Available online at http://www.ijabbr.com

International journal of Advanced Biological and Biomedical Research

Volume 2, Issue 5, 2014: 1398-1416

Reinforcement Learning in Neural Networks: a Survey

Ahmad Ghanbari[1], Yasaman Vaghei*[2], Sayyed Mohammad Reza Sayyed Noorani[2]

[1]Faculty of Mechanical Engineering, and Mechatronics Research Laboratory, School of Engineering Emerging Technologies, University of Tabriz, Tabriz, Iran
[2]Mechatronics Research Laboratory, School of Engineering Emerging Technologies, University of Tabriz, Tabriz, Iran

ABSTRACT

In recent years, researches on reinforcement learning (RL) have focused on bridging the gap between adaptive optimal control and bio-inspired learning techniques. Neural network reinforcement learning (NNRL) is among the most popular algorithms in the RL framework. The advantage of using neural networks enables the RL to search for optimal policies more efficiently in several real-life applications. Although many surveys investigated general RL, no survey is specifically dedicated to the combination of artificial neural networks and RL. This paper therefore describes the state of the art of NNRL algorithms, with a focus on robotics applications. In this paper, a comprehensive survey is started with a discussion on the concepts of RL. Then, a review of several different NNRL algorithms is presented. Afterwards, the performances of different NNRL algorithms are evaluated and compared in learning prediction and learning control tasks from an empirical aspect and the paper concludes with a discussion on open issues.

Key words: Artificial Neural Networks, Reinforcement Learning, Adaptive Networks, Survey.

1. INTRODUCTION

Reinforcement learning (RL) is a widely used machine learning framework in which an agent tries to optimize its behavior during its interaction with its initially unknown environment to solve sequential decision problems that can be modeled as Markov Decision Processes (MDPs) (Lewis et al., 2009). In RL, the learning agent tries to maximize a scalar evaluation (reward or control cost) and modify the policies through actions. Hence, RL is an efficient framework for solving complex learning control problems (Sutton, 1998). In recent years, RL has been studied in several different fields such as neural networks (NN), operations research, and control theory (Busoniu et al., 2010; Chen et al., 2011). Moreover, RL can be seen as adaptive optimal control (Sutton et al., 1992). Studies on human brain reveal that RL is a major human learning mechanism in basal ganglia (Werbos, 2009). The main goal of the researches is to develop NNRL agents, which are able to survive and optimize the system's behavior during their interaction with the environment. According to the importance of function approximation and generalization methods in RL, they have been a key research interest recently (Ernst et al., 2005; Barto, 2004). The main components of the RL algorithm are the state signal, action signal and the reward signal, which are demonstrated in Fig.1.

Corresponding Author E-mail: y.vaghei91@ms.tabrizu.ac.ir

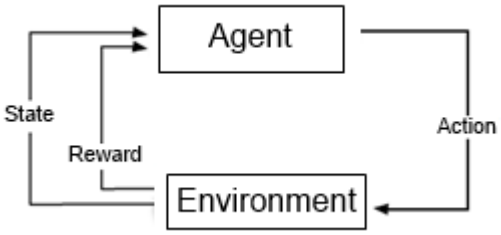

Figure 1: Reinforcement Learning (Sutton, 1998)

There are three main elements in RL schemes (Sutton, 1998):
a) The agent, which predicts the future reward in order to increase the reward's value with value functions. In many applications with or without having the model of the environment, value function implementation is preferred.
b) The policy, one of the major elements in RL, which determines the behavior of the agent over the operation time and it may be stochastic or deterministic.
c) The reward function, which demonstrates each particular time action reward value. The reward (total reward) is generally defined as the sum of the rewards over time. If the action leads to the goal, the reward will increase. Conversely, the reward will be decreased if an action distracts the agent. Immediate or delayed rewards may be employed by assuming a discount factor for the rewards over time.
Today, adaptive control has been widely used in nonlinear parameterized systems (Chen et al., 2011). NNs have obtained many applications and have attracted a great deal of attention in RL due to their ability to deal with complex and nonlinear systems. Recently, adaptive control algorithms for robots manipulators have been studied by (Wang, 2010; Adams, 2008). Sun et al. (1998) studied a stable NN-based adaptive control approach. Moreover, discrete-time adaptive tracking control for robotic manipulators has been studied in (Li HX, 2007). Also, many RL-based control systems have been applied to robotics. For example, an instance-based learning algorithm was applied to a real robot in a corridor-following task in (Smart, 2000) obtaining good results. In (Carreras et al., 2003), an underwater robot learns simple behaviors using a modified Q-Learning algorithm. Value function methodologies also demonstrated satisfying results in many applications but when dealing with high dimensional domains, the convergence is not guaranteed due to the lack of generalization among continuous variables (Sutton, 1998).
Despite the fact that many surveys exist on the RL algorithms, the growing popularity of NNRL algorithms motivated us to survey these algorithms on their own right. In this paper, we will mainly focus on recent developments in RL algorithms with neural network function approximation. The applications of recent NNRL algorithms will also be analyzed and the advantages and disadvantages of different algorithms will be compared. Thus, the paper is generally structured as follows. Section 2 introduces the RL concepts. NNRL methods and their applications are introduced in Section 3. In the end, conclusions and future outlook are provided in Section 4.

2. Reinforcement Learning concepts

Typically, in machine learning, the environment is formulated as a Markov decision processes (MDPs). According to (Sutton, 1998; Busoniu et al., 2010; Peters, 2008), a MDP consists of a set of states, S, and a set of actions denoted by a. Associated with each action, there is a state transition matrix $P(a)$

and a reward function $r: S \times A \rightarrow R$, where $r(x, a)$ is the expected reward for doing the action a in state x. A policy is a mapping $\pi: S \rightarrow A$ from states to actions. This policy is both stationary and deterministic. The RL's goal is to find policy π, which maximizes the expected value of a specified function, f, of the immediate obtained rewards while following the policy π. This expected value is defined in Eq.1.

