EMPEROR Soccer Simulation 2D Team Description Paper 2023

Erfan Fathi , and Soroush Mazloum

EmperorTeamRCSS@gmail.com
fathye897@gmail.com
mazloomsoroush@gmail.com

Abstract. This Team Description Paper introduces the overview of recent works done in EMPEROR team. Recently we have been working on several effective algorithms for optimizing different actions. Some of these actions which are presented more later are through pass which is important for better team performance, defending strategy that has the most effect on the result of a game and decision making that chooses the best action for the situation.

Keywords: Perceptron Neural Network, Entropy, Soccer Simulation 2D

1 Introduction

EMPEROR team members were gathered in 2022. The target was participating in IranOpen 2022. The result of that participation was getting the 6th place in Soccer Simulation 2D-Starter league. One year later, in IranOpen 2023, we managed to get the 1st place in Soccer Simulation 2D-Starter league.

Note: We are using Cyrus2D Base [1] as our base for Soccer Simulation 2D. Cyrus2D Base is created by merging Helios base (Agent2D) with Glider2D base and applying features from Cyrus2021, the champion of RoboCup2021 in Soccer Simulation 2D league.

2 Related work

Now, we are going to check some articles published by other Soccer simulation 2D teams. HELIOS developed "Player's MatchUp" algorithm for exchanging players' positions during the game for better team performance [2]. CYRUS uses opponent's pass prediction for marking and teammate's pass prediction for unmarking [3]. Hades2D improved players' dribble with splitting the generated sector and scoring them for the best decision [4]. Tehran2D improved defense by developing a block algorithm [5]. Persepolis optimized its offensive strategy by randomly placing players in the soccer 2D field and training them for the best attack [6]. MT2022 used HFO (Half Field Offense) for training and testing their shoot algorithm [7]. Apollo2D developed Tree structure model and A* search for chain pass [8]. Alice uses Monte Carlo tree search algorithm to find the best chain action possible [9].

3 Trough pass

3.1 Through pass summery

Through pass is a good way to break into the opponent's danger area but it's impossible for the player with the ball to calculate all the possible passes in one cycle (0.1 seconds). To tackle this problem we used messaging between the player that has the ball and the other players. The players except the player with the ball calculate the possible points to receive a through pass and the acceleration needed for the ball to reach that point exactly when the player reaches there. Then the player sends this information to the player with the ball.

In fact, the incentive of this idea is to be distinctive. We know that there is a limited period of time to review each order. This has created innovation in the configuration of the orders of the EMPEROR's team.

Messaging in the team provides an opportunity to compensate for the lack of review time. So, instead of the player with to ball going through the process, we can proceed with the player who does not have the ball. Then, by messaging between the players, the information needed to pass is sent to the player with the ball.

In the last step, all the eligible players who sent message to the player with the ball should be prioritized. The EMPEROR team uses **Perceptron Neural Network Algorithm** for this task.
Now let's explain each step more:

First step: As we mentioned, all the steps are calculated by the player who may receive the pass. Firstly, the player must choose some points that are effective for offense and going deep in opponents' defense (See Fig. 1 and Fig. 2). Then the player should check the pass routes so opponent players won't be able to intercept the ball.

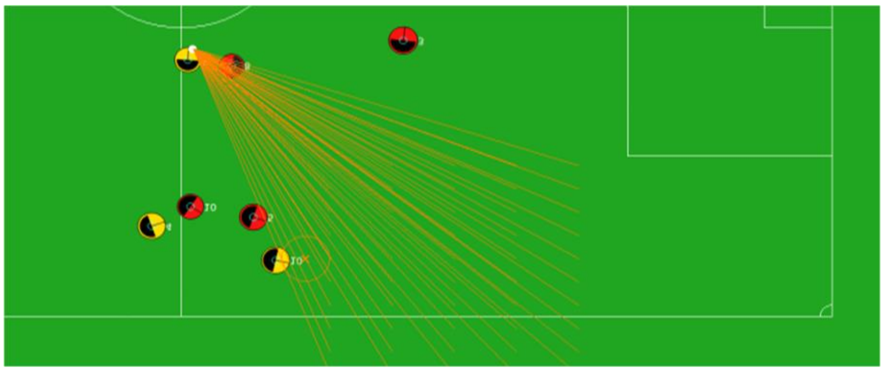

Fig. 1. Points checked by player no. 11.

