Estimation of Players' Actions in Soccer Matches Based On Deep Autoencoder

Jorge L. Copete, Junichi Suzuki, Quan Wei, Ryo Iwaki, Nobutsuna Endo, Hiroki Mori, Yukie Nagai, Minoru Asada
Department of Adaptive Machine Systems, Graduate School of Engineering,
Osaka University, Osaka, Japan
robocup-spl@ams.eng.osaka-u.ac.jp

Abstract

The ability to estimate future events is essential for biological systems to adapt themselves to uncertain situations efficiently. Similarly, the abilities to estimate tactical movements of opponents and to perform in an unpredictable manner against opponents in robot soccer competitions are required in the best interest of the robot players. However, there is still a need of approaches that estimate robot players' future actions while dealing with complex dynamic conditions when their behaviors have been altered because of mutual interactions. In this study, we propose a method based on a deep autoencoder to estimate actions of soccer players in an environment with multiple agents and dynamical conditions. We carried out experiments employing data from the RoboCup 2D Soccer Simulation League and showed the validity of the proposed method. Our results suggest that computational models using deep architectures may be key to developing new skills for robots using low-level representations.

1 Introduction

In human soccer the ability to understand the tactics of opposing teams is crucial to modifying a team's own actions, thus achieving better performance. In real soccer, for example, while a player who has an offensive role might try to move forward and kick the ball directly into the goal, a midfielder might try to pass the ball to an offensive player who is near the opponent's goal. Human players deal with these situations by estimating the future action of each player on the opposing team. In the context of RoboCup soccer competitions, therefore, learning to estimate the future actions of other players is also required in the best interest of the robot players. One of the main challenges of estimating the actions of robot soccer teams is that the actions of the players during a match depend on several complex factors, including both teams' strategies and the skills and role of each player on both teams. Therefore, in order to estimate the behavioral patterns of multiple agents in a dynamic environment, we need to employ a learning process that considers all of these factors simultaneously.

Among previous works, Visser et al. [1] proposed classifying the behaviors of the opposing team by inferring the behaviors that players can perform in advance. Bowling et al. [2] proposed adapting to the opponent by observing our own team 's effectiveness rather than observing the opponent 's behavior. For that purpose, they used multiple team plans that are appropriate for different opponents and situations, and rewarded the plans during the match based on the actual results. Recently Trevizan and Veloso [3] proposed classifying opponents' strategies by studying the similarity of the strategies of two teams by employing feature vectors of the distances between players and the ball. Yasui et al. [4] proposed a dissimilarity function that shows the difference between opponents ' deployments at two different times, and extends it to the difference between those of two different time intervals. However, these initial approaches, which classified players ' behaviors by finding a strategy from a set of known strategies or learned a strategy assuming that a team's own behaviors are independent of the opposing teams' behaviors, can be regarded as top-down oriented and thus are not sufficiently robust to deal with complex dynamic conditions when the players' behaviors have been altered because of their mutual interactions. Therefore, there is still a need of approaches employing bottom-up procedures to provide low-level behavioral features (e.g., anticipation of players' motions) that are needed at higher-level representations to model teams' strategies.

In this study, we proposed a method to estimate players' actions that does not rely on a priori knowledge of teams' actions or strategies. Specifically, we proposed a method to estimate the trajectory of players during ball motion and the direction of the ball after being kicked. We used dynamical prediction (that is, a mathematical model describing a physical system that changed over time) to estimate the future position of both players and ball. This estimation is desired because both the players and the ball move; they are dynamic. In our model, we employed deep neural networks to represent the complex dynamics of temporal sequences of player positions and a one-hidden-layer neural network to estimate the orientation of the ball movement. We applied our model to data from the RoboCup 2D Soccer Simulation League to assess our prediction method. The rest of the paper is organized as follows: In section 2, we explain our proposed method. The experimental settings for the RoboCup 2D Soccer simulator are explained in section 3. Section 4 presents the results of our experiments. In section 5, our conclusions are given and future research possibilities are discussed.