$$J(\pi) = E\{f(r_1, r_2, \ldots)|\pi\} \quad (1)$$

Generally, this function is either the discounted sum of rewards or the average reward received (Bertsekas, 2007). MDP can be solved by RL without explicit specification of the transition probabilities. Decisions are made at discrete time intervals in discrete-time MDPs. However, decisions can be made at any time the decision maker chooses for continuous-time MDPs. Continuous-time MDP can better model the decision making process for a system that has continuous dynamics in contrast to discrete-time MDP (Vamvoudakis, 2010; Hanselmann et al., 2007). Learning prediction and learning control are the two main tasks in RL. As pointed out by Sutton and Barto (Sutton, 1998; Sutton, 1998[a]), the three basic methods to deal with the RL problems are dynamic programming (DP) [26, 27, 30], Monte Carlo (MC) Methods (Sutton, 1998) and temporal difference (TD) methods (Brartke, 1996; Maei et al., 2010). DP has been well developed even in 1994 (Bradtke, 1994). A DP algorithm implicitly begins at the goal state and propagates the information backwards. However, requiring a complete model of the environment lead the researches to do further studies on other methods. Moreover, due to the more complexity and the computation limitation, these algorithms have slow convergence. MC methods use experience to only approximate the value function, encountered during an episode of problem solving. Although these methods do not require a model of the environment, they are difficult to be implemented online due to the fact that the value function is evaluated at the end of each time step (Vlassis et al., 2009). In addition to this, in MC methods a severe penalty may incur and its negative effect will be back propagated to the start point. TD methods enable us to combine the advantages of DP and MC methods at the same time. In these methods, the value function is updated at each time step and they are completely model free. Q-learning, state-action-state-reward (SARSA) (Rummery, 1994), actor-only methods (Konda, 2003; Williams, 1992), critic-only methods (Melo et al., 2008; Gordon, 1995), and actor-critic methods are five popular methods of RL. Q-learning is an off-policy method invented by Watkins in which the value function is updated separately from the current policy. $Q(s_t, a_t)$ Denotes the value of choosing a_t while the environment is in state s_t. The complete equation of value update in Q-learning is as follows.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (2)$$

In Eq. 2, γ is the discount factor and α is the step size or learning rate. Applying Q-learning techniques that use a propositional Q-value generalization to tasks, which have an unknown number of objects or are largely defined by the relations between available objects, may be difficult. In SARSA algorithm, the on-policy TD method, the value function is estimated for the current policy. It is an algorithm for learning a Markov Decision Process (MDP), presented in Eq.3.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (3)$$

Particularly, actor-only methods deal with a parameterized family of policies which has the benefit of generating a spectrum of continuous actions; however, the implemented optimization methods (policy gradient methods) have the disadvantage of high variance in the estimates of the gradient, which results in slow learning (Konda, 2003).

Critic-only methods use temporal difference learning and have a lower variance in the expected returns estimates (Boyan, 2002). These methods usually work with discrete action space. As a result, this approach is not able to find the true optimum (Melo et al., 2008; Gordon, 1995).

The actor–critic algorithms have been shown to be more effective than value function approximation (VFA), and policy search in online learning control tasks with continuous spaces (Vamvoudakis, 2010). Actor-critic methods provide the advantages of actor-only and critic-only methods by computing continuous actions without the need for optimization procedures on a value function and supplying the actor with low-variance knowledge of the performance, at the same time. This leads to a faster learning process (Berenji, 2003).The convergence properties of actor-critic methods usually are more satisfying than critic-only methods (Konda, 2003). Fig. 2 demonstrates the overall scheme of the actor-critic RL.

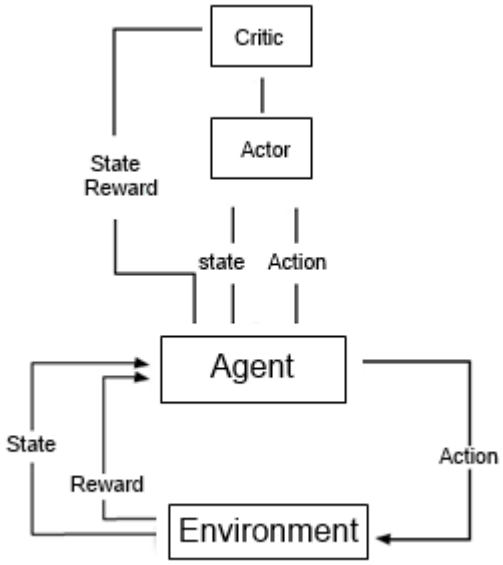

Figure 2: Overall scheme of actor-critic reinforcement learning

A recent study by Tang et al. has been done on trajectory tracking of an n-link robot manipulator. The proposed controller consists of two neural networks as the actor and critic with a satisfying tracking error. However, the effect of input nonlinearities, such as dead-zone input and hysteresis has not been considered in this paper (Tang, 2013). Farkaš et al.(2012) investigated a two-layer perceptron, actor-

critic architecture, and an echo-state neural-network based modules that were trained in different ways on the iCub robot action learning (point, touch, and push). They found that the trained model is able to generalize well in case of action-target combination with randomized initial arm positions and also adapt its behavior to sudden changes during motor execution (Farkaš et al., 2012). In another recent study, Bhasin et al. combined robust integral of the sign of the error with the actor-critic architecture to guarantee the asymptotic tracking of the nonlinear system with faster convergence. However, this controller does not ensure optimally (Bhasin et al., 2011). In addition, some of the recent studies were devoted to central pattern generators (CPGs) RL. Nakamura et al. used the CPG-actor-critic method for RL of a biped robot and demonstrated that the proposed method enabled the robot to walk stably and also adapt to the environment (Nakamura et al., 2007).

3. Neural Network Reinforcement Learning Methods

Almost all RL algorithms involve value function estimation. The most basic way to build this value function is updating a table, which contains a value for each state-action pair. However, this method is not practical for large scale problems. Task with a very large number of states require the generalization capabilities of function approximators to deal with. Hence, NNs can be used to reduce the complexity. Various studies have been done in this field. Recently, Miljkovic' et al. used Q-learning and SARSA coupled with neural networks for visual control of a robot manipulator. The stability and effectiveness of the method were investigated and it was shown that the proposed method is able to provide high accuracy (Miljkovic', 2013). Also, Stingu and Lewis used a radial basis function neural network based on the work of (Sarangapani, 2006) to model the Q-function and control policy in a quadrotor unmanned aerial vehicle (UAV) and obtained good results (Stingu, 2010).

3.1. Feed Forward Neural Network Reinforcement Learning (FFNNRL)

Feed forward neural networks (FFNN) are a particular case of function approximates that can be combined with RL algorithms. Tesauro's backgammon player is the most spectacular success of this technique (Tesauro, 1995), which could reach the level of human masters after months of self-play. Basically, multi-layer feed forward neural networks (multi-layer perception (MLP)) consist of an input, a number hidden layers and an output layer. The network is trained by comparing its outputs with the given target values and adapting the weights in order to minimize the error (White, 1989). The experiments reveal that a FFNN is able to approximate a value function more accurately than a normalized Gaussian network with many less weights, especially in high dimensional state space. This enables the FFNN to deal with some significantly complex problems (Johansson et al., 1991). However, a disadvantage of FFNNs is their lack of recurrence, which limits the incorporation of inter-temporal dependencies. Moreover, these neural networks are able to perform a pattern matching from inputs to outputs, which doubts their application to (non-Markovian) dynamical systems (Bebis, 1994).

