2D BEHAVIOR BASED SOCCER TEAM - A NEURAL NETWORK APPROACH

Felipe Lira Santana Silva*, João Paulo de Almeida Barbosa†, Guilherme Sousa Bastos*

* *Av. BPS, 1303, Bairro Pinheirinho*
Universidade Federal de Itajubá
Itajubá - MG - Brasil

† *Praça Marechal Eduardo Gomes, 50 - Vila das Acacias*
Instituto Tecnológico de Aeronáutica
São José dos Campos - SP - Brasil

Emails: felipe.lira.ss@gmail.com, barbosajoaopa@gmail.com, sousa@unifei.edu.br

Abstract— RoboCup is an internatinal robotics competition created to stimulate artificial intelligence research all over the world. In a RoboCup's 2D Soccer Simulation League, a server simulates a soccer game with 11 players and 1 coach for each side. This server saves after each game two log files, that can be analyzed with Data Mining techniques to extract information about the game. The software developed in this work can read and analyze more than 30 log files in less than 2 hours, so it is possible to change the defensive behavior as soon as the game ends. A feed forward artificial neural network is then trained to recognize opponents' shoot and pass probabilities, and its output is used as sensors to activate specific defensive behaviors to suppress opponents' scoring ability. Practical tests showed a 17% decrease in opponents' scoring when using the new Expertinos team.

Keywords— Artificial Neural Network, Simulation 2D, RoboCup.

1 Introduction

RoboCup is an international robotics competition created as an attempt to stimulate robotics and artificial intelligence (AI) research. The RoboCup Soccer part of the competition has several categories: Small and Medium League, Simulation League, Humanoid League, and others. All these leagues are focused in a soccer match as research topic and test environment. RoboCup's challenging and ultimate goal is to create a totally autonomous soccer team composed only by robot soccer players, and capable of winning a soccer game against the current FIFA World Cup winner until 2050 (*RoboCup Objective*, n.d.).

RoboCup's Simulation League has two categories: 2D and 3D Soccer Simulation. While the 3D category is basically a simulated Humanoid League with virtual NAO robots as players, the 2D soccer simulation is very similar to a real soccer game. RoboCup provides a server called rcssserver, which simulates the soccer field, game modes, and physical models for the players and the ball. It also simulates a vision and body sensor model, stamina management, players' actions, coach decisions, and the referee model. Also, after each game, the server saves two log files that can be used to replay the match later.

These log files can be read and converted to xml, txt and even flash files. Therefore, anyone can have access to detailed data from each one of the 6000 cycles of a regular game. Some works (Karimi and Ahmazadeh, 2014; Oliveira et al., 2009; Davi et al., 2010) have successfully implemented Data Mining techniques to explore patterns of different opponents and to improve the offensive strategy of 2D simulated soccer teams.

The complexity of programming individual agents to win a soccer match has lead researches in fields such as AI algorithms, Multi-agent systems, Behavior Based Robots and Data Mining techniques. Also, some teams have released their base code allowing other researchers to focus only on the decision-making algorithm. The most known base teams are the UvA Trilearn, agent2D, and HELIOS.

This work presents a data mining program using a feed forward artificial neural network (ANN) to improve the defensive algorithm of the Expertinos team from Universidade Federal de Itajubá, Brazil. The main objective is to be able to analyze the log files from regular games during the competition, and to use the collected information to improve the defensive behavior of each player.

The article is organized as follows: In section 2 it is shown some data mining related work that have already improved a 2D soccer team. In section 3 it is described the background of ANN used in this work. In section 4 it is discussed the problem definition. In section 5 the methodology behind the program and the ANN modeling are explained. In section 6 are presented the results. Finally, in section 7 this work is concluded.

2 Related work

RoboCup server not only simulates a 2D soccer match, but it also saves all the information of each game in two log files called rcg and rcl files. Therefore, a data mining approach can extract from these files important information about the teams. Even though it is old data, it can show patterns of teams' offensive and defensive strategies. It also can be used to train the AI algorithm

since it contains the players' states for each cycle of the game. As a result, several researches have done data mining on RoboCup log files.

This work was based on Karimi 2014 (Karimi and Ahmazadeh, 2014) article. Karimi analyzed 43 different log files to predict the players' own and opponent behavior using a C4.5 algorithm to classify the data. With the resulted decision tree generated by the classification algorithm she improved the Agent2d-3.2.2 base team's goal score in 251.39% with 64.13% confidence.

