The Two Facets of the Exploration-Exploitation Dilemma

Kaifu Zhang, Wei Pan
Tsinghua University, Beijing, China
{zkf03, bow03}@mails.tsinghua.edu.cn

Abstract

This paper proposes an algorithm to better solve the exploration-exploitation dilemma faced by model-less reinforcement learning agents. The main contribution is twofold: (1) The two facets of the exploration-exploitation dilemma are distinguished: in some cases, the agent faces a non-stationary environment, therefore it needs to choose the best moment to explore in order to adapt to the changes; in some other cases, the agent faces a relatively large state-action space, and it therefore needs to choose the most promising subset of states/actions to explore. In this two-facet framework, we compared the relative advantage and limitations of two previously proposed algorithms in difference situations. (2) We unified these two algorithms to produce the new algorithm which works fairly well in all testing situations.

1. Introduction

Reinforcement learning (RL) is a widely studied learning algorithm that has found wide applications [1]. A reinforcement learner faces the problem of choosing an optimal *action* under each *state*. By estimating the value for each <*state*, *action*> pair through a trial-and-error process, the learner can learn the optimal actions without keeping a model of the environment. In a series of recent studies, the reinforcement learning algorithm has been extended to the multi-agent settings [6, 11].

The exploration-exploitation dilemma (See [1, 2, and 8] for reviews, E-E dilemma thereafter) is a famous challenge for reinforcement learners. As an agent starts to accumulate some knowledge about the environment, it seems plausible to exploit the current knowledge in order to improve the immediate payoffs. However, sticking to the currently best option may cause the agent to ignore some potentially better options that haven't been explored yet.

In this paper, we will distinguish the two facets of the E-E dilemma. In a series of earlier works [2, 4], the researchers investigated the E-E dilemma in reinforcement learning tasks concerning a large state-action space, for example, the *robot navigation task* (the test case used in [2, 4, 5] and a number of other studies). In such a task, a robot is expected to find an 'optimal' path from the start point to the destination in a large maze. Because there are many states (usually represented by x-y coordinate in the two dimensional maze), and the robot has at least four action choices at each states, it would be costly to uniformly explore every state-action combination. Therefore, it would be plausible to choose a *subset* of state-action combinations to explore. This was called the *efficient* exploration in [2, 4]. In their work, the robot navigation is used as the benchmark to test the algorithms. A large but *stationary* maze (that doesn't change during the entire learning episode) is used.

As the reinforcement learning algorithm has been extended to the multi-agent settings [6, 11], the assumption of a stationary environment usually no longer holds. An agent will have to adapt to the changing behaviors of its peers, who are also learning just like the agent itself does. In this situation, a new form of E-E dilemma emerged: it's plausible for an agent to constantly explore the environment in order to integrate the most recent changes into its knowledge of the world; however, such exploration should not be done excessively for consideration of the immediate payoffs. The *two-person strategic game* is one of the most often used benchmarks to simulate such a multi-agent, non-stationary environment (as is the case in [6, 8, and 9] and a number of other works). In such a task, a learning agent plays a repeated matrix game with another learning or non-learning opponent. In [8, 9], the authors tested whether their proposed learning algorithms can work well when the opponent holds strategies that change with time. In such a non-stationary setting, it would be plausible that the learning agent can choose the *right moment* to explore: the agent can adapt to a changing opponent in a timelier fashion, while keeping exploration to a minimal when the opponent doesn't change.

These two facets of the E-E dilemma have quite similar forms of statement, and are not always distinguished in previous studies. A number of algorithms have been claimed to solve the E-E dilemma quite well. In this work, however, we will demonstrate that algorithm that works well with one facet of the E-E dilemma may fail to successfully attack the other facet. In another word, a RL with those algorithms that works well in a non-stationary small scale problem may fail to efficiently explore the state-action space in a large scale problem, and vice versa.