3.2. Recurrent Neural Network Reinforcement Learning (RCNNRL)

The identification of the problem's dynamics and the optimal control problem are the RCNNRL's two different tasks to fulfill. In these networks, the back propagated error from the reward clusters updates the weights of the control network. This follows the idea that the system tends to learn a policy that maximizes the sum of expected future rewards given the system dynamics modeled. The principal

architecture of the recurrent control neural network is based on the recurrent neural network for the Markovian state space reconstruction (Zidong et al., 2006). In both RCNNRL's tasks, a recurrent neural network is trained on the similar sets of training patterns with the identical weight extended backpropagation. In either case, the optimal system identification plays an important role in the RCNNRL and forms the basis for learning the optimal policy. In addition to this, the operation of the RCNNRL continues until a sufficient generalization performance is achieved in both tasks. The optimal control problem is connected to policy gradient methods and does not use the value function. It rather searches directly within in the policy space and policy improvement is done by reward's gradient. In fact, we can assume the RCNN as a virtual MC policy gradient approach or an unfolded network, which virtually determines the value of the current policy and improves it according to the gradient of the reward function. Therefore, an explicit Monte-Carlo estimation (Binder, 2010), which would require a generation of new data-samples, is avoided because it increases the RCNN's data-efficiency.

3.2.1. Boltzman Neural Network Reinforcement Learning

This method's formulation maps past rewards into the future actions through a Gibbs (or Boltzmann) probability distribution by closely following Bell's algorithm (Bell, 2001). Boltzmann learning underlies an artificial neural network model known as the Boltzman machine, which allows the existence of hidden units to model data distributions much more complex than those which can be learned by Hopfield networks (Rojas, 1996). The length of time and the computational power required to train the network are this method's major limitations. These are due to the connections between visible and hidden units in the network. However, this type of learning offers potential in agent-based simulation models. Bass et al. investigated the Boltzman learning in ecology and agent-based simulation models (Bass, 2008).

3.2.3. Simple Recurrent Neural Network Reinforcement Learning

Elman and Jordan NNs are two major types of simple recurrent networks. The Elman NN is a semi-recursive neural network using the back propagation-through time learning algorithm to find patterns in sequences of values. According to Michael I. Jordan, Jordan networks are similar to Elman networks. The main difference of the Jordan network is that output layer feeds the context units instead of the hidden layer. The context units in a Jordan network have a recurrent connection to themselves and with no other nodes on that connection. They are also referred to as the state layer (Cruse, 1997). Basically, simple recurrent neural network reinforcement learning algorithms are trained with gradient descent based learning algorithms, with variants of backpropagation. A major drawback for the cognitive plausibility of backpropagation is that it is a supervised scheme, in which a teacher has to provide a fully specified target answer. In a recent research, Gruning showed that the learning behaviors of Elman backpropagation and its reinforcement variant are very similar in online learning tasks, which is a major advantage of this method (Gruning, 2007). Kasiran et al (2012)'s research main purpose was to develop Elman and Jordan recurrent neural networks, and equipping them with Q-Learning to predict the probabilities of mobile phone churning rates. They tested both Elman and Jordan recurrent neural networks algorithms using data gathered from mobile phone users and found that Jordan recurrent neural networks had shown to perform better in churn prediction (Kasiran et al., 2012).

3.2.4. Long Short Term Memory (LSTM) Neural Network Reinforcement Learning

According to the literature, many approaches have difficulties to deal with complex problems, long-term dependencies between relevant past events, and choosing the best action. This is an important shortcoming because we doubt about short-term dependencies in realistic partially observable RL tasks. This can simply be solved by remembering the last few observations and/or actions. In this method, some obvious problems with fixed size history window have been reported. If the information to be remembered falls outside the history window, the agent will not be able to use it. (Ring, 1993; McCallum, 1996) could increase the capacity to represent long-term dependencies in variable history window approaches.

These algorithms start with an initially unknown environment and zero history. They increase the depth of the history window in each step, based on gathering statistics. Learning long time lag dependencies virtually become impossible when there are no short-term dependencies to build on. Even if it worked, it would be a very time consuming process. The entire history in model-free approaches is based on memory bits (Lanzi, 2000) or recurrent neural networks, which can extract and represent just the relevant information for an arbitrary amount of time. However, learning to extract and represent information from a long time ago both for model-based and for model-free approaches is difficult, which lies in discovering the correlation between a part of information from the past and the moment at which this information becomes relevant. This is as an instance of the general difficulty of learning long-term dependencies in time series data (Schmidhuber, 2002). Long Short-Term Memory (LSTM) recurrent neural network is one particular solution to this problem. The implementation of this network in a variety of supervised time series learning tasks has proven to work well in the literature. In a recent work by Shiraga et al.(2002), a neural network model with incremental learning ability was applied and correctly acquired input-output relations were stored into long-term memory. Their results certify that the proposed model could acquire proper action-values (Shiraga et al., 2012). Policy gradient methods, evolution strategies or genetic algorithms are the RL LSTM training methods (Schmidhuber et al., 2007). Model-free RL-LSTM using Advantage (A) learning and directed exploration methods have been studied by Bakker in a T-maze task, as well as in a difficult variation of the pole balancing task. Bakker demonstrated that this method is able to solve non-Markovian tasks with long-term dependencies between relevant events. Moreover, Bakker described back propagation through an LSTM neural network critic for RL tasks in partially observable domains (Bakker, 2007). This work illustrated the advantage of being able to learn high-dimensional and/or continuous actions.