2 Dynamical estimation of players' actions

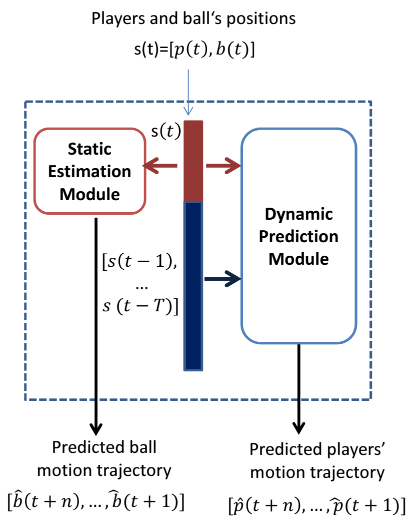
In soccer competitions, not only are teams without strategies weak opponents to play against, but also teams using strategies based on repetitive actions are predictable and thus possible to defeat. During soccer matches players perform actions such as running in certain trajectories, passing the ball, keeping the ball as far away as possible from the opponents, and kicking the ball to the opposing goal. Additionally, if the opposing team is in possession of the ball, players are expected to attempt to regain control of the match. Therefore, the ability to estimate the trajectory of other players and the ball is critical for soccer players. In this investigation, we considered two basic actions of soccer players —moving in the field and kicking the ball —and proposed a method to estimate the trajectory of players during ball motion and the direction of the ball after being kicked.To do this, we used the following two assumptions:

- The trajectory of players can be anticipated by learning to estimate future states based on the past states
- The trajectory of a kicked ball can be predicted by learning to estimate future states from the current state

Figure 1 shows a schematic diagram of the method that we proposed. We used a deep autoencoder as a temporal sequence predictor to account for the first assumption, and a feed-forward neural network for the second one. Players are not initially assigned any particular role or strategy; instead their action patterns are learned from raw data consisting of players' positions and ball position. In our implementation, the direction of players' motions and ball's motions are calculated from the estimated positions. We used motion direction but not position because the neural networks were expected to learn to generalize motion patterns but not to calculate exact positions.

2.1 Estimation of actions under dynamic conditions based on deep neural networks

In order to account for the dynamic action of players moving in the field, we proposed using deep neural networks (DNNs) which have the ability to represent complex functions. DNNs have been shown to outperform state-of-the-art machine learning algorithms in many applications [5], [6]. In particular, we referred to a previous work by Noda et al. [7], who reported the advantage of employing autoencoders based on deep neural networks for predicting temporal data sequences. In this approach, two deep neural networks are stacked in a mirrored structure, and the output layer of the neural network is connected in a closed-loop manner to the input layer to generate the prediction of future steps, as shown in Figure 2. The input-output mappings of the network are defined as follows:



Players and ball's positions
s(t)=[p(t), b(t)]
s(t)
Static Estimation Module
Dynamic Prediction Module
[s(t − 1), ... s (t − T)]
Predicted ball motion trajectory
[b̂(t + n), ..., b̂(t + 1)]
Predicted players' motion trajectory
[p̂(t + n), ..., p̂(t + 1)]

Figure 1: Estimation of static and dynamical actions: the last T steps for training a deep neural network, and the last step for training a feed-forward neural network.

$$\hat{u}_t = f(r_t), \quad (1)$$

$$\hat{r}_t = f^{-1}(u_t), \quad (2)$$

where $\mathbf{r}(t)$, $\mathbf{u}(t)$, and $\hat{\mathbf{r}}(t)$ are the vectors representing the input data, the corresponding vector feature, and the reconstructed data, respectively. $\mathbf{f}(.)$ represents the transformation mapping from the input layer to the central hidden layer of the network and $\mathbf{f}^{-1}(.)$ represents the transformation mapping from the central hidden layer to the output layer.