Our current work will focus the *model-less* reinforcement learning. A modeless reinforcement learner keeps no model of the environment and other agents, the only observation it can make is the actual rewards received. In previous studies, there are a number of model-based reinforcement learning algorithms that have claimed to overcome the E-E dilemma [3, 4, 5, 7 and 10], such as the famous DYNA-Q algorithm and the Prioritized Sweeping algorithms. Some of them use several priorly-available world models (stereotypes [10]), and select from them by comparing the real observation with the stereotypes, in a Bayesian manner [7, 10]. In most cases, the availability of a world model (or model of other agents) requires at least some knowledge a priori, and can not always be satisfied. Therefore, we may want to balance the E-E dilemma with only the minimal information: the Q-values, observed rewards, and other model-independent information [1, 9].

This paper is structured as follows: in the second section, we will introduce two previously proposed algorithms, and design two robot navigation testing scenarios which we will use later. The robot will be expected to navigate in non-stationary mazes that change with time. In the third section, we will present motivational results to access the performances of the previous algorithms with our proposed testing scenarios. In the fourth section, we will unify the two previous algorithms to produce the integrated framework in the following section, and give the test results. We will end this paper with outlook on future works.

2. Background

2.1. The E-E Dilemma: Large State-Action Space

When the concept of the E-E dilemma was first coined, the two facets of E-E dilemma were not explicitly distinguished. In the famous review of Thrun [2], it was shown that the time cost of uniform exploration will rise exponentially with the scale of the problem (often represented by the number of states), and it's therefore important to conduct *efficient* exploration: choosing a subset from the state-action space to explore. This formation of the E-E dilemma has also been adopted in some subsequent works, such as [4].

A number of algorithms have been proposed to select the 'efficient subset'. Famous examples include the interval estimation technique [11], the counter-based exploration technique and the recency-based exploration technique [2, 4]. These modeless algorithms are usually concerned with assigning an *exploration bonus* to a specific subset of state-actions, in order to make them more favorable for exploration.

In the current study, we will consider one of these algorithms, namely the *Recency-Based Exploration* technique (RBE). This algorithm has been tested in [2, 4], where the agent is supposed to find an optimal path in a large, but stationary maze. The results have shown that agents with this exploration technique will be able to find the optimal path, rather than sticking to suboptimal solutions.

RBE is based on the standard Q-learning algorithm. Each state-action pair is given an exploration bonus in addition to the Q value:

$$Q'(s,a) = Q(s,a) + \alpha * R(s,a)$$

This Q' will be later used in the Boltzman action selection schema:

$$P(s,a) = e^{Q'(s,a)/T} \Big/ \sum_{a'} e^{Q'(s,a')/T}$$

The $R(s,a)$ depicts how much steps ago the action was chosen. Thus, for an action which has not been chosen for a long time, its probability to be chosen will increase since R and Q' is increased. Thus, actions with small Q values won't be 'starved': each action will be picked once in a while.

2.2. The E-E Dilemma: Non-Stationary Environment

How to keep learning in a non-stationary environment has been a difficult problem. It has been first attacked in the earlier works of Sutton and Moore [1, 3], and becomes a hot topic with the development of multi-agent reinforcement learning. The term of E-E dilemma has been used again to describe this scenario.

Some authors applied the algorithms introduced in the last section to this type of non-stationary environment. The RBE algorithm has been applied to dynamic environment by Dayan [5] and Zhu [8], and has been shown to have rather good performance.

A number of 'dedicated' algorithms have also been proposed. Instead of using action specific exploration bonus, these algorithms are concerned with choosing the *right moment* to explore. In the current study, we will consider the DAE (Detect and Explore) algorithm [9]. This algorithm bases itself on a simple Q-learner with the Boltzman action selection schema. By observing the latest rewards, the DAE algorithm performs a hypothesis testing procedure to exam whether the current estimation of Q-values will fall within a certain confidence interval of the memorized Q-values. If a change has been detected with some confidence, the learner will adaptively raise the exploration temperature. Therefore, the latest information can be integrated into its knowledge of the world.

