Classifier With Genetic Algorithms Deals With Rare Actions

Yu-Hsin Lin
Department of Computer Science
National Chiao Tung University
HsinChu City 300, Taiwan
lewislin3@gmail.com

Zheng-Wen Chen
Department of Computer Science
National Chiao Tung University
HsinChu City 300, Taiwan
akxeshomio@gmail.com

ABSTRACT

Learning Classifier System(LCS), Extend Classifier system(XCS) ..... are Classifiers with rules and genetic algorithms include. It is shown that these kind of classifiers can have high accuracy in lots of scenarios. However, these kinds of classifiers often overlook actions that have a low percentage, which we call that minor actions, since it has a lower chance to be trained in the population of rules. The result shows that these rules often have low fitness. Even doesn't exist in the population of rules. In this work, we just try to build a simple classifier system to deal with some scenarios that include minor actions. We propose setting a bound of amount of every action in the population to make sure every action would exist in our rule. The experimental result shows that by adding a bound to the population of the rule in each action can give minor actions a bigger chance to survive and produce higher fitness rules.

General Terms

Algorithm, Design, Parameter

Keywords

Classifier System, Genetic Algorithm, Evolutionary Computation, Mutation, Minor Actions, Imbalanced Dataset

1. INTRODUCTION

Learning from imbalanced data sets is part of the most challenging issues in the field of machine learning. In these data sets, the sample's share of the data far exceeds other categories. In this case, the classification algorithm does not lose its high accuracy even if the classification algorithm completely ignores the samples from the subcategories. If you do not have to learn the algorithm carefully, the final model will usually favor the main class[1]. This may be a very large problem in most cases. Detection of unlabeled minor samples, such as minor samples infected with rare diseases, is the main target of prediction. Statistics and machine learning have taken different approaches to address data imbalances.

Learning Classifier Systems(LCS) is a learning method that uses evolutionary algorithms like genetic algorithm to create rule-base classifier. Varieties of classifier system that is based on LCS including Extend Classifier System(XCS), Zero Classifier System(ZCS) and more. Like most methods in machine learning area that classifies, LCS often overlooks minor actions in imbalanced data sets. To deal with minor actions, many methods were given to deal with imbalanced data set. In some research, they pick out the minor actions and train it separately. Others may adjust the input data by increasing the amount of minor actions before training. We think it is inappropriate because we may not know what actions are minor actions when we use LCS. Therefore, we propose using a bound of amount of actions to improve the learning of minor actions.

This paper is organized as follows. In Section II, We give the description of our classifier algorithm. In Section III, we present the concept of our action amount bound. We show experimental results on the multiplexer problem and give discussions in Section IV. Finally, we present our guideline and summarize our contributions of this paper in Section V.

2. EASY CLASSIFIER (ESC)

In this project, we set up an easy classifier (ESC) that includes genetic programming. The ESC gets input from the environment at the beginning of every turn. If there isn't any population in the population set that matches the input, the classifier random creates a population to match the input. The classifier collects every population that matches the input of

the match set. The classifier then choose an action from the match set. Environment updates fitness of the population of the match set. The last step of the classifier, we use genetic algorithms in the population set to create new populations. *The flow of ESC is showed in Figure 1.*

