Fraud Detection in Banking Using Deep Reinforcement Learning

Abdelali El Bouchti†, Ahmed Chakroun†, Hassan Abbar† and Chafik Okar*
†National School of Business and Management, Hassan 1st University, Settat, Morocco
*School of Technology, Hassan 1st University, Berrechid, Morocco
Email: a.elbouchti@gmail.com, chakroun.phd@gmail.com, hassan.abbar@gmail.com and okar.chafik@uh1.ac.ma

***Abstract*—Deep learning and machine learning are hot topics in the financial services nowadays. They allow financial entities to define products and segment clients, efficiently manage risk and detect fraud in banks. The theory of Deep Reinforcement Learning (DRL) was originally motivated by animal learning of sequential behavior, but has been developed and extended in the field of machine learning as an approach to Markov decision processes. Recently, a number of financial risk analysis and fraud detection studies have suggested a relationship between reward-related activities in the brain and functions necessary for DRL. Regarding the history of DRL, we introduce in this article the theory of DRL and present two applications in banking. Then we will discuss possible implementations.**

***Index Terms*—Deep Learning; Reinforcement Learning; Risk Management; Banking.**

I. Introduction

Fraud detection is the recognition of symptoms of fraud where no prior suspicion or tendency to fraud exists. Examples include insurance fraud, credit card fraud and accounting fraud. Risk management in banking has been transformed over the past decade, largely in response to regulations that emerged from the global financial crisis and the fines levied in its wake. But important trends are afoot that suggest risk management will experience even more sweeping change in the next decade.

The change expected in the risk function's operating model illustrates the magnitude of what lies ahead. Today, about 50 percent of the function's staff are dedicated to risk-related operational processes such as credit administration, while 15 percent work in analytics. McKinsey research suggests that by 2025, these numbers will be closer to 25 and 40 percent, respectively.

Technological innovations continuously emerge, enabling new risk-management techniques and helping the risk function make better risk decisions at lower cost. Big data, machine learning, and crowdsourcing illustrate the potential impact.

Big data. Faster, cheaper computing power enables risk functions to use reams of structured and unstructured customer information to help them make better credit risk decisions, monitor portfolios for early evidence of problems, detect financial crime, and predict operational losses. An important question for banks is whether they can obtain regulatory and customer approval for models that use social data and online activity.

Machine learning. This method improves the accuracy of risk models by identifying complex, nonlinear patterns in large data sets. Every bit of new information is used to increase the predictive power of the model. Some banks that have used models enhanced in this way have achieved promising early results. Since they cannot be traditionally validated, however, self-learning models may not be approved for regulatory capital purposes. Nevertheless, their accuracy is compelling, and financial institutions will probably employ machine learning for other purposes.

Reinforcement learning (RL) is a natural extension of the R-W model to enable description of sequential learning by means of the conventional delta rule. Over the last decade in the field of neuroscience, the theory of RL has been extended to allow for explanations of not only lower-order neural activities related to rewards but also higher-order brain activities involved in decision making, although their evidence is still controversial. Reinforcement learning has also been recognized as an approach to Markov decision processes which were studied in the 1950s in the field of control theory, and which subsequently have become a major research topic in machine learning. In the field of artificial intelligence, RL is now attracting particular attention, because adaptability to and optimal control in complicated, dynamic and even unknown environments could be described by the RL theory. Paying attention to such a history of RL, in the first part of this article, we introduce the theory of RL and present two engineering applications. In the second part, we discuss possible implementations in the brain.

RL is a learning scheme to acquire an action sequence, which maximizes the sum of rewards accumulated over the future, and it has been reported that the reward (reinforcer) is also represented in the brain. In 1954, Olds and Milner first identified the brain region responsible for processing a reward, which is defined, in operant psychology, as a stimulus that increases the probability of operant or instrumental responses that precede or are contingent upon the stimulus. They applied direct electrical stimulation to the limbic system of rats when the rats happened to press a lever, and found that the rats became very willing to lever-press.

