IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received January 9, 2018, accepted February 14, 2018, date of publication February 23, 2018, date of current version March 12, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2808266

An Adaptive Strategy Selection Method With Reinforcement Learning for Robotic Soccer Games

HAOBIN SHI[1], ZHIQIANG LIN[1], KAO-SHING HWANG[2], (Senior Member, IEEE), SHIKE YANG[1], AND JIALIN CHEN[1]

[1]School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China
[2]National Sun Yat-sen University, Kaohsiung 80424, Taiwan

Corresponding author: Haobin Shi (shihaobin@nwpu.edu.cn)

This work was supported in part by the National Research and Development Plan of China under Grant 2017YFB1001900, in part by the Fund of National Ministries under Grant 2016ZC53022, and in part by the Fundamental Research Funds for the Central Universities under Grant 3102017JSJ0005.

ABSTRACT Robotic soccer games, which have become popular, require timely and precise decision-making in a dynamic environment. To address the problems of complexity in a critical situation, policy improvement in robotic soccer games must occur. This paper proposes an adaptive decision-making method that uses reinforcement learning (RL), and the decision-making system for a robotic soccer game is composed of two subsystems. The first subsystem in the architecture for the proposed method criticizes the situation, and the second subsystem implements decision-making policy. Inspired by the support vector machine (SVM), a situation classification method, which is called an improved SVM, embeds a decision tree structure and simultaneously addresses the problems of a large scale and multiple classifications. When a variety of situations that are collected in the field are classified and congregated into the tree structure, the problem of local strategy selection for each individual class of situations over time is regarded as a RL problem and is solved using a Q-learning method. The results of simulations and experiments demonstrate that the proposed method allows satisfactory decision-making.

INDEX TERMS Robotic soccer, support vector machines, reinforcement learning, Q learning.

I. INTRODUCTION

As multi-agent systems (MAS), robotic soccer games have recently become a popular means for the study of multi-agent technologies, such as cooperation, collaboration, and coordination [1], [2]. In a coordination process, each agent taking assignments in a team must cooperate with others while facing competition [3], [4]. In robotic soccer games, a decision-making system must deal with a variety of situations in confrontation and be able to adjust its strategies adaptively to achieve the purposes of coordination [5]. In [6], the decision-making system for a robotic soccer game is composed of two subsystems: a situation classification (SC) and a strategy selection (SS). The SC evaluates uncertain environmental information and situations that agents encounter. The SS selects an appropriate strategy for a certain situations classified by the SC. Many methods have been proposed for the implementation of the SC mechanism. Expert systems can evaluate situations and has been widely utilized [7], [8], though their performance depends heavily on expert knowledge of robotic soccer games. Besides, it is difficult to acquire sufficient domain knowledge in a dynamic environment. Some methods to increase the capacity of expert knowledge, such as fuzzy decision trees [9] and self-organizing fuzzy decision trees [10], are also used. These methods combine a fuzzy logic with a decision tree's structure to render the process of the SC that is more similar to human manners. These methods have problems such as the need for huge amounts of memory and slow generation speed. Some variants of artificial neural networks are feasible alternatives, such as fuzzy neural networks (FNN) [11], wherein the neural network is combined with fuzzy logic to realize a more intelligent SC. However, an accurate SC requires sufficient training samples and the parameter adjustment process is complicated. As demonstrated in [12], an incremental learning algorithm with an SVM has a good performance in generalization, because it does not depend on all the training data, but a subset named support vector.

A Bayesian network is used in SS methods [13]. The Bayesian network is sensitive to *a priori* knowledge. Nevertheless, it is difficult to acquire knowledge beforehand and the method may perform poorly when an agent is in a dynamic environment. Reinforcement learning (RL) has become more commonly used in SS [14]. For example, in [15], a multi-strategy decision-making system with RL is proposed for robot soccer games. A better action can be granted after an iterative learning process. Recently, methods combining RL and deep learning are popular for decision-making. Agents with deep RL can learn the optimal policy by a manner of end-to-end learning similar to the way of human learning [16]. For learning in different situations, the method proposed in [17] can achieve adaptive decision-making. It also has many advantages, such as adaptability, robustness, and versatility. In many RL problems, however, a large state or action spaces results in an infeasible value function estimation. Therefore, combining the neural network with RL is an effective scheme that has been proposed to solve this problem [18], [19]. In the combination, the neural network approximates the value function. But its learning speed is slow and the performance in local generalization is poor in online learning. It has some limitations in practical applications.

This paper elaborates the state space modeling of a robot soccer system. For representation of dynamic operations, high abstraction is defined as several factors constituting of the dimensions of the state space to accommodate the characteristic of a hybrid system such as the robot soccer games. The obtained unified state space representation allows the efficient and accurate dynamic operations of the entire hybrid system working under dynamic and static conditions. Moreover, this paper proposes an RL decision-making method based on a tree structure where each leave is a state classified by a multi- classification technique. Based on the aggregated state space, a model-free RL method, Q-learning, is applied to a robotic soccer game [14]–[16]. Prior to execution of Q-learning, an improved support vector machine (ISVM) with a particle swarm optimization algorithm (PSO) for classification, named PSO-SVM, is proposed to construct a decision tree as the state space for RL [20]–[22]. With the ISVM, its situation evaluation result is close to the actual condition in robotic soccer games, the accurate and timely situation evaluation is the necessary guarantee for efficient strategy selection. When a variety of situations collected in the field are classified and congregated into a tree structure, the problem of local strategy selection for each individual class of situations over time is regarded as an RL problem and is solved using Q-learning [23]–[26]. With the RL, the adaptive strategy selection can be achieved. Because of its adaptivity, it is usually preponderant robotic soccer games.

This paper is arranged into four sections: Following the introduction, the decision-making method is presented in Section II. The SC subsystem classifies situations in a continuous domain into a discrete state space using the ISVM. The SS subsystem selects a specific strategy using the adaptive decision-making algorithm with RL (ADMA-RL). To illustrate the performance of the proposed method, comparisons between the proposed method and competitors in simulations and experiments are demonstrated and discussed in Section III. Section IV presents the conclusion.

II. THE ADAPTIVE STRATEGY DECISION METHOD FOR ROBOTIC SOCCER

To address decision-making problems in a dynamic environment, such as in a robotic soccer game, this paper proposes an adaptive decision method, where the ISVM is used to collect and classify the environmental situational information that is constituted by the defined evaluation factors and the ADMA-RL chooses the proper strategy adaptively.