When employing deep autoencoders to make predictions, the input data is fed into the neural network as a contiguous segment of T steps. The inputs corresponding to Tin steps ($Tin < T$) are filled with previous input data, and the rest of the inputs corresponding to $T - Tin$ steps are filled with the outputs from the closed-loop data, as shown in Figure 3. In this study, the input data segment contains the positions of the players and the ball for T steps in Cartesian coordinates. Hence, for an input size of T where T_{in} corresponds to the size of previous input data, the input to the network at time t is defined as follows:

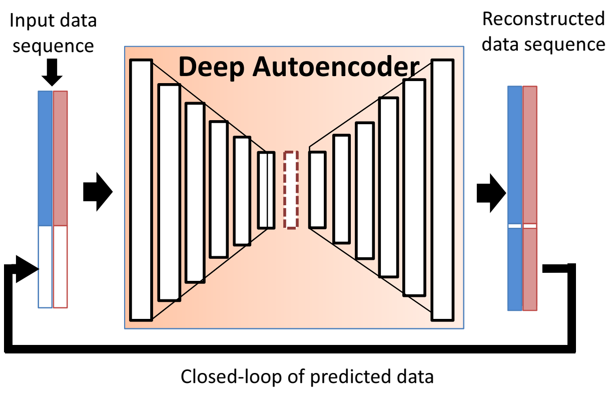


Figure 2: Deep autoencoder for temporal sequence prediction.

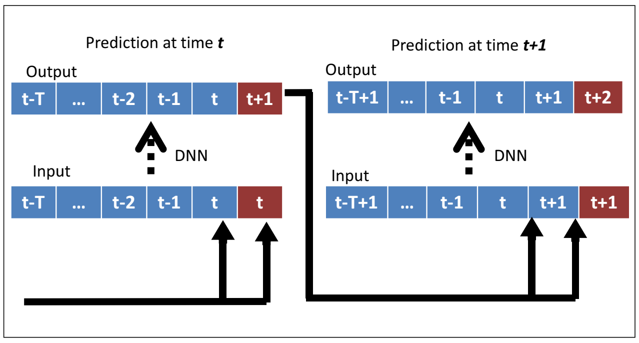


Figure 3: Scheme for recursive input-output loop.

$$s(t) = (b_{t_1}, \hat{p}_{t_1}, b_{t_1}, \hat{p}_{t_1}), \tag{3}$$

$$t - T_{in} + 1 < t_1 < t, \tag{4}$$

$$t + 1 < t_2 < t + (T - T_{in}), \tag{5}$$

where $\mathbf{b}(t)$ and $\mathbf{p}(t)$ are the vectors representing the ball's position and the players' positions, respectively.

2.2 Estimation of actions under static conditions based on a one-hidden-layer neural network

In this module the current state corresponds to the positions of the players when a player kicks the ball, and the future state is the position of the ball when it stops. We employ a one-hidden-layer neural network that is trained to estimate the direction of the ball, as shown in Figure 4. The inputs of the network are the position of the ball and the players in Cartesian coordinates at time t, and the output is the estimated position of the ball w steps ahead:

$$i_t = f(p_t, b_t), \tag{6}$$

$$\hat{b}_{t+1} = f(i_t), \tag{7}$$

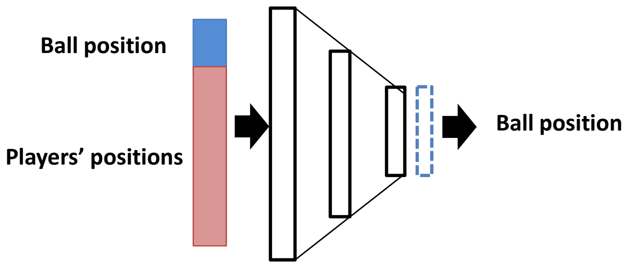


Figure 4: One-hidden-layer neural network for ball direction prediction.

where $\mathbf{b}(t)$, $\mathbf{p}(t)$, and $\mathbf{i}(t)$ are the vectors representing the ball's position, the players' positions, and the input data, respectively.