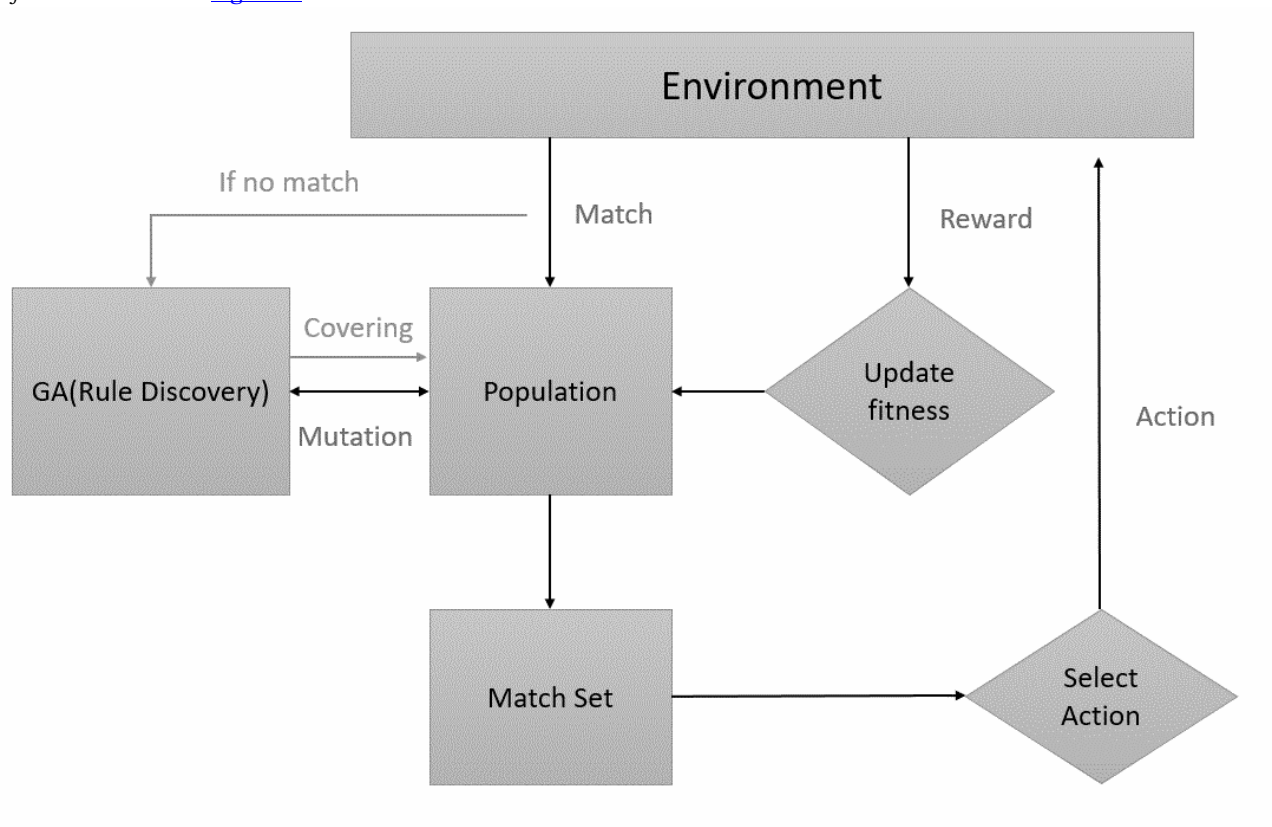

***Figure 1:** The flow of ESC in a run*

2.1 Rules

In the population set of ESC, the populations contains three parts. Rules, actions, and parameters. For the rules, there will be a string of bit, including don't care. The rule decides if the population matches the input. The action represents the action that the population gives if it matches the input. For the parameter part, the parameter will be list[Table 1] below.

***Table 1:** The structure of a population*

Rules	A string of bits.
Actions	The action that a population represents.
Fitness F	To show how good a population is, which will be update and use in GA.
Match time MT	The counter that a population is sent to th match set.
Success time ST	The counter that a population gets reward from environment.
Error time ER	The counter that a population gets penalty from environment.

2.2 Fitness Update

The method that we update fitness is that we multiply the original reward/penalty by the success rate/error rate of the rule in the match set. For example, in a scenario of multiplexer, we give award to the correct rule in the match set 1. If the rule is having 80% of success rate, the fitness of the rule would be multiply by 1 * (1+80%). The calculate of the match time, success time and error time can be reset after an amount of match time.

2.3 Genetic Algorithm Update

In every run of this classifier, the GA will produce N new rules from the population by selecting rules from the population and doing crossover and mutation to create new rules. The GA random selects two population from the population set with the same action, than creates a new rule. The population with new rule will replace the old population from the classifier with the lowest fitness.

3. POPULATION RATE (PR)

In lots of classifier systems, they often overlooks low percentage actions and low ratio of populations that has the action will exist in the population set. In ESC however, we add a parameter that protects every action that has a population in the population set, which called population rate PR. To make sure every action exist a minimum amount of rule, the PR limits the amount of rule with every action. It can make sure every action occupy a certain percentage of population. By make sure giving the minimum amount of size to every population, we can let low ratio action have more chance to train and produce a better population.

We Implement the action bound by checking the PR of actions every time we kill a rule in the population. If the action's PR is lower than the bound, the rules with this action will not be choose even if they own a very low fitness. We use this method to keep the minor actions.

4. EXPERIMENT AND RESULT

4.1 6-1 Multiplexer

Multiplexer is a common scenario to check if a classify method works. In ESC, we take 20000 runs to get the result below. *Figure 2* represents the correct rate of ESC running multiplexer. The correct rate is calculated every 100 runs.

