

# PASS EVALUATING IN SIMULATED SOCCER DOMAIN USING ANT-MINER ALGORITHM

Mohammad Ali Darvish Darab

*Qazvin Azad University*

*Mechatronics Research Laboratory, Qazvin Azad University, Qazvin, Iran*

Mosalam Ebrahimi

*Qazvin Azad University*

*Mechatronics Research Laboratory, Qazvin Azad University, Qazvin, Iran*

Mohammad Reza Shamshiri

*Qazvin Azad University*

*Mechatronics Research Laboratory, Qazvin Azad University, Qazvin, Iran*

## ABSTRACT

This paper utilizes Ant-Miner, the first Ant colony algorithm for discovering classification rules, in a simulated soccer domain to enhance teamwork abilities among agents. The method is used to create appropriate rules using collected training data to estimate whether passing the ball to a particular teammate will result in a success or not. The quality of each constructed rule is considered as estimation for a successful pass. The results showed that the constructed rules with this method are more reliable and accurate in comparison to the widely used constructed rules generated by C4.5

## KEYWORDS

RoboCup Soccer Simulation, Classification Rule Discovery, Multiagent System

## 1. INTRODUCTION

In Multiagent systems such as RoboCup Soccer Simulation, collaboration between independent agents has a critical role since each agent should decide and act independently. Because of inherent complexity of such a system there is much interest in using machine learning techniques to help deal with this complexity (Stone, 1998).

Passing the ball as a basic collaborative behavior in a soccer match has a great influence on the performance of a team as a whole. So, it seems it's necessary for a team to have a good policy about it. One of the major and basic problems regarding this issue is the estimation of whether or not a pass to a particular teammate will succeed that is called Pass Evaluation (Stone, 2000). If an agent estimates the success rate of a pass in a given situation, then he might use it to make a better decision. In other words, if an agent can realize that the receiver of his pass can not collect the ball successfully, he could make another decision (e.g. Dribble Shoot, etc.).

In this paper we propose a method to obtain a measure to estimate the success rate of a pass (Evaluate a Pass) using Ant-Miner (Ant Colony-based Data Mining) algorithm. Here we use this algorithm to discover appropriate rules from gathered training data for Pass Evaluation. Once these rules are discovered we use them to classify a given situation of a pass into three classes of Success, Failure, Miss. Considering the predicted class and the rule used for this prediction, we estimate the success rate of the pass. In our results it was revealed that the discovered rules using this method are more reliable and accurate in comparison with the wildy used constructed rules generated by C4.5 (Quinlan, 1993).

## 2. ANT-MINER ALGORITHM

An Ant Colony system involves simple agents (ants) that cooperate with each other to achieve an emergent, unified behavior for the system as a whole, producing a robust system capable of finding high-quality solutions for problems with a large search space. In the context of rule discovery, an Ant Colony system has the ability to perform a flexible, robust search for a good combination of logical conditions involving values of the predictor attributes (Parpinelli et al, 2002).

As mentioned before the method introduced in this paper for discovering classification rules to evaluate a pass is based on Ant-Miner algorithm. Ant-Miner outputs an ordered set of rules in the form of below:

```
IF <conditions> THEN <class>
```

This algorithm discovers appropriate rules by a number of ants that incrementally construct/modify rules (Parpinelli et al, 2002). Before starting to apply Ant-Miner algorithm on training data, continuous attributes should be discretized since current version of Ant-Miner can not handle continuous attributes directly. After discretizing, there will be a set of terms for each attribute which the best ones should be discovered and used to construct appropriate rules.

Each ant starts with an empty rule (a rule without any term in its antecedent) and adds one term at a time to its current partial rule. The choice of the term to be added to the current partial rule depends on both heuristic function and on the amount of pheromone associated with each term (Parpinelli et al 2002).

Heuristic function for a term is a measure of entropy associated with that term that states the ability of this term to improve the predictive accuracy of the rule. After each ant constructed its rule there is an increase of the pheromone of all terms contained in the rule and a decrease of the pheromone of unused terms as simulating pheromone evaporation.