II. Theory of Deep Reinforcement Learning

A. The Rise of DRL

Many of the successes in DRL have been based on scaling up prior work in RL to high-dimensional problems. This is due to the learning of low-dimensional feature representations

978-1-5090-3989-0/17/$31.00

and the powerful function approximation properties of neural networks. By means of representation learning, DRL can deal efficiently with the curse of dimensionality, unlike tabular and traditional non-parametric methods [11]. For instance, convolutional neural networks (CNNs) can be used as components of RL agents, allowing them to learn directly from raw, highdimensional visual inputs. In general, DRL is based on training deep neural networks to approximate the optimal policy π^* , and/or the optimal value functions V^*, Q^* and A^* .

Following our review of RL, the next part of the survey is similarly partitioned into value function and policy search methods in DRL. In these sections, we will focus on stateof-the-art techniques, as well as the historical works they are built upon. The focus of the state-of-the-art techniques will be on those for which the state space is conveyed through visual inputs, e.g., images and video. To conclude, we will examine ongoing research areas and open challenges.

B. Value-based Reinforcement Learning

As in Fig. 1, we have a controller (or agent) and a system (or environment) to be controlled (or interacted with). At a discrete time t, the controller emits a control signal, often called an action, a_t, based on the system's state x_t. The controller's function from a state x_t to an action a_t is called a policy, π, and represented as $a_t = \pi(x_t)$ or $a_t \sim \pi(a_t/x_t)$ for a deterministic or stochastic policy, respectively. The system is assumed to be probabilistic and memory-less, i.e., Markov, so that the state transition probability for reaching a next statex' from a previous state x by an action a is given by $P(x'/x, a)$. A deterministic system is regarded as a special case of the probabilistic system. We assume that when the system receives an action a_t at a state x_t, the controller receives a scalar response $r_t = r(x_t, a_t)$, often called a reward, from the system, which represents the instantaneous goodness of the action at a_t the state x_t. We further define the return:

$$R_t = \sum_{s=0}^{\infty} \gamma^s r_{t+s} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + ... \quad (1)$$

which represents a discounted summation of rewards to be received in the future starting from the state x_t at time t. Here, the discount factor γ is a meta- parameter to avoid the divergence of the summation. Using the above setting and definitions, the problem here is to obtain the 'optimal' policyπ^* that maximizes the return from any state. This is an instance of a Markov decision process (MDP).

Although we deal in this section with deterministic policies for ease of explanation, a similar setting can be constructed for stochastic policies. Since the return (1) depends on the actual sequence of states and actions, we define its expectation with respect to possible sequences:

$$V^{\pi}(x) = E\left[R_t/x_t = x\right] = Q^{\pi}(x, \pi(x)) \quad (2)$$

$$Q^{\pi}(x, a) = E\left[R_t/x_t = x,\ a_t = a\right] = r(x, a) + \gamma \int_{x'} P(x'/x, a) V^{\pi}(x') dx' \quad (3)$$

The function $V^{\pi}(x)$ represents the expected return when starting from a state x_t and employing a fixed policy 7r, and is called the value function. Similarly, the function $Q^{\pi}(x, a)$ represents the expected return when selecting an action at a_t the current state x_t and employing a policy π in subsequent states, and is called the action-value function or Q-function. Using these definitions, the above MDP is equivalent to the problem of obtaining the optimal policy π^* that maximizes the value function $V^{\pi}(x)$ for every state x.

Maximization of the value function (2) with respect to policy π is a func- tional optimization, and seems difficult at first glance. Bellman's optimization principle tells us, however, that this optimization problem is equivalent to obtaining the optimal value function that satisfies at every state x the following optimal Bellman equation:

$$V^*(x) = \max_a Q^*(x, a) \quad (4)$$

$$Q^*(x, a) = r(x, a) + \gamma \int_{x'} P(x'/x, a) V^{\pi}(x') dx' \quad (5)$$

This is a function optimization problem for each state, and seemingly an easier problem than the functional optimization. Using the optimal value function, the optimal policy we seek is given by $\pi^*(x) = \arg\max_a Q^*(x, a)$. When the state and action spaces are discrete, the optimization problem (3) can be solved efficiently by dynamic programming (DP) methods. The famous RL algorithm called Q- learning is a stochastic version, i.e., the stochastic approximation based on actual sampling, of such DP methods.

