Test Plan and Results

Overall Test Plan

For testing MORL, we have both automated and manual testing. For the automated testing, we have set up unit tests using the Python unittest module. These unit tests cover every method in our Multilearn and QLearn classes, where the majority of our functionality lies. We plan to add to these unit tests as necessary as the project progresses. Our manual tests take the form of our OpenAI Gym examples. These examples act as integration tests for our code in addition to creating easily demonstrable examples of our codes functionality for people interested in using MORL.

Test Case Descriptions

TC1.1	Test Case Identifier (A number or unique name)
TC1.2	Purpose of Test
TC1.3	Description of Test
TC1.4	Inputs
TC1.5	Expected Outputs and Results
TC1.6	Normal/abnormal/boundary case indication
TC1.7	Blackbox/whitebox test indication
TC1.8	Functional/performance test indication
TC1.9	Unit/integration test indication

TSQ-1.1 **Test Sequential QLearn 1 (getQ)**
TSQ-1.2 Ensure QLearn.getQ returns correct Q-value for state-action pair
TSQ-1.3

- Manually sets the q value of a particular state-action pair
- Calls getQ on that state and a never-before-seen-state
- Checks to make sure Q-values of both states are correct

TSQ-1.4 Inputs: (State-Action Pair: Tuple(int, int) = Q-Value: Float)
TSQ-1.5 Expected Output: getQ(visited_state) = Q-Value, getQ(unvisited_state) = 0.0
TSQ-1.6 Normal
TSQ-1.7 Whitebox
TSQ-1.8 Functional
TSQ-1.9 Unit

TSQ-2.1 **Test Sequential QLearn 2 (learnQ)**
TSQ-2.2 Ensure QLearn.learnQ updates Q-Values of states-action pairs correctly
TSQ-2.3

- Calls learnQ with arguments for a specific state-action pair and reward
- Tests to ensure that state-action pair's q-value is equal to the reward
- Calls learnQ again with arguments for same state-action pair and new reward

- Tests to ensure that state-action pair's q-value is correctly updated

TSQ-2.4 Inputs: (State (int), Action (int), Reward (float), Reward+GammaMod (float)

TSQ-2.5 Expected Output: QLearn.q[(State, Action)] = 5.0, 5.5 (Whitebox, no actual expected output.

TSQ-2.6 Normal

TSQ-2.7 Whitebox

TSQ-2.8 Functional

TSQ-2.9 Unit

TSQ-3.1 **Test Sequential QLearn 3 (choose_action_egreedy)**

TSQ-3.2 Ensure QLearn.choose_action_egreedy returns either a random action or none depending on epsilon.

TSQ-3.3

- Set Epsilon to 0.0
- Test if choose_action_egreedy returns None
- Set Epsilon to 1.0
- Test if choose_action_egreedy returns anything other than None

TSQ-3.4 Inputs: State (int)

TSQ-3.5 Expected Output: QLearn.choose_action_egreedy = None, Int

TSQ-3.6 Normal

TSQ-3.7 Blackbox

TSQ-3.8 Functional

TSQ-3.9 Unit

TSQ-4.1 **Test Sequential QLearn 4 (choose_action)**

TSQ-4.2 Ensure QLearn.choose_action returns an action and the q-table

TSQ-4.3

- Call choose_action
- Test if it is not equal to None
- Call choose_action with return_q = True
- Test if qdict is not equal to None

TSQ-4.4 Inputs: State (int)

TSQ-4.5 Expected Output: QLearn.choose_action = Action (int) OR Action (int), Q-Table (dict)