A. A SITUATION CLASSIFICATION BASED ON ISVM

In robotic soccer, situations change and diversify so dynamically that an SC method with an SVM is proposed to assess the robot's ability to control the ball [27]. In general, an SVM constructs a hyper plane that is used for classification, regression, or other tasks [28]. Intuitively, good separation is achieved by the hyper plane that has the largest distance to the nearest training-data point for any class (the so-called functional margin), since the greater the margin, the less is the generalization error for the classifier. Assuming that there is a training dataset of k points of the form: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_k, y_k)$ where $\mathbf{x}_i$ is m dimension's vector and y_i are either 1 or -1, each y_i indicates the class to which point, $\mathbf{x}_i$, belongs. As shown in Fig. 1, the maximum-margin hyper plan'' must be found to divide the group of points, $\mathbf{x}_i$, for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyper plane and the nearest point, $\mathbf{x}_i$, from either group is maximized.

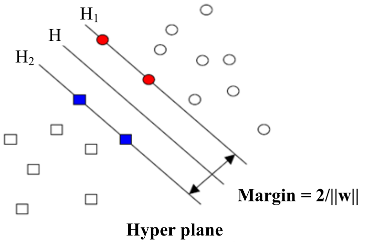


FIGURE 1. The hyper plane for the SVM.

While the optimal hyper plane is being determined, any hyper plane can be written as the set of points, $\mathbf{x}_i$, that satisfies:

$$\mathbf{w} * \mathbf{x}_i + b = 0 \tag{1}$$

where $\mathbf{w}$ is the (not necessarily normalized) normal vector to the hyper plane. The parameter, $b/||\mathbf{w}||$, determines the offset for the hyper plane from the origin along the normal vector, $\mathbf{w}$, $\mathbf{x}_i$ denotes the input vector.

Using the criterion for the maximum-margin hyper plane, the optimization problem for the SVM can be rewritten as:

$$\begin{cases} \text{minimize}: \ \frac{1}{2} * \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^{N} \zeta_i \\ \text{subject}: \ y_i(\mathbf{x}_i * \mathbf{w} + b) \geq 1 - \zeta_i \text{ and } \zeta_i \geq 0 \end{cases} \tag{2}$$

where ζ_i is a slack variable, C is the weight of outliers.

Using the trained weight vector $\mathbf{w}$, the classification function for the SVM is

$$f(\mathbf{x}) = \operatorname{sgn}(\mathbf{w} * \mathbf{x} + b) \tag{3}$$

A SVM makes a judgment for the current situation in the game by learning and it avoids over-learning, a dimensional disaster, or local minima [28]. In the SVM for this paper, an inseparable plane is mapped into a high dimensional plane using a kernel function [29]. However, a traditional SVM has disadvantages when large-scale data must be classified and when there are multiple classifications, but the proposed ISVM performs well to a multi-classification problem for large-scale data in dynamic environments. The SC is executed using an ISVM that has n_s descriptions of the environment.

B. DATA PREPARATION – THE DEFINITIONS OF EVALUATION FACTORS

During learning tasks, different high-level factors have a trade-off, in terms of rewards and penalties, such as minimizing the amount of time and effort that is spent on a task while maximizing the performance, in terms of the amount of time required. To describe a reward function in a task, the factors that matter for task completion must be decided exactly, and each of the factors' contribution to the reward must also be described exactly. In other words, the definitions of factors eventually constitute the dimension of the feature space in classification as well as the state space in RL [30]. These factors are referred to as environmental state features or just features. Using expert knowledge and experience, several key factors for situations in a robotic soccer game can be defined for decision- making. The factors are categorized as global and local factors.

Factor 1: Current score difference:

$$S^t = S_m^t - S_o^t \tag{4}$$

where S_m^t denotes the user's current score and S_o^t denotes the opponent's current score.

Factor 2: Remaining competition time normalization:

$$\eta^t = (T_{\max} - t)/T_{\max} \tag{5}$$

where $T_{\max}$ denotes the longest time slice number for one competition and t denotes the time that the current game has run.

Factor 3: Ball position, X^t:

To a certain extent, the ball's position determines the current decision. If the ball is in the opponent's penalty area, an aggressive strategy is taken; if the ball is in the user's penalty area, a defensive strategy is chosen to avoid losing the ball.

Factor 4: Ball approaching time difference:

$$t_A = t_{Ah} - t_{Ao} \tag{6}$$

where t_{Ah} denotes the fastest time taken to control the ball and t_{Ao} denotes the fastest time taken for the opponent to control the ball. This fastest time is calculated using the position of players.

Factor 5: Number difference of both offensive robots:

$$n_A = n_{Ah} - n_{Ao} \tag{7}$$

where n_{Ah} denotes the number of the user's offensive robots. n_{Ao} denotes the number of the opponent's offensive robots. The offensive robots indicates robots which have offensive capability. All robots have an offensive capability only within a certain range of the ball. This range is derived using actual match experience.

Factor 6: Distance difference between the ball and the both offensive robots:

$$d_A = d_{Ah} - d_{Ao} \tag{8}$$

where d_{Ah} denotes the average distance between the user's offensive robots and the ball, i.e. $d_{Ah} = \sum_{i=1}^{n_{Ah}} d_i/n_{Ah}$ where d_i denotes the distance between the ball and the user's i^{th} offensive robot; d_{Ao} denotes the average distance between the user's offensive robots and the ball, i.e. $d_{Ao} = \sum_{j=1}^{n_{Ao}} d_j/n_{Ao}$, where d_j denotes the distance between the ball and the opponent's i^{th} offensive robot.

Factor 7: The difference in the number of robots on both sides of the ball:

$$n_B = n_{Bh} - n_{Bo} \tag{9}$$

where n_{Bh} denotes the number of the user's robots on both sides of the ball and n_{Bo} denotes number of the opponent's robots on both sides of the ball. The first two are regarded as global factors and the others are local factors.

In other words, the current score difference, the remaining competition time normalization, the ball position, X^t, the ball approaching time difference, t_A, the number difference of both offensive robots, n_A, the distance difference between the ball and the both offensive robots, d_A, and the difference in the number of robots on both sides of the ball, n_B, are selected as features to represent a variety of situations occurring in the field.

Simulated experiences are collected in a simulation platform initially and the situation labels for all data are marked empirically according to the playback.

Because every feature has different dimensions, standardization is necessary. Max-min standardization is used as standardization method. Assure min_A, max_A are the minimum value and the maximum value of a feature, A, respectively. The standardization is executed by:

$$v_i' = (v_i - min_A)/(max_A - min_A) \tag{10}$$

The value v_i of feature A is mapped to the range of [0, 1].

Because data collected from the game field is nonlinearly separable, a kernel function is used to transform the data to another linear separable space. The selection of kernel function is important. Thus, A Radial Basis Functions (RBF) is selected as the kernel function for this paper [31].