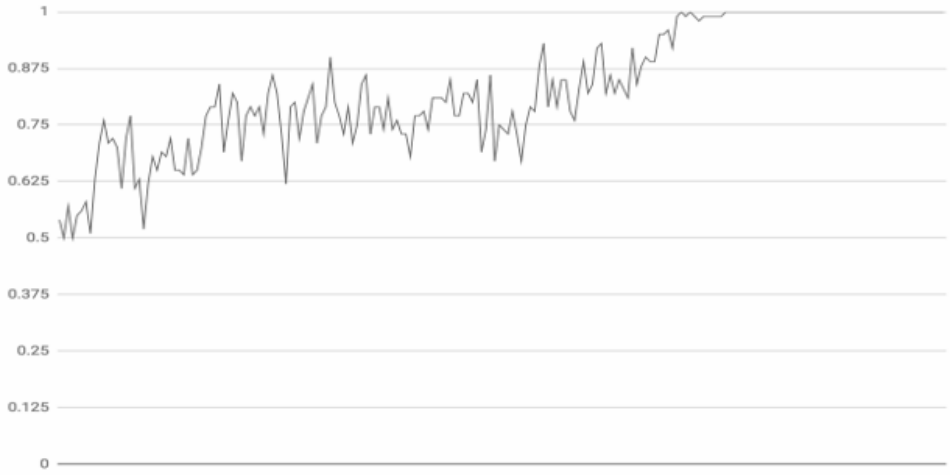

***Figure 2:** Correct rate of 6-1 multiplexer in ESC (20000 run)*

In the result, we can see that ESC can 100% recognize 6-1 multiplexer around 15000 runs, showing that ESC is a way that really can classifies a problem with different actions.

4.2 Simple Scenario

To make sure if PR works, we first came up an really simple scenario that the action only depends on the first bit of the input. (0#####->0, 1#####->1) The result is obvious that the ESC can recognize this scenario in a very short of time. Figure 3 shows the correct rate of ESC running this scenario in 10000 runs.

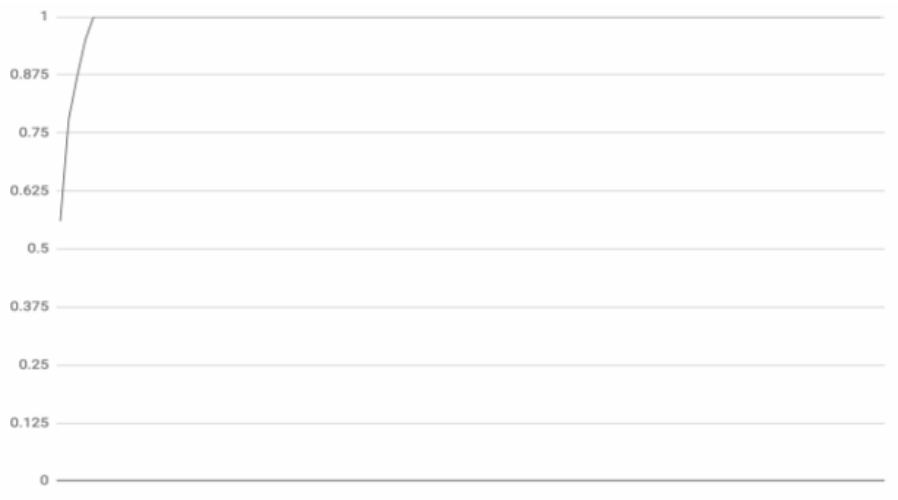

Figure 3: Correct rate of Simple scenario in ESC (10000 run)

4.3 Adding low ratio action

In this part of the experiment, we add one special rule to the scenario in 4.2. If the bit string contains all 0, instead of outputting 0, we change it to 2, making 2 in this scenario a low ratio rule (1.5%).

Figure 4 is the result of this scenario with ESC without PR, and Figure 5 is the result of this scenario with ESC with PR. we can found that the one without PR have a flowing correct rate around 98% which makes sense since it always gets an error when the answer is 2. On the other hand, the second result which using PR has a little difference at the end of the experiment that can have 100% of correct rate.