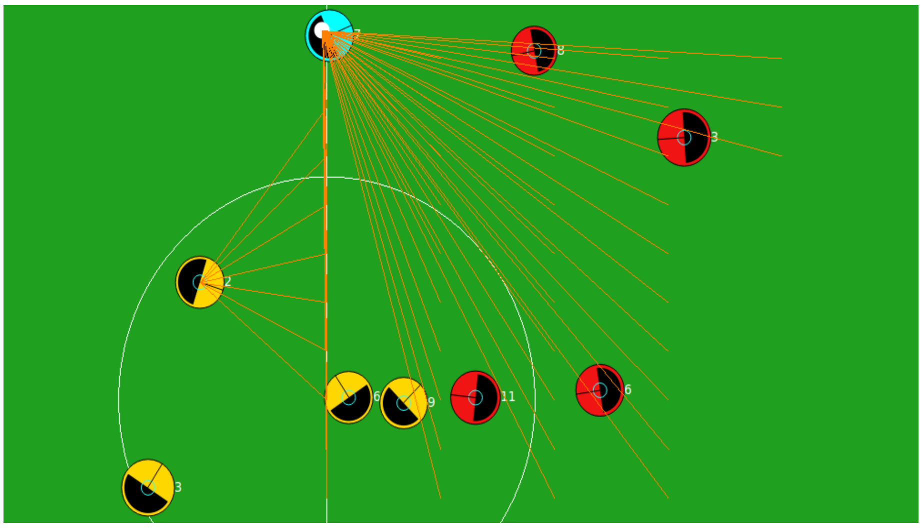

Fig. 2. Points checked by player no. 9 and player no. 2.

Second step: Sometimes many players find too many safe positions to pass. Well, the simple thing that can be done in these situations is to apply our priorities to the obtained items. But doing this as a method in the field of algorithmic thinking can make the team's playing the same in most cases and this is the big problem that we want to solve by using the Perceptron Neural Network algorithm.

First, we need to consider weights for all our values. For example, the preference of our thinking is to choose the most forward player... which of course, the possibility of correctness of this thinking can be effective in some teams, but we want to optimize the work steps.

To draw conclusions from this algorithm, we must save all the thoughts and ideas of the opposing team from the beginning of the game (using constant variables). After scoring, we start to check the existing situations. Now, using the saved information about the type of movement of the opposing team's defense and the densest points of their pitch, let's change the initial scoring (using the points-to-errors chart).

In this way, it is possible to choose the right player so that he does not face problems in the next steps and does not lead to the loss of the ball.

Here is a simple test that finds the best weight of our value (see Fig. 3 and Fig. 4).