3.2.5. Hierarchical Neural Network Reinforcement Learning

The hierarchical soft-modular architecture is the analogy with the hierarchical organization of basal ganglia and motor cortex in biological systems (Luppino, 2000). In this architecture, the system is allowed to autonomously decide the sophistication of the computational resources to perform different manipulation behaviors based on object features and task complexity (Baldassarre, 2002). Recent applications of the hierarchical RL combine the advantages of multiple tasks. For instance, Ciancio et al (2011) demonstrated that the combination of a hierarchical approach and the actor-critic method is a valuable tool for the study of the development of rhythmic manipulation skills to search the parameters of a set of CPGs, having different degrees of sophistication (Ciancio et al., 2011). Furthermore, Caligiore et al. proposed a system, which combines the mixture of experts architecture with the neural network actor-critic architecture trained with the TD (λ) RL algorithm on a 2D robotic arm for reaching (Johansson et al., 2010). In addition to this, Snel et al. studied the robust CPGs for embodied hierarchical RL on a dynamically simulated hexapod robot. They showed the positive effect of low-level controllers with a degree of instability as terrain difficulty increases and adaptively to

environmental disturbances becomes more important (Snel et al., 2011) There are three main approaches to hierarchical RL; the option formalism of, the hierarchies of abstract machines (HAMs) approach of (Parr, 1998), and the MAXQ framework of (Dietterich, 2000). Although these approaches were developed relatively independently, they have many elements in common. Particularly, they all rely on the theory of semi-Markov decision processes to provide a formal basis (Andrew, 2003).

3.2.5.1. Option

An option or a Markov option sets action probabilities based solely on the current state of the core MDP. The implementation of Semi-Markov options allows more flexibility and also the policies are able to set action probabilities based on the entire previous states. Moreover, Semi-Markov options terminate after a pre-specified number of time steps, and most importantly, they are needed when policies over options are considered. Value functions for option policies can be defined in terms of value functions of semi-Markov flat policies (Eq. 4).

$$V^{\pi} = E\{r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{\tau-1} r_{t+\tau} + \cdots | \xi(\pi, s, t)\} \quad (4)$$

Where $\xi(\pi, s, t)$ is the event of π being initiated at time t in s, r is the reward value in each step and γ is the discount factor. Similarly, one can define the option-value function for a policy μ over options o (Eq. 5).

$$Q^{\mu}(s, o) = E\{r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{\tau-1} r_{t+\tau} + \cdots | \xi(o\mu, s, t)\} \quad (5)$$

Where $o\mu$ is the semi-Markov policy that follows o until it terminates after τ time steps and then continues according to μ.

The main goal of the options framework is to permit one to add temporally-extended activities to the repertoire of choices available to an RL agent, while not precluding planning and learning at the finer grain of the core MDP at the same time.

3.2.5.2. Hierarchies of Abstract Machines (HAMs)

The basic idea of HAM approach is defining the policies of a core MDP as programs which execute based on their own states as well as the current states of the core MDP. The RL method's most important strength in this context is its ability to reduce discrete time Semi-MDP without performing any explicit reduction on it (Andrew, 2003).

3.2.5.3. MAXQ Value Function Decomposition (MAXQ)

MAXQ is another approach, developed by (Dietterich, 2000) to hierarchical RL. This approach also relies on the theory of Semi-MDPs and a hierarchy of Semi-MDPs is created whose solutions can be learned simultaneously (Andrew, 2003).

3.2.6. Stochastic Neural Network Reinforcement Learning

Transforming a deterministic neural network to a stochastic neural network is done by applying transfer functions or network weights, which can be determined stochastically. Generally, better descriptions for this method are provided by using stochastic synapse neural networks, which have been studied in (Borkar, 2008; Borkar, 1997). They found that NNs with stochastic synapses are able to perform classification and reinforcement tasks on a par with the Boltzmann machines. Samli (2012) has investigated the properties of stochastic neural networks and the examples of using this type of neural networks for optimization problems.

Kernel-based stochastic reinforcement-learning is a decision policy learning method for a set of sample transitions. This method is impractical for large problems due to the approximator's size increment with the number of transitions. Barreto et al.(2011) introduced a novel algorithm to improve the scalability of KBRL. They empirically demonstrated the ability of the proposed approach to compress the information contained in KBRL's model (Barreto et al., 2011).

3.3. Associative Memory Neural Networks Reinforcement Learning

The idea of associative memories has been obtained from psychological theories of human and animal learning. Associative memories store information by learning correlations between different state-actions. In this method, strong correlations are reinforced and then the stored information is distributed in the sense that it is shared amongst numerous correlations between individual features. Each state-action pair location has a particular address. The information is retrieved by returning the contents of a given address. However, there is no need for the address and its contents to share any features in common. On the other hand, it may still be possible to retrieve the information in a distributed memory if the cue is sufficiently similar to the associated state-action pair. In a recent study, Bose et al. designed an associative separate memory with feedback that is capable of on-line temporal sequence learning and is able to store and predict online sequences of various types and length (Bose et al., 2006). The bi-directional associative memory (BAM) NN has been extensively studied since its introduction by Kosko (1988). Today, most of BAM models became able to store and recall all the patterns in a learning set in order to modify the learning rule or coding procedure. This resulted in both storage capacity, performance increment, and the number of spurious states decrement (Wang et al., 1990; Wang, 1996). Genetic algorithms and perceptron learning have also been studied in (Shen, 2005; Du et al., 2005; Leung, 1994). In addition, the implementation of expanded pattern pairs was included in further researches (Wang, 1996; Shen, 2005). Chartier et al. (2006) proposed an online learning BAM, in which the connections weights are iteratively developed in order to converge to a stable solution. Researches could also increase the BAM learning and recall performances using a hybrid model with the general regression neural network (Bohte et al., 2002).

3.4. Spiking Neural Networks Reinforcement Learning

The most accurate models of biological nervous system are spiking neural networks (SNN) due to their huge potential in information processing. Synaptic weights optimization is the learning objective of SNN. Gradient is the key information that enables learning in structures like SNN. Today, various ways of gradient estimation exist in learning methods. In the spike propagation algorithm [94], the relation between the weights and the spike occurrence time form the basis of gradient estimation. On the other hand, the ReSuMe algorithm (Ponulak et al., 2010) brings the actual spike timing to the desired one in order to adjust the weights of the network. However, this approach is limited to single