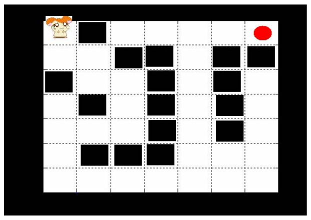
The DAE algorithm has been tested in a two-person gaming scenario. A learning agent with the DAE algorithm played the coordination game with a non-stationary opponent [9]. Such a testing scenario has also been used in [8], where the RBE algorithm is tested.

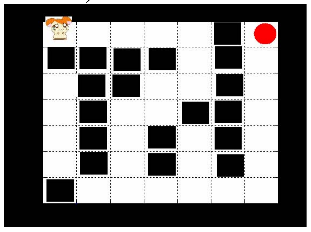
2.3. Non-Stationary Maze: Incorporating the Two Facets of the E-E Dilemma

In this section, we will introduce the testing scenarios we will use to test the abovementioned algorithms. We will consider a *robot navigation task in a non-stationary maze*: a scenario that has a large space-action space as well as temporal non-stationarity.

The maze is represented by a two dimensional matrix, including passable grids and blocks. The agent is expected to find an optimal path from the start point (the mice) to a destination point (the red point). However, during the learning episode, the maze can be abruptly changed to another form. Therefore, the agent has to adapt to such changes.

We will consider three 7-by-7 mazes, as shown in figure 1:


a) the maze M1


b) the maze M2

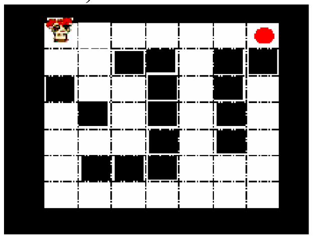

c) the maze M3

Figure 1. The mazes used to construct the testing scenarios

Based on the three maze configurations in figure 1, two testing scenarios are established here.

The first scenario (referred as S1) starts in the form of M1, after 80 learning epochs, the maze changes to M2. The second scenario (referred as S2) starts in the form of M1, after 80 learning epochs, the changes to M3. The two scenes are illustrated in figure 2.

A similar scenario has been proposed as the testing scenario in the insightful early work of Dayan [5]. His model based reinforcement learning algorithm DYNA-Q is tested. In the current research, we will use these benchmarks to test our modeless learning algorithms.

The two testing scenarios embrace the two facets of the E-E dilemma. In the first scenario, the structure of the maze undergoes a *major* change at the 80th epoch, and it will take some effort to gain the most up-to-date information. However, such changes are easy to detect: the blocks along the originally optimal path change. Therefore, the major challenge is: *how to facilitate exploration to adapt to the major change of environment in a timelier manner?* In the second scenario, only one block is removed at the 80th epoch, and there is relatively little to update. However, the change happens on an originally un-favored grid, and it would be hard to detect such change if the agent simply follows the optimal path. Therefore, the major challenge for S2 is: *how to facilitate exploration to detect the changes of an infrequently visited state,* when the state-action space is large?

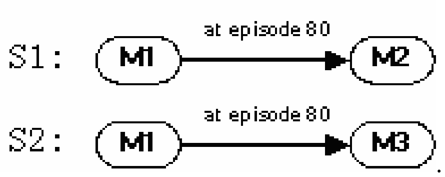


Figure 2. The two testing scenarios

3. Motivational Results and Discussion

In this section, we will discuss the performances of the abovementioned two algorithms in the two testing scenarios. All the learning algorithms will be based on the standard Q-learning algorithm (The performance of simple Q learning algorithm is also provided as a reference point.) The Q values will be updated as:

$$Q(s_t,a_t) = 0 + \gamma * V(s_{t+1}),\ \text{for successful state transition}$$
$$Q(s_t,a_t) = -2 + \gamma * V(s_t),\ \text{if bump into the wall}$$
$$where\ V(s_{t+1}) = \max_a Q(s_{t+1},a) \quad \gamma = 0.999$$

When the agent reaches the destination point, a reward of 10 will be received. A Boltzman action selection mechanism will be used. The Boltzman Temperature will be initialized to 9, and discounted by 0.9 at each time step

For RBE algorithm, each step, the recency values of unvisited state-action pairs will be increased by 1; all the recency values will be discounted by 0.8 each epoch.