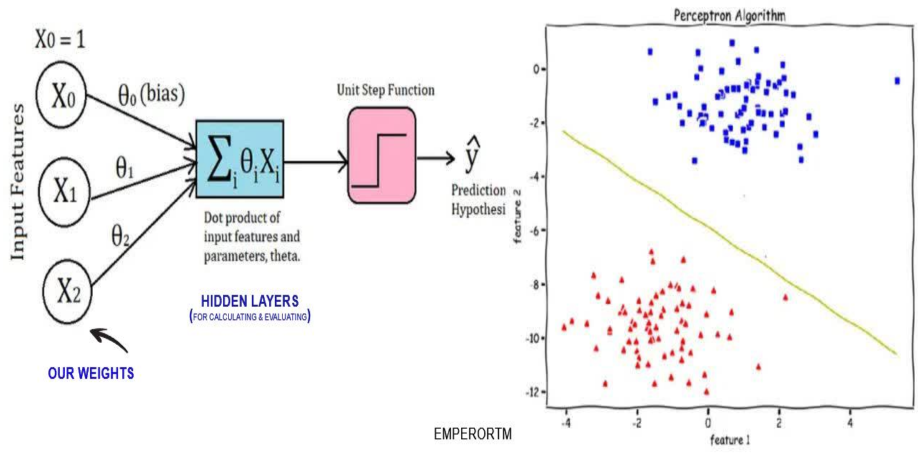

Fig. 3. This is how it calculates and evaluates our data.

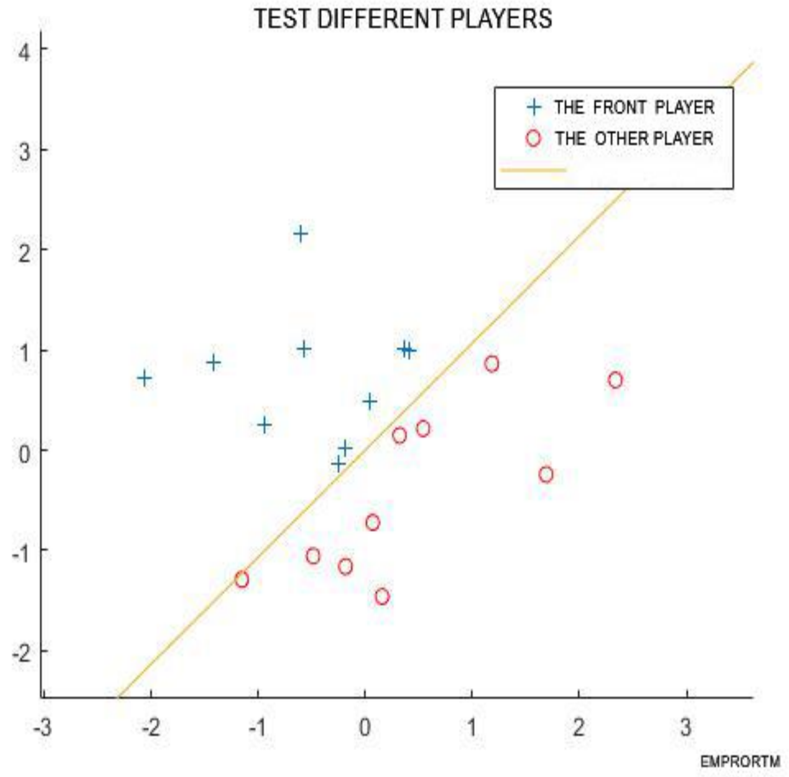

Fig. 4. This is how it calculates and evaluates our data.

By doing the above checks, you can get an acceptable result for your priorities in a game.

Third step: The player should now send the coordinate of the pass and the acceleration needed to apply to the ball to the player with the ball. This part is a bit tricky as there is no parser to request for a throw pass which also sends the acceleration beside the coordinate. In order not to write another parser for through pass we found out a solution to that problem:

In this method, player uses the below algorithm to send the acceleration with Y coordinate.

X, Y of the coordinate:

$$X = 23 \rightarrow Output = 23$$
$$Y = 12 \rightarrow 12 * 100 + a(for\ example\ 2.3) * 10 \rightarrow Output = 1223$$

final coordinate:

$$Vector2D\ (23, 1223);$$

As you have seen, we transfer acceleration and speed in this way. X is remained the same, but Y has changed.

When the player with the ball receives this information, he does the reverse of this algorithm to extract the data.

3.2 Pass security

For checking the pass security, player calculates the inertia point of the ball and makes sure no opponent will intercept the ball in the middle of the way. This way may take longer but it is safer. Moreover, inertia point helps us to calculate the acceleration needed to apply to the ball.

