# XCS-CR: Determining Accuracy of Classifier by its Collective Reward in Action Set toward Environment with Action Noise

Takato Tatsumi
The University of Electro-Communications
Japan
tatsumi@uec.ac.jp

Tim Kovacs
University of Bristol
United Kingdom
Tim.Kovacs@bristol.ac.uk

Keiki Takadama
The University of Electro-Communications
Japan
keiki@hc.uec.ac.jp

## ABSTRACT

Accuracy based Learning Classifier System (XCS) prefers to generalize the classifiers that always acquire the same reward, because they make accurate reward predictions. However, real-world problems have noise, which means that classifiers may not receive the same reward even if they always take the correct action. For this case, since all classifiers acquire multiple values as the reward, XCS cannot identify accurate classifiers. In this paper, we study a single step environment with action noise, where XCS's action is sometimes changed at random. To overcome this problem, this paper proposes XCS based on Collective weighted Reward (XCS-CR) to identify the accurate classifiers. In XCS each rule predicts its next reward by averaging its past rewards. Instead, XCS-CR predicts its next reward by selecting a reward from the set of past rewards, by comparing the past rewards to the collective weighted average reward of the rules matching the current input for each action. This comparison helps XCS-CR identify rewards that result from action noise. In experiments, XCS-CR acquired the optimal generalized classifier subset in 6-Multiplexer problems with action noise, similar to the environment without noise, and judged those optimal generalized classifiers correctly accurate.

## CCS CONCEPTS

• **Computing methodologies → Rule learning**;

## KEYWORDS

XCS, accuracy criteria, reward, alternative noise

**ACM Reference Format:**
Takato Tatsumi, Tim Kovacs, and Keiki Takadama. 2018. XCS-CR: Determining Accuracy of Classifier by its Collective Reward in Action Set toward Environment with Action Noise. In *GECCO '18 Companion: Genetic and Evolutionary Computation Conference Companion, July 15–19, 2018, Kyoto, Japan,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205651.3208271


*GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan*

ACM ISBN 978-1-4503-5764-7/18/07…$15.00
https://doi.org/10.1145/3205651.3208271

## 1 INTRODUCTION

It is useful to understand the characteristics of data by expressing data as if-then rules and revealing important elements within each data element. If-then rules using data elements as a condition part are easy to understand. There is the rule generalization as a way to clarify important elements. The rule generalization can ignore the data elements that do not affect the evaluation of the if-then rule and retain the data elements that affect the evaluation.

Learning Classifier Systems (LCSs) [4] are a family of evolutionary machine learning techniques that can generalize if-then rules that are called classifiers. LCSs find the optimal combination to generalize rules by genetic algorithm (GA) [3] and evaluate the generalized rules by reinforcement learning (RL) [8]. The generalized classifier is highly interpretable for humans, compared with other machine learning methods such as support vector machine and neural networks [13]. Among LCSs, XCS (Accuracy-based LCS) [12] is the current mainstream classifier system. XCS generalizes classifiers from the accurate classifiers by the GA and by the generalization mechanism called subsumption. In XCS, a classifier that always obtains a consistent reward is defined as an accurate classifier.

In real-world problems, however, it is difficult for XCS to acquire such accurate classifiers because the given input does not necessarily represent the situation correctly. For example, bad sensor performance or mistyping of data may be considered. Since the incorrect input given to XCS is different from the state of the environment, the acquired rewards may be significantly different. In order to make XCS applicable to various problems, it is necessary to realize stable rule generalization even in the above environments. In such a situation, it is very hard for XCS to generalize classifiers by subsuming the classifiers because many classifiers are not accurate due to the unstable inputs. This indicates that it is also hard for XCS to reduce the number of the low generality classifiers by the subsumption mechanism in XCS. From this difficulty, we need a new XCS that can acquire the generalized classifier rules in the environments with uncertain inputs.

As a first step toward achieving the above objectives, this paper assumes the environments with action noise, that is the environments where XCS's actions (also called outputs) are sometimes changed. When the input is changed, the degree of influence on the reward differs depending on the input value and changed element. This paper simplifies the noise model by changing the output instead of the input, because changing the output is guaranteed to have an effect on the reward. Regardless of whether noise is added to inputs or outputs, classifiers acquire rewards when the provide an answer to an input. In this paper, we study problems with 2

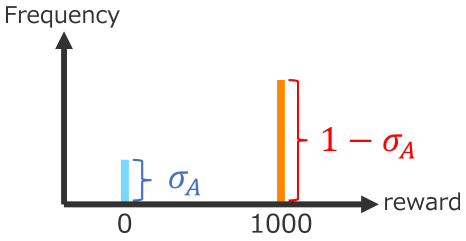



**Figure 1: Action noise**

actions and we use a reward of 1000 for correct answers and 0 for incorrect answers. We control the amount of action noise with $\sigma_A$, the probability of changing the output of the learner. Figure 1 shows the reward that is acquired by the accurate classifier with the action noise $\sigma_A$. The accurate classifier acquires the original reward (1000) with the probability $1-\sigma_A$, and acquires the reward different from the original reward (0) with the probability $\sigma_A$. Note that because accurate classifiers sometimes acquire the reward for incorrect answers, it is difficult to find the classifiers that should be judged the accurate when the noise is removed. Even when noise is added to both the input and the output, since the influence of the noises on acquired reward are the same, the simplification of this paper (using only action noise) is appropriate.