3 Experimental settings

For the static condition, we employed a one-hidden-layer neural network using as input data the position of the ball and the players at a frame before the player kicked the ball, and as output data the position of the ball once it stopped. The size of the input layer was 46 neurons, the hidden layer was 46 neurons, and the output layer was 2 neurons. For the dynamical condition, we employed a deep neural network with 8 hidden layers, composed of an encoder and a decoder whose structures are mirror images of each other. The input and output layers each contain 230 neurons. The number of neurons in the hidden encoder layers is 250, 150, 80, and 30. The decoder layers contain 30, 80, 150, and 250 neurons. The activation functions are linear functions for the hidden layers and logistic functions for the output layer. For this experiment, we implemented the proposed modules in the Python library, Theano, based on available implementations for DNNs [8].

The data we collected from the simulator of the RoboCup 2D Soccer Simulation League included the field position of 22 players (11 players for each team) and the position of the ball. We ran the software using the HELIObase and WEBase teams. The log data of the matches was recorded and processed to obtain the positions of the ball and the players in Cartesian coordinates (x, y) at each frame of the match. The length of each match was about 3000 frames. The position data (x, y) was normalized between 0 and 1 considering the original size of the field, which was 40 $\times$ 105 pixels. We recorded the log data of ten matches to be used as training data, and of one match for test data.

To train the deep neural network we used the entire raw data set, and for the one-hidden-layer neural network, we extracted the sets of frames of when players kicked the ball and of when the ball stopped moving. For testing, we extracted from the raw data those segments of the match in which the ball was detected as

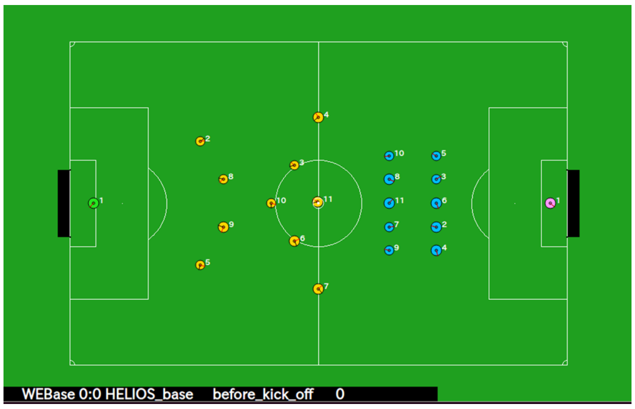


Figure 5: RoboCup 2D Soccer Simulation League.

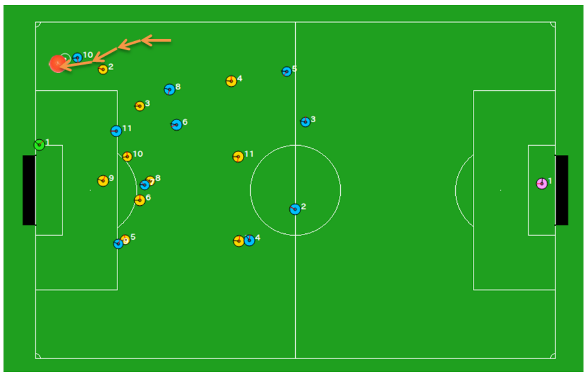


Figure 6: Example of ball motion that meets the requirements of a dynamic condition.

moving. An example of this is shown in Figure 6, where the motion of the ball is represented by arrows. That is, we focused our analysis on dynamical conditions, assuming that the motion of the ball caused players to move to other positions. For our purposes, the ball must have moved at an average speed of greater than 4 pixels per frame (field size is 40 × 105 pixels) to be considered in motion. To calculate the average speed and other results during training and testing, we used a time window of 5 frames, with the target frame included. The deep neural network was trained employing the stochastic gradient descent method, and the one-hidden-layer network was trained using the back-propagation method.