The DAE algorithm has a sampling length of 3. The Boltzman temperature will be raised to 9 once a change has been detected.

The lower bounds of Boltzman Temperature for the DAE algorithm and the RBE algorithm are both 0.05.

The learning process goes for 200 epochs. In each epoch, the number of steps taken for the agent to reach the destination is recorded. If the agent fails to reach the destination in 5000 steps, the epoch terminates and the number 5000 is recorded.

3.3. Simple Q-learner, RBE and DAE in S1

The results of two simple Q learners are shown in figure 3:

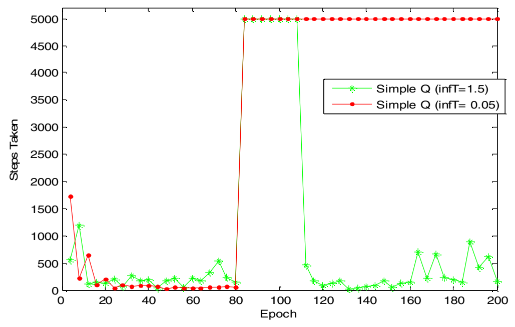


Figure 3. The E-E dilemma in S1: simple Q-learners

These two simple Q learners differ in their *lower bounds* of the exploration temperature (inf (T)). As can be seen, the learner with inf (T) = 0.05 cannot adapt to the changes, due to its relative smaller degree of exploration. Therefore, it cannot reach the goal after the 80th epoch (steps taken remains at 5000). The learner with inf (T) = 1.5 is able to learn the new maze, and it can find the new optimal path at the 120th epoch. However, due to its relatively high level of exploration, it cannot consistently choose the optimal path when the system doesn't change: it takes more steps to reach the goal during the epochs 20-80.

Next we will enhance the simple Q learner with the DAE and RBE mechanisms respectively, and access the performance of these algorithms. The results are shown in figure 4:

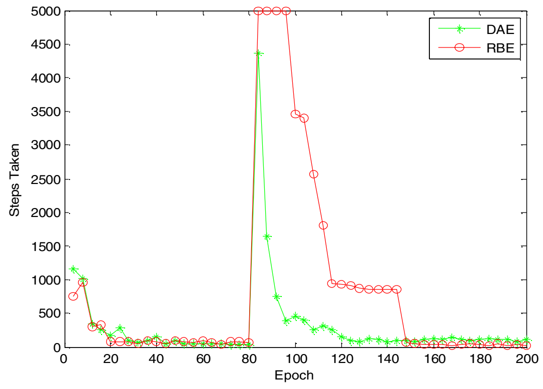


Figure 4. DAE and RBE in S1

As can be seen, both the DAE and RBE algorithm enable the learner to find the new optimal path after the maze changes. Moreover, they enable the learner to choose the (almost) optimal path consistently when the maze doesn't change (during epochs 20-80 and epochs 150-200). In a sense, both algorithms can help the learner to solve the E-E dilemma in such a non-stationary scenario.

However, the learner with the DAE mechanism can adapt to the changes in a timelier manner. The new optimal path has been found between epochs 120-140. And the learner with the RBE algorithms can only do this by the 150th epoch.

In this scenario, the structure of the maze undergoes a major change. The Q values for many states need to be reevaluated. Since the DAE algorithm can raise the exploration temperature, it enables the agent to traverse these modified states rather fast, and acquire the new Q values. Though the RBE algorithm steadily explores, it does the exploration with a smaller Boltzman temperature. Therefore, it will take the RBE learner longer to reevaluate all the modified Q values.