XCS judges the classifier whose range of the acquired rewards is smaller than a certain range (threshold) as accurate. But, as described above, even the accurate classifier acquires multiple value rewards in the environment where noise is added to the action, so all classifiers are determined as inaccurate. Since the subsumption does not occur, the number of the classifiers does not decrease. XCS$\mu$ is proposed by Lanzi *et al.* [6] as XCS applicable to the stochastic environment which is a type of noisy action environment. The stochastic environment is the maze environment with slip. Since the agent that searches for the maze slips at a certain probability, there are cases where the agent cannot go in the direction he wants to go. XCS$\mu$ subtracts the prediction error of all classifiers by the minimum value of prediction error among the classifier set. XCS$\mu$ eliminates uncertainty and makes it possible to identify the accurate classifiers. Since the stochastic environment is a multi-step problem, we do not know whether XCS$\mu$ is effective in the single-step problem covered in this paper. In the XCS$\mu$ paper, the evaluation of XCS$\mu$ was only the number of steps up to the goal and there was no mention of the generalization of the classifier.

To overcome this problem, this paper proposes a new XCS, XCS-CR (XCS based on Collective Reward) that can reduce the number of classifiers even in the noisy action environment. Unlike XCS, XCS-CR does not judge the accuracy of the classifier directly from the *actual* reward acquired by the classifier. Instead, XCS-CR judges the accuracy using an *estimate* of reward. If the action has been changed by noise, the actual reward will be the wrong reward for this input and action. However, the estimated reward may be the correct reward for this input and action. That is, the estimated reward can correct noise in the reward.

The estimated reward is determined by comparing the collective reward for each action. The collective reward is calculated for an action from the classifiers that match the current input. The estimated reward is the reward whose action has the highest collective reward. If a small amount of action noise biases the reward acquired by a classifier, XCS-CR can still accurately judged the accuracy of the classifier by using the estimated reward.

The rest of this paper is organized as follows. Section 2 introduces other XCSs that can handle the environments with uncertainty in input, action, or reward. Section 3, 4, and 5 explain the mechanism of XCS, XCS$\mu$, and XCS-CR, respectively. Section 6 provides the noisy multiplexer problem. Section 7 conducts the experiments. Section 8 discusses the experimental results. Finally, our conclusion is given in Section 9.

## 2 RELATED WORK

Various works have tackled to improve the performance of XCS in several uncertain environments previously.

The first uncertainty is related to an "input" from an environment. An example includes a Partially Observable Markov Decision Process (POMDP) environment. In POMDP, an environmental input is uncertain and missing parts of the information are required to distinguish states instantly. To tackle this type of environment, Lanzi *et al.* [7] and Webb *et al.* [11] proposed XCS with memory to solve the POMDP maze problems. It is necessary to know in advance how many previous internal states are needed.

The second uncertainty is related to an "action" to an environment. A representative study is XCS$\mu$, but details are described in Section 4.

The final uncertainty is related to a "reward" from an environment. An example includes the environment where a reward varies such as evaluations by a human in Interactive Evolutionary Computation [9]. In such an environment, the consistent reward is not guaranteed to be obtained even if the same action is performed as the action for the same input. Due to this feature, all classifiers are determined to be inaccurate, which prevents XCS from generalize the classifiers. To tackle this type of environment, we proposed XCS-MR [10] that dynamically determines the accuracy criterion from the sample standard deviation of the obtained rewards.

Reinforcement learning includes part of supervised learning. There are studies that corresponded with UCS which extended XCS to supervised learning, but this research focuses on reinforcement learning and is not treated the supervised learning methods.

## 3 ACCURACY-BASED LEARNING CLASSIFIER SYSTEM (XCS)

### 3.1 Overview

XCS is composed of 1) performance component, 2) reinforcement component, and 3) rule-discovery component as shown in Figure 2. These components evolve a set of classifiers in $[P]$.

### 3.2 Classifier and its generalization

The classifier in XCS is composed of the condition (if) part, action (then) part, prediction ($p$), prediction error ($\epsilon$) that is the difference between the prediction and reward ($P$), fitness ($F$), and numerosity ($n$). An LCS acquires knowledge by evolving classifiers to fit multiple environmental conditions. When the condition part is represented by a bit string with the fixed length composed of 0 and 1,

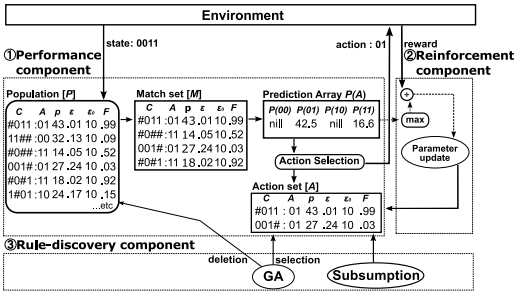



**Figure 2: Learning mechanism of XCS**

XCS generalizes the classifiers by using the symbol # representing "don't-care". For example, "10###" matches eight inputs.