layered networks. In (Fiete, 2006) the gradient is estimated according to random perturbations of membrane potentials in neurons. Moreover, in (Florian, 2007) the random perturbations of firing thresholds in neurons estimate the gradient. Razvan V. Florian (Florian, 2005) presented a mechanism, in which learning has been achieved by synaptic changes that depend on the firing of pre and postsynaptic neurons, and that are modulated with a global reinforcement signal. They also discovered a biologically-plausible RL mechanism that works by spike-timing-dependent plasticity (STDP) modulation with a global reward signal. A learning method based on perturbation of synaptic conductance was proposed by Suszynski et al., which was slow and had large variance (Suszynski, 2013). In another study, Seung considered the hypothesis that the randomness of synaptic transmission is harnessed by the brain for learning and demonstrated that a network of hendonistic synapses can be trained to perform a desired computation by administrating reward appropriately (Sebastian, 2003). Also, Brea et al. derived a tractable learning rule for the synaptic weights towards hidden and visible neurons that leads to optimal recall of the training sequences. They have illustrated that learning synaptic weights towards hidden neurons improves the storing capacity of the network significantly (Brea et al., 2011). Christodoulou et al. investigated the application of RL on spiking neural networks in a demanding multi-agent setting and showed that their learning algorithms achieved to exhibit 'sophisticated intelligence' in a non-trivial task (Christodoulou, 2010). Also, they revealed that they could enhance the learning through strong memory for each agent and firing irregularity. Potjans et al(2009). presented a spiking neural network model that implements actor-critic temporal-difference learning by combining local plasticity rules with a global reward signal. This network could solve a nontrivial gridworld task with sparse rewards and learned with a similar speed to its discrete time counterpart and attained the same equilibrium performance (Potjans et al., 2009). Moreover, in Frémaux et al.'s model, the critic learns to predict expected future rewards in real time in continuous time actor-critic framework with spiking neurons. They have shown that such architecture can solve a Morris water-maze-like navigation task, the acrobat, and the cartpole problems (Frémaux et al., 2013).

3.5. Cascading Neural Networks Reinforcement Learning

There are two key tasks in the cascade-correlation architecture (Fahlman, 1990). First, the implementation of this architecture grows the network and adds new neurons when it is required to solve the problem. Second, each new added neuron is trained separately. In the former task, the learning algorithm starts with a network, which only consists of the input and the output layers and does not provide any hidden layers. The absence of hidden neurons in this network enables learning by a simple gradient descent algorithm applied for each output neuron individually. The whole process is continued until the desired network accuracy is obtained. Covariance maximization between the network's and the neuron's outputs is the goal of this method (Nissen, 2007). The cascade-correlation architecture variants and its important applications were surveyed recently (Balazs, 2009).

3.6. Neuro-Fuzzy Neural Networks Reinforcement Learning

Recently, combining the learning abilities of neural networks and fuzzy systems has become one of the major active research areas. This integration enhances the usefulness of both of the algorithms. However, precise data for training and learning are usually difficult or expensive to obtain or even not really available for some real-world applications. Therefore, many researches have been done on neuro-fuzzy RL. For instance, fuzzy adaptive learning control network (Lin, 1991), back-propagation fuzzy systems (Wang, 1992), adaptive neuro-fuzzy inference systems (Roger, 1993), fuzzy associative memory (Kosko, 1992), and fuzzy hyper rectangular composite neural networks (Su, 1997; Su, 1999) have been studied widely in the past decade. Self-generating neuro-fuzzy systems tune the parameters

of the system and construct the incremental architecture of the whole system through learning. The development of these systems was obtained from a machine learning system that learns syntactically simple string rules, called classifiers (Goldberg, 1989). In a recent study by Shah et al. a novel on-line sequential learning evolving neuro-fuzzy model design for RL was proposed and the performance of the approach was investigated through the simulation results of a two-link manipulator (Shah, 2013). Moreover, Jinlin et al. proposed a neuro-fuzzy RL controller for velocity tracking control of a vehicle, manipulated by a robot driver. As a result, the robot driver embodied driving behavior of a skilled driver as operating the pedals, especially in the alternative switch process of the two pedals (Jinlin et al., 2009).

4. Conclusions and future work

As an interdisciplinary area, the combination of reinforcement learning (RL) and neural networks algorithms has changed the face of modern optimal control significantly and became a key research interest in various domains such as robotics, control theory, operational research, and finance. In this paper, we have presented an overview on the state of the art neural network reinforcement learning (NNRL) algorithms. In addition to this, recent developments in the performance of these algorithms were evaluated and compared in learning prediction and learning control tasks. According to the literature, it is clear that using NNRL is not yet a straightforward undertaking but rather requires a certain amount of skill to select the best algorithm. Moreover, choosing a good parameterization for the policy, which highly influences the performance after learning, also remains for further investigations. Although many valuable advanced studies have been done in the past decade, there are still some open research problems. New NNRL algorithms development with fast convergence abilities in online learning process and generalization in high-dimensional continuous spaces are desirable. The research on the combination of these methods as data-driven or learning control techniques for complex dynamical systems and high dimensional environments still need to be explored. Also, NNRL methods for multi-agent systems and control tasks are important topics for future work.

References

- Lewis, F. L., & Vrabie, D., “Reinforcement learning and adaptive dynamic programming for feedback control”, Circuits and Systems Magazine, 09(3), pp. 32–50, 2009.
- L. Busoniu, R. Babuska, B. De Schutter, D. Ernst, “Reinforcement Learning and Dynamic Programming Using Function Approximators”, CRC Press, NY, 2010.
- Williams, J. K., “Reinforcement learning of optimal controls”, In S. E. Haupt, A. Pasini, & C. Marzban (Eds.), Artificial intelligence methods in the environmental sciences, Springer, pp. 297–327, 2009
- Cs. Szepesvri, “Algorithms for Reinforcement Learning”, Morgan and Claypool, 2010.
- X. Xu, “Reinforcement Learning and Approximate Dynamic Programming”, Science Press, Beijing, 2010.
- Sutton, R., Barto, A. G., & Williams, R. J., “Reinforcement learning is direct adaptive optimal control”, In Proceedings of the American control conference, pp. 2143–2146, 1992.