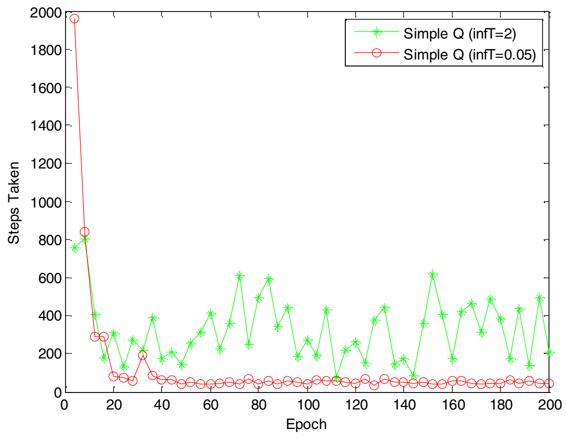
3.3. Simple Q-learner, RBE and DAE in S2

The performances of the abovementioned simple Q learners are shown in figure 5. To better describe the dynamics in this scenario, we introduce a new performance indicator: the *estimation error* of the Q values. The estimation error is defined by:

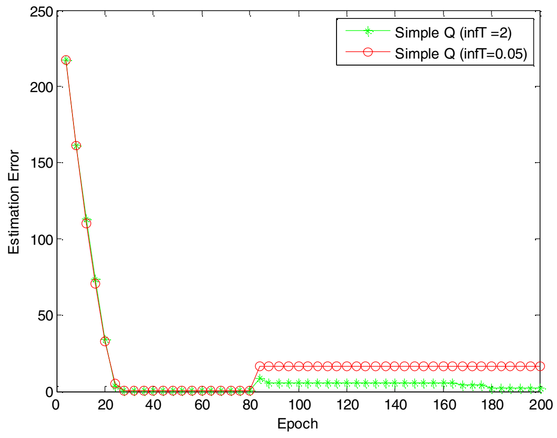
$$Error = \sum_{s \in \text{optimal path}} \left| V^{*}(s) - V(s) \right|$$

Where V* is the correct V values, and V is the estimated V values. As the learning process goes on, we expect this estimation error to decrease in value. And when the maze changes at the 80th epoch, we expect this value to undergo a sudden increase.

In the second scenario, the agent is still able choose the original path after the 80th epoch, and its *steps taken* to reach the goal won't go to 5000 after the 80th epoch, even if the new optimal path is not detected. Therefore, we need to introduce this estimation error as the performance indicator to help us better evaluates the performance of the learner.