4 Defense

4.1 Defense Layers

Layers: Our defense line is basically made of three layers. Each layer is responsible to do certain jobs.
First layer is the back layer that is made by players no .2 – 5.
Second layer (middle layer) is made by players no. 6 - 8.
The final (front layer) is made by players no. 9 - 11.

Layer 1: This layer is responsible for marking opponent's forward players and not letting them to scape and receive a through pass.

Note: When marking opponents’ forward player, our player should go to a X coordinate, less than opponent player's X coordinate so if opponent player wants to scape, our player has time to react.

Layer 2: This layer prevents opponent from dribbling forward and if needed, intercepts them (See Fig. 5). Also it often presses opponent's players (See Fig. 6) and helps layer one in *inside penalty area marking*. Players in this layer spend the most energy compared to players in other layers because they take part in *both* defense and offense. So it is necessary for them to move with a less dash power (They have to move slower).

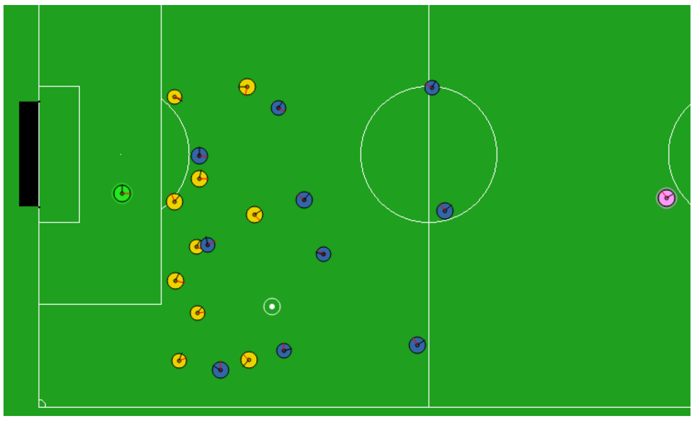

Fig. 5. You can see our middle line is helping defense line.

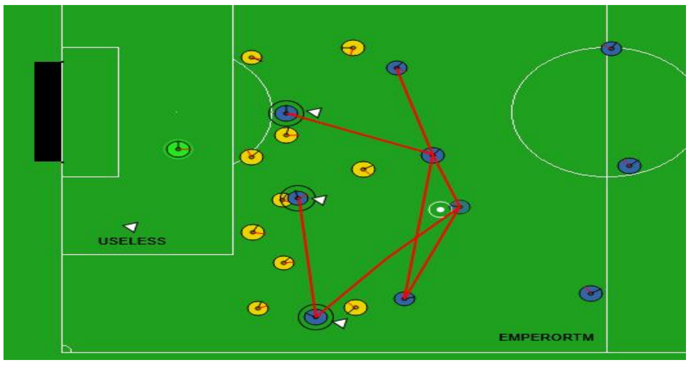

Fig. 6. This figure shows how our defense makes it difficult for opponent to get near our goal.

Layer 3: This layer has to block passes from opponent's defense/middle line to forward line. When defending, players in this line move with less dash power rate to recover from our previous attack.

Note: Defense positions are calculated with simple math operator when ball X is more than -30.

4.2 Defending near our goal

Obviously, it will be critical if ball gets near our goal. So positioning will be more important than before. As a result we manually wrote positions for the situations in which ball's X is less than -30. This method makes is harder for opponents' positioning near our goal. (See Fig. 7)