The data mining solution was also successfully implemented in Oliveira 2009 (Oliveira et al., 2009). Oliveira used a multilayer perceptron network to analyze the data and predict the best conditions to score a goal. He successfully increased goals scored by 77%.

Other important work was done in the paper "Game Theory-based Data Mining Technique for Strategy Making of a Soccer Simulation Coach Agent" (Fard et al., 2007). While most of researches have focused on the players, this paper analyzed the data to improve coach's performance using Data Mining theory, Deterministic Finite State Automata (DFA), and Game theory. The idea is to use the privileged coach agent to construct a knowledge-base of the game and opponent's strategy, and share it among the other players.

All these works use the agent2D base code, which is divided in different behaviors, so all the teams use by default Behavior Based Robotics theory. Behavior Based Robots basically use sensors to activate specific and pre-programmed behaviors to react to the environment. This theory is implemented in multi-robot soccer teams (Werger, 1999), unmanned ground vehicles formation (Balch and Arkin, 1998), and many other applications. Behavior based theory will not be explained in this work because the focus is the ANN, and information about it can be found in the cited papers.

3 Technical background

An Artificial Neural Network (ANN) is a pattern recognition mathematical tool that works by mimicking the human brain structure. Basically it is composed by a mathematical model of several brain's neurons linked to each other by weights that can be modified to achieve a desirable output value.

It has been chosen as main learning tool for this work because of its great generalization power and its low computational cost to operate once it has been trained.

In this section it is explained the basic architecture of a feed forward ANN, and the backpropagation algorithm used to train it.

3.1 Training

The ANN training with the backpropagation algorithm is done by feeding the ANN with an input data to find its outcome, and then backpropagate the error through the network to adjust its weights and biases values. Therefore, let I be the input vector, WI the first weight matrix, H the hidden neurons' values vector, WO the output weight matrix, BI the biases vector for the hidden layer's neurons, BO the output neuron bias, and Y the network output, the training algorithm is (Nielsen, 2015):

1. Input the training data
2. For each training example in the training set:
   (a) Feed Forward:
   $$y = \sigma\left(WO \cdot \sigma\left((WI \cdot I) + BI\right) + BO\right) \quad (1)$$
   (b) Output Error: Usually a quadratic cost function is used as shown in equation (2).
   $$C = \frac{1}{2}\sum_j (y_j - output_j)^2 \quad (2)$$
   Where y_j is the training output from the training set. Then, for the output neuron the error δ^l is calculated by equation (3), and for the hidden layer it is calculated by equation (4)
   $$\delta_1^2 = \bigtriangledown C \cdot \sigma'(output) \quad (3)$$
   (c) Backpropagate the error.
   $$\delta^1 = \left((WO)^T \cdot \delta_1^2\right) \cdot \sigma'(\delta_1^2) \quad (4)$$
3. The weights and biases are updated with equations (5), (6), (7), and (8).
   $$WI = WI - \eta(I \cdot \delta^1) \quad (5)$$
   $$WO = WO - \eta(H \cdot \delta^2) \quad (6)$$
   $$BI = BI - \eta(\delta^1) \quad (7)$$
   $$BO = BO - \eta(\delta^2) \quad (8)$$

Where η is the learning rate.

4 Problem definition

During a soccer match in the RoboCup server, the players have to make fast decisions all the time based on limited sensor data and the condition of teammates and opponents. Different actions and behaviors depend on whether the agent has the possession of the ball or not, its position on the field, and the strategy to perform cooperative actions. For this reason, many teams uses base codes that already have some intelligence implemented, so they can focus the work on specific improvements.

The HELIOS base team used in this work has already a good decision making algorithm for cooperative actions, specially when attacking. Its code leaves little room for deep modifications without broking its *chain action* and ending up with worse performance. Moreover, there are already many works on offensive strategy such as (Davi et al., 2010; Oliveira et al., 2009; Karimi and Ahmazadeh, 2014).

In this work, there are two major problems to improve the defensive strategy of the base team.

- The development of a program capable of analyzing log files and training the ANN in a fair amount of time.
- To implement the changes in the defensive behavior of HELIOS base code, which had to be separated in smaller behaviors to have a better relationship with the ANN outputs. In this part it is important to emphasize that many changes in the defensive behavior can have a negative impact on offensive moves. When the agent runs with the ball or make a dribble, it loses the ball possession during a short period of time, this activate the defensive behavior, where actions such as *intercept* or *move* are used.