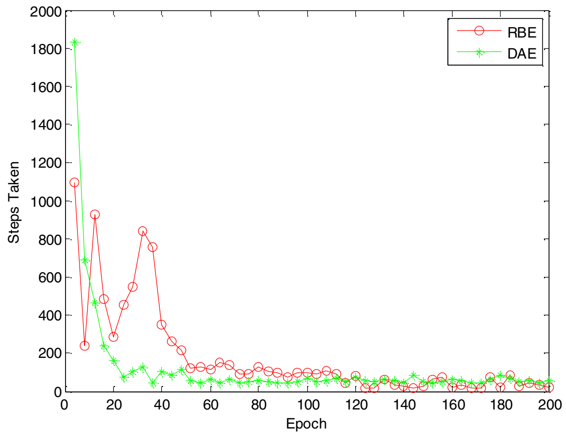


a) Steps taken to reach the goal

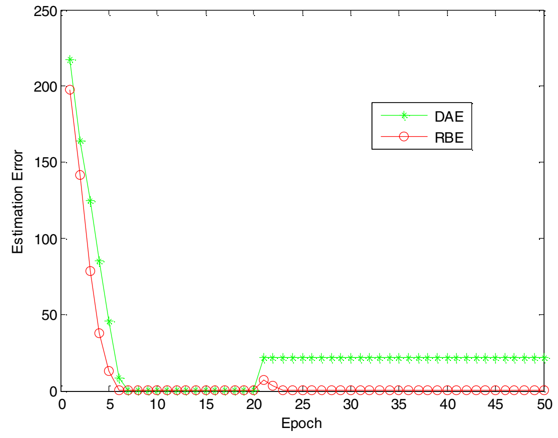


b) Estimation error

Figure 5. The E-E dilemma in S2: simple Q-learners



a) Steps taken to reach the goal



b) Estimation error

Figure 6. DAE and RBE in S2

The results give us the same message as in S1. The learner with inf (T) = 2 takes more steps to reach the goal throughout the learning epochs, due to its high level of exploration. And the learner with inf (T) = 0.05 seems to have dominantly better performance. However, a closer look at the estimation error implies that the learner with inf (T) = 0.05 actually cannot detect the change of the maze throughout the learning episode. Though the learner with the higher exploration temperature can detect this short cut, its high level of exploration prevents it from consistently choosing this optimal path.

Ideally, a learning algorithm should be able to detect the new optimal path -- a short cut. And it must be able to keep exploration to an acceptable level in order to *utilize* this short cut.

The performances of the DAE algorithm and the RBE algorithm are shown in figure 6. As shown in figure 6 b), the estimation error of the DAE algorithm remains high after the 80th epoch. The DAE algorithm has actually been reduced to a simple Q learner: the agent with the DAE algorithm cannot *detect* the new short cut, and it therefore cannot enhance its level of exploration. Meanwhile, the RBE algorithm is able to detect the new shortcut.

As shown in figure 6 a), during the first 80 epochs, the DAE agent takes slightly less steps to reach the goal than the RBE agent. This is because the DAE agent (with the same exploration temperature) explores less than the RBE agent when the environment doesn't change. However, after the 80th epoch, the RBE agent is able to reach the destination in fewer steps than the DAE agent: because the RBE agent has learnt the new maze, it can take the newly emerged shortcut instead of taking the originally optimal (now sub-optimal) route.

4. An Integrated Framework

4.1. The Two Facets of the E-E Dilemma and the Comparative Advantages of the Algorithms

In section three, we tested the performance of two previously proposed algorithms in two testing scenarios. The rationale is that each of the algorithms attacks one facet of the E-E dilemma.

The DAE algorithm is able to speed up exploration if the changes have been detected, and is therefore able to learn the new information in a timelier manner (S1). By using the recency-based exploration bonus, the RBE algorithm ensures that each state-action pair in the state-action space will be visited once in a while, and therefore ensuring the detection of changes in relatively less visited states even if the state-action space is large (S2). On the other hand, both algorithms have their relative disadvantages.

In this section, we will provide an integrated algorithm, which unifies the two abovementioned algorithms in a common framework. Our results will show that this integrated framework actually incorporate the comparative advantages of both the algorithms, and can work quite well in both the testing scenarios.

4.2. Algorithm description: RB-DAE

We will name our integrated framework the RB-DAE algorithm, which stands for *Recency-Based Detect and Explore*. Since both RBE algorithm and DAE algorithm have been based on the simple Q-learning algorithm with a Boltzman action selection schema, they can be put into a common framework without much modification.

An agent with the RB-DAE algorithm keeps two key additional memories apart from the Q-values: the recency values and the *cumulative errors*. The recency values work in the same way as in the original RBE exploration schema, as shown in section 2. The cumulative error for k observations for the state-action pair (s_t, a) is defined as:

$$error(s_t, a, k) = \sum_{i=1}^{k} r_k(s_t, a) + \gamma * V_k(s_{t+!}) - Q_k(s_t, a)$$

This cumulative error can be used to perform a hypothesis testing procedure to exam whether the *maximal likelihood estimation* of Q-value *derived from the latest k observations* is within a *confidence interval* of the corresponding Q-value stored in memory. The confidence interval can be derived by estimating the *variance* of the latest k observations. If the estimated Q value is not within the calculated confidence interval, we believe a change has been detected, and the exploration temperature will be consequently raised.