TSQ-4.6 Normal

TSQ-4.7 Blackbox

TSQ-4.8 Functional

TSQ-4.9 Unit

TSQ-5.1 **Test Sequential QLearn 5 (learn)**

TSQ-5.2 Ensure QLearn.learn successfully updates Q-values

TSQ-5.3

- Call learn

	● Test using getQ to see if Q-value was updated
TSQ-5.4	Inputs: State (int), Action (int), Reward Tuple (from openai.gym), Next State (int)
TSQ-5.5	Expected Output: QLearn.getQ(State, Action) = Reward
TSQ-5.6	Normal
TSQ-5.7	Whitebox
TSQ-5.8	Functional
TSQ-5.9	Unit
TSQ-6.1	**Test Sequential QLearn 6 (train)**
TSQ-6.2	Ensure QLearn.train runs no more than specified iterations
TSQ-6.3	
	● Call train
	● Test if returned number of iterations is equal to defined max iterations (1)
TSQ-6.4	Inputs: State (int), Environment (Class), Max Iterations (int, optional)
TSQ-6.5	Expected Output: Iterations (int)
TSQ-6.6	Normal
TSQ-6.7	Blackbox
TSQ-6.8	Functional
TSQ-6.9	Unit
TSQ-7.1	**Test Sequential QLearn 7 (train_step)**
TSQ-7.2	Ensure QLearn.train_step returns next step and done bool
TSQ-7.3	
	● Call train_step
	● Check if new_state is not None and if done is False
TSQ-7.4	Inputs: State (int), Environment (Class)
TSQ-7.5	Expected Output: New State (int), Done (bool)
TSQ-7.6	Normal
TSQ-7.7	Blackbox
TSQ-7.8	Functional
TSQ-7.9	Unit
TSM-1.1	**Test Sequential MultiLearn 1 (alpha)**
TSM-1.2	Ensure Multilearn.alpha getter and setter work correctly
TSM-1.3	
	● Instantiate Multilearn with certain alpha
	● Test to see if alpha values are correct
TSM-1.4	Inputs: Learning Rate Alpha (float)
TSM-1.5	Expected Output: Learning Rate Alpha (float)
TSM-1.6	Normal
TSM-1.7	Blackbox
TSM-1.8	Functional
TSM-1.9	Unit

TSM-2.1 **Test Sequential MultiLearn 2 (gamma)**
TSM-2.2 Ensure Multilearn.gamma getter and setter work correctly
TSM-2.3

- Instantiate Multilearn with certain gamma
- Test to see if gamma values are correct

TSM-2.4 Inputs: Learning Rate Gamma (float)
TSM-2.5 Expected Output: Learning Rate Gamma (float)
TSM-2.6 Normal
TSM-2.7 Blackbox
TSM-2.8 Functional
TSM-2.9 Unit

TSM-3.1 **Test Sequential MultiLearn 3 (epsilon)**
TSM-3.2 Ensure Multilearn.epsilon getter and setter work correctly
TSM-3.3

- Instantiate Multilearn with certain epsilon
- Test to see if epsilon values are correct

TSM-3.4 Inputs: Learning Rate Epsilon (float)
TSM-3.5 Expected Output: Learning Rate Epsilon (float)
TSM-3.6 Normal
TSM-3.7 Blackbox
TSM-3.8 Functional
TSM-3.9 Unit

TSM-4.1 **Test Sequential Multilearn 4 (getQ)**
TSM-4.2 Ensure Multilearn.getQ returns correct Q-value for state-action pair
TSM-4.3

- Manually sets the q value of a particular state-action pair
- Calls getQ on that state and a never-before-seen-state
- Checks to make sure Q-values of both states are correct

TSM-4.4 Inputs: (State-Action Pair: Tuple(int, int) = Q-Value: Float)
TSM-4.5 Expected Output: getQ(visited_state) = Q-Dict, getQ(unvisited_state) = Q-Dict (0)
TSM-4.6 Normal
TSM-4.7 Whitebox
TSM-4.8 Functional
TSM-4.9 Unit

TSM-5.1 **Test Sequential Multilearn 5 (choose_actions)**
TSM-5.2 Ensure Multilearn.choose_actions returns a list of actions and a set of Q-tables
TSM-5.3

- Call choose_actions
- Test if it is not equal to None

- Call choose_action with return_q = True
- Test if qdict is not equal to None

TSM-5.4 Inputs: State (int)
TSM-5.5 Expected Output: Multilearn.choose_actions = List(Int)
TSM-5.6 Normal
TSM-5.7 Blackbox
TSM-5.8 Functional
TSM-5.9 Unit

TSM-6.1 **Test Sequential Multilearn 6 (choose_action_maxutil)**
TSM-6.2 Ensure Multilearn.choose_action_maxutil returns either an action of max utility or none depending on epsilon.
TSM-6.3

- Set Epsilon to 0.0
- Test if choose_action_maxutil returns None
- Set Epsilon to 1.0
- Test if choose_action_maxutil returns anything other than None

TSM-6.4 Inputs: State (int)
TSM-6.5 Expected Output: Multilearn.choose_action_maxutil = None, Int
TSM-6.6 Normal
TSM-6.7 Blackbox
TSM-6.8 Functional
TSM-6.9 Unit