In an SVM model, the penalty parameter, C, and the kernel parameter, γ, are important. The penalty parameter, C, describes the tradeoff between the ratio of the sample for error classification and the complexity of the algorithm. In terms of the choice of the penalty parameter, this paper uses a grid search method [32].The Gaussian kernel function is taken as an example and one set of parameters are chosen, such as $C = \{2^{-10}, 2^{-9}, \ldots\}, \gamma = \{2^{-10}, 2^{-9}, \ldots, 2^{9}, 2^{10}\}$. In this way, C and γ constitute a two-dimensional grid. The data is trained and tested with every combination (C, γ) using the ISVM and the accuracy of the results for every combination is recorded. The combination with the highest accuracy is chosen as the penalty parameter.

C. SITUATION CLASSIFICATION

The SC for a robotic soccer game is a multi-classification problem with large-scale data. It includes situation sensing and state aggregation. Vectors with the values of defined factors representing the features of the game accommodate the sensory information on situation encountered. The process of ISVM takes charge of state aggregation. For large-scale data training, due to the limitations of computer memory capacity, a conventional SVM cannot solve this large-scale data problem. This paper proposes a training method using a SVM, which uses a PSO algorithm to find the optimal solution. To solve a multi-classification problem, conventional methods, such as "1-a-1" and "1-a-r" [33], have high training complexity and low classification accuracy. For a multi-classification problem, a multiple classification method that uses a decision tree (DT) is proposed [34].

For a PSO, the feasible solution of the problem corresponds to the position of a particle in the search space. Each particle has an adaptive value that is determined by the objective function and has a velocity vector that determines the position of the particle at the next moment. The search process is optimized using a group of randomly initialized particles in an iterative way.

Assume that in the m-dimensional search space, there is a population $\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}\}$ that has n particles. The attribute of i^{th} particle and the state of i^{th} particle at time $t+1$ are:

$$v_i^{t+1} = v_i^t + c_1 r_1 (p_i^t - x_i^t) + c_2 r_2 (p_g^t - x_i^t) \tag{11}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{12}$$

where $i \in [1, n]$; r_1 and r_2 are random numbers that are uniformly distributed on the interval (0,1); c_1 and c_2 are learning factors and $c_1 = c_2 = 2$. p_i^t is individual extreme of i^{th} particle at time t, p_g^t is global extreme of the whole population at time t. The principle for a PSO is shown in Fig. 2.

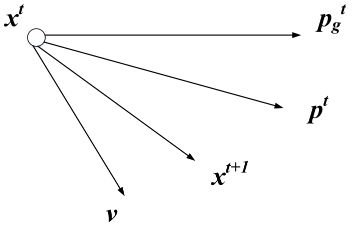


FIGURE 2. The principle of a PSO.

In essence, a SVM determines a support vector's coefficient, which is a dual problem that is solvable using quadratic programming algorithms, such as the Newton method or the conjugate gradient method. These methods are limited by the capacity of the computer and cannot solve this dual problem for large data sets. This paper proposes a method that uses a PSO to address this quadratic programming problem. Every support vector corresponds to a particle in the PSO. The process of finding a support vector's coefficients is treated as an optimization problem which is tackled by a PSO in this work. As for the fault detection and diagnosis always seen in the data process of support vector machines, the PCA-SVM and Cross- Validation to discover the fault [35], [36] can be applied to the support vector machines to alleviate the influence of defects.

The steps for the realization of a PSO-SVM are shown in Algorithm 1.

Algorithm 1 PSO-SVM

```
Definition
$X'$ := the support vector set in the training sample set
$X$ := thewhole particle population,which consists of $X'$
$T_{\max}$ := the maximum number of iterations
$T$ := the global position threshold
$F(x)$ := the fitness function
$F_i^t(x)$ := the fitness value of $i^{th}$
Initialization
$c_1$ ←Initial parameters for particle swarm;
$c_2$ ←Initial parameters for particle swarm;
$v$ ←Initial velocity matrix;
$p_i$ ←Initial optimal position for $i^{th}$ particle;
$p_g$ ←Initial global optimal position;
$F(x) \leftarrow \frac{1}{m}\sum_{i=1}^{m}(f_i - y_i)^2$
$t \leftarrow 1$;
Repeat $t++$
calculate the fitness value $F_i^t(x)$ of $i^{th}$ particle by $F(x)$;
compare $p_i$ and $p_g$ with $F_i^t(x)$, update $p_i$ and $p_g$;
update the state of the whole particle population by
  (11)(12)
obtain a new set of parameters in SVM
until $t > T_{\max}$ or $P_g > T$
```

A decision tree classifier is also known as a multistage classifier. It uses a binary tree classifier to transform a complex

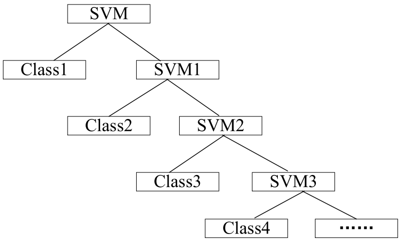


FIGURE 3. The architecture for the DT-SVM.

multi-classification problem into several simple classification problems. The architecture for a decision tree is used for a SVM (DT-SVM), as shown in Fig. 3. For a multiple-class, $K(K > 2)$ classification question, this algorithm constructs K-1 two-class classifiers in the training stage. The samples for the m^{th} classifier consist of sample subsets that have a classification label. Changing the sample label for a class by +1 changes the label for the other classes by -1.

$$y_i = \begin{cases} +1, & (y_i = m) \\ -1, & (y_i > m) \end{cases} \tag{13}$$

According to (3), the decision function of m^{th} SVM classifier is:

$$f_m(x) = \mathrm{sgn}(\sum_{i=1}^{m} \alpha_i^m y_i^m K(x_i^m * x) + b^m) \tag{14}$$

For unknown samples, the decision output value in each of the two classes of the SVM classifiers is calculated to make classification decision using (15).

$$f(x) = \begin{cases} m, f^1(x) = f^2(x) = \ldots = f^{m-1}(x) = -1, \\ \qquad\qquad\qquad\qquad\qquad f^m(x) = +1 \\ K, f^1(x) = f^2(x) = \ldots = f^{K-1}(x) = -1 \end{cases} \tag{15}$$

With the structure of decision trees, the robustness of the ISVM is improved further. For unknown samples, we can use formula (15) to provide its category value.

The process that uses a decision tree [21] and PSO-SVM for a multiple-classification question is shown in Algorithm 2.

Algorithm 2 ISVM

```
Definition
  $T$ := training the sample set
  $K$ := the number of environment descriptions
  $T_m$ := the $m^{th}$ positive samples set
  $f_m(x)$ := the decision function for $m^{th}$ classifier
$m \leftarrow 1$;
Repeat $m++$
  The $m^{th}$ class samples are selected from $T$ as positive
    samples set $T_m$
  The remaining samples form negative set $T - T_m$
  Utilize PSO-SVM to train on set $T$ and obtain the $f_m(x)$.
  $T \leftarrow T - T_m$
until $m > K - 1$
```

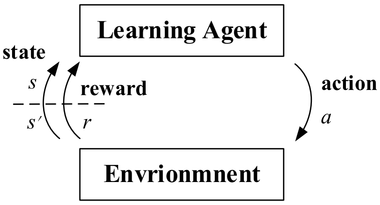


FIGURE 4. The architecture for reinforcement learning.