The Ant-Miner algorithm is as follow:

Alg 1. High-level psuedocode of Ant-Miner

```
TrainingSet = {all training cases};
DiscoveredRuleList = [ ]; /* rule list is initialized with an empty list */
WHILE (TrainingSet > Max_uncovered_instances)
t = 1; /* ant index */
j = 1; /* convergence test index */
Initialize all trails with the same amount of pheromone;
  REPEAT
    i: Ant starts with an empty rule and incrementally constructs a
      classification rule Rt by adding one term at a time to the current rule. It
      continues until adding any other term will cause (the number of covered
      cases < Min_instances_per_rule) or all attributes are already part of
      current rule;
    ii: Prune rule Rt;
    iii: Update the pheromone of all trails by increasing pheromone in the
      terms followed by Ant t (proportional to the quality of Rt) and decreasing
      pheromone in the other (simulating pheromone evaporation);
  IF (Rt is equal to Rt - 1) then
    j = j + 1;
  ELSE j = 1;
  END IF
  t = t + 1;
  UNTIL (t >= No_of_ants) OR (j >= No_rules_converg)
    Choose the best rule Rbest among all rules Rt constructed by all the ants;
    Add rule Rbest to DiscoveredRuleList;
    TrainingSet = TrainingSet - {set of cases correctly covered by Rbest};
  END WHILE
```

1: (*No\_of\_ants*): The maximum number of iterations the Repeat-Until loop of the algorithm will perform.

2: (*Min\_instances\_per\_rule*): During the construction phase, a term cannot be added to the current partial rule if this would make the number of covered cases fall below this number. The algorithm thereby excludes possible solutions which cover very few cases.

3: (*Max\_uncovered\_instances*): The point at which rule discovery will stop. Each iteration of the While Loop reduces the size of the training set and the number of cases will finally fall below this number.

4: (*No\_rules\_converg*): The alternative stopping condition for the Repeat-Until loop is if this number of identical rules is produced in sequence by the ants.

### 3. ANT-MINER ALGORITHM FOR PASS EVALUATING

In this paper we aim to discover appropriate rules that work as a function like  $\varphi(x, y)$  where  $x$  and  $y$  are the last state of the world from passer and receiver perspective respectively. The output of such a function is the predicted classification for these inputs. Considering the calculated quality of the rule used for this prediction we estimate the success rate of the pass as a number in the range of  $[0, 1]$ . To achieve this we should follow the steps below:

- 1: Recognizing the world state attributes that might be relevant (e.g. Distance and direction of the passer to the receiver).
- 2: Defining a constrained training scenario and gathering enough training data.
- 3: Discretizing continuous attributes.
- 4: Applying Ant-Miner Algorithm on gathered training data to discover appropriate rules.

#### 3.1 Training Scenario and Collecting Training Data

Pass action requires two different agents. A passer must kick the ball towards a receiver, who must collect the ball. Although the execution of a pass is not difficult in the open field, it becomes more complicated in the presence of opponents that they also try to intercept the pass. Albeit we assume opponents are equipped with the same ball-interception capability as our agents since it's an effective parameter on evaluating a pass.

We used the training process to gather training data which is used in (Stone, 2000). The players are placed randomly in the field and the passer announces its intention to pass. Then the teammates reply with their views of the field. The passer chooses a receiver randomly among its options. Then the passer announces to whom it's passing and opponents and receiver attempt to collect the ball. The training instance is recorded as a Success if the receiver can collect the ball successfully, a Failure if one of the opponents intercept the pass, and a Miss otherwise.

Also, we recorded the values of 174 attributes for each training instance as they are used in (Stone, 2000) too. Half of these attributes are from passer perspective and half of them are from receiver perspective. It seems these attributes have enough potential for evaluating a pass where Ant-Miner also has the ability to prune irrelevant ones (Parpinelli et al, 2002).

#### 3.2 Discretizing Continuous Attributes

As mentioned before, in order to apply Ant-Miner on training set we should discretize continuous attributes such as metric data (e.g. the distance between passer and receiver). We used C4.5-Disc (Dougherty et al, 1995) since it's a supervised discretization method which it makes the use of the instance label information while performing discretization (Kovahi et al, 1996). C4.5-Disc is an entropy-based discretization method that applies C4.5 to each continuous attribute separately (Dougherty et al, 1995). It takes a reduced data set containing just the values of each attribute and the associated classification as input. Then it generates a decision tree which the nodes are considered as threshold values for discretizing that continuous attribute. We used Weka<sup>1</sup> system to apply C4.5 on these reduced data sets. Below you can see the tree generated by Weka for one of these reduced data sets, the Passer-Opponent1-Distance (the nearest opponent to the passer) attribute.

---

<sup>1</sup> Weka is a Java implementation of a collection of commonly-used machine learning algorithms, including C4.5. Weka was developed and is maintained by staff at The University of Waikato, New Zealand. Available from <http://www.cs.waikato.ac.nz/ml/weka>

```

J48 pruned tree
-----
passer-opponent1-distance > 22.3: 5 (702.58/278.58)
passer-opponent1-distance <= 22.3
| passer-opponent-distance <= 11.2
| | passer-opponent-distance <= 5.3: 1 (507.6/294.3)
| | passer-opponent-distance > 5.3: 2 (886.58/591.51)
| | passer-opponent-distance > 11.2
| | | passer-opponent-distance <= 14.9: 3 (904.18/649.04)
| | | passer-opponent-distance > 14.9: 4 (1073.18/797.04)
Number of Leaves: 5
Size of the tree: 9