- P.J. Werbos, “Intelligence in the brain: a theory of how it works and how to build it”, Neural Networks, pp. 200–212, 2009.
- D. Ernst, P. Geurts, L. Wehenkel, “Tree-based batch mode reinforcement learning”, Journal of Machine Learning Research 6, pp. 503–556, 2005.
- A.G. Barto, T.G. Dietterich, “Reinforcement learning and its relationship to supervised learning”, in: J. Si, A. Barto, W. Powell, D. Wunsch (Eds.), Handbook of Learning and Approximate Dynamic programming, Wiley-IEEE Press, New York, 2004.
- Chen M, Ge SS and Ren BB, “Adaptive tracking control of uncertain MIMO nonlinear systems with input constraints”, Automatica, 47, pp. 452–465, 2011.
- Li TS, Wang D and Feng G, ”A DSC Approach to robust adaptive NN tracking control for strict-feedback nonlinear systems”, IEEE Transactions on System Man, and Cybernetics – Part B: Cybernetics, 40, pp. 915–927, 2010.
- Li ZJ, Ge SS, Adams M and Wijesoma WS, “Adaptive robust output-feedback motion/force control of electrically driven nonholonomic mobile manipulators”, IEEE Transactions on Control Systems Technology, 16(6), pp. 1308–1315, 2008a.
- Li ZJ, Ge SS, Adams M and Wijesoma WS , “Robust adaptive control of uncertain force/motion constrained nonholonomic mobile manipulators”, Automatica, 44, pp. 776–784, 2008b.
- Yang C, Ganesh G, Albu-Schaeffer A and Burdet E, “Human like adaptation of force and impedance in stable and unstable tasks”, IEEE Transactions on Robotics, 27, pp. 918–930, 2011.
- Sun FC, Sun ZQ and Woo PY, “Stable neural-networkbased adaptive control for sampled-data nonlinear systems”, IEEE Transactions on Neural Networks, 9(5), pp. 956–968, 1998.
- Sun FC, Li L, Li HX and Liu HP, “Neuro-fuzzy dynamic-inversion-based adaptive control for robotic manipulators- discrete time case”, IEEE Transactions on Industrial Electronics, 54(3), pp. 1342–1351, 2007.
- R. Sutton, A.G. Barto, “Reinforcement Learning. An Introduction”, MIT Press, Cambridge MA, 1998.
- W. Smart, L. Kaelbling, “Practical reinforcement learning in continuous spaces”, International Conference on Machine Learning, ICML, 2000.
- M. Carreras, P. Ridao, A. El-Fakdi, “Semi-online neural-Q-learning for real-time robot learning”, IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, Las Vegas, USA, 2003.
- L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst, “Reinforcement Learning and Dynamic Programming Using Function Approximators”, CRC Press, 2010.
- J. Peters and S. Schaal, “Natural Actor-Critic,” Neurocomputing, 71, pp. 1180–1190, 2008.
- D. P. Bertsekas, “Dynamic Programming and Optimal Control”, vol.2, 3rd ed. Athena Scientific, 2007.

- K. G. Vamvoudakis and F. L. Lewis, "Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem," Automatica, 46(5), pp. 878–888, 2010.
- T. Hanselmann, L. Noakes, and A. Zaknich, "Continuous-Time Adaptive Critics", IEEE Transactions on Neural Networks, 18(3), pp. 631– 647, 2007.
- R. Sutton, "Learning to predict by the method of temporal differences", Machine Learning 3, pp. 9–44, 1988.
- Bertsekas, D., "Neuro-dynamic programming. In Encyclopedia of optimization", pp. 2555–2560, 2009.
- R. Enns, J. Si, "Helicopter trimming and tracking control using direct neural dynamic programming", IEEE Transactions on Neural Networks, 14 (4), pp. 929–939, 2003.
- S.J. Brartke, A. Barto, "Linear least-squares algorithms for temporal difference learning", Machine Learning, 22, pp. 33–57, 1996.
- H.R. Maei, C. Szepesvri, S. Bhatnagar, D. Precup, R.S. Sutton", Convergent temporal-difference learning with arbitrary smooth function approximation", in: J. Laferty, C. Williams (Eds.), Advances in Neural Information Processing Systems, 22, MIT Press, Cambridge, MA, USA, 2010.
- H. Zhang, L. Cui, X. Zhang, Y. Luo, "Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method", IEEE Transactions on Neural Networks, 22 (12), pp. 2226–2236, 2011.
- S. Bradtke, "Incremental Dynamic Programming for On-Line Adaptive Optimal Control", Ph.D. thesis, University of Massachusetts, Computer Science Dept. Tech. Rep., pp. 94–62, 1994.
- Vlassis, N., Toussaint, M., Kontes, G., and Piperidis, S.,"Learning model-free robot control by a Monte Carlo EM algorithm", Autonomous Robots, 27(2), pp. 123–130, 2009.
- G. A. Rummery and M. Niranjan, "On-Line Q-Learning Using Connectionist Systems", Cambridge University, Tech. Rep. CUED/FINFENG/ TR 166, 1994.
- C. Watkins, P. Dayan, "Q-learning", Machine Learning, 8, pp. 279–292, 1992.
- V. R. Konda and J. N. Tsitsiklis, "On Actor-Critic Algorithms", SIAM Journal on Control and Optimization, 42(4), pp. 1143–1166, 2003.
- R. J. Williams, "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning", Machine Learning, 8, pp. 229–256, 1992.
- F. S. Melo, S. P. Meyn, and M. I. Ribeiro, "An Analysis of Reinforcement Learning with Function Approximation", in Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, pp. 664–671, 2008.
- G. J. Gordon, "Stable Function Approximation in Dynamic Programming", in Proceedings of the 12th International Conference on Machine Learning, Tahoe City, USA, pp. 261–268, 1995.

- S. Bhatnagar, "An actor-critic algorithm with function approximation for discounted cost constrained Markov decision processes", Systems & Control Letters, 59(12), pp. 760–766, 2010.
- B. Kim, J. Park, S. Park, and S. Kang, "Impedance Learning for Robotic Contact Tasks Using Natural Actor-Critic Algorithm", IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, 40(2), pp. 433–443, 2010.
- V. R. Konda and J. N. Tsitsiklis, "On Actor-Critic Algorithms", SIAM Journal on Control and optimization, 42(4), pp. 1143–1166, 2003.
- J. Peters and S. Schaal, "Natural Actor-Critic", Neurocomputing, 71, pp. 1180–1190, 2008.
- [43] J. A. Boyan, "Technical Update: Least-Squares Temporal Difference Learning," Machine Learning, 49, pp. 233–246, 2002.
- K.G. Vamvoudakis, Frank L. Lewis, "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem", Automatica, 46 (5), pp. 878–888, 2010.
- H. R. Berenji and D. Vengerov, "A Convergent Actor-Critic-Based FRL Algorithm with Application to Power Management of Wireless Transmitters", IEEE Transactions on Fuzzy Systems, 11(4), pp. 478–485, 2003.
- L. Tang and Y. Liu, "Adaptive neural network control of robot manipulator using reinforcement learning", Journal of Vibration and Control (online version), June 2013.
- Farkaš,T. Malík and K. Rebrová, "Grounding the meanings in sensorimotor behavior using reinforcement learning", frontiers in neurobotics, 6(1), pp.1-13, 2012.
- Sh. Bhasin , N. Sharma, P. Patre , W. Dixon, "Asymptotic tracking by a reinforcement learning-based adaptive critic controller", Journal of Control Theory Application, 9(3), pp. 400-409, 2011.
- Y. Nakamura, T. Mori, M. Sato, Sh. Ishii, "Reinforcement learning for a biped robot based on a CPG-actor-critic method", Neural Networks, 20, pp.723-735, 2007.
- Z. Miljkovic', M. Mitic', M. Lazarevic', B. Babic', "Neural network Reinforcement Learning for visual control of robot manipulators", Expert Systems with Applications, 40, pp.1721-1736, 2013.
- Sarangapani, J., "Neural network control of nonlinear discrete-time systems", USA: Taylor & Francis, 2006.
- Stingu, P., & Lewis, F, "Adaptive dynamic programming applied to a 6dof quadrotor", In B. Igelnik (Ed.), Computational modeling and simulation of intellect: Current state and future perspectives. IGI Global, 2010.
- G. Tesauro, "Temporal difference learning and TD-Gammon. Communications of the ACM", 38(3), pp.58–68, March 1995.
- H. White, "Learning in Artificial Neural Networks: A Statistical Perspective", Neural computation, 1(4), pp. 425-464, 1989.