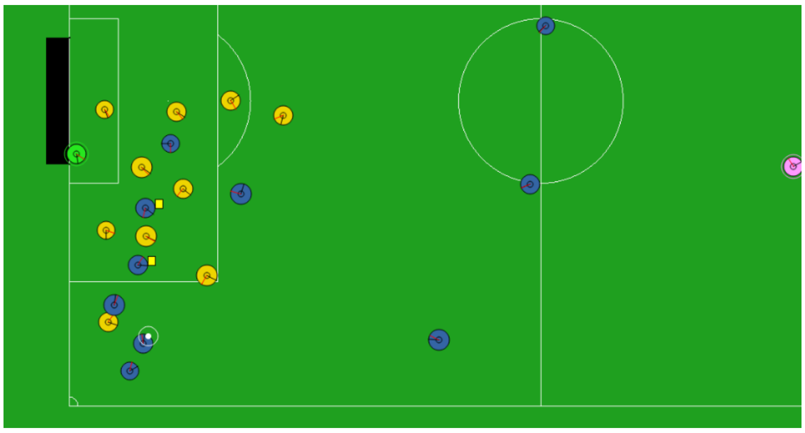

Fig. 7. We manually entered positions for the players.

5 Decision making

5.1 Entropy

As mentioned above, the same thinking is not effective against different teams. Let's look at a simple example to explain how to make a decision in different situations to **pass** or **dribble**.

Entropy is a method to determine the degree of purity, the amount of disorder or impurity (correctness of our decision) of a set of samples.

To explain more, if we have a set of data such as **<u>*S*</u>** that a certain feature divides them into **<u>*C*</u>** different classes, then the entropy of the set **<u>*S*</u>** is equal to:

$$Entropy(S) = \sum_{i=1}^{c} - pi\ log2\ pi$$

As a simple example:

$$C1 \quad 2 \qquad P(C1) = \frac{2}{6} \qquad\qquad P(C2) = \frac{4}{6}$$

$$C2 \quad 4 \quad Entropy = -\left(\frac{2}{6}\right) log2 \left(\frac{2}{6}\right) - \left(\frac{4}{6}\right) log2 \left(\frac{4}{6}\right) = 0.92$$

Note: If our entropy value is close to zero, it means that our purity value is at maximum, but if our entropy value is close to one, at most half of our value is impure.

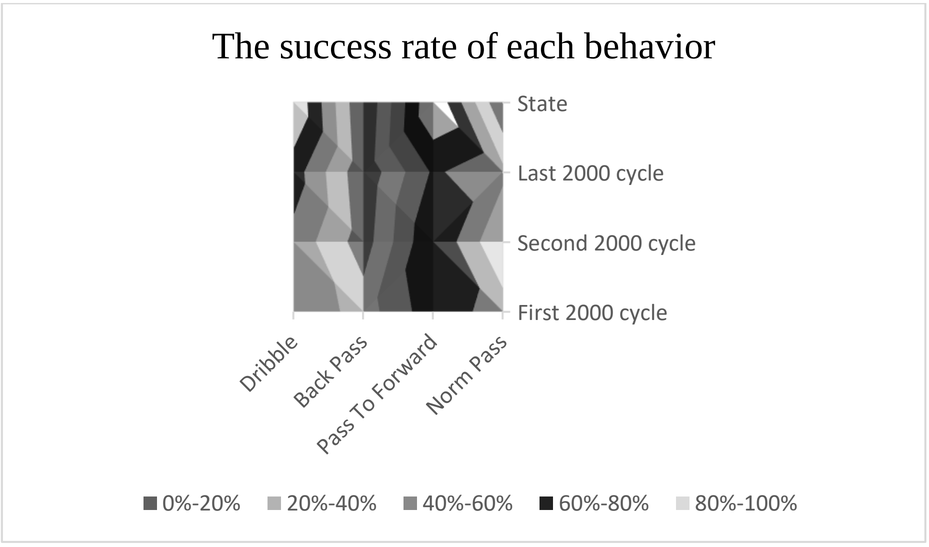

Fig. 8. Success rate of each behavior.