D. ADAPTIVE DECISION-MAKING ALGORITHM USING RL (ADMA-RL)

Reinforcement learning is a machine learning algorithm that is used to determine an optimal action-selection policy for a Markov Decision Process (MDP) [26]. In this algorithm, agents interact with the environment and select a specific action for a specific state. A trial-and-error method is used to gather experience and to train the policy. A policy is a rule that a learning agent follows to select possible actions for the state in which it finds itself. When the policy has been learnt, agents use it to make a series of optimal action sequences to achieve the goal. The process for agents interacting with the environment is shown in Fig. 4. An agent perceives the current state, $s \in \mathbf{S}$, from the environment and makes a decision action using the learnt policy. S is the state set and A is the action set. When the agent takes an action, a, a state transition occurs from s to s'. A reward, r, is used to evaluate whether the action is good. If an implicit or explicit reward function induces a good reward, the agent strengthens the possibility of the selected action and vice versa. Q-learning, which is a model-free action-dependent heuristic dynamic programming method, does not require knowledge of the plant model, but utilizes experience that is derived from on-line state-action data for the controlled system. Q-Learning is a model-free reinforcement learning algorithm that is used to learn in discrete Markov decision process environments. Q-Learning allows learning agents to interact with the environment, even though the learning agents have no prior knowledge. The trial-and-error method is used to gather experience and to train the action (decision) policy. When the policy has been trained, the learning agent uses it to make a series of optimal action sequences to achieve the goal. During the interaction, as shown in Fig. 4, a Q-learning agent perceives the current state, s_t, from the environment when taking action $\mathrm{a_t}$. The agent then gets the reward r_t. The value of the state-action pairs Q-value for the agent is then recoded. The Q-value is updated by Eq. (16).

The update criterion for the *Q-Value* is:

$$Q^t(s, a) = (1 - \alpha)Q^{t-1}(s, a) + \alpha(r + \gamma \max_{a'} Q^{t-1}(s', a')) \tag{16}$$

where $\alpha(0 \leq \alpha \leq 1)$ is the learning rate and $\gamma(0 \leq \gamma \leq 1)$ is the discount factor. When the agent has trained for a sufficient number of episodes, the estimated Q-values approach the optimal values. The agent develops an optimal or a semi-optimal policy using these approximated Q-values.

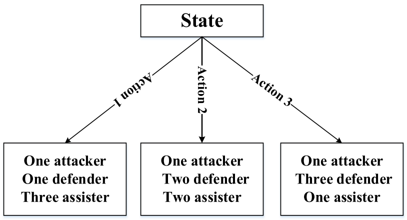


FIGURE 5. **The actions for strategy selection.**

For an SC based on ISVM, there are n_s states. For more efficient decision-making, there are three actions, which indicate different patterns of players' roles, in a state that is similar to a leaf in the decision tree, as shown in Fig. 5.

$$r = \begin{cases} 100 & \text{if } \textit{get score} \\ -100 & \text{if } \textit{lose score} \\ 100 * (\sum\limits_{i=1}^{K} \sqrt{{}^m x_i^2 + {}^m y_i^2} \\ \quad - \sum\limits_{i=1}^{K} \sqrt{{}^o x_i^2 + {}^o y_i^2})/\sqrt{x_b^2 + y_b^2} & \textit{otherwise} \end{cases} \tag{17}$$

The reward function is defined by three conditions: getting a point, losing a point and a normal reward. Generally speaking as formula (17).

Where $({}^m x_i, {}^m y_i)$ is the coordinate of the user's i^{th} robot and K denotes the number of the user's robots on the field, $({}^o x_i, {}^o y_i)$ is the coordinate of the opponent's i^{th} robot and K denotes the number of the opponent's robots on the field and (x_b, y_b) is the coordinate of ball.

Semi-Markov decision processes (SMDPs) are used in modeling stochastic control problems arising in Markova dynamic systems [37]. A sojourn time in a state is defined as the time cost by RL to transit from one state to the next state and is a general random variable. In a state, the agent may take a serial of similar actions before transiting into the next state. Although the robot soccer game is regarded as an SMDP, the intrinsic characteristic of Q-learning is a model-free method which captures the environmental feedbacks by physically interacting with the environment. In such a way, the uncertainty resulting from modeling can be diminished purposely in this case [38], [40]. The strategy that is used in this paper is different to that for a traditional Q-Value updating strategy, where

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha(r + \gamma \max_{a_{t+1}} Q_t(s_{t+1}, a_{t+1})) \tag{18}$$

where s_t represents the state in the t^{th} time slice; α is learning rate; γ is discount rate; and s_{t+1} is the next state of s_t for a_t and r is immediate reward.

Assuming that in one epoch, the state, **s**,is transformed into $\mathbf{s}'$ after λ learning cycle by the same actions as shown in Fig. 6, the action value updating strategy in this paper is:

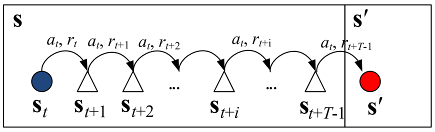


FIGURE 6. **An illustrative diagram of a semi-MDP.**

$$Q_{t+\lambda-1}(s, a) = r_{t+\lambda-1} + \gamma \max_{a'} Q(s', a') \tag{19}$$

$$Q_{t+i}(s, a) = r_{t+i} + \gamma Q_{t+i+1}(s, a), 0 \leq i \leq \lambda - 2 \tag{20}$$

$$Q_{t+\lambda}(s, a) = Q_t(s, a) + \alpha(\sum\nolimits_{k=0}^{\lambda-1} Q_{t+k} \Big/ \lambda - Q_t(s, a)) \tag{21}$$

The learning system is then executed and the corresponding ADMA-RL is shown in Algorithm 3.

