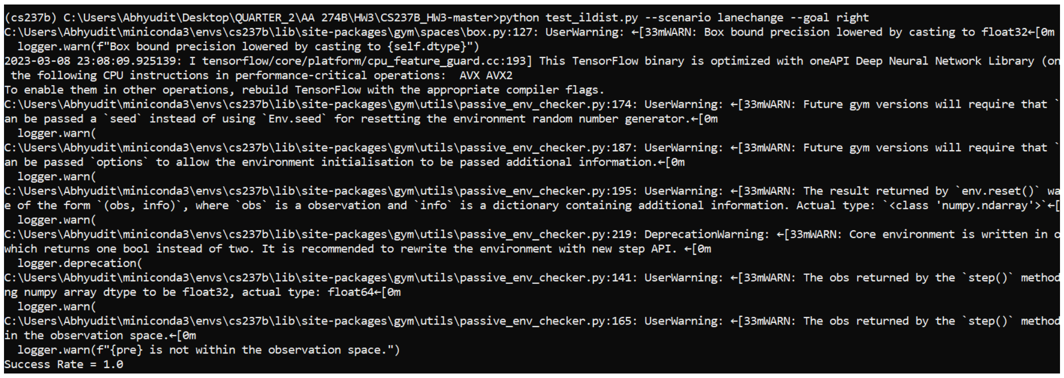
My success rates are:

(a) For the "left" goal: Success = 1.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_ildist.py --scenario lanechange --goal left
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 22:42:31.011473: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 1.0
```

(b) For the "right" goal: Success = 1.



```
(cs237b) C:\Users\Abhyudit\Desktop\QUARTER_2\AA 274B\HW3\CS237B_HW3-master>python test_ildist.py --scenario lanechange --goal right
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\spaces\box.py:127: UserWarning: ←[33mWARN: Box bound precision lowered by casting to float32←[0m
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
2023-03-08 23:08:09.925139: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (on
 the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:174: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed a `seed` instead of using `Env.seed` for resetting the environment random number generator.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:187: UserWarning: ←[33mWARN: Future gym versions will require that `
an be passed `options` to allow the environment initialisation to be passed additional information.←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:195: UserWarning: ←[33mWARN: The result returned by `env.reset()` wa
e of the form `(obs, info)`, where `obs` is a observation and `info` is a dictionary containing additional information. Actual type: `<class 'numpy.ndarray'>`←[
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:219: DeprecationWarning: ←[33mWARN: Core environment is written in o
which returns one bool instead of two. It is recommended to rewrite the environment with new step API. ←[0m
  logger.deprecation(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:141: UserWarning: ←[33mWARN: The obs returned by the `step()` method
ng numpy array dtype to be float32, actual type: float64←[0m
  logger.warn(
C:\Users\Abhyudit\miniconda3\envs\cs237b\lib\site-packages\gym\utils\passive_env_checker.py:165: UserWarning: ←[33mWARN: The obs returned by the `step()` method
in the observation space.←[0m
  logger.warn(f"{pre} is not within the observation space.")
Success Rate = 1.0
```

(v) Enforcing constraints on the robot action is important from a human-robot interaction perspective because it ensures safety and reliability in the interaction. If the robot's steering or throttle is too large, it may result in erratic movements, which could cause harm to humans or a serious accident. Also, human users may not feel comfortable or safe interacting with a robot with such unpredictable behavior. Hence, by enforcing constraints on the robot's actions, we can ensure that it operates within a safe and predictable range of behavior, making it easier for humans to interact with it and reducing the risk of accidents or injuries. This can help build trust between humans and robots, which is essential for successful human-robot interaction.

(vi) Filled in the missing parts of `shared_autonomy.py`.

(vii) I feel that the robot is not really helpful. It is not able to confidently improve the lanechange performance with the inputs I'm providing. For example, when I steer the car to change my lane to right, it also steers towards the right causing the car to go off road. I think this happens because it predicts (correctly) my intent to go right, and so decides to steer right. So both me and the robot steer in the same direction causing the car to sometimes go off road. However, I can still drive the car without any issues, because the maximum steering provided by the robot is only 0.05.