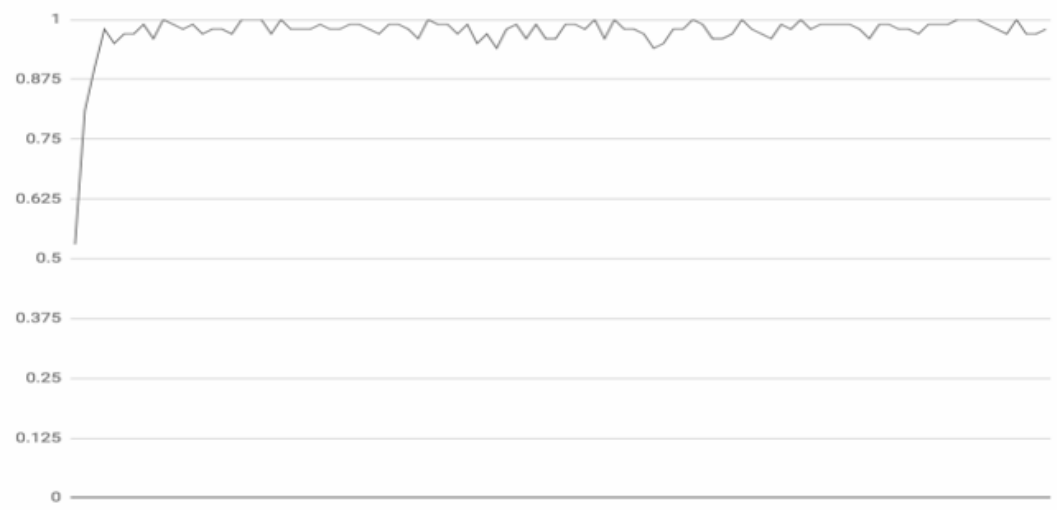

Figure 4: *Correct rate of simple scenario adding low ratio rule without PR in ESC (10000run)*

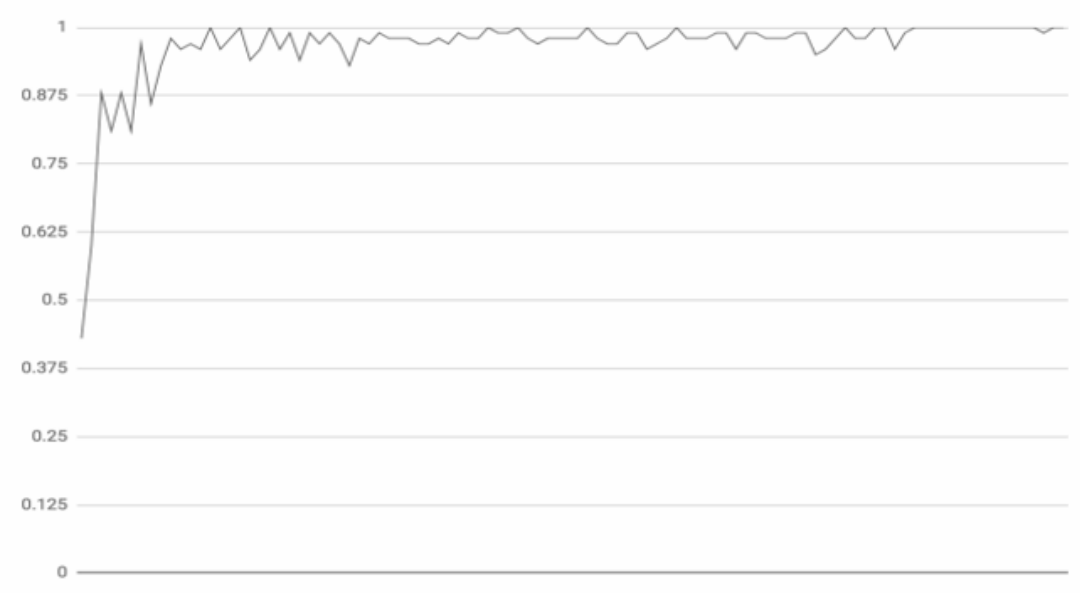

Figure 5 ***Correct rate of simple scenario adding low ratio rule with PR in ESC (10000run)***

Since the difference isn't that obvious, we try to expand the running time to 50000 run to watch if the PR actually works. In the picture below, we can see that without PR in Figure 6, the correct rate still floating around 98% while the ESC with PR in figure 7 can 100% recognize every input including low ratio action 2.

