# Learning Soccer Drills for the Small Size League of RoboCup

Carlos Quintero[1], Saith Rodríguez[1], Katherín Pérez[1], Jorge López[1], Eyberth Rojas[1], and Juan Calderón[1,2]

[1] Universidad Santo Tomás, Colombia
{carlosquinterop,saithrodriguez,andrea.perez,
jorgelopez,eyberthrojas}@usantotomas.edu.co
[2] University of South Florida, Tampa FL, USA
juancalderon@mail.usf.edu

**Abstract.** This paper shows the results of applying machine learning techniques to the problem of predicting soccer plays in the Small Size League of RoboCup. We have modeled the task as a multi-class classification problem by learning the plays of the STOx's team. For this, we have created a database of observations for this team's plays and obtained key features that describe the game state during a match. We have shown experimentally, that these features allow two learning classifiers to obtain high prediction accuracies and that most miss-classified observations are found early on the plays.

**Keywords:** Machine Learning, RoboCup SSL, Learning soccer drills, Neural Networks, Support Vector Machines

## 1 Introduction

One of the big challenges in robotics have been on how to provide the robots with the capability of efficiently using the available information to make decisions without human intervention [5, 13]. In a sense, this process is related to the way humans do it. First, there is a step of raw data acquisition from the environment; then, these data are processed and finally a decision is made usually based on a specific reasoning structure or a previous similar experience.

The Small Size League (SSL) of the RoboCup international initiative has become a key scenario to push forward the state of the art in this matter as it provides an ideal framework where all the information of the current game state is available to a global controller that acts as a human coach with the additional capability of being able to transmit customized instructions to each player in the field [16]. Under this scenario, the ability of a team to predict or understand its opponent's strategies during a game seems crucial.

The approach that we propose in this paper resembles that of the human soccer in the following way: a group of experts (coaching group) of a given team is in charge of studying and learning, before the game, the strategies or plays that the opponent team has utilized against other teams in previous games. Based on this information, their task is to define specific strategies or actions to defend the team against such plays if these are used during the game. Furthermore, the team must have the ability of predicting which of the studied strategies will be implemented by the opponent team.

Other researchers have pointed out the importance of such ability mostly in the simulation league [12], where the conditions and gameplay are in general different than those found in the SSL. The SSL gameplay is usually fast due to the velocities of the robots and the ball and the relatively small size of the field. We have taken this into consideration in order to propose a learning framework in which the strategies to be learned from the opponent team correspond to soccer drills. These are plays that teams usually execute after a free kick command is issued from the referee box. In this work, we model the situation as a classification problem and use machine learning techniques to perform the task. We also evaluate the performance of the learning machines on unseen data and assess through simulated experiments whether the machines are able to correctly predict the drills or not.

The remainder of this paper is as follows: The following section shows the work that has been carried out by other researchers in similar environments. Then, we show the details of our approach by describing how we model the problem as a classification task, then, showing the set of experiments that were designed to obtained a dataset suitable for the problem and finally describing how we implemented the automatic learners. Section 4 shows the results of such experiments and finally we conclude and comment on future work.

## 2 Related Work

Researchers all over the globe have strived for providing their robotic soccer teams with the ability of autonomously recognize the opponent's strategies. It seems that all researchers agree in the fact that such information may be highly valuable in order to define the strategy that the team should carry out in a given game situation. However, these works highly vary in the concept that they try to learn and the modeling tools utilized to describe the robotic soccer challenge. This section briefly shows some of these works and highlight the similarities and differences between our work and theirs.

Certain authors have considered to use reinforcement learning as modeling tool to learn multi-agent strategies [1–3]. In these works, the idea is to model the soccer challenge as a state space for sequential decision making based on achieving specific goals. A notable work is shown in [4], where the authors propose learning how to play keepaway soccer using reinforcement learning. Finally, in [5], the authors propose a hierarchical learning methodology to learn skill actions, dynamic role assignment and action selection.