By putting $a_t = \pi(x_t)$ into (2), we see

$$V^{\pi}(x_t) \approx r(x_t, a_t) - (V^{\pi}(x_t) - \gamma V^{\pi}(x_{t+1})) \quad (6)$$

is approximately satisfied. Since we have ignored probabilistic nature of the state transition in the right-hand-side, the equality holds probabilistically. The difference between the right-hand-side and left-hand-side:

$$\delta_t \equiv r(x_t, a_t) + \gamma V^{\pi}(x_{t+1}) \quad (7)$$

is called the temporal difference (TD) error, and its expectation should approach zero as learning of the value function proceeds. Based on the conventional delta (Widrow-Hoff) rule, such learning can be performed as to make the TD error small:

$$V^{\pi}(x_t) \ := V^{\pi}(x_t) + \alpha_c \delta_t \quad (8)$$

is the famous TD-learning If the system is stationary, the learning coefficient α_c is often reduced as learning proceeds, so as to realize the stochastic approxiimation of the value function (2). Note that this TD-learning is to obtain the value function for a certain policy π, which is not necessarily optimal, and therefore, the policy improvement should be performed outside of the TD-learning (6). If V^{π} for the policy π is approximated by TD-learning, then the policy can be improved by using a good approximation to the value function; this is the policy iteration method. A more intelligent method performs policy learning by updating a specific utility function

based on the TD error, concurrent with value learning. The actor-critic method, the first RL method, implemented such a concurrent learning scheme, and is apparently much more efficient than simple policy iteration.

C. Partially Observable MDP

In the previous section, we assumed that the state variable x is completely observable and obeys a Markov process. In reality, however, it is often the case that only part of the real state is observable with the remainder forming an associated unobservable state variable. Therefore, we assume for this case that the real state variable of the system $s\epsilon S$ obeys an underlying Markov process $P(s'/s,a)$ but that this process is intrinsically unobservable. Instead the controller (or agent) can access an observation variable $x\epsilon\chi$ according to an observation process $P(x/s)$. The reward condition is defined similarly to that for the MDP in section 2.1. A partially observable MDP (POMDP) is a problem of obtaining the optimal policy in such a partially observable environment. Since the controller cannot observe directly the system state s in a POMDP, an ideal controller has to estimate the system state s from the history of observations and to decide the action based on the estimation. When the controller estimates the system state s_t at time t based on the history of past observable variables (observation and action), $Ht - x_t, a_{t-1}, x_{t-1}, ..., a_1, x_1$, as a probability dis- tribution $P(s_t/H_t)$, the distribution is called a belief state and represented as $b_t\epsilon B$. A belief state is a probability distribution of states s, $b_t = P(s_t/H_t)$. If we know the Markov transition and observation processes, a belief state can be calculated according to the incremental Bayes formula:

$$b_{t+1}(s_{t+1}) = \frac{P(x_{t+1}/s_{t+1})\sum_{s_t\epsilon S} P(s_{t+1}/s_t,a_t)b_t(s_t)}{P(x_{t+1}/a_t,H_t)} \tag{9}$$

which represents the 'state transition' of belief states.

The objective of a POMDP controller is to obtain the optimal policy π^* that maximizes the value function for the system state, $V^\pi(s)$, for every state. Since the controller cannot directly observe the state s, however, this is almost impossible. Instead, we consider maximizing its expectation with respect to belief:

$$V^\pi(b_t) = \sum_{s_t\epsilon S} b_t(s_t)V^\pi(s_t) \tag{10}$$

for every belief state. The optimization problem for MDPs was a functional optimization. Since the value function for POMDPs is itself a functional, a function of belief states, the optimization problem here is more difficult, and a solution often needs some approximations. Moreover, we must calculate sum- mations over the system state $s\epsilon S$ as in equations (7) and (8), which becomes intractable as the number of possible states, $\mid s \mid$, increases, even in a discrete- state space. It should be noted that the denominator in equation (7) is more concretely described as

$$P(x_{t+1}/a_t,H_t) = \sum_{s_{t+1}\epsilon S} P(x_{t+1}/s_{t+1}) * \sum_{s_t\epsilon S} P(s_{t+1}/s_t,a_t)b_t(s_t) \tag{11}$$

and its calculation requires taking sums twice over states; this calculation may also be intractable.