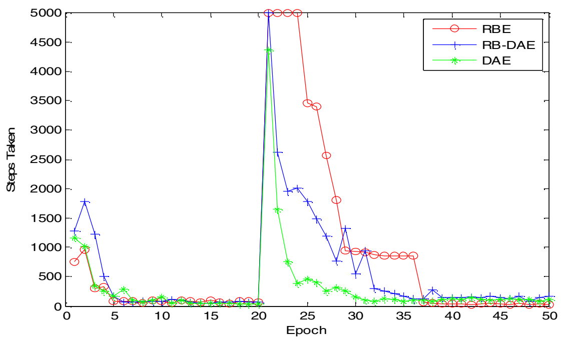
For more technical details about RBE and DAE, refer to [2] and [9] respectively. An algorithmic description is also given below in figure 7:

```
1: Initialize Q(s,a) to random values,
   Receny values R(s,a)=0
   Cumulative Error X(s,a)=0
2: Repeat for each episode:
3:    Set $s_t$ to the start point,
      initialize Boltzman temperature $T$,
4:    Variance $\mathrm{Var}^* = 0$
5:    Repeat for each step of the episode:
6:        calculate $Q'(s,a) = Q(s,a) + \alpha * R(s,a)$
7:        Choose $a_t$ using Boltzman policy derived from $Q'$,
          $R(s_t,a_t) = 0$, for all other (s,a) , $R(s,a) = R(s,a) + 1$
          decrease Boltzman temperature $T$
8:        Observe the reward r, Update Q value
           Update the cumulative error :
           $Y = r_k(s_t,a_t) + \gamma * V_k(s_{t+!}) - Q_k(s_t,a_t)$,
           $X(s_t,a_t) = X(s_t,a_t) + Y$,
           $\mathrm{Var}^*(s_t,a_t) = \mathrm{Var}^*(s_t,a_t) + Y^2$
11:       if the current episode is the nkth episode,
          then $\mathrm{Var}(s_t,a_t) = \frac{1}{n-1}\mathrm{Var}^*(s_t,a_t)$
           Update threshold value $\sqrt{n} * \Phi(\frac{\alpha}{2}) * \mathrm{Var}(s_t,a_t)$
12:       if the current exploration temperature T = inf (T),
           if $|X(s_t,a_t)| > [\sqrt{n} * \Phi(\frac{\alpha}{2}) * \mathrm{Var}(s_t,a_t)]^2$
           Reset learning rate and exploration temperature
13:        Perform state transition
13:    End of a step
14:End of an episode
```

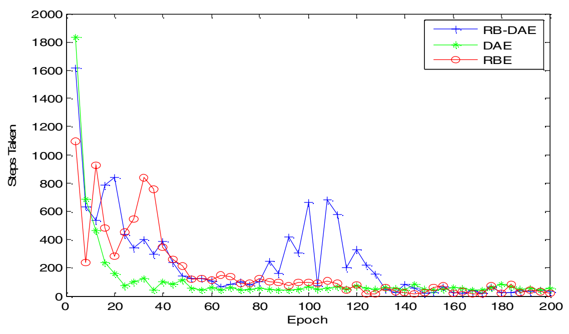
Figure 7. The RB-DAE algorithm framework

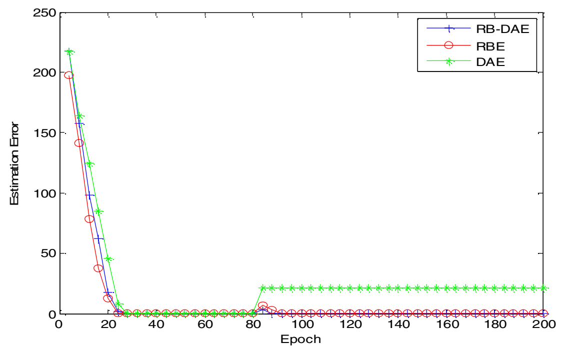
4.3. Results

The RB-DAE algorithm is tested with the two proposed testing scenarios. In this section, we will compare the performance of an agent using the RB-DAE algorithm with agents using the RBE or DAE algorithms alone. The results are shown in figure 8.