Other common approach is to use Case-Based Reasoning (CBR), a strategy that uses recorded behaviors to compare with the current situation based on a given similarity measure. CBR approaches have been used for action selection in the four-legged league of RoboCup [6], to recognize game state, team formations and positioning of the goalie [7], the behavior of agents in the RoboCup Simulation league [8], action selection [9] and other classification goals [10]. Improvements on these techniques have also been proposed in [11, 9]. All these approaches share with our approach the use of historical data to predict unseen behaviors. However, their approach relies on finding a similarity between past and current situations instead of aiming at learning the underlying relationship in the data like ours.

Closest to our work we can find approaches where machine learning techniques such as decision trees, neural networks and SVMs are used as classifiers in order to predict adversary classes [13], opponent formations [14, 15], strategies [16], game situations for each actor within the field [17] and actions [18]. The greatest similarity between our work and theirs is the use of features that describe the game state within the field. However, the criteria to be learned is different. To the best of our knowledge, there is no work that aims at learning drills for the RoboCup Small Size League in the way we propose here.

## 3 Our approach

The Small Size League (SSL) is a highly traditional and widely known soccer league in the RoboCup initiative. In the league, a global vision system is in charge of acquiring the information of the current game state. This information is processed by a software framework whose main role is deciding in real time the actions that each robot should perform in the next period of time. Finally, this information is transmitted wirelessly to each team player who must be capable of successfully execute the commanded actions.

In the current state of the league each team is made of 6 players capable of moving up to 3m/s into a 4m x 6m field using a standard golf ball that reaches velocities of up to 8m/s. These features provide the league with a highly dynamic and fast gameplay unique among the other leagues within the RoboCup initiative. This scenario has defined a very specific game style in which the ball remains within the field for short periods of time. A large proportion of the game relies on what is known as drills in human soccer: sets of plays that start with a direct kick, free kick or corner kick that human teams create during the training sessions that are used later in the official games and seek to surprise the opponent team to score a goal.

In the SSL, the idea remains the same: a set of plays previously defined by a programmer to perform a sequence of actions that may lead to score a goal. However, in the SSL they play a more significant role since there are more opportunities to perform a drill. Furthermore, a high amount of the goals scored are achieved as consequence of the execution of a drill.

Based on this observation we have proposed to build a learning machine capable of predicting the opponent's drills during a SSL game. For this, we have modeled the problem as a multiclass classification task where each class corresponds to one drill and each observation corresponds the state of the game in the current time frame represented as a set of features. In this framework, each time frame is labeled with the corresponding drill and the learner must assign drills to new unseen frames.

## 3.1 Learning Soccer Drills

As discussed above, we have modeled the problem of learning soccer drills in the SSL of RoboCup as a classification problem, initially aimed at learning some plays of the STOx's team. We have identified and selected six drills for this team's participation in RoboCup 2013 that we considered could demonstrate a proof of concepts for the problem of learning drills. In general, each drill is characterized by a sequence of movements performed by the robots after the referee issues the order. The identified drills of the STOx's team are be described next:

- **Direct Kick (DK)**: Two attackers move towards the opponent area and stand in front of the goalie. Two defenders remain in their own area. The attacker kicks directly towards the goalie.
- **Pass to Opposite Attacker (POA)**: Two attackers move towards the opponent area and stand in front of the goalie. Two defenders initially remain in their own area together with their goalie. However, one of them moves forward and stands in the opposite side of the ball. Next, the two attackers split up, the defender goes back and the kicker passes the ball to the opposite attacker. See Fig. 1.
- **Pass to Peer Attacker (PPA)**: Two attackers move towards the opponent area and stand in front of the goalie. Two defenders initially remain in their own area together with their goalie. However, one of them moves forward and stands in the opposite side of the ball. Next, the two attackers open up, the defender goes back and the kicker pass the ball to the attacker closest to it. Unlike the POA, when the two attackers open up, the peer attacker also moves backward in order to receive the pass properly.
- **Pass to Opposite Defender (POD)**: Two attackers move towards the opponent area and stand in front of the goalie. Two defenders initially remain in their own area together with their goalie. However, one of them moves forward and stands in the opposite side of the ball. Next, the kicker pass the ball to such defender.
- **Pass to Middle Defender (PMD)**: Two attackers move towards the opponent area and stand in front of the goalie. Two defenders initially remain in their own area together with their goalie. However, one of them moves forward and stands ahead the middle of the field. Next, the kicker pass the ball to such defender. See Fig. 2.
- **Five Attackers (FA)**: All the team members move forward to the opponent's area and get ready to receive the pass. The kicker pass the ball to the robot that is farther away.