The final program developed in this work can be used in future competitions as a built-in behavior to improve the teams' performance in real time during the competitions by recognizing not only the opponent agent behavior, but also weakness, strategies, and trained moves of both teams.

5 System design

This work is separated in four phases: Log conversion and pre-processing, data mining of the resulted files to build a players' state database that the ANN can work with, the neural network training, and finally the code implementation on the base team.

5.1 Log files conversion

The rcg files saved by RoboCup server were converted using a modified version of *rcg2txt* program inside the package *Librcsc 4.0.0* necessary to run the HELIOS base team. The original program output uses too much unnecessary space and it is hard to analyze because its information identifier is inside brackets. This program is installed in Ubuntu automatically when installing the package and compiling the team. The txt file generated has the following structure:

- First line is the source file name.
- After it, a state line shows: The current cycle, game mode, left score, and right score for each cycle.
- Then the ball line contains: Current cycle, ball's x and y position, ball's x and y velocities.
- Lastly, the next 22 lines show all the players information: Side, number, current cycle, position and velocities, body and head angles, stamina and the last action.

5.2 Data Mining

In order to analyze the data, three programs were developed: `gameDB` reads the txt files and creates a data base, `playerState` determines all states of each player the way it is received from the RoboCup server and with the base team's world model, and `logAnalyzer` is the function that recognizes dribble, pass and shoot actions taken during the game and if they were successful or not.

The function `logAnalyzer` classify the actions using the information provided by the created database, and the decision tree shown in Figure 2. When an opponent gets the ball, all actions are considered to be unsuccessful by the algorithm.

A graphical interface was developed to play the old games. Moreover, using the files generated by `logAnalyzer` as an input to `LogPlayer`, it is possible to analyze if the algorithm is correctly classifying the actions as shown in Figure 1.

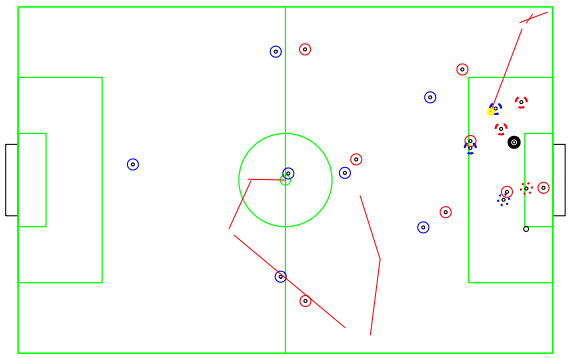

Figure 1: *Log Player*

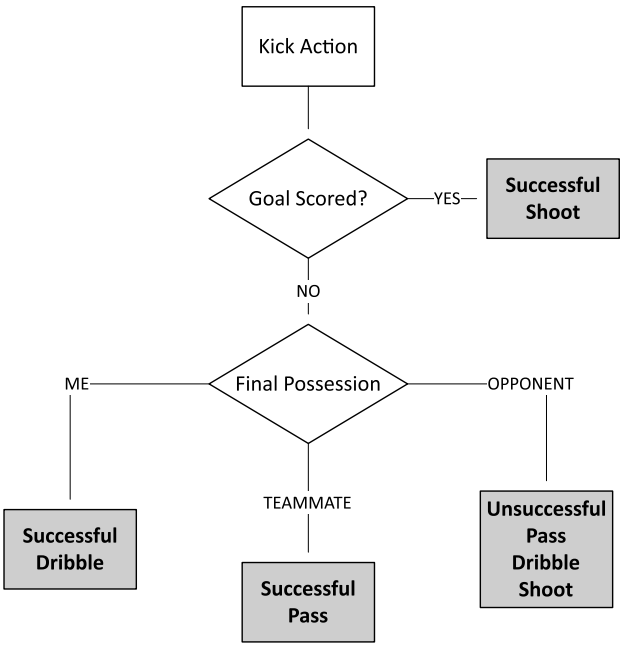

Figure 2: *Action Classification Tree*

The red lines represent successful passes, the yellow circle is a successful dribble and when a player scores a goal, a green line indicates the player's angle and position. When a successful action is recognized, the program stores a true value for the cycle in which the action has started. This true value indicates that in the specified state a successful action was taken. This value does not represent action's likelihood to happen, it is just a symbolical number for training purposes.

After all the data mining process, the information used as state is a normalized vector containing the following values in this specific order: self x position, self y position, self x velocity, self y velocity, self body angle, self head angle, self stamina, nearest teammate x and y position, nearest teammate to ball x and y position, nearest opponent x and y position, nearest opponent to ball x and y position, ball's x and y position, ball's x and y velocity, and current cycle.