4 Results

4.1 Evaluations of estimation of ball direction

We conducted the first experiment to estimate ball direction using the extracted data corresponding to the positions of the ball and the players when a player was close to kicking the ball. The results of the experiment are shown in Figure 7 and Figure 8. Figure 7 shows examples of estimation of ball motion direction for several kicking players in different positions in the field. The read and blue arrows indicate the actual and the estimated directions of the ball, respectively. For analysis purposes, we decided that a estimation would be considered successful if the maximum difference between the estimated direction and the actual one was 15 degrees. Figure 8 shows the number of successful and failed estimations of ball direction for each player. For this experiment, the average successful estimation rate was 84.1%. The results demonstrate that our model was effective at estimating the future ball direction under several conditions. These results suggest that the kicking actions of players of both teams of the RoboCup 2D Soccer Simulation League are highly predictable and could be used by both teams to improve their ability to anticipate their opponents' shots.

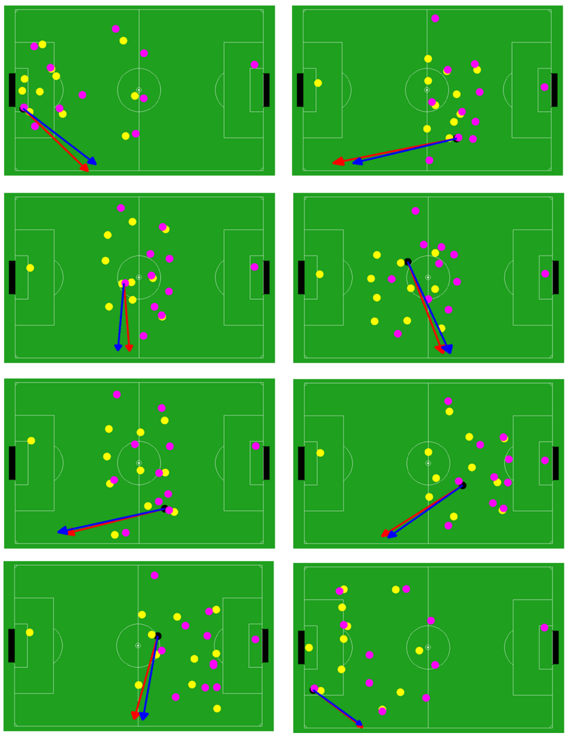


Figure 7: Examples of ball direction estimations for several players in different positions in the field. Red and blue arrows indicate the actual and the estimated directions of the ball, respectively.

4.2 Evaluation of estimation of direction of players' movements

We conducted the second experiment using the positions of the ball and the players. We considered two cases when analyzing the estimation results: moving players and stationary players. Players were only considered to have moved if the distance between the median of the past positions and the median of the estimated positions was greater than the motion threshold, which was arbitrarily set to 3.7 pixels. We defined eight possible directions that a player could move toward. We set 45 degrees as the maximum allowable difference between the estimated direction and the actual direction to consider a estimation as successful. Otherwise, the estimation was

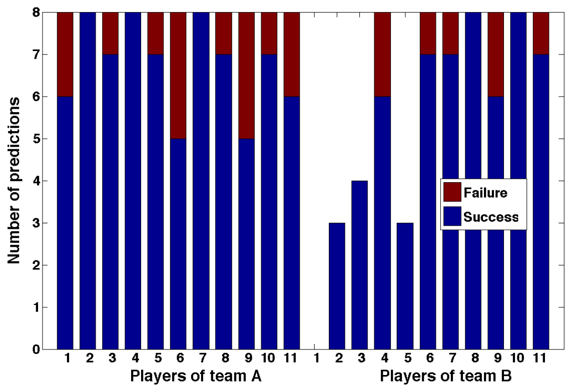


Figure 8: Success rates of ball direction estimation for players kicking the ball.