Fig. 1 and Fig. 2 show the most important steps of drills **POA** and **PMD** respectively as described above. The arrows show the movement of a robot or the ball from one figure to the next. This specific sequence of steps will allow

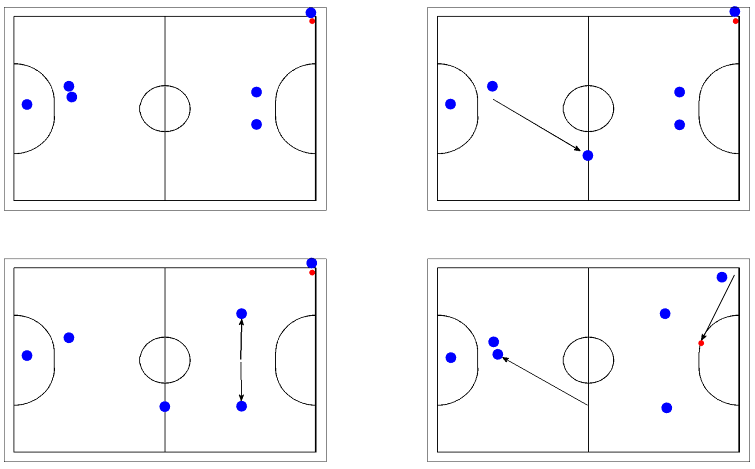

**Fig. 1.** Sequence of **Pass to Opposite Attacker** drill. The robots initially form in attack position. Then, one defender moves towards the center of the field. Next, the two attackers move apart of each other, the defender goes back and the kicker kicks the ball towards the opposite attacker.

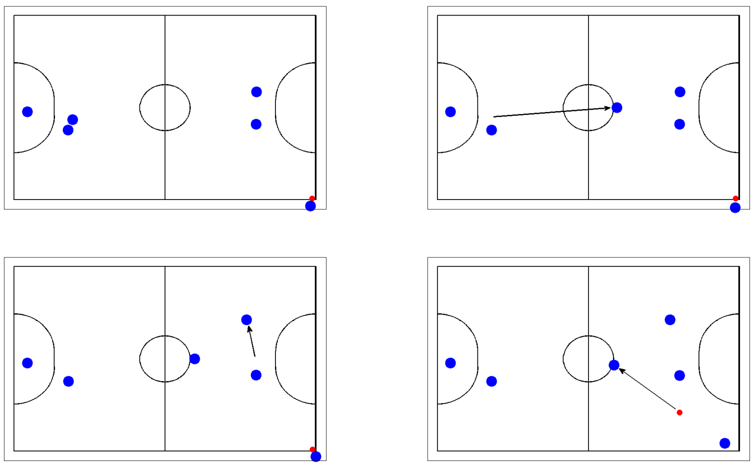

**Fig. 2.** Sequence of **Pass to Middle Defender** drill. The robots initially form in attack position. Then, one defender moves ahead the middle of the field. Next, the opposite attacker slightly moves and the kicker kicks the ball towards the defender in the middle.

the automatic classifier to predict the corresponding drill during a game. Notice that this approach requires at first human intervention in order to identify and label each frame's observation within the logs of the opponent team. Although this may seem burdensome, it mimics the behavior of human coaches before playing a game in real soccer. Furthermore, automatic identification and recognition of drills may be possible through the use of unsupervised learning techniques that find similarities in the data. However, such possibility requires further exploration.