5.3 Artificial Neural Network design and training

As an initial phase, all the necessary program to create and to train the ANN was developed and a simple problem was used to evaluate the network's capabilities.

This simple problem is a pass situation shown in Figure 3, with a central player with the ball who need to make a pass. There is a teammate (player in blue) and an opponent (player in orange) placed in the field. In order to make a successful pass, the central player has to observe both positions to decide if it is better to keep the ball, or to make a pass.

An ANN was created to identify when a pass would be successful or not, with a program generating random positions for both players and analyzing when the opponent was able to intercept the ball. The ANN created had four inputs with the coordinates x and y of each player in relation to the central player, a hidden layer with 10 neurons and an output layer classifying a successful pass as 1, and failure as 0. It was trained with 100 random cases and tested with another 100 different cases and had a 93% accuracy rate.

This simple problem showed that a simple ANN can classify actions based on other players states.

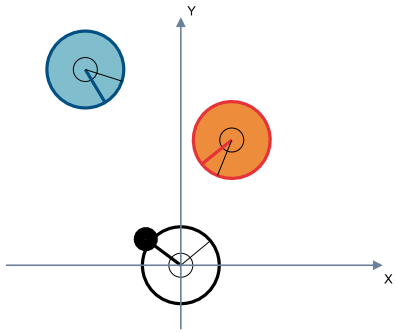

Figure 3: *Simple Problem to test the ANN*

Later, two neural networks were created inside the team's defensive behavior, one analyzes the pass likelihood of opponents, and the second recognizes when an opponent is likely to kick the ball towards the goal in the defensive field.

The input data is: self x and y position, nearest opponent to agent x and y position, and nearest opponent to the ball x and y position. Figure 4 shows the ANN structure used in this work.

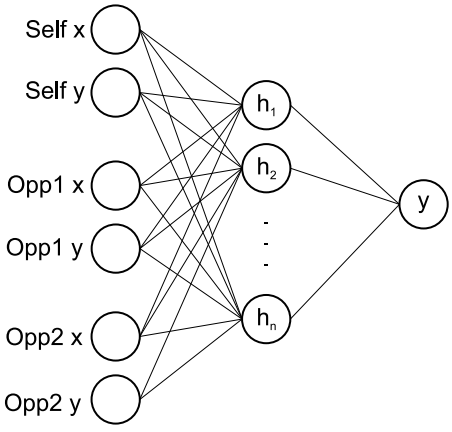

Figure 4: *Final ANN*

Each training set has 74626 examples extracted from 30 games played during the XIV Latin American Robotics Competition (LARC 2015). Only the states of players with ball possession were taken in order to narrow down the

training set. Otherwise, the training set would have more than 3 million states to analyze. The training set has 4386 successful pass states, and 118 successfull shoot states. The training algorithm used is the backpropagation algorithm.

Several tests were performed then to determine the size of each ANN. Each one was trained with 100 epochs and with multiple of 6 hidden neurons. The results are shown in Tables 1 and 2.

Table 1: *Pass test results for different numbers of hidden neurons*

Number of Hidden Neurons	True positives
6	592
12	692
24	832
48	1041
96	1071

Table 2: *Shoot test results for different numbers of hidden neurons*

Number of Hidden Neurons	True positives
6	56
12	86
24	78

In Table 1 the results do not stop to improve, but with 96 neurons only a 2,8% improvement was achieved. As a result, the number of hidden neurons chosen for the pass' ANN is 48.

In contrast, the results for the shoot ANN actually got worse with more neurons, so the number of hidden neurons was established in 12.

These numbers may not be the optimal value to solve the problems, but they are acceptable for this work and the limited time available for each phase.

5.4 Code implementation

Inside the HELIOS base code there is a class called Bhv_BasicMove, this class is called whenever the player has no ball possession, and the defensive algorithm is implemented inside this class.

The ANN structure was built inside the class. Its output is calculated every different cycle. In order to not interfere in the offensive moves, the decision making based on the network outcomes only have results in the defensive field. Each ANN was used as a sensor to activate the intercept defensive behavior and a marking behavior, both already implemented in the base team.

During the games it was possible to notice a dangerous region right in front of the penalty area, so this region received an extra weight when deciding about whether to intercept or not the ball. Before this modification, the defenders would only intercept when probability of recovering the ball was greater than 80%. The behaviors used were: `Body_Intercept`, `Body_GoToPoint`.