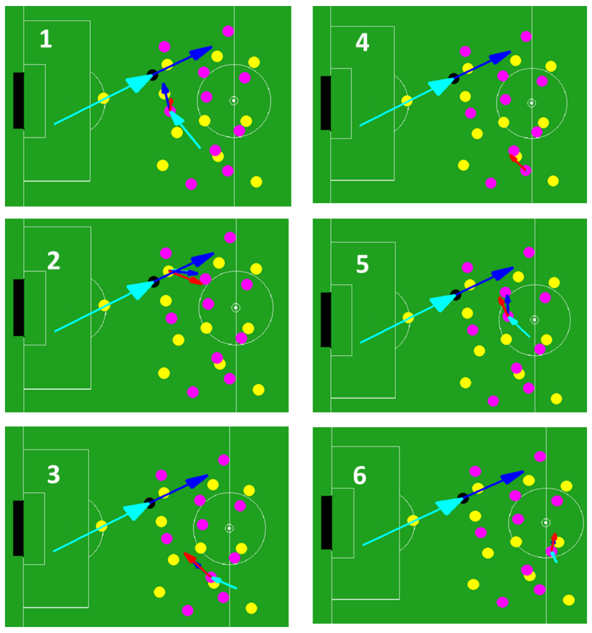


Figure 9: Examples of correct estimations by several players in different positions in the field. The yellow and pink points represent the players' positions for each team, and the black point represents the ball. The cyan arrow pointing to the ball indicates the previous direction of the ball and the cyan arrow pointing to a player indicates the previous direction of that player. The blue arrows originating at the ball and the players indicate the actual future direction of the ball and of the players. The red arrow indicates the estimated direction of the player.

considered a failure.

Figure 9 and Figure 10 show examples of correct estimations and failed estimations of the direction of the motion of several players for the same frame.

In Figure 9 we observe examples of estimation results for six different players using our system, including cases in which the direction before and after the estimation did not change significantly, as well as cases in which players were previously stationary and then started to move (image 4), or were moving in one direction and then changed direction (image 2). Figure 10 shows examples of failed estimations. These included cases where the actual motion of the player was not significant enough to be considered moving, although the estimated direction was correct (image 7), where the players stayed practically stationary, so the estimations were considered failures, but the magnitude of the estimation was small (image 9 and 10), or where the estimated direction of the player differed from the actual, but the estimated direction followed the same direction as the ball (image 8). Figure 11 shows that, in general our system was able to effectively infer an average of 50.6% and 69.9% of the future directions of players of team A and B, respectively. This means that our approach was able to learn and generalize motion patterns from the raw data and then apply that knowledge to new scenarios. Addition-

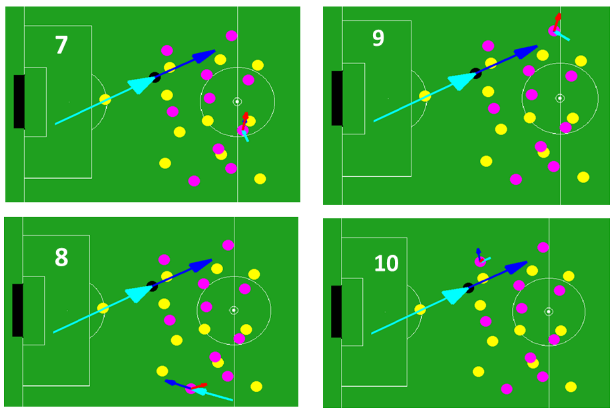


Figure 10: Examples of incorrect estimations by several players in different positions in the field. The interpretation of this figure is identical to that of Figure 9