Corresponding to the optimal Bellman equation (3) in MDPs, the optimal value function in POMDPs, $V^*(b)$, satisfies the Bellman equation for belief states:

$$V^*(b) = \max_a Q^*(b,a) \tag{12}$$

$$Q^*(b,a) = r(b,a) + \gamma \int_{b'} P(b'/b,a)V^\pi(b')db' \tag{13}$$

Because a belief state should be a sufficient statistic which adequately represents the information contained in the history of past observations, the stochastic process on belief states should be Markov, and hence $P(b'/b,a)$ is stationary. Since the process on belief states becomes an MDP, this POMDP formulation is called a belief-state MDP.

The objective in a belief-state MDP is to maximize the functional $Q^*(b,a)$ with respect to the variable a for each belief state b, as can be seen in (10), but there are two difficulties:

1) The value function $V^*(b)$ and Q-function $Q^*(b,a)$are both functionals, and it is difficult to represent them.
2) The update of belief states requires application of the incremental Bayes formula, equation (7), but this calculation involves summing over states, which is difficult.

Some methods, for example, the witness algorithm, have been proposed for solution of belief-state MDPs based on exact belief states, but they can be applied only to problems whose state space is relatively small. Recently, various methods to approximate the value function of belief states or to approximately represent belief states have been studied extensively.

D. Policy-based Reinforcement Learning

Value-based RL methods, such as Q-learning and TD-learning, were very popular several years ago, but some of those methods led to unstable learn- ing, especially when function approximators were used, partly because a small change in the value function may result in a significant change in the optimal policy. On the other hand, policy-based methods like Williams' REINFORCE attempted to improve policy based on observed sequences of rewards. Since the methods of the latter type do not use value functions, their learning, although stable, proceeded extremely slowly. Very recently, however, some policy-based methods, which incorporate value learning have been proposed which result in both stable and efficient learning.

Let a policy π be stochastic and parameterized by a parameter θ as π^θ By assuming the Markov system has a stationary distribution of states, $D^\theta(x)$, the stationary distribution of

state-action pairs is given by $D^\theta(x,a) = D^\theta(x)\pi^\theta(a/x)$. The expectation of reward with respect to the stationary distribution is then given by

$$\rho(\theta) \equiv \int_{x,a} r(x,a) D^\theta(x,a) dadx \equiv < r(x,a) >_\theta$$

In the current and the next sections, an MDP is defined so as to obtain the parameter θ that maximizes the expected reward $\rho(\theta)$. Although this problem is slightly different from the maximization problem of reward accumulation in section 2.1, a similar discussion to the following may be made if we use the reward accumulation as the objective function.

Let the return be the sum of the differences between the reward and the expected reward, from a state x_t at time t over all future states:

$$R_t \equiv \sum_{s=0}^{\infty} (r_{t+s} - \rho(\theta))$$

With this definition, the value function under a policy π^θ is given by

$$V^\theta(x) = E\left[R_t / x_t = x\right] = \int_a \pi^\theta(a/x) Q^\theta(x,a) da \qquad (14)$$

$$Q^\theta(x,a) = E\left[R_t / x_t = x, a_t = a\right] = (r(x,a) - \rho(\theta)) + \int_{x'} P(x'/x,a) V^\theta(x') dx' \qquad (15)$$

Note that the value function V^θ and the Q-function Q^θ here are different by $-\rho(\theta)$ from those in section 2.1.