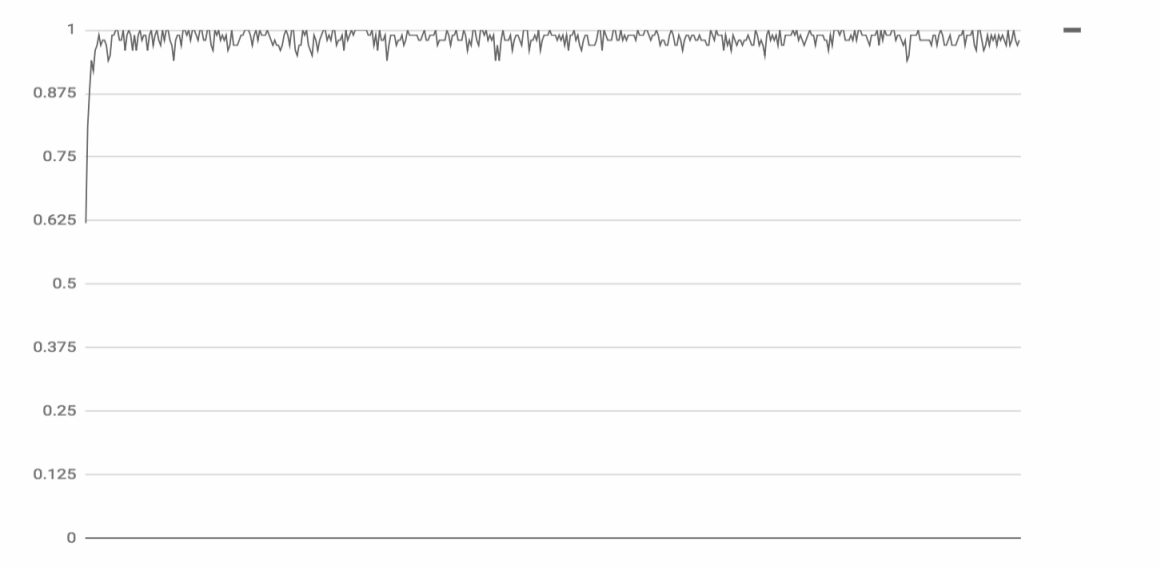

Figure 6: ***Correct rate of simple scenario adding low ratio rule without PR in ESC (50000run)***

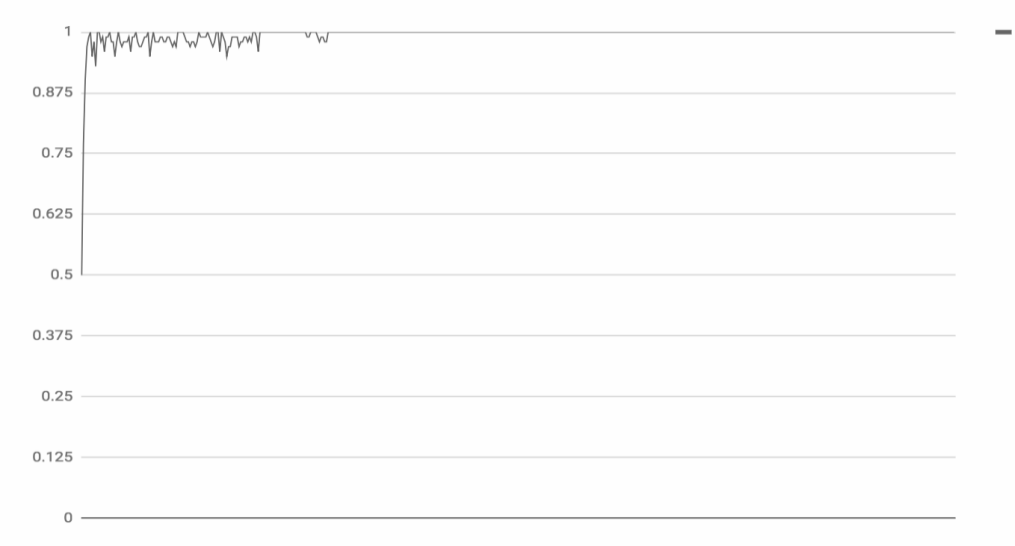

Figure 7: Correct rate of simple scenario adding low ratio rule with PR in ESC (50000run)

5. CONCLUSION

In the experiment, we can found that ESC works on multiplexer and scenario with low ratio action, which the algorithm seems to be useful in classifying. Also, the parameter PR can not only use in ESC but other classifier systems too. To combine PR with other methods that deals with low ratio actions, classifiers may have better performance.

REFERENCE

[1] Hooman Sanatkar, Saman Haratizadeh. An XCS-Based Algorithm for Classifying Imbalanced Datasets. International Journal of Intelligent Information Systems. Vol. 4, No. 6, 2015, pp. 101-105. DOI: 10.11648/j.ijiis.20150406.12