Algorithm 3 ADMA-RL

```
Definition
$\mathbf{s}_t$ := calculated current state by $ISVM$
$a_t$ := current selected decision scheme with maximum Q-value
$T_{\max}$ := the maximum competition time
max() := finding the action with maximum Q-value
$t \leftarrow 1$;
Repeat $t++$
    $\mathbf{s}_t \leftarrow$ discretizing_state_space;
    $r \leftarrow$ reward_function();
    $a_t \leftarrow$ max($Q$(s,a));
    updating Q-value by (19)(20)(21);
until $t > T_{\max}$
```

III. SIMULATIONS AND EXPERIMENTS

Comparisons are shown to demonstrate the efficiency of the proposed method. Firstly, the parameters, C and γ, in the SVM are selected using a grid search method [32]. Then, "1-a-1", "1-a-r" [33] and the ISVM are compared to demonstrate that the ISVM has better classification accuracy and a shorter classification time. In order to demonstrate the practicality and efficiency of RL, the proposed AMDA-RL is tested using a robotic soccer platform. A decision-making method that uses a fuzzy neural network (DM-FNN) [11], a decision-making method that uses a Bayesian SOM neural network (DM-BSOM) [39] and a decision-making method that uses an Adaptive Strategy Selection Method (DM-AD) are then compared in robotic soccer games to prove the efficiency of the DM-AD. Finally, the proposed method is used for a robotic soccer game and the results are presented.

The experiments are implemented using a robotic soccer simulation platform where there are ten two-wheeled robots - five user's robots and five opponent's robots - as shown in Fig. 7(a). The size of this platform is 1000pixel∗640pixels and the time for each competition is $T_{\max} = 400s$.The Parameters for simulation platform are shown in Table 1.

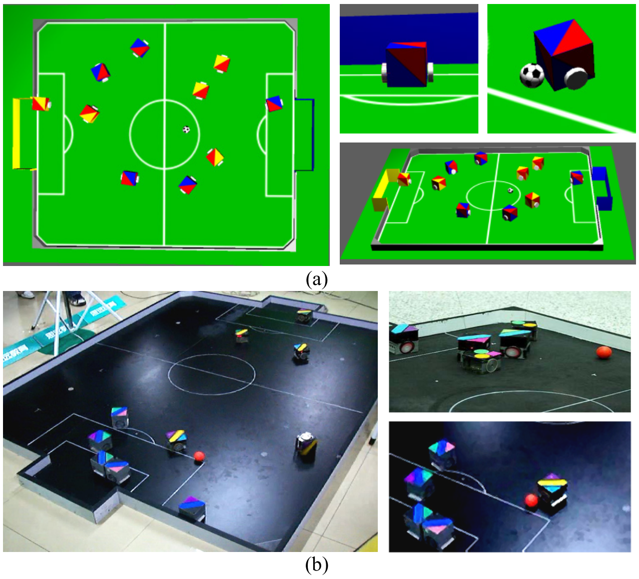


FIGURE 7. The experimental platforms. (a) The simulation platform. (b) The real environment.

TABLE 1. Parameters for simulation platform.

Category	Value
size (pixels)	1000*640
Time (sec.)	400s

During the test, the game uses three decision-making models: AO (aggressive offense), CD (conservative defense) and MP (moderate play). In order to demonstrate the practicality of the proposed method, a real environment is show in Fig. 7(b). In Fig. 7(b), there are ten real wheeled robots: five user's robots and five opponent's robots. The user's robots compete with the opponent's.

A. TEST RESULT FOR THE ADMA-RL

For the SVM, the penalty parameters, C *and* γ, in the RBFs are important. This paper uses the grid search method to select them. Through the selection of this method and with expert knowledge and experience, When $C = 512, \gamma = 2$, the test accuracy is 98%. Changing the parameter again has little influence to SC. Therefore,$C = 512, \gamma = 2$ are chosen as the experimental parameters for ISVM. The parameters for ISVM are shown in Table 2.

TABLE 2. Parameters for ISVM.

Category	Value
C	512
γ	2

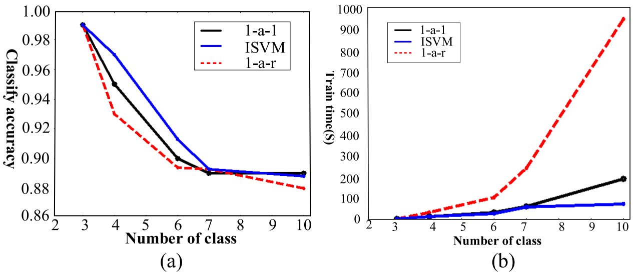


FIGURE 8. A comparison of the performance of the three algorithms. (a) Classify accuracy. (b) Train time.

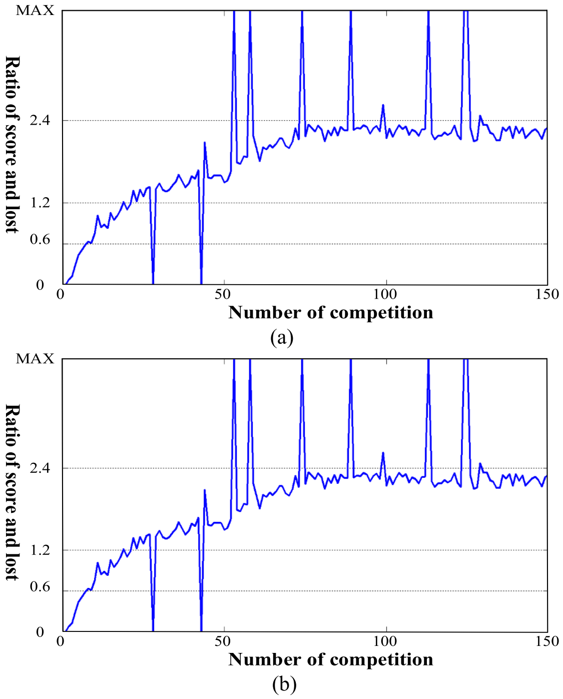


FIGURE 9. Ratios over competitions. (a) The change in the ratio for the DM-AD and the AO. (b) The change in the ratio for the DM-AD and the CD.

Fig. 8(a) shows the comparisons between 1-a-1, 1-a-r, and ISVM in accuracy. From the figures it can be observed that the classification accuracy for all three methods decreases, when the number of classification classes increases from three to ten, but the classification accuracy for the ISVM is higher than that for the other two methods. Fig. 8(b) shows that the training time for the ISVM is shorter than that the counter-parts' and less sensitive to the increment of the number of the classes. This is because all samples are used in every training cycle for the 1-a-r method, its training time is longest. The training time of the 1-a-1 method is similar to the ISVM when the number of classes is below seven, but, beyond seven, its training time is obviously longer than ISVM. The ISVM can excel over SVM in terms of computational complexities. Comparatively, ISVM has higher classification accuracy and faster training speed, especially in the case of large-scale data.

To demonstrate the efficiency of the proposed method in robot soccer games, the AO and the CD were used as the opponents, respectively, during learning. Each competition was 400s and the ratios of the points won and lost over the games are shown in Fig. 9. The ratio for scores and losses was calculated. When points lost is 0, "MAX" denotes the ultimate ratio in the Figure. The intrinsic parameters for RL are shown in Table 3.

TABLE 3. Intrinsic parameters for RL.