6 Results

With the new Expertinos team (Expertinos2015) and the old one (Expertinos), 90 matches were played against the binary of ITANdroids from RoboCup 2015. The results of these games are shown in Tables 3 and 4.

Table 3: *Expertinos vs ITAndroids*

	Expertinos	ITAndroids
Goals scored	98	209
Average of goals	1.077	2.297
Wins	14	63

Table 4: *Expertinos2015 vs ITAndroids*

	Expertinos2015	ITAndroids
Goals scored	86	173
Average of goals	0.945	1.901
Wins	16	54

The new team decreased the number of goals taken by 36 goals, or 17.22%. The number of goals scored by Expertinos2015 also decreased due to the fact that modifications in the defensive behavior interfere in offensive moves.

Expertinos2015 team have also participated in LARC 2015 where it got the fourth place. During the first day of games, Expertinos2015 had little offensive modifications. The neural network defensive algorithm was implemented in the second day, and one good result was achieved playing against ITAndroids as shown in Table 5, in the first day Expertinos2015 lost the game with a score of 2 goals against 5 of ITAndroids, and during the second day in a second game against ITA the team with improved defensive strategy tied the match.

Table 5: *Expertinos versus ITAndroids*

Team	Scored	Against
Itandroids	5	2
Expertinos	2	5
Itandroids	1	1
Expertinos2015	1	1

Analyzing Table 4 it is possible to see that Expertinos2015 lowered the opponent goals by approximately 17% comparing to its base team. On the other hand, the offensive part of the team had

no improvements. This is exactly the expected behavior of this team since there was no modification in its offensive strategy.

7 Conclusion

In this work a data mining approach with a feed forward neural network solution was used to improve the defensive strategy of Expertinos' RoboCup 2D simulated soccer team. The main goal was to develop an automated program that could read rcg files converted to txt, extract player's states and actions information, and train two neural networks to recognize opponent's successful shoot and pass probabilities. The outputs from the neural network activated different defensive behaviors in order to anticipate dangerous movements on the defensive field. The team was tested in XIV LARC 2015 competition extracting the training data from the log files of the competition games improving group phase results. When the software was used to play against ITAndroids, a decrease of 17% in opponent's scoring was observed comparing to the old Expertinos team, with no relevant improvement in offensive behavior.

8 Future work

One of the great problems in this work was the code implementation in HELIOS base team. This team has already a heavy algorithm for the soccer decision making problem, so all the changes have to be well studied and tested to certify a improvement has been made. Therefore, during the LARC competition the idea of developing a Brazilian base team came up as a future work in this field.

Also, the neural net presented here is at an early development stage, the goal for this net is to be able to recognize the opponent's strategy in real time, changing the players' behavior several times during the same match.

9 Acknowledgments

The author acknowledges UNIFEI (Universidade Federal de Itajubá) for all the knowledge and support provided during the Control and Automation undergraduate program, and the advisor Guilherme Sousa Bastos.

References

Balch, T. and Arkin, R. C. (1998). Behavior-based formation control for multirobot teams, *IEEE transactions on robotics and automation* **14**(6): 926–939.

Davi, C. d. L., Adeodato, P. J. and Gonçalves, P. M. (2010). Improving reinforcement learning algorithms by the use of data mining techniques for feature and action selection, *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, IEEE, pp. 1863–1870.

Fard, A. M., Salmani, V., Naghibzadeh, M., Nejad, S. K. and Ahmadi, H. (2007). Game theory-based data mining technique for strategy making of a soccer simulation coach agent., *ISTA*, Vol. 2007, Citeseer, pp. 54–65.

Karimi, M. and Ahmazadeh, M. (2014). Mining robocup log files to predict own and opponent action, *International Journal of Advanced Research in Computer Science* **5**(6).

Nielsen, M. A. (2015). Neural networks and deep learning, `http://neuralnetworksanddeeplearning.com/index.html`. Free online book.

Oliveira, R., Adeodato, P., Carvalho, A., Viegas, I., Diego, C. and Ing-Ren, T. (2009). A data mining approach to solve the goal scoring problem, pp. 2347–2352.

RoboCup Objective (n.d.). `http://www.robocup.org/about-robocup/`. Accessed: 2015-11-07.

Werger, B. B. (1999). Cooperation without deliberation: A minimal behavior-based approach to multi-robot teams, *Artificial Intelligence* **110**(2): 293–320.