## 3.3 Mechanism of XCS

*3.3.1 Performance component.* XCS selects one action and executes it. The algorithm in this component is summarized as follows: (i) the classifiers in $[P]$ whose condition part matches the current input are stored in the Match Set $[M]$. (ii) the Prediction Array is calculated by prediction of these classifiers and actions in $[M]$. The prediction of action $a_i$ is calculated by the following equation where $cl$ represents a classifier.

$$P(a_i) = \frac{\sum_{cl_k \in [M] | a_i} cl_k.p \times cl_k.F}{\sum_{cl_l \in [M] | a_i} cl_l.F} \tag{1}$$

(iii) an action is selected according to the prediction array. The classifiers in $[M]$ that have the selected action configure the Action Set $[A]$, which executes the action for the environment and receives the reward $P$. After receiving the reward, the reinforcement component is executed, and the evolution component is executed after a certain time.

*3.3.2 Reinforcement component.* The classifiers in $[A]$ have their parameters updated in this component as follows: (i) the prediction $p$ is updated from the obtained reward $P$ as follows.

$$cl.p \leftarrow cl.p + \beta(P - cl.p) \tag{2}$$

The variable $\beta$ is the learning rate and contributes the learning speed. (ii) the error $\epsilon$ that is the difference between $P$ and $cl.p$ is updated as follows.

$$cl.\epsilon \leftarrow cl.\epsilon + \beta(|P - cl.p| - cl.\epsilon) \tag{3}$$

However, when the number of updates of the classifier ($cl.exp$) is less than $1/\beta$, Equations (2) and (3) are replaced with Equations (4) and (5), respectively.

$$cl.p \leftarrow cl.p + (P - cl.p)/cl.exp \tag{4}$$

$$cl.\epsilon \leftarrow cl.\epsilon + (|P - cl.p| - cl.\epsilon)/cl.exp \tag{5}$$

These operations increase the learning speed at the beginning of learning. (iii) the fitness $F$ is calculated on the basis of accuracy $\kappa$ as follows.

$$\kappa(cl) = \begin{cases} 1 & \text{if } \epsilon < \epsilon_0 \\ \alpha \left(\frac{\epsilon}{\epsilon_0}\right)^{-\nu} & \text{otherwise} \end{cases} \tag{6}$$

In this equation, $\epsilon_0$ ($\epsilon_0 > 0$) is a constant variable that indicates the accuracy criterion. When $cl.\epsilon$ is smaller than $\epsilon_0$, the classifier is accurate. The variable $\alpha$ ($0 \leq \alpha \leq 1$) and $\nu$ ($\nu > 0$) control the reduction rate of the accuracy. Relative accuracy of the classifier $\kappa'$ is then calculated.

$$\kappa'(cl) = \frac{\kappa(cl) \times cl.n}{\sum_{x \in [A]} \kappa(x) \times x.n} \tag{7}$$

(iv) The fitness $F$ is updated as follows.

$$cl.F \leftarrow cl.F + \beta(\kappa'(cl) - cl.F) \tag{8}$$

*3.3.3 Rule-discovery component.* GA evolves the classifiers in $[A]$. The algorithm in this component is summarized as follows: (i) two parent individuals are selected according to the ratio of their fitness as a selection probability. (ii) two child individuals are generated by crossing these parent individuals. (iii) the elements of the condition part of these child individuals are mutated with probability $\mu$. (iv) when the number of the classifiers in $[P]$ exceeds the parameter N, the low fitness classifiers are deleted preferentially.

XCS has a subsumption mechanism to integrate the classifiers with a low generality into more generalized (more # in the condition part) classifiers. The classifiers whose experience $exp$ exceeds $\theta_{sub}$ and are judged to be accurate ($\kappa(cl) = 1$) can subsume classifiers having their condition part included in the condition part of the subsuming classifier. The numerosity of the subsumed classifier is added to the classifier that subsumes.

# 4 XCS$\mu$

## 4.1 Architecture of XCS$\mu$

The classifier of XCS$\mu$ is composed of the condition part, action part, prediction $p$, prediction error $\epsilon$, fitness $F$, and numerosity $n$ as with the classifier of XCS. In addition, XCS$\mu$ classifier has a parameter $\mu$ estimating the minimum prediction error.

## 4.2 Mechanism of XCS$\mu$

The most parts of the learning component of XCS$\mu$ are the same as the mechanism of XCS (Figure 2). The difference from XCS is the parameter updating in the reinforcement learning component. In this section, the different part is described in detail.

XCS$\mu$ calculates the minimum prediction error $\overline{\mu}$ of the classifiers that match the current input. The new parameter $cl.\mu$ is updated as follows.

$$cl.\mu \leftarrow cl.\mu + \beta_\epsilon(\overline{\mu} - cl.\mu) \tag{