- E.M. Johansson, F.U. Dowla, and D.M. Goodman, " Backpropagation Learning for Multilayer Feed-Forward Neural Networks Using the Conjugate Gradient Method", International Journal of Neural Systems, 02(291), 1991.
- Bebis, G., Georgiopoulos, M., "Feed-forward neural networks", Potentials, IEEE, 13(4), pp. 27-31, 1994.
- Zidong Wang, Yurong Liu, Li Yu, Xiaohui Liu, "Exponential stability of delayed recurrent neural networks with Markovian jumping parameters", Physics Letters A, 356, Issues 4–5, pp. 346–352, 2006.
- Kurt Binder, Dieter W. Heermann, "Monte Carlo Simulation in Statistical Physics: An Introduction", Springer, 2010.
- Bell, Ann M., "Reinforcement Learning Rules in a Repeated Game", Computational Economics, 18(1), August 2001.
- Raul Rojas, "Neural Networks - A Systematic Introduction", Springer-Verlag, Berlin, New-York, 1996
- B. Bass, T. Nixon, "Boltzmann Learning", Reference Module in Earth Systems and Environmental Sciences Encyclopedia of Ecology, pp.489-493, 2008.
- Holk Cruse, "Neural Networks As Cybernetic Systems: Science Briefings" , George Thieme Verlag, 1997.
- Gruning, A., "Elman backpropagation as reinforcement for simple recurrent networks. Neural Computation", 2007.
- Kasiran, Z., Ibrahim, Z., Syahir Mohd Ribuan, M. , "Mobile phone customers churn prediction using elman and Jordan Recurrent Neural Network ", 7th International Conference on Computing and Convergence Technology (ICCCT), pp. 673 – 678, 2012.
- Ring, M. B., "Learning sequential tasks by incrementally adding higher orders", In C. L. Giles, S. J. Hanson, & J. D. Cowan (Eds.), Advances in neural information processing systems, 5 ,pp. 115-122. San Mateo, California: Morgan Kaufmann Publishers, 1993a.
- Ring, M. B., "Two methods for hierarchy learning in reinforcement environments", In J. A. Meyer, H. Roitblat, & S. Wilson (Eds.), From animals to animats 2: Proceedings of the second international conference on simulation of adaptive behavior, pp. 148-155, 1993b.
- McCallum, R. A., "Learning to use selective attention and short-term memory in sequential tasks", In From animals to animats 4: Proceedings of the fourth international conference on simulation of adaptive behavior. Cambridge, MA: MIT Press, 1996.
- Lanzi, P. L., "Adaptive agents with reinforcement learning and internal memory", In J.-A. Meyer, D. Floreano, H. L. Roitblat, & S. W. Wilson (Eds.), From animals to animats 6, Proceedings of the sixth international conference on simulation of adaptive behavior, pp. 333-342, Cambridge, MA: MIT Press, 2000.
- Schmidhuber, J., "Optimal ordered problem solver", Technical report No. IDSIA- 12-02, Manno-Lugano, Switzerland: IDSIA, 2002.

- N. Shiraga, S. Ozawa, and S. Abe, "A Reinforcement Learning Algorithm for Neural Networks with Incremental Learning Ability", Proceedings of the 9th International Conference on Neural Information Processing (ICONIP), 5, pp. 2566 – 2570, 2002.
- J. Schmidhuber, D. Wierstra, M. Gagliolo, F. Gomez., "Training Recurrent Networks by Evolino", Neural Computation, 19(3), pp. 757–779, 2007
- B. Bakker, "Reinforcement learning by backpropagation through an LSTM model/critic", IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL), pp.127-134, 2007.
- G. Luppino, G. Rizzolatti, G. "The Organization of the Frontal MotoCortex", News in physiological sciences, 15, pp. 219-224, 2000.
- G. Baldassarre" A modular neural-network model of the basal ganglia's role in learning and selecting motor behaviours", Journal of Cognitive Systems Research, 3, pp. 5–13, 2002.
    - L. Ciancio, L. Zollo, E. Guglielmelli, D. Caligiore and G. Baldassarre, "Hierarchical Reinforcement Learning and Central Pattern Generators for Modeling the Development of Rhythmic Manipulation Skills", IEEE International Conference on Development and Learning (ICDL), 2, pp. 1-8, 2011.
- Johansson, B., !ahin, E. & Balkenius, C., "A Bioinspired Hierarchical Reinforcement Learning Architecture for Modeling Learning of Multiple Skills with Continuous States and Actions", International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems, 2010.
- M. Snel, Sh. Whiteson, and Y. Kuniyoshi, "Robust Central Pattern Generators for Embodied Hierarchical Reinforcement Learning", In ICDL: Proceedings of the First Joint IEEE International Conference on Development and Learning, pp. 1–6, August 2011.
- Parr, R. and Russel, S., "Reinforcement learning with hierarchies of machines", Advanced Neural Information Processing Systems: Proceedings of the 1997 conference, Cambridge, MA:MIT Press, 1998.
- Dietterich, T. G., "Hierarchical reinforcement learning with the maxq value function decomposition", Journal of Artificial Intelligence Research, 13, pp. 227-303, 2000.
- Andrew G. Barto, Sridhar Mahadevan, "Recent Advances in Hierarchical Reinforcement Learning", Discrete Event Dynamic Systems: Theory and Applications, 13, pp. 41-77, 2003
- V.S. Borkar, "Stochastic Approximation: A Dynamical Systems Viewpoint", Cambridge University Press, 2008.
- V.S. Borkar, "Stochastic approximation with two time scales", Systems & Control Letters 29 (5), pp. 291–294, 1997.
- R. Samli, "Stochastic Neural Networks and Their Solutions to Optimization Problems", International Journal of Electronics; Mechanical and Mechatronics Engineering, 2(3), pp.(293-297), 2012
- S. Barreto, D. Precup, J. Pineau, "Reinforcement Learning using Kernel-Based Stochastic Factorization", Advances in Neural Information Processing Systems, 24, pp. 720-728, 2011