In the above-defined MDP, the differential of the expected reward with respect to the parameter is given by

$$\frac{\partial \rho(\theta)}{\partial \theta} = \int_{x,a} D^\theta(x,a) Q^\theta(x,a) \psi^\theta(x,a) dadx = < Q^\theta(x,a) \psi^\theta(x,a) >_\theta \qquad (16)$$

where $\psi^\theta(x,a)$ is a compatible function vector defined as

$$\psi^\theta(x,a) = \frac{\partial \log \pi^\theta(a/x)}{\partial \theta} = \frac{1}{\pi^\theta(a/x)} \frac{\partial \pi^\theta(a/x)}{\partial \theta} \qquad (17)$$

Since θ is usually a parameter vector, $\psi^\theta \equiv \left\{\psi_i^\theta\right\}$ is a set (vector) of compatible function elements. Equation (12) is called the policy gradient theorem (for a brief proof, see Appendix A), and it states that if the correlation between the Q- function and the compatible function, given by the right-hand-side of equation (12), can be estimated, the policy can be (locally) optimized by modifying it in that direction. This is a policy gradient method, which is different in concept from the value-based RL methods presented in section 2.1.

E. Policy-gradient Actor-critic Learning

In this section, an elegant policy gradient learning method, policy-gradient actor-critic learning, is introduced. Let a policy π have an n-dimensional parameter 0. If we define an inner product $\ll ., . \gg_\theta$ of two functions, $q_1(x,a)$ and $q_2(x,a)$, by

$$\ll q_1(x,a), q_2(x,a) \gg_\theta = \int_{x,a} D^\theta(x,a) q_1(x,a) q_2(x,a) dadx$$

and a set of basis functions, $\Psi = \left\{\psi_i^\theta / i = 1, ..., n\right\}$, given by equation (13) automatically from the policy's parameterization, then $Q^\theta(x,a)$ can be projected onto a function space spanned by Ψ as

$$\Pi_\Psi Q^\theta(x,a) = \arg \min_{\hat{Q} \epsilon \{\Psi\}} \left\| Q^\theta - \hat{Q} \right\|_\theta = w^T \psi^\theta(x,a) \equiv Q^w(x,a) \qquad (18)$$

where $\|.\|_\theta$ denotes the norm induced by the inner product above, and T is the transpose. Since $Q^w(x,a)$ is in the space spanned by Ψ, it is represented as a linear combination of the basis functions. From equations (12) and (14), we have

$$\frac{\partial \rho(\theta)}{\partial \theta} = < Q^\theta \psi^\theta >_\theta = < (\Pi_\Psi Q^\theta) \psi^\theta >_\theta = < Q^w \psi^\theta >_\theta$$

This leads to the fact that the policy gradient is not sensitive even if the Q- function is projected by Π_Ψ,, indicating that the policy gradient can be estimated without any bias if we approximate the Q-function as a linear combination of the basis functions in Ψ. As an instance of TD-learning (equation (6)), then, parameter w of the Q-function (or critic) can be updated by

$$\triangle w = \alpha_c \delta_t \psi^\theta(x_t, a_t)$$

where δ_t it is a TD error:

$$\delta_t \equiv r(x_t, a_t) - \hat{\rho} - (Q^w(x_t, a_t) - Q^w(x_{t+1}, a_{t+1}))$$

This TD error defined for the Q-function is different from that in equation (5). The TD-learning for the Q-function is sometimes called SARSA. The parameter for the policy (or actor) can be adjusted so as to approximate equation (12) from actual samples:

$$\triangle \theta = \alpha_a Q^w(x_{t+1}, a_{t+1}) \psi^\theta(x_{t+1}, a_{t+1}) \qquad (19)$$

This is policy-gradient actor-critic learning. Note that the update in equation (15) can be accelerated by using the eligibility trace, which encourages the prop- agation of rewards along sample sequences. It has been proved that policy-gradient actor-critic learning converges under certain conditions such as employ- ing appropriate scheduling of the learning coefficients, α_c and α_a. The approximate Q-function in equation (14), $Q^w(x,a)$, is n-dimensional. However, since the dimensionality of the real Q-function (11b) is usually high, reflecting the non- linearity of the target system, the convergence is very slow due to the variance stemming from the model difference in (14) if the dimensionality of the approxi- mate Q-function is low. To avoid such variance, then, we may construct a richer set of basis functions, $\Phi \equiv \{\phi_i / i = 1, ..., m, \phi_i = \psi_i, i = 1, ..., n, m > n\}$ so that the Q-function is represented in that function space, i.e., $Q^{\tilde{w}}(x,a) = \tilde{w}^T \phi(x,a)$.