Identified	Value
n_s	10
n_a	8
γ	0.75
α	0.70

TABLE 4. Confrontation result.

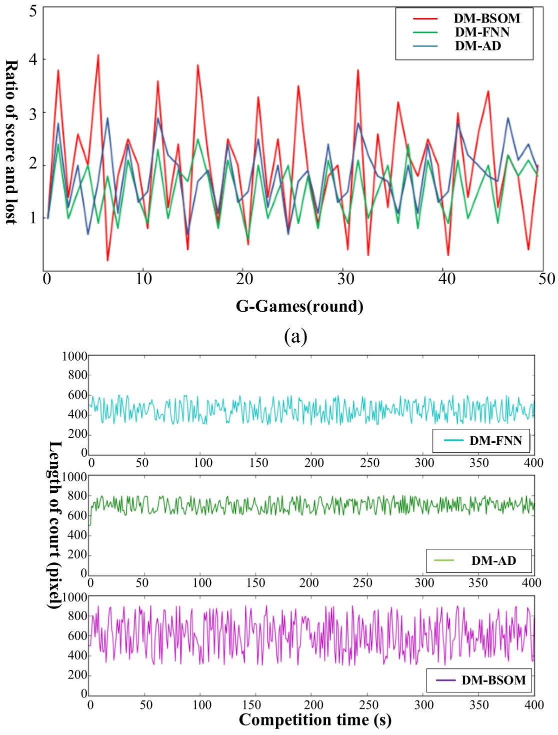
Confrontation Factories	DM-FNN	DM-BSOM	DM-AD
Total ball-controlling rate	40.9%	60.7%	70.2%
Degree of threat to gate	30/855	70/1589	50/1202
Total number of losing balls	20	58	30

Fig. 9 shows that when the ADMA-RL competes with the AO, the ratio rises gradually and reaches a plateau around the 70th competition. When the ADMA-RL competes with the CD, the ratio rises gradually and reaches a plateau around the 60th competition. It is seen that along when continuous learning is used, ADMA-RL becomes more intelligent and gets more scores in one competition. The efficiency of the ADMA-RL is proven.

B. A COMPARISON OF DECISION-MAKING METHODS

The DM-BSOM, DM-FNN and DM-AD were evaluated in 50 robotic soccer games and each competition was 400s. These models competed with the MP model. The results are shown in Table 4 and Fig. 10. Table 4 shows the confrontation result. Fig. 10(a) shows score ratio for each match, which shows the effect of the strategy and the model. In Fig. 10(a), the x-axis is the number of competitions and y-axis is the score ratio for each match. In Fig. 10(b), the x-axis is time and the y-axis is the ball's abscissa. In robotic soccer the ball's position shows the team's competitiveness and ability to defend.

Table 4 and Fig. 10 show that the DM-AD, DM-FNN and DM-BSOM have an advantage in robotic soccer decision-making over the MP model. However, the model that uses DM-FNN requires much manual intervention. It is difficult for DM-FNN to achieve self-adaptive learning and it cannot predict situations precisely. It is always in a defensive state and has only a short time to control the ball or choose a conservative competition strategy in confrontations. This results in a lower score ratio and offers no threat to an opponent. Although DM-BSOM is more aggressive and scores better, it uses regions so it is sensitive to regional divisions. Therefore, it chooses different offensive and defensive strategies frequently for different regions, which can lose the match because there is poor stability. However, the proposed method applied the SC and the SS to decision making for robotic soccer games. In the SC, the ISVM with the PSO as an accelerator is applied to decision tree structure. The accurate and timely situation classification of the ISVM are critical for efficient strategy selection so as that the SS can have a better ability for self-adaptive learning and chooses strategies based on a concise discrete state space. This results in good stability and a balanced offensive and defensive strategy, compared with the other two methods.



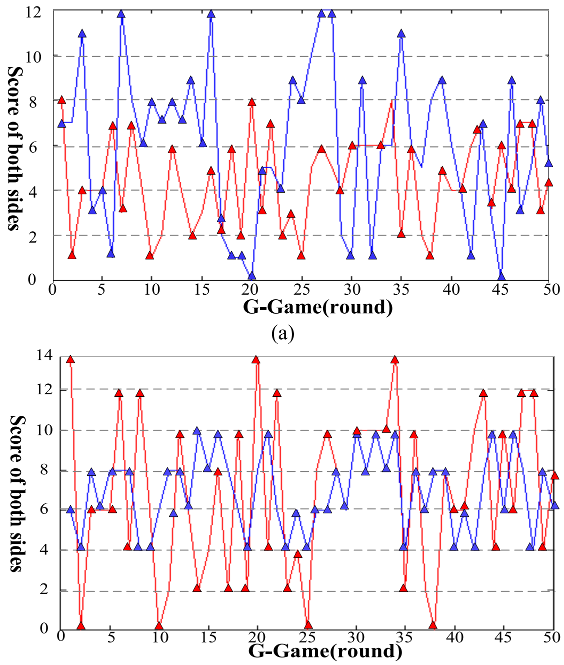
(a)

(b)

FIGURE 10. The score ratio and average trajectories for the ball. (a) The ratio of the score and losing. (b) The average of the trajectories.

C. EXPERIMENTS

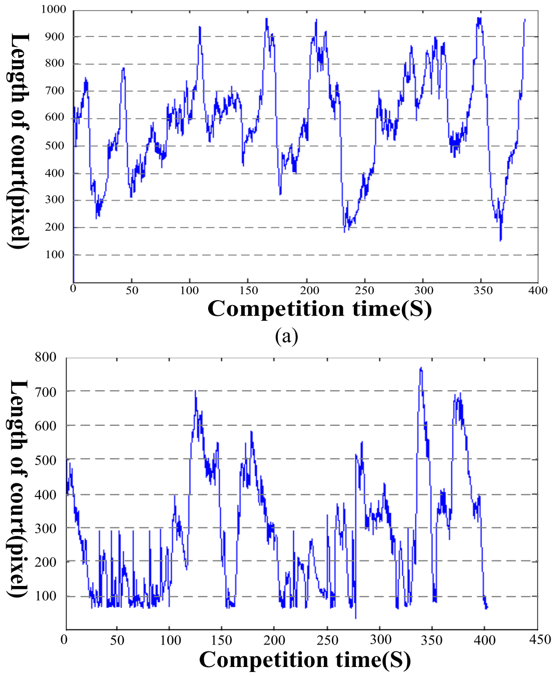
The decision method was applied to an actual game for two types of situations: advantages and disadvantages. Two teams competed with the opponent team. The opponent team used the traditional method, which selects strategy based on the position of the ball. The statistical score difference between the two sides is about 50 competitions. The result is shown in Fig. 11, where blue line denotes the team that uses the adaptive decision method and the red line denotes the team that uses the traditional method. The x-axis denotes the number of competitions and the y-axis denotes the score for both sides. Fig. 11(a) shows that when there are disadvantages, the team that uses the adaptive decision method has a larger score difference than the team that uses the traditional method. The team that uses the adaptive decision method can lose because it takes an aggressive strategy to make a score. This is the only way to make a score in a short time. Fig. 11(b) shows that when there are advantages, the team that uses an adaptive decision method uses a conservative strategy to control a situation, unlike the traditional team. This shows that the learning system performs well in both



(a)

(b)

FIGURE 11. **The difference in the score. (a) The score difference in disadvantages. (b) The score difference in advantages.**



(a)

(b)

FIGURE 12. **The trajectories for the ball. (a) The trajectories of the ball in advantages. (b) The trajectories of the ball in disadvantages.**

disadvantageous and advantageous situations. The satisfactory results that are obtained demonstrate that the learning system performs well.