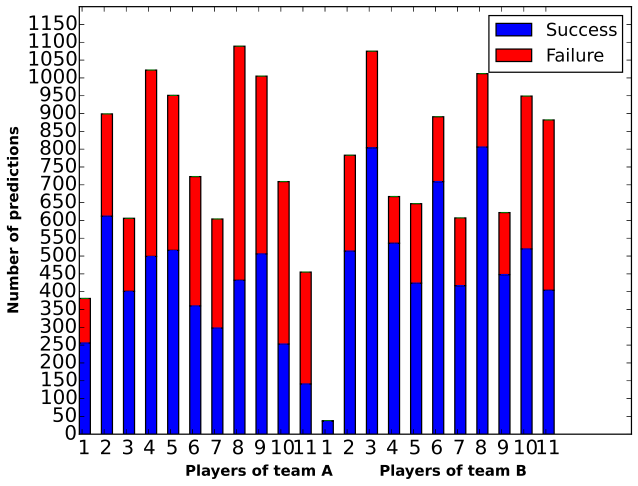


Figure 11: Results of motion direction estimation for each player.

ally, we see the correspondence between the fact that team B (HELIOBase, with 8 goals) had better performance than team A (WEBase, with 0 goals) during the matches, and the fact that the estimation by the players on team B was significantly higher than the number of correct estimations by the players of team A. It may suggest that the actions of the players on team B include strategic patterns and thus are more predictable. To summarize, these results indicate that our model was able under several conditions to learn motion patterns of the players and then apply that knowledge to infer future states.

5 Conclusions

In this study we proposed a method to estimate the action of soccer players and conducted experiments to measure this predictability using data from the RoboCup 2D Soccer Simulation League. The experimental results demonstrate that the future movement directions of the players and of the ball can be estimated successfully to a great extent; consequently, our method was proven to be able to estimate low-level predicted actions that could be adopted to represent teams' strategies. We are considering that task (i.e., adopting low-level actions for strategies' representation) as a future work for this study. Further work is required to classify the roles of the players by exploiting the capability of DNNs to learn high-level representations from raw features, to assess the ability to estimate players' trajectories for different sizes of the time window, and to validate our approach using data from the RoboCup Standard Platform and Middle Size leagues. Finally, we expect that introducing approaches based on DNNs like ours that estimate actions in a bottom-up manner opens the door to developing new skills for robots in RoboCup competitions.

6 Acknowledgements

This work was partially supported by the JSPS Grant-in-Aid for Specially Promoted Research (24000012).

References

[1] Visser, Ubbo, and Hans-Georg Weland. Using Online Learning to Analyze the Opponents Behavior, RoboCup 2002: Robot Soccer World Cup VI. Springer Berlin Heidelberg, 2003.

[2] Bowling, Michael, Brett Browning, and Manuela Veloso. Plays as Effective Multiagent Plans Enabling Opponent-Adaptive Play Selection, Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling, 2004.

[3] Trevizan, Felipe W., and Manuela M. Veloso Learning Opponents Strategies in the RoboCup Small Size League, International Conference on Autonomous Agents and Multiagent Systems, Springer, 2010.

[4] Yasui, Kotaro, Kunikazu Kobayashi, Kazuhito Murakami,and Tadashi Naruse. Analyzing and Learning an Opponent's Strategies in the RoboCup Small Size League, RoboCup 2013: Robot World Cup XVII, Lecture Notes in Computer Science, Volume 8371, 2014, pp 159-170.

[5] Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. Reducing the Dimensionality of Data With Neural Networks, Science 313.5786, 2006, pp 504-507.

[6] Bengio, Yoshua. Learning Deep Architectures for AI, Foundations and Trends in Machine Learning 2.1, 2009, pp 1-127.

[7] Noda, Kuniaki, et al. Multimodal Integration Learning of Object Manipulation Behaviors Using Deep Neural Networks, 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems IEEE, 2013.

[8] Chapelle, Olivier, and Dumitru Erhan. Improved Preconditioner for Hessian Free Optimization, NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011.