### 3.2 The Dataset

We have built a dataset especially suited for the task at hand by using the grSim simulator. Although we do not show experiments in the real robots, we claim that there is little difference with respect to build the classifer with simulated data since the features used to perform the training are attributes such as positions and distances that are well emulated by the simulator. Furthermore, the simulator also considers the gaussian noise related to the artificial vision system.

Our program automatically begins the data acquisition immediately after a free kick command is issued from the Referee Box and outputs a log file with the information of each experiment. For each experiment, we have stored key features that describe the current game state and that provides enough information to allow the classifier perform its task. The features stored within the log files are mentioned below with the number of related features in brackets:

- Position of the ball (2)
- Position of each robot within the field (12)
- Orientation angle of each robot (6)
- Linear and angular velocities of each robot (18)
- Pairwise distances between robots (15)
- Distance between each robot and the ball (6)
- Time frame (1)

Some of these features are directly obtained from the information given by the simulator, such as the position of the ball and the robots as well as their orientation angles and time frame. The remaining features were computed based on the others in order to facilitate the classification task, such as distances between the robots and between the robots and the ball. A common practice in many team's drills is to create plays that depend on the opponent's positions, i.e., if one opponent has high chance of intercepting the pass, then the pass will go to a different player. We have performed experiments using the opponent's positions (not shown here) with a dynamic behavior and verified that these additional features make no difference in terms of the classifier's prediction accuracy. However, might be used to create dynamic defense strategies. For more complex plays sets, the opponent's positions could also be added to improve the learning capability of the classifier.

We have performed 24 repetitions of each drill with each defense strategy; 3 from the top of the field and 3 from bottom for a total of 144 runs. Each run records data at a rate of 60 frames per second.

### 3.3 Automatic Learners

Machine Learning (ML) is a discipline of computer science and statistics that seeks to build systems capable of improving their performance based on historical data. Usually, a learning machine is designed to perform tasks such as regression or classification by finding the underlying relationship within the data. Our approach consists on using the built dataset to create automatic learners capable of correctly classify observed frames into drills. For this, we have decided to perform experiments using two traditional ML techniques, namely Support Vector Machines (SVM) and Neural Networks (NN) to finally choose the one with highest prediction accuracy.

The artificial neural networks are well stablished learning machines that have been succesfully used in a variety of applications where historical data is available. They are systems made of connected nodes capable of approximating nonlinear functions of their weighted inputs. Training algorithms are usually designed to find suitable values for the connection weights according to a desired criteria. We have chosen a feed-forward multilayer perceptron architecture for the NN with one hidden layer and sigmoid activation function to perform the task. Additionally, the training process is performed using the Levenberg-Marquardt method, an optimization algorithm that solves nonlinear least square problems to fit a parameterized function to measured data. Its main advantage is that it combines two minimization methods: the gradient descent method and Gauss-Newton method.

The SVMs have become of paramount importance in the statistical learning community for the last two decades as well as in a variety of application areas where automatic classification, ranking, regression and pattern recognition tasks are required. The underlying concept behind SVMs is to find the hyperplane that separates instances of two different classes with maximum margin, i.e., the largest distance between the hyperplane and the closest training data point. More formally, the SVM algorithm solves the following optimization problem:

$$\arg\min_{w,\zeta,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\zeta_i \tag{1}$$

subject to $y_i(wx_i - b) \geq 1 - \zeta_i$, where $\zeta_i > 0 \forall i = 1, \ldots n$ are slack variables that allow missclassification of data points for not linearly separable data, $w$ is a vector that defines the slope of the separating hyperplane and $C$ is a regularization parameter that defines a balance between large margin and the amount of missclassificated data.

