Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 61 (2015) 361 – 366

Procedia Computer Science

Complex Adaptive Systems, Publication 5
Cihan H. Dagli, Editor in Chief
Conference Organized by Missouri University of Science and Technology
2015-San Jose, CA

An Improved Continuous-Action Extended Classifier Systems for Function Approximation

Omid Saremi[a]*, Masoud Shariat Panahi[a], Amin Sabzehzar[b]

[a] *Mechanical Engineering Department, University of Tehran, Tehran, Iran*
[b] *Mechanical Engineering Department, University of Nevada, Reno, Nevada, USA*

Abstract

Due to their structural simplicity and superior generalization capability, Extended Classifier Systems (XCSs) are gaining popularity within the Artificial Intelligence community. In this study an improved XCS with continuous actions is introduced for function approximation purposes. The proposed XCSF uses "prediction zones," rather than distinct "prediction values," to enable multi-member match sets that would allow multiple rules to be evaluated per training step. It is shown that this would accelerate the training procedure and reduce the computational cost associated with the training phase. The improved XCSF is also shown to produce more accurate rules than the classical classifier system when it comes to approximating complex nonlinear functions.

Peer-review under responsibility of scientific committee of Missouri University of Science and Technology

Keywords: Improved XCSF; Classifier Systems; Function approximation

1. Introduction

Learning classifier systems were first introduced by John Holland in 1992[1]. The early variants used a set of randomly generated classifiers (rules) that were gradually evolved into more specialized rules using Genetic Algorithms. In 1994 Wilson introduced ZCS which improved the performance of LCS and reduce its computational cost[2]. In 1995 Wilson proposed a new way to calculate the classifier's fitness which was a quantitative measure of a rule's desirability. He presented eXtended Classifier Systems (XCS) in which accuracy plays a major role in

* Corresponding author. Tel.: +98-935-558-2736
E-mail address: omidsaremi@ut.ac.ir

Peer-review under responsibility of scientific committee of Missouri University of Science and Technology
doi:10.1016/j.procs.2015.09.160

computing classifier fitness[3]. Later studies mostly focused on XCS and its improvements like XCSR[4], TCS[5], and etc.

XCSF is one of modified version of XCS, which presented by Wilson in 2002[6]. It evolves classifiers represent piecewise linear approximations of reward that is commonly the problem solution or the function value[7]. It firstly had been used as approximator[6]. Later, Lanzi et al[8, 9] have done some generalizations on XCSF prediction updating method. Some advanced approximators such as polynomials[10], neural networks[11] and … were studied by them. Other studies were conducted on combining several types of conditions like ellipsoids[12], convex hull[13] and … with computed prediction. They showed that an adequate combination of approximator and representation results in having a more competitive XCSF and conducts it to have superior noise-robust performance[14].

In this study, a new simple low cost method in order to increase XCSF performance and decrease the algorithm error as well as its convergence time is suggested. In the next sections, a brief description of XCSF is given. In the section 3, an improvement on XCSF will be introduced. Section 4 is dedicated to improved algorithm results and discussing them. And at last, conclusion will be presented in section 5.

2. The original XCSF

As expressed before XCSF is an extent to eXtended Classifier Systems. It is distinguished from XCS in three respects[6]: i) using continuous integers instead of binary coding in condition part, (this method has also been used for XCSI, too)[15]. ii) A vector of weights ω is used to compute classifier prediction. And at last, iii) the weights ω are updated instead of the classifier prediction.

XCSF includes a population of rules. Each rule contains condition part, action part, calculated prediction, classifier error, classifier fitness, classifier experience and its numerosity. Unlike XCS that uses binary genes, XCSF uses two real genes for each input. Forming condition part is done using upper and lower limits of each interval. Therefore, u_i and l_i (the upper and lower limits of the i^{th} interval) are used to bound each interval.

Another section of a classifier condition part is weight vector. This vector is used to calculate each classifier prediction. By presenting an example, classifiers which cover the inputs; form the Match set [M]. Each classifier prediction is calculated using linear approximation method, which is brought as below:

$$h(x) = \omega.x' \tag{1}$$

Where ω is the weight vector $(\omega_0, \omega_1, ..., \omega_n)$ and x' is the input vector x augmented by a constant x_0, $(x_0, x_1, ..., x_n)$. In the case of function approximation, the function can be mapped by calculated prediction. So, the action part is omitted according to prediction mapping.

Like XCS, prediction array in which each rule could be selected to form Action set [A] with the probability extracted from its prediction, is composed of classifiers with calculated prediction. A difference between XCSF and XCS, which has a great impact on convergence period, is XCSF disability to form subsets in prediction array. This problem causes the algorithm to select just one classifier for Action set [A] formation. In order to overcome this problem, a solution, which is the main idea of this paper, will be presented.

Applying the Action set to the environment results a reward for updating the rule properties such as classifier error and fitness. In order to update each classifier error and experience, the following formulas (like in XCS) are used:

$$\exp_i = \exp_i + 1 \tag{2}$$

$$\varepsilon_i = \begin{cases} \varepsilon_i + \dfrac{(|R - P_i| - \varepsilon_i)}{\exp_i} & \text{if } \exp_i < \dfrac{1}{\beta} \\ \varepsilon_i + (|R - P_i| - \varepsilon_i) & \text{if } \exp_i \geq \dfrac{1}{\beta} \end{cases} \tag{3}$$