```

### 3.3 Applying Ant-Miner Algorithm on Training Data

After discretizing the continuous attributes, we have a set of ordinal attributes that can be used by Ant-Miner algorithm. Let  $term_{ij}$  be a rule condition of the form  $A_i = V_{ij}$ , where  $A_i$  is the  $i$ th attribute and  $V_{ij}$  is the  $j$ th value of the domain of  $A_i$ . The probability that  $term_{ij}$  is chosen to be added to the current partial rule is given by equation (1): (Parpinelli et al, 2002)

$$P_{ij}(t) = \frac{\tau_{ij}(t) \cdot \eta_{ij}}{\sum_i^a \sum_j^{b_i} \tau_{ij}(t) \cdot \eta_{ij}}, \forall i \in I \quad (1)$$

Where:

- $\eta_{ij}$  is the value of a heuristic function for  $term_{ij}$ ;
- $\tau_{ij}(t)$  is the amount of pheromone currently available (at time  $t$ ) in the position;
- $i, j$  of the trail(term) being followed by the ant;
- $a$  is the total number of attributes; (here 174 attributes);
- $b_i$  is the total number of values on the domain of attribute  $i$ ;
- $I$  are the attributes not yet used by the ant.

The heuristic function  $\eta_{ij}$  is a measure of the predictive power of  $term_{ij}$ . The higher the value of  $\eta_{ij}$  it is, the more relevant for classification and so the higher probability to add to the current partial rule. Ant-Miner heuristic function uses the normalized, information-theoretic function which is: (Parpinelli et al, 2002)

$$\eta_{ij} = \frac{\log_2(k) - InfoT_{ij}}{\sum_i^a \sum_j^{b_i} \log_2(k) - InfoT_{ij}} \quad (2)$$

Where:

- $k$  is the number of classes; (here there is three classes of Success, Failure and Miss .So it's fixed on 3)
- $a$  is the total number of attributes;
- $b_i$  is the number of values in the domain of attribute  $i$ ;
- $InfoT_{ij}$  is a measure of entropy (amount of information) associated with  $term_{ij}$ . (Cover T.M, 1991)

After an ant finished its rule construction, it should prune the rule since it can improve rule simplicity and eliminate irrelevant terms. The basic idea is to start with the full rule and iteratively remove one term at a time from the rule while this process improves the quality of the rule according to the equation (4).

At the beginning of an AntRun the pheromones of all terms are initialized and given equal values which are the inverse of the total number of terms (Parpinelli et al, 2002). Then after an ant finished its rule construction and pruning, the amount of pheromone of unused terms is decreased by dividing the value of each  $\tau_{ij}$  by the summation of all  $\tau_{ij}$  and the pheromone values of the terms found in the rule are increased according to the following equation: (Parpinelli et al, 2002)

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \tau_{ij}(t) \cdot Q \quad (3)$$

Where  $Q$  is the quality of the rule which can be calculated by: (Parpinelli et al, 2002)

$$Q = \frac{TP}{TP + FN} + \frac{TN}{FP + TN} \quad (4)$$

Where:

- *TP* (true positives) is the number of cases covered by the rule that have the class predicted by the rule;
- *FP* (false positives) is the number of cases covered by the rule that have a class different from the class predicted by the rule;
- *FN* (false negatives) is the number of cases that are not covered by the rule but that have the class predicted by the rule;
- *TN* (true negatives) is the number of cases that are not covered by the rule and that do not have the class predicted by the rule.

We use the quality of a rule calculated by equation (4) as our measure for pass evaluation. More precisely, whenever the inner loop in Alg.1 is finished and a rule is constructed we calculate the quality of the rule by equation (4). Since the value of *Q* is always within [0, 1] and also it's a measure of predictive power of the rule, we can consider it as the probability of the correct classification for that instance.

As we said before there are four user-input parameters that should be tuned manually. We fixed these parameters as follows:

- Number of Ants (*No\_of\_ants*) = 4000;
- Minimum number of cases per rule (*Min\_instances\_per\_rule*) = 110;
- Maximum number of uncovered cases in the training set (*Max\_uncovered\_instances*) = 120;
- Number of rules used to test convergence of the ants (*No\_Rules\_Converg*) = 15.

We used 110 for *Min\_instances\_per\_rule* because this parameter acts as a control on the amount of overfitting allowed to the training data. It means, the greater the value of this threshold, the more general the rule antecedents created by ants are forced to be. The other values also were tuned manually. It seemed these values are good enough for such a training data.