Finally, the SVMs have been enhanced to perform nonlinear classification through the use of what is known as the kernel trick. The idea is to find a mapping between the original data space and a feature space, usually in a higher dimension with the idea that the hyperplane in the transformed space becomes a nonlinear surface in the original space. Such mapping is achieved by using kernel functions; a set of functions that define the new transformed space. Some common kernels used in general classification and regression problems include the polynomial kernel, the gaussian kernel and the hyperbolic tangent kernel. The gaussian kernel has the following relation:

$$k(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2) \text{ for } \gamma > 0 \tag{2}$$

An important issue that needs to be solved in the classification learning framework is that of choosing the correct parameters for the automatic learners. For the case of the NN, the number of neurons in the hidden layer needs to be decided. For the SVMs, the regularization parameter $C$ and kernel parameter $\gamma$ require a tuning process. This process is carried out by using $k$-fold cross validation, a statistical procedure used to evaluate the generalization capability of a statistical analysis in a predictive model. The entire dataset is partitioned into $k$ equal size subsamples and $k-1$ of those subsamples are used for training. The remaining subsample is used to test the classifier's generalization capability. This process is repeated $k$ times with each subsample used once as validation data and the idividual results are averaged. Cross validation avoids the possibility of overfitting the model which happens when the model complexity is too high compared to the complexity of the data, leading to poor generalization capabilities. The importance of the cross validation procedure is that the generalization error is calculated in data that have not been used during the training process.

## 4 Results

We begin by performing an unsupervised study of the features used to describe the game state as presented in Section 3.2 using the technique of Principal Component Analysis (PCA). In a PCA analysis the idea is to define a transformation from the original variables to a new set of variables with the constraint that they are orthogonal among them. We have applied the PCA algorithm on the entire dataset and found that 37 of the transformed variables are capable of explaining the 99.12% of the variance within the data. This means that there is redundancy between the original variables and that a smaller subset of new features (37 out of 60) is enough to explain the information in the data. Notice that this is to be expected since certain features were computed based on others.

We have performed the training of the NN and the SVM with the built dataset of the STOx's drills using the Neural Network Toolbox of MATLAB and LIBSVM [19] respectively. We have used 10-fold cross validation in order to assess the generalization capability of each classifier together with a grid search on their respective parameters. For the NN, the parameter to be tuned is the number of neurons in the hidden layer. In the case of the SVM, the grid search is on the $C$ and $\gamma$ parameters and included the following values: $C = [2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5}, 2^{7}, 2^{9}]$ and $\gamma = [2^{-7}, 2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5}]$.

The training and testing procedures are performed for every combination of parameters for each classifier and the results with highest generalization accuracy are chosen in Table 1. On one hand, the training accuracy corresponds to the proportion of data points correctly classified on the data used to perform the training process. For the NN this value is related to the nonlinear function found after the Levenberg-Marquardt algorithm is executed and the quality of the optimal solution. In the case of the SVMs, this value is related to the balance between the parameters $C$ and $\gamma$. On the other hand, the generalization accuracy is the proportion of data points correctly classified when using the test dataset (i.e., data points that were not used during the training phase). High values of the generalization accuracy ensure that there is no overfit on the constructed model.

**Table 1.** Results of the training and testing processes for the SVM and NN classifiers.

| | **SVM** | **NN** |
|---|---|---|
| **Algorithm Parameters** | $C = 512$<br>$\gamma = 0.0078$<br>nSV = 19637 | nNeurons = 20<br>MSE = 0.2048 |
| **Training Accuracy** | 96.69% | 82.44% |
| **Generalization Accuracy** | 91.45% | 81.96% |

It is noteworthy that the SVM attains a higher generalization accuracy by achieving 91.45% compared to that of the NN which achieves 81.96%. However, these results show that both classifiers are capable of correctly predict unseen frames.

We have also calculated the prediction accuracy of each classifier per class. In the Table 2 we can see that the frames of certain drills are easier to predict than others. For instance, it seems clear that the drill **FA** is significantly easier than all other drills. This makes sense since the configuration of the players is unique among the drills. There is no other drill in which all players move close to the opponent goal. On the contrary, drills **POA**, **PPA** and **POD** are more similar in terms of the robot's position at the beginning of the drill. In all of them, the initial position of the two attackers is very similar as well as the position of the defender that moves towards the opponent zone. This is reflected in their lower prediction accuracy for both classifiers.