Table. 1. The success rate of each behavior

ENTROPY *(data)*	First 2000 cycles	Second 2000 cycles	Last 2000 cycles	State
Dribble	(6/10) 60%	(5.4/10) 54%	(7/10) 70%	TRUE
Back Pass	(3/10) 30%	(1/10) 10%	(0.5/10) 5%	FALSE
Pass To Forward	(7.3/10)73%	(8/10) 80%	(6.3/10) 63%	TRUE
Norm Pass	(5/10) 50%	(2/10) 20%	(4.5/10) 45%	FALSE

To illustrate:

$$Entropy = -(\frac{6}{10})log2(\frac{6}{10}) - (\frac{4}{10})log2(\frac{4}{10}) \approx 0.97$$

As it is clear from the above entropy result, for the *dribble success rate in the first 2000 cycles*[1], less than half of our situations are unsuccessful. So, team's priorities need to be changed.

5.2 Information gain

The information gain of a behavior = the amount of entropy reduction that is achieved by separating samples through this feature.

The information gain **Gain(S, A)** for a similar feature **A** compared to a set of examples **S** is defined as follows:

$$Gain(S,A) = Entropy(S) - \sum_{\upsilon \in Values(A)} \frac{|S\upsilon|}{|S|} Entropy(S\upsilon)$$

In this way our team can make an acceptable decision during the game.

6 Future ideas

1. Storing information in certain files about opponent behaviors for use during the game.
2. Reinforcement learning in payers' attack system.

References

1. CYRUS team. Cyrus2DBase. [Online] Available: https://github.com/Cyrus2D/Cyrus2DBase
2. Yamaguchi, M., Kuga, R., Omori, H., Fukushima, T., Nakashima, T., & Akiyama, H. (2021). *HELIOS 2021 Team Description Paper.*
3. Zare, N., Firouzkouhi, A., Amini, O., Sarvmaili, M., Sayareh, A., Ramezani Rad, S., Matwin, S., & Soares, A. (2022). *CYRUS Soccer Simulation 2D Team Description Paper 2022*.
4. Esmaelifar, S., Rokni, R., Esmaelifar, S., Akhondi, F., Rajabi, A., & Pourmoghaddam, R. (2022). *Hades2D soccer 2D simulation Team Description Paper.*

5.Khanjari, S. (2022). *Tehran2D Team Description Paper.*

6.Noohpisheh, M., Shekarriz, M., Zaremehrjardi, F., Khademi Ardekani, F., & Khorsand, S. (2021). *Persepolis Soccer2D Simulation Team Description Paper 2021.*

7.Guan, L., Chen, Q., Wang, J., Xiang, H., Meng, S., Wang, C., Xu, H., Fang, H., & Chen, S. (2022). *MT2022 Team Description Paper.*

8.10. Li, Y., Lin, C., Yu, S., Chen, Y., Jie, X., & Jiang, M. (2022). *Apollo2D Team Description Paper.*

9.Wang, F., Guo, Y., Zheng, X., & Liu, X. (2021). *Alice2021 Team Description Paper.*

10. Zare, N., Sayareh, A., Sarvmaili, M., Amini, O., Soares, A., & Matwin, S. (2021). *CYRUS Soccer Simulation 2D Team Description Paper.*

11. Gabel, T., Sommer, F., Breuer, S., & Godehardt, E. (2019). *FRA-UNIted Team Description Paper2019.*

12. Marian, S., Luca, D., Sarac, B., & Cotarlea, O. (2021). *OXSY 2021 Team Description.*

13. Cheng, Z., Guo, J., Ren, Y., Tang, Y., & Rong, L. (2022). *YuShan2022 Team Description Paper for RoboCup 2022*.

14. Ahgayari, R., Shahbazy, S., Iranmanesh, M., & Nikfetrat, A. (2022). *IraNad Soccer 2D Simulation Team Description Paper 2022.*

15. Noohpisheh, M., Shekarriz, M., Zaremehrjardi, F., & Karimi, M. (2022). *Persepolis Soccer 2D Simulation Team Description Paper 2022.*

16. Akiyama, H., Nakashima, T., & Hatakeyama, K. (2022). *HELIOS 2022 Team Description Paper.*