## 4. EXPERIMENTAL RESULTS

We used the constructed rules by Ant-Miner for evaluating a pass skill in RoboCup Soccer Simulation2D domain. To do so, we used CMUnited-99 low-level source code and implemented the constructed rules as Pass Evaluation layer. This way we made the results of two method comparable since such an evaluation depends on not only the teammate's and opponent's position, but also their abilities to receive or intercept the pass (Stone, 2000).

In order to test the predictive accuracy of the constructed rules, we ran 5000 trials and recorded the results. The results were compared with the results in (Stone, 2000) (see Table 1). We form this experiment with four opponents and one passer and three teammates. Albeit these results were obtained under a condition of forced passing which means the passer was forced to pass the ball when it was the ball possessor. Obviously, in a full game situation it can choose another action (such as Dribble, Shoot, etc).

Table 1. The result of 5000 trials which the passer uses the constructed rules by Ant-Miner. In each column the predicted success rate and its associated number are given. Also the real results are given in percent

	Estimation of Success Rate					
	C4.5			Ant-Miner		
	0.8-0.9	0.7-0.8	0.6-0.7	0.8-0.9	0.7-0.8	0.6-0.7
Number	1050	3485	185	1318	2913	391
Success%	79	63	58	82	69	64
Failure%	15	29	31	12	23	33
Miss%	5	8	10	6	8	3

Also, we investigated the results of the passes in a full game situation within 2500 passes using constructed rules. The results are given in Table 2. Since just the successful passes are ideal in a real game, we recorded just the number of successful passes and their associated estimation of success rate. At each step the passer estimates the success rate of a pass to potential receivers and chooses the best one based on its associated success rate.

Table 2. The result of 2500 trials in 15 full games .In the leftmost column the estimation of success rate of the passes using the constructed rules by Ant-Miner algorithm are given whereas the rightmost column shows the real success rate of the passes in percent

<b>Estimation of success rate</b>	<b>No of Passes in 15 games</b>	<b>No of success</b>	<b>Success percentage</b>
0.85 – 1.0	507	418	82.4%
0.75 – 0.85	1099	788	71.7%
0.65 – 0.75	894	547	61.1%

## 5. CONCLUSION

Previously empirical comparison across several domains between Ant-Miner and C4.5 has shown that they are competitive with respect to predictive accuracy. But to our knowledge it was the first time that these two methods were compared in a complex and dynamic domain such as RoboCup Soccer Simulation. In this paper we applied Ant-Miner algorithm to obtain a measure for evaluating a pass which means to estimate of whether a pass to a particular teammate will succeed or not. Having such knowledge in a soccer match can contribute to the decision of to which player to pass or of whether to pass, dribble, shoot, etc. Using Ant-Miner, we constructed appropriate rules using collected training data to classify an instance of a pass situation into three classes of Success, Failure and Miss. Then we calculated the quality of a constructed rule based on its sensitivity and specificity and considered it as a measure to evaluate a pass. Then we went through a series of practical experiments and compared it with C4.5, since it's a traditional method for the purpose of rule discovery among a data set and moreover is used to evaluate a pass. Our result showed that Ant-Miner can construct more reliable and accurate rules even in such a dynamic and real-world domain.

## ACKNOWLEDGEMENT

This Research was supported in part by Mechatronics Research Laboratory at Qazvin Azad University. The authors are thankful to Dr. Mousakhani and Dr. Haghghat for their Supporting.

## REFERENCES

- Cover, T.M and Thomas J.A, 1991. *Elements of Information Theory*. John wiley & Sons Publishers, New York, USA.
- Dougherty, J. et al, 1995. Supervised and Unsupervised Discretization of Continuous Features. *Proceeding of the Twelfth International Conference on Machine Learning*. Tahoe City, CA, pp. 194-202.
- Kovahi, R and Sahami M, 1996. Error-Based and Entropy-Based Discretization of Continuous Features. *Proceeding of 2<sup>nd</sup> International Conference Knowledge Discovery and Data Mining*. AAAI Press, CA, pp. 114-119.
- Holden, N and Freitas, A, 2004. Web Page Classification with an Ant Colony Algorithm. *Parallel Problem Solving from Nature*. Springer-Verlag, USA, pp. 1092-1102.
- Parpinelli, R. et al, 2002. An Ant Colony Algorithm for Classification Rule Discovery. *Data Mining: Heuristic Approach*. Idea Group Publishing, pp. 191-208.
- Quinlan, J.R, 1993. *Programs for Machine Learning*. Morgan Kaufmann Publishers, San Francisco, USA.
- Stone, P and Veloso, M, 1998. A Layered Approach to Learning Client Behaviors in the RoboCup Soccer Server. *Applied Artificial Intelligence (AAI)*, Vol. 12, No 2, pp. 165-188.
- Stone, P, 2000. *Layered Learning in Multi-Agent Systems: A Winning Approach to Robotic Soccer*. MIT Press, USA.