In the meantime, the linear weight vector w in equation (14) is given as a least mean square solution:

$$w = \arg \min_w \left\| Q^\theta(x,a) - Q^w(x,a) \right\|_\theta$$

leading to

$$w = < Q^\theta(x,a) \psi^\theta(x,a) >_\theta < \psi^\theta(x,a) \psi^\theta(x,a) >_\theta^{-1} \qquad (20)$$

Here, the numerator is identical to the policy gradient, equation (12), and the denominator is a positive semi-definite matrix, so w is in the same direction as the policy gradient. Therefore, the actor's parameter θ can be modified by making the modification proportional to w, $\triangle\theta = \alpha_a w$, instead of equation (15). Because the denominator is formally the Fisher information matrix of the parametric stochastic policy π^θ:

$$\mathcal{F}_{i,j} = \left\langle \frac{\partial \log \pi^\theta(a/x)}{\partial \theta_i} \frac{\partial \log \pi^\theta(a/x)}{\partial \theta_j} \right\rangle$$

equation (16) is in the same form as the natural gradient; this actor-critic learning is then called natural actor-critic. By using the second-order differential, learning may be expected to be accelerated, especially in the region where the policy gradient is small.

III. Challenges in Deep Reinforcement Learning

It is instructive to emphasise some challenges faced in RL:

- The optimal policy must be inferred by trial-and-error interaction with the environment. The only learning signal the agent receives is the reward.
- The observations of the agent depend on its actions and can contain strong temporal correlations.
- Agents must deal with long-range time dependencies: Often the consequences of an action only materialise after many transitions of the environment. This is known as the (temporal) credit assignment problem [16].

We will illustrate these challenges in the context of an indoor robotic visual navigation task: if the goal location is specified, we may be able to estimate the distance remaining (and use it as a reward signal), but it is unlikely that we will know exactly what series of actions the robot needs to take to reach the goal. As the robot must choose where to go as it navigates the building, its decisions influence which rooms it sees and, hence, the statistics of the visual sequence captured. Finally, after navigating several junctions, the robot may find itself in a dead end. There are a range of problems, from learning the consequences of actions, to balancing exploration versus exploitation, but ultimately these can all be addressed formally within the framework of RL.

IV. Risk Management / Fraud Detection

The customer and the banks while dealing with each other will always try to cover the risk factor. To identify, quantify and control the risk factor is always an area of concern for every business organization. In commercial lending, risk assessment is usually an attempt to quantify the risk of loss to the lender while making a particular lending decision. Data mining technique helps to distinguish borrowers who repay loans promptly from those who don't [8]. It also helps to predict when the borrower is at fault, whether providing loan to a particular customer will result in bad loans etc. Such techniques come under the category of credit risk, where we wish to check the behavior of the prospective customers [10]. Bank executives by using Data mining technique can also analyze the behavior and reliability of the customers while selling credit cards too.

While dealing with banks, the customers and the banks have the chances of falling an easy prey to the frauds. So both the parties wish to be secure while dealing with each other. The data mining techniques can help them to detect and hence prevent frauds. The data mining techniques will help the organization to focus on the ways and means of analyzing the customer data in order to identify the patterns that can lead to frauds [4,8].

V. Conclusion

The most intriguing point of Deep Reinforcement Learning (DRL), regardless of whether it is model-free or model-based, is to realize prediction of future reward in terms of Bellman-like self-consistent equations. Whether this prediction is really implemented in the brain or not is still controversial. Such a self-referential structure, if exists, has many implications; DRL may be a basis not only for decision making in a complicated world containing multiple agents, but also for recognition of the 'self' in such a world. The theory of DRL will thus grow by incorporating concepts and knowledge from various research fields, such as machine learning, control theory and fraud detection in banking.