The x-axis denotes times and the y-axis denotes the ball's abscissa position. 500 is midfield and values less than 500 on the y axis denote the opponent's area.

Fig. 12(a) shows that, when there are advantages, the trajectory of the ball always fluctuates above the middle line of the field, the horizontal line of 500 pixels. This means that the conservative strategy moves the ball to the opponent's half field so as to avoid losing a point. Fig. 12(b) shows that when there are disadvantages, the trajectory of the ball fluctuates around the region of 100-400 pixels. That means that the user's team uses the aggressive strategy that the situation requires. Only this match strategy achieves a balance in attack and defense.

IV. CONCLUSION

Problems such as uncertain knowledge representation and high dimensional data representation exist in decision-making methods. This paper proposes an Adaptive Strategy Selection Method with Reinforcement Learning for Robotic Soccer Games. In the SC subsystem, the ISVM, where the PSO is used to speed up the training rate of the SVM, is an efficient situation classification. The main objective of the ISVM is to evaluate situations and the rewards. All of the learning uses reinforcement learning. The ADMA-RL enacts the learning process in the SS subsystem so decisions are made adaptively as the environment changes. The ADMA-RL approach allows more effective learning in the SS subsystem. Using RL processes, the proposed method learns the best strategies based on the identified compositions of factors that represent a variety of the situations in the field. To demonstrate the effectiveness of the proposed method, experiments using simulation and a real environment are performed. The players' roles are designed to assign roles as actions using the learnt policy. The strategy is composed of role assignment that defines various numbers of attackers, defender, and assistants. All of these are independent of each other. If one displays a fault, it is easily replaced by another. Dynamic role assignment means that for the proposed approach, the role of each robot is changeable. The learned policy decides all of the robots' roles. All of the robots' roles can be changed according to the situation on the field. More flexible situation aggregation means that the ISVM method can be used for factor classification and immediate rewards. The aggregation eliminates interference from noise.

In the robot soccer filed, the commands asserted from the learning system to individual robots are actually carrying the mixture of event-triggered (strategy selection) and time-triggered (robot motor command) task sets, which communicate over protocols consisting of both static and dynamic phases. In the future work, we should work on how to partition and schedule the system functionality into time- triggered and event-triggered domains and the optimization of parameters corresponding to the communication protocol.

REFERENCES

[1] M. J. Wooldridge, "An Introduction to Multi-Agent Systems," *Wiley Sons*, vol. 4, no. 2, pp. 125–128, 2009.

[2] K.-S. Hwang, S.-W. Tan, and C.-C. Chen, "Cooperative strategy based on adaptive Q-learning for robot soccer systems," *IEEE Trans. Fuzzy Syst.*, vol. 12, no. 4, pp. 569–576, Aug. 2004.

[3] J. P. Mendoza, J. Biswas, P. Cooksey, R. Wang, and S. Klee, "Selectively reactive coordination for a team of robot soccer champions," in *Proc. Assoc. Adv. Artif. Intell. Conf. (AAAI)*, 2016, pp. 3354–3360.

[4] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 427–438, Feb. 2013.

[5] S. Barrett and P. Stone, "Cooperating with unknown teammates in complex domains: A robot soccer case study of ad hoc teamwork," in *Proc. Assoc. Adv. Artif. Intell. Conf. (AAAI)*, 2015, pp. 2010–2016.

[6] H. Shi, Z. Lin, S. Zhang, X. Li, and K.-S. Hwang, "An adaptive decision-making method with fuzzy Bayesian reinforcement learning for robot soccer," *Inf. Sci.*, vols. 436–437, pp. 268–281, Apr. 2018.

[7] Z. Xu, K. Gao, T. M. Khoshgoftaar, and N. Seliya, "System regression test planning with a fuzzy expert system," *Inf. Sci.*, vol. 259, no. 3, pp. 532–543, 2014.

[8] W. Tao and G. Zhang, "Trusted interaction approach for dynamic service selection using multi-criteria decision making technique," *Knowl.-Based Syst.*, vol. 32, no. 8, pp. 116–122, 2012.

[9] C.-Y. Fan, P.-C. Chang, J.-J. Lin, and J. C. Hsieh, "A hybrid model combining case-based reasoning and fuzzy decision tree for medical data classification," *Appl. Soft Comput.*, vol. 11, no. 1, pp. 632–644, 2011.

[10] H. P. Huang and C. C. Liang, "Strategy-based decision making of a soccer robot system using a real-time self-organizing fuzzy decision tree," *Fuzzy Sets Syst.*, vol. 127, no. 1, pp. 49–64, 2002.

[11] K. G. Jolly, R. S. Kumar, and R. Vijayakumar, "A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits," *Robot. Auto. Syst.*, vol. 57, no. 1, pp. 23–33, 2009.

[12] Y. Qin and H. R. Karimi, "A mahalanobis hyperellipsoidal learning machine class incremental learning algorithm," *Abstract Appl. Anal.*, vol. 2014, no. 1, pp. 1–5, 2014.

[13] A. T. Misirli and A. B. Bener, "Bayesian networks for evidence-based decision-making in software engineering," *IEEE Trans. Softw. Eng.*, vol. 40, no. 6, pp. 533–554, Jun. 2014.

[14] Y. Hu, Y. Gao, and B. An, "Accelerating multiagent reinforcement learning by equilibrium transfer," *IEEE Trans. Cybern.*, vol. 45, no. 7, pp. 1289–1302, Jul. 2015.

[15] K. S. Hwang, Y. J. Chen, and C. H. Lee, "Reinforcement learning in strategy selection for a coordinated multirobot system," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 6, pp. 1151–1157, Nov. 2007.

[16] N. D. Nguyen, T. Nguyen, and S. Nahavandi, "System design perspective for human-level agents using deep reinforcement learning: A survey," *IEEE Access*, vol. 5, pp. 27091–27102, 2017.

[17] T. Mori and S. Ishii, "Incremental state aggregation for value function estimation in reinforcement learning," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 5, pp. 1407–1416, Oct. 2011.

[18] L. A. Prashanth and S. Bhatnagar, "Reinforcement learning with function approximation for traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 412–421, Jun. 2011.

[19] B. Chen, A. Zhang, and L. Cao, "Autonomous intelligent decision-making system based on Bayesian SOM neural network for robot soccer," *Neurocomputing*, vol. 128, pp. 447–458, Mar. 2014.

[20] S. Abe, "Fuzzy support vector machines for multilabel classification," *Pattern Recognit.*, vol. 48, no. 6, pp. 2110–2117, 2015.

[21] K. C. Lin, K. Y. Zhang, Y. H. Huang, and J. C. Hung, "Feature selection based on an improved cat swarm optimization algorithm for big data classification," *J. Supercomput.*, vol. 72, no. 8, pp. 3210–3221, 2016.

[22] S. Shilaskar, A. Ghatol, and P. Chatur, "Medical decision support system for extremely imbalanced datasets," *Inf. Sci.*, vol. 384, pp. 205–209, Apr. 2017.

[23] S. Barrett and P. Stone, "Cooperating with unknown teammates in complex domains: A robot soccer case study of ad hoc teamwork," in *Proc. Assoc. Adv. Artif. Intell. Conf. (AAAI)*, 2015, pp. 2010–2016.

[24] J. L. Lin *et al.*, "Gait balance and acceleration of a biped robot based on Q-learning," *IEEE Access*, vol. 4, pp. 2439–2449, 2016.

[25] H. Wang, X. Wang, X. Hu, X. Zhang, and M. Gu, "A multi-agent reinforcement learning approach to dynamic service composition," *Inf. Sci.*, vol. 363, pp. 96–119, Oct. 2016.

[26] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.

[27] S. Nadarajah and K. Sundaraj, "Vision in robot soccer: A review," *Artif. Intell. Rev.*, vol. 44, no. 3, pp. 289–310, 2015.

[28] A. Ukil, "Support vector machine," *Comput. Sci.*, vol. 1, no. 4, pp. 1–28, 2002.

[29] S. Amari and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Netw.*, vol. 12, no. 6, pp. 783–789, 1999.

[30] J. Mei, M. Liu, H. R. Karimi, and H. Gao, "Logdet divergence-based metric learning with triplet constraints and its applications," *IEEE Trans. Image Process.*, vol. 23, no. 11, pp. 4920–4931, Nov. 2014.

[31] A. Mehrsai *et al.*, "Application of learning pallets for real-time scheduling by the use of radial basis function network," *Neurocomputing*, vol. 101, no. 10, pp. 82–93, 2013.

[32] J. A. Walsh, P. K. Romano, B. Forget, and K. S. Smith, "Optimizations of the energy grid search algorithm in continuous-energy Monte Carlo particle transport codes," *Comput. Phys. Commun.*, vol. 196, pp. 134–142, Nov. 2015.

[33] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[34] E. Montañés, J. Barranquero, and J. DÃez, "Enhancing directed binary trees for multi-class classification," *Inf. Sci.*, vol. 223, no. 2, pp. 42–55, 2013.

[35] F. Wu, S. Yin, and H. R. Karimi, "Fault detection and diagnosis in process data using support vector machines," *J. Appl. Math.*, vol. 8, pp. 1–9, Mar. 2014.

[36] S. Yin *et al.*, "Study on support vector machine-based fault detection in tennessee eastman process," *Abstract Appl. Anal.*, vol. 2, pp. 1–8, Apr. 2014.

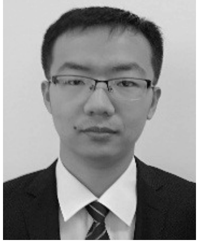
[37] N. Marchenko and C. Bettstetter, "Cooperative ARQ with relay selection: An analytical framework using Semi-MDP processes," *IEEE Trans. Veh. Technol.*, vol. 63, no. 1, pp. 178–190, Jan. 2014.

[38] Y. Wang, Y. Xia, H. Shen, and P. Zhou, "SMC design for robust stabilization of nonlinear Markovian jump singular systems," *IEEE Trans. Autom. Control*, vol. 63, no. 1, pp. 219–224, Jan. 2018.

[39] S. Ribaric and T. Hrkac, "A model of fuzzy spatio-temporal knowledge representation and reasoning based on high-level Petri nets," *Inf. Syst.*, vol. 37, no. 3, pp. 238–256, 2012.

[40] H. Shi, X. Li, K.-S. Hwang, W. Pan, and G. Xu, "Decoupled visual servoing with fuzzy Q-learning," *IEEE Trans. Ind. Informat.*, vol. 14, no. 1, pp. 241–252, Jan. 2018.

HAOBIN SHI received the Ph.D. degree from Northwestern Polytechnical University, China, in 2008. He is an Associate Professor with the School of Computer Science, Northwestern Polytechnical University, and a Visiting Scholar with the Electrical Engineering Department, National Sun Yat-sen University, Taiwan. His research interests include intelligent robots, decision support systems, artificial intelligence, multi-agent systems, and machine learning. He is the Director of the Chinese Association for Artificial Intelligence.



ZHIQIANG LIN received the B.S. degree from the School of Computer Science, Northwestern Polytechnical University, China, in 2016, where he is currently pursuing the master's degree. His research interests include intelligent control, electrical engineering, machine learning, and embedded system design.

KAO-SHING HWANG (M'93–SM'09) received the M.M.E. and Ph.D. degrees in electrical and computer engineering from Northwestern University, Evanston, IL, USA, in 1989 and 1993, respectively. He was with National Chung Cheng University, Taiwan, from 1993 to 2011. He was the Deputy Director of the Computer Center from 1998 to 1999, the Chairman of the Electrical Engineering Department from 2003 to 2006, and the Director of the Opti-mechatronics Institute of the University from 2010 to 2011. He is a Professor with the Electrical Engineering Department, National Sun Yat-sen University, and an Adjunct Professor with the Department of Healthcare Administration and Medical Informatic, Kaohsiung Medical University, Taiwan. He has been a fellow of the Institution of Engineering and Technology. His research interests include methodologies and analysis for various intelligent robot systems, machine learning, embedded system design, and ASIC design for robotic applications.

SHIKE YANG is currently pursuing the master's degree with the School of Computer Science, Northwestern Polytechnical University, China. His research interests include intelligent control, electrical engineering, machine learning, and embedded system design.

JIALIN CHEN received the B.S. degree from the School of Software and Microelectronics, Northwestern Polytechnical University, China, in 2016, where he is currently pursuing the master's degree with the School of Computer Science. His research interests include intelligent control, electrical engineering, machine learning, and embedded system design.