- J. Bose, S. B. Furber, and J. L. Shapiro., "An associative memory for the on-line recognition and prediction of temporal sequences", CoRR , 2006
- Kosko, "Bidirectional associative memories," IEEE Transactions on Systems, Man and Cybernetics, SMC-18(1), pp. 49–60, Jan./Feb. 1988.
- Y. F.Wang, J. B. Cruz, Jr., and J. H. Mulligan, Jr., "Two coding strategies for bidirectional associative memory", IEEE Transactions on Neural Networks, 1(1), pp. 81–92, Mar. 1990.
- Z. Wang, "A bidirectional associative memory based on optimal linear associative memory", IEEE Trans. Comput., 45(10), pp. 1171–1179, Oct. 1996.
- Shen and J. B. Cruz, Jr., "Encoding strategy for maximum noise tolerance bidirectional associative memory," IEEE Transactions on Neural Networks, 16(2), pp. 293–300, Mar. 2005.
- S. Du, Z. Chen, Z. Yuan, and X. Zhang, "Sensitivity to noise in bidirectional associative memory (BAM)," IEEE Transactions on Neural Network, 16(4), pp. 887–898, Jul. 2005.
- C.-S. Leung, "Optimum learning for bidirectional associative memory in the sense of capacity," IEEE Transactions on Systems, Man, and Cybernetics, 24(5), pp. 791–795, May 1994.
- S. Chartier and M. Boukadoum, "A bidirectional heteroassociative memory for binary and grey-level patterns", IEEE Transactions on Neural Networks, 17(2), pp. 385–396, Mar. 2006
- S. Chartier, M. Boukadoum, and M. Amiri, "BAM Learning of Nonlinearly Separable Tasks by Using an Asymmetrical Output Function and Reinforcement Learning", IEEE TRANSACTIONS ON NEURAL NETWORKS, 20(8), 2009.
- S. M. Bohte, J. N. Kok, and H. L. Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons", Neurocomputing, 48, pp. 17–37, 2002.
- F. Ponulak and A. Kasinski, "Supervised learning in spiking neural networks with resume: Sequence learning, classication, and spike shifting," Neural Computation, 22, pp. 467–510, 2010.
- R. Fiete and H. S. Seung, "Gradient learning in spiking neural networks by dynamic perturbation of conductances", Physical Review Letters, 97(4), 2006.
- R. V. Florian, "Reinforcement learning through modulation of spike timing- dependent synaptic plasticity," Neural Computation, 19, pp. 1468–1502, 2007.
- R. V. Florian, "A reinforcement learning algorithm for spiking neural networks", Seventh International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2005.
- P. Suszynski, P. Wawrzynski, "Learning population of spiking neural networks with perturbation of conductances", Proceedings of International Joint Conference on Neural Networks, Dallas, Texas, USA, 2013.
- H. Sebastian Seung, "Learning in Spiking Neural Viewpoint Networks by Reinforcement of Stochastic Synaptic Transmission", Neuron, 40, pp. 1063–1073, 2003.

- J. Brea,W. Senn and J. P. Pfister, “Sequence learning with hidden units in spiking neural networks”, Advances in Neural Information Processing Systems Conference, 2011.
- Christodoulou, A. Cleanthous, “Spiking Neural Networks with Different Reinforcement Learning (RL) Schemes in a Multiagent Setting”, Chinese Journal of Physiology, 53(6), PP. 447-453, 2010.
- W. Potjans, A. Morrison, M. Diesmann, “A spiking neural network model of an actor-critic learning agent”, Neural Computation, 21(2), pp. 301-339, 2009
- Frémaux N, Sprekeler H, Gerstner W , ”Reinforcement Learning Using a Continuous Time Actor-Critic Framework with Spiking Neurons”, PLoS Computer Biology, 9(4), 2013.
- S. E. Fahlman and C. Lebiere,” The Cascade-Correlation Learning Architecture”, D. S. Touretzky (ed.), Advances in Neural Information Processing Systems, 2, pp. 524–532, 1990.
- S. Nissen, “Large Scale Reinforcement Learning using Q-SARSA and Cascading Neural Networks”, MSc Thesis, University of Copenhagen, 2007.
- G. Balazs. "Cascade-Correlation Neural Networks: A Survey". Technical Report, University of Alberta, December 2009.
    - -T. Lin and C. S. G. Lee, “Neural-network-based fuzzy logic control and decision system,” IEEE Transactions on Computers, 40(12), pp. 1320-1336, 1991.
- L. -X. Wang and J. H. Mendel, “Back-propagation fuzzy systems as nonlinear dynamic system identifiers”, Proc. IEEE International Conference on Fuzzy Systems, San Diego, pp. 1163-1170, 1992.
- J. -S. Roger Jang, “ANFIS: adaptive-network-based fuzzy inference systems”, IEEE Transactions on Systems, Man, and Cybernetics, 23(3), pp. 665-685, 1993.
    - Kosko, “Neural Networks and Fuzzy Systems: A Dynamical system Approach to Machine Intelligence”, Prentice-Hall, Englewood Cliffs, NJ, 1992.
- M. C. Su, “Identification of Singleton fuzzy Models via fuzzy hyper-rectangular composite NN, in Fuzzy Model Identification: Selected Approaches”, H. Hellen doorn and D. Driankov, Eds., Springer-Verlag, pp. 193-212, 1997.
- M. C. Su, C. –W. Lin, and S. –S. Tsay, “Neural-network-based Fuzzy Model and its Application to Transient stability Prediction in Power System”, IEEE Transactions on Systems, Man, and Cybernetics, 29(1), pp. 149-157, 1999
    - Goldberg, “Genetic Algorithm in Search,Optimization, and machine Learning”, Addison-Welsley, MA, 1989.
- H. Shah, M.Gopal, “A Reinforcement Learning Algorithm with Evolving Fuzzy Neural Networks”, Proceedings of the International Conference on Systems, Control and Informatics, 2013
- X. Jinlin, Z. Weigong , G. Zongyang, “Neurofuzzy velocity tracking control with reinforcement learning ”, International Conference on Electronic Measurement & Instruments (ICEMI), pp. 465-468, 2009.