TSM-7.1 **Test Sequential Multilearn 7 (choose_action_random)**
TSM-7.2 Ensure Multilearn.choose_action_random returns either a random action or none depending on epsilon.
TSM-7.3

- Set Epsilon to 0.0
- Test if choose_action_random returns None
- Set Epsilon to 1.0
- Test if choose_action_random returns anything other than None

TSM-7.4 Inputs: State (int)
TSM-7.5 Expected Output: Multilearn.choose_action_random = None, Int
TSM-7.6 Normal
TSM-7.7 Blackbox
TSM-7.8 Functional
TSM-7.9 Unit

TSM-8.1 **Test Sequential Multilearn 8 (choose_action_vote)**
TSM-8.2 Ensure Multilearn.choose_action_vote returns either an action of via votes or none depending on epsilon.
TSM-8.3

- Set Epsilon to 0.0
- Test if choose_action_vote returns None
- Set Epsilon to 1.0
- Test if choose_action_vote returns anything other than None

TSM-8.4 Inputs: State (int)
TSM-8.5 Expected Output: Multilearn.choose_action_vote = None, Int
TSM-8.6 Normal
TSM-8.7 Blackbox
TSM-8.8 Functional
TSM-8.9 Unit

TSM-9.1 **Test Sequential Multilearn 9 (choose_action_egreedy)**
TSM-9.2 Ensure Multilearn.choose_action_egreedy returns either a random action or none depending on epsilon.
TSM-9.3

- Set Epsilon to 0.0
- Test if choose_action_egreedy returns None
- Set Epsilon to 1.0
- Test if choose_action_egreedy returns anything other than None

TSM-9.4 Inputs: State (int)
TSM-9.5 Expected Output: Multilearn.choose_action_egreedy = None, Int
TSM-9.6 Normal
TSM-9.7 Blackbox
TSM-9.8 Functional
TSM-9.9 Unit

TSM-10.1 **Test Sequential Multilearn 10 (choose_action)**
TSM-10.2 Ensure Multilearn.choose_actions returns an action and a set of Q-tables
TSM-10.3

- Call choose_action
- Test if it is not equal to None
- Call choose_action with return_q = True
- Test if qdict is not equal to None

TSM-10.4 Inputs: State (int)
TSM-10.5 Expected Output: Multilearn.choose_action = Int
TSM-10.6 Normal
TSM-10.7 Blackbox
TSM-10.8 Functional
TSM-10.9 Unit

TSM-11.1 **Test Sequential Multilearn 11 (filter)**
TSM-11.2 Ensure Multilearn.filter returns a filter
TSM-11.3

- Call filter
- Test if it is not equal to None

TSM-11.4 Inputs: QArray (dict), State (int)
TSM-11.5 Expected Output: Multilearn.filter = List
TSM-11.6 Normal
TSM-11.7 Blackbox
TSM-11.8 Functional
TSM-11.9 Unit

TSM-12.1 **Test Sequential Multilearn 12 (learn)**
TSM-12.2 Ensure Multilearn.learn successfully updates Q-values
TSM-12.3

- Call learn
- Test using getQ to see if Q-value was updated

TSM-12.4 Inputs: State (int), Action (int), Reward Tuple (from openai.gym), Next State (int)
TSM-12.5 Expected Output: Multilearn.getQ(State, Action) = Reward
TSM-12.6 Normal
TSM-12.7 Whitebox
TSM-12.8 Functional
TSM-12.9 Unit

TSM-13.1 **Test Sequential Multilearn 13 (train)**
TSM-13.2 Ensure Multilearn.train runs no more than specified iterations
TSM-13.3

- Call train
- Test if returned number of iterations is equal to defined max iterations (1)

TSM-13.4 Inputs: State (int), Environment (Class), Max Iterations (int, optional)
TSM-13.5 Expected Output: Iterations (int)
TSM-13.6 Normal
TSM-13.7 Blackbox
TSM-13.8 Functional
TSM-13.9 Unit

TSM-14.1 **Test Sequential Multilearn 14 (train_step)**
TSM-14.2 Ensure Multilearn.train_step returns next step and done bool
TSM-14.3

- Call train_step
- Check if new_state is not None and if done is False

TSM-14.4 Inputs: State (int), Environment (Class)
TSM-14.5 Expected Output: New State (int), Done (bool)
TSM-14.6 Normal
TSM-14.7 Blackbox
TSM-14.8 Functional

TSM-14.9 Unit