References

[1] Badre, D., Frank, M.J., 2012. Mechanisms of hierarchical reinforcement learning in cortico-striatal circuits 2: evidence from fMRI. Cereb. Cortex. 22, 527536.

[2] Barto, A.G., Mahadevan, S., 2003. Recent advances in hierarchical reinforcement learning. Discrete Event Dyn. Syst. 13, 341379.

[3] Barto, A.G., Sutton, R.S., Brouwer, P.S., 1981. Associative search networka reinforcement learning associative memory. Biol. Cybern. 40 (3), 201211.

[4] Behrens, T., Woolrich, M., Walton, M., Rushworth, M., 2007. Learning the value of information in an uncertain world. Nat. Neurosci. 10, 12141221.

[5] Bellman, R., 1957. Dynamic Programming. Princeton University Press, Princeton.

[6] Bogacz, R., McClure, S.M., Li, J., Cohen, J.D., Montague, P.R., 2007. Short-term memory traces for action bias in human reinforcement learning. Brain Res. 1153, 111121.

[7] Botvinick, M.M., Niv, Y., Barto, A.C., 2009. Hierarchically organized behavior and its neural foundations: a reinforcement learning perspective. Cognition. 113, 262280.

[8] Courville, A.C., Daw, N.D., Gordon, G.J., Touretzky, D.S., 2003. Model uncertainty in classical conditioning. Adv. Neural Inf. Process. Syst. 16, 977984.

[9] Dayan, P., Daw, N.D., 2008. Decision theory, reinforcement learning, and the brain. Cogn. Affect. Behav. Neurosci. 8, 429453.

[10] Dayan, P., Long, T., 1998. Statistical models of conditioning. Adv. Neural Inf. Process. Syst.117123.

[11] Dearden R., Friedman N., Russell S., 1998. Bayesian Q-learning. In: John Wiley & Sons Ltd, pp. 761768.

[12] Frank, M.J., Badre, D., 2012. Mechanisms of hierarchical reinforcement learning in corticostriatal circuits 1: computational analysis. Cereb. Cortex. 22, 509526.

[13] Gershman, S., Pesaran, B., Daw, N., 2009. Human reinforcement learning subdivides structured action spaces by learning effectorspecific values. J. Neurosci. 29, 1352413531.

[14] Kalman, R.E., 1960. A new approach to linear filtering and prediction problems. J. Basic Eng. 82, 3545.

[15] Killeen, P.R., 2011. Models of trace decay, eligibility for reinforcement, and delay of reinforcement gradients, from exponential to hyperboloid. Behav. Process. 87 (1), 5763.

[16] Madan Lal Bhasin, “Data Mining:A Competitive Tool in the Banking and Retail Industries”, The Chartered Accountant October ,2006
[17] Richard S Sutton and Andrew G Barto. Reinforcement Learning: An Introduction. MIT Press, 1998.
[18] Ribas-Fernandes, J.J.F., Solway, A., Diuk, C., McGuire, J.T., Barto, A.G., Niv, Y., et al., 2011. A neural signature of hierarchical reinforcement learning. Neuron. 71, 370379.
[19] Russell, S., Zimdars, A.L., 2003. Q-decomposition for reinforcement learning agents. Proceedings of ICML-03.
[20] Sutton, R.S., Precup, D., Singh, S., 1999. Between MDPs and semiMDPs: A framework for temporal abstraction in reinforcement learning. Artif. Intell. 112, 181211.
[21] von Neumann, J.V., Morgenstern, O., 1947. Theory of Games and Economic Behavior, second ed. Princeton University Press, Princeton, NJ.
[22] Wilson, R.C., Niv, Y., 2011. Inferring relevance in a changing world. Front. Human Neurosci. 5, 189.
[23] Yuille, A., Kersten, D., 2006. Vision as Bayesian inference: analysis by synthesis? Trends Cogn. Sci. 10, 301308.