a) Testing RBE, RB-DAE and DAE in S1





b) Testing RBE, RB-DAE and DAE in S2

Figure 8. Experiments results

As can be seen, in the first testing scenario, the RB-DAE algorithm is able to adapt to the new maze structure in a timelier manner: slower than the DAE alone, but much faster than the RBE algorithm alone. Meanwhile, in the second testing scenario, the agent using the RB-DAE algorithm is able to find the optimal shortcut, though the increased temperature at the 80th epoch invoked a few epochs of high exploration (the 'steps taken' value goes up).

Actually, the integrated framework is in the middle of its two precedent algorithms. In the extreme cases, it cannot outperform both of its precedents. However, it's proved to be a more general algorithm that has fairly good performance – no matter which facet is responsible for the E-E dilemma in the current scenario.

5. Conclusions

In this paper, we differentiated the two facets of the Exploration-Exploitation Dilemma for reinforcement learning agents. We investigate two previously proposed algorithms which had been claimed to solve the E-E dilemma. Our results showed their comparative advantages in different scenarios where different facets are responsible for the E-E dilemma. We proposed an integrated framework that is able to work fairly well in both the testing scenarios.

Finally, we propose several future directions for our work: First, applying our algorithm to more complex scenarios. Plausible test cases include stochastic maze (rather than deterministic maze), or multi-agent settings. Real-world applications are usually characterized by these complex elements. Second, it will be plausible to compare our algorithm with the model-based approaches of reinforcement learning. Model-based algorithms [3, 4] have proved to work better when the E-E dilemma is a major challenge for the learner. We believe these future directions can shed more light on the theoretical significance of our proposed RB-DAE algorithm, and better polish it for engineering applications as well.

6. References

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning*, MIT Press, Cambridge MA, 1998.
[2] S. B. Thrun, "Efficient exploration in reinforcement learning", Technical Report CMU-CS-92-102, School of Computer Science, Carnegie Mellon University, 1992.
[3] A. W. Moore and C. G. Atkeson, "Prioritized sweeping: Reinforcement learning with less data and less time", Machine Learning, vol. 13, pp.103--130, Oct 1993.
[4] M.A. Wiering and J. Schmidhuber, "Efficient model-based exploration", Proc. 5th SAB, pp. 223-228, 1998.
[5] P. Dayan, T. J. Sejnowski, "Exploration Bonuses and Dual Control", Machine Learning, vol. 25(1), pp. 5-22, 1998.
[6] Y. Shoham, R. Powers, and T. Grenager, "Multi-agent reinforcement learning: a critical survey", Technical report, Computer Science Department, Stanford University, 2003.
[7] N. K. Jong and P. Stone, "Bayesian Models of Nonstationary Markov Decision Problems", IJCAI 2005 workshop on Planning and Learning in A Priori Unknown or Dynamic Domains, Aug 2005.
[8] S. Zhu, D.H Dana, "Overcoming Non-stationarity in Uncommunicative Learning", PhD Thesis, Rutgers University, 2002.
[9] K. Zhang, "Learn to Coordinate with Generic Non-Stationary Opponents", Proc. IEEE International Conference on Cognitive Informatics, 2006.
[10] D Denzinger, J Hamdan, "Improving Modeling of Other Agents using Tentative Stereotypes and Compactification of Observations", Proc. IEEE IAT'04, pp. 106-112, 2004.
[11] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning", Proc. ICML'94, Morgan Kaufmann Press, 1994, pp. 157–163.
[12] L. P. Kaelbling, *Learning in Embedded Systems*, MIT Press, Cambridge MA, 2006.