**Table 2.** Prediction Accuracy of each classifier per class.

| **Drill** | **SVM (%)** | **NN (%)** |
|---|---|---|
| **DK** | 94.75 | 76.81 |
| **POA** | 89.17 | 78.03 |
| **PPA** | 88.35 | 82.99 |
| **POD** | 86.01 | 73.60 |
| **PMD** | 93.79 | 82.78 |
| **FA** | 98.2 | 95.81 |

Finally, we have calculated the evolution of missclassified frames in one example of each drill by computing the accumulated number of errors per frame during the episodes in order to analyze the prediction pattern of each drill. These results are shown in Fig. 4.

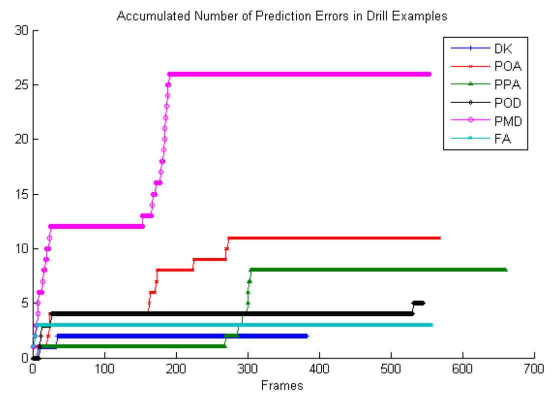

**Fig. 3.** Accumulated number of errors per frame in one example of each drill.

It is noteworthy that most classification errors occur mostly at the initial frames and no more after certain point of the episodes. It seems that there is a point where the drill becomes completely clear to the classifier which makes sense since all drills have similar frames at the beggining and become different after certain time. This information is highly valuable since it may be involved in the creation of dynamic defense strategies and it confirms that there is enough time for the team to defend against the executed drill.

A video showing episodes of each drill and the prediction performed by the SVM can be found in [20].

## 5 Conclusions and Future Work

We have demonstrated experimentally that it is possible to predict the play that certain team will perform in a SSL game of RoboCup by using historical data. We have done this by using machine learning techniques and modeling the task as a multi-class classification problem of learning the frames that are part of the drills. We have achieved high generalization accuracy with two learning algorithms namely support vector machines and neural networks using a thoroughly tuned process.

These results were achieved by using a dataset build from the strategies of the STOx's team in the Small Size League of RoboCup in 2013. However, we claim that the approach can be seamlessly extended to learn the drills of all teams that have participated in any RoboCup tournament for which data logs are available. Certain challenges still remain, such as managing the imbalance in the number of data points per class due to the fact that some drills may be more common than others.

At this point, the framework requires human intervention in order to know the number of drills of each team and label the data according to the logs. As future work, we plan to design a learning strategy capable of automatically obtained such information in order to assess the work. Also, the next step of this work consists on defining strategies to defend against the predicted drills and implement them during games in order to evaluate the impact of this approach on real games. Finally, a process to evaluate the features used to predict the drills is also of interest.

## Acknowledgements

This work has been funded by "Octava Convocatoria Interna de Proyectos de Investigación FODEIN 2014 #048" at Universidad Santo Tomás Colombia, entitled "Desarrollo de un entorno para la validación de técnicas de robótica cooperativa".

## References

1. Salustowicz, R. P., Wiering, M. A., Schmidhuber, J.: Learning Team Strategies: Soccer Case Studies. Machine Learning 263–282 (1998).
2. Stone, P.: Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer. MIT Press, United States, (2000).
3. Bianchi, R. A., Ribeiro, C. H., Costa, A. H.: Heuristic Selection of Actions in Multiagent Reinforcement Learning. In: Proceedings of the 20th international joint conference on Artificial intelligence, pp. 690–696. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA (2007).
4. Stone, P., Sutton, R.: Keepaway Soccer: A Machine Learning Testbed. In: Ansgar, B., Adam, J., Itsuki, N., Yasutake, T. (eds.) RoboCup 2005: Robot Soccer World Cup IX. LNCS, vol. 4020, pp. 93–105. Springer, Heidelber (2006).
5. Duan, Y., Liu, Q., Xu, X.: Application of reinforcement learning in robot soccer. Engineering Applications of Artificial Intelligence, 936–950, (2007).
6. Alankar, K., Nebell, B,. Stantontl, C., Williams, M A.: Case Based Game Play in the RoboCup Four-Legged League. In: Polani, D., Browning, B., Bonarini, A., Yoshida, K. (eds.) RoboCup 2003: Robot Soccer World Cup VII. LNCS, vol. 3020, pp.739–747. Springer, Heidelberg (2004).
7. Marling, C., Tomko, M., Gillen, M., Alexander, D., Chelberg, D.: Case-based reasoning for planning and world modeling in the robocup small size league. In: IJCAI workshop on issues in designing physical agents for dynamic real-time environments, Acapulco (2003)
8. Wendler, J., Bach, J.: Recognizing and predicting agent behavior with case based reasoning. In: Daniel, P., Brett, B., Andrea, B., Kasuo, Y. (eds.) RoboCup 2003.LNCS, vol. 3020, pp. 729–738.Springer, Heidelberg (2004)

9. Ros, R., Lopez De Mntaras, R. L., Arcos, J. L., Veloso, M.:Team playing behavior in robot soccer: A case-based reasoning approach. In: Rosina, O.W., Michael, M.R. (eds) Case-Based Reasoning Research and Development. LNCS, vol 4626,pp. 46–60. Springer, Heidelberg (2007).
10. Steffens, T.: Adapting similarity-measures to agent types in opponent modelling. In: Workshop on Modeling Other Agents from Observations at AAMA, pp. 125–128. New York (2004)
11. Ahmadi, M., Lamjiri, A. K., Nevisi, M.M., Habibi, J., Badie, K.:Using two-layered case-based reasoning for prediction in soccer coach. In: International Conference of Machine Learning, Models, Technologies and Applications, pp. 181–185.CSREA Press, Las Vegas (2003)
12. Lattner, A.D., Miene, A., Visser, U., Herzog, O.:Sequential pattern mining for situation and behavior prediction in simulated robotic soccer. In:Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds) RoboCup 2005: Robot Soccer World Cup IX. LNCS, vol4020,pp. 118–129.Springer, Heidelberg (2006)
13. Riley, P., Veloso, M.: On Behavior Classification in Adversarial Environments. In: Parker, L.E., Bekey, G., Barhen, J. (eds) Distributed Autonomous Robotic Systems, pp. 371–380.Springer, Tokyo (2000)
14. Visser, U., Drcker, C., Hbner, S., Schmidt, E., Weland, H.:Sequential pattern mining for situation and behavior prediction in simulated robotic soccer. In: Stone, P., Balch, T., Kraetzschmar, G. (eds) RoboCup 2000: Robot Soccer World Cup IV. LNCS, vol2019,pp. 391–396. Springer, Heidelberg (2006)
15. Faria, B.M., Reis, L.P., Lau, N., Castillo, G.: Machine Learning algorithms applied to the classification of robotic soccer formations and opponent teams. In: 2010 IEEE Conference on Cybernetics and Intelligent Systems , IEEE Press, pp. 344–349. Singapore (2010)
16. Trevizan, F., Veloso, M.: Learning Opponent's Strategies In the RoboCup Small Size League. In: Proceedings of the AAMAS 2010 Workshop on Agents in Real-time and Dynamic Environments, pp. 45–52. Toronto (2010)
17. Konur, S., Ferrein, A., Lakemeyer, G.: Learning Decision Trees for Action Selection in Soccer Agents. In: Proceedings of the ECAI-04 Workshop on Agents in Dynamic and Real-time Environments, IOS Press. Valencia (2004)
18. Ledezma, A., Aler, R., Sanchis, A., Borrajo, D.: Predicting Opponent Actions by Observation. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004: Robot Soccer World Cup VIII. LCNS, vol 3276, pp. 286–296, (Heidelberg), (2005).
19. Chang, C-C., Lin, C-J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology. 1–27 (2011). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
20. STOx's team webpage. `http://www.stoxs.org/index.php/en/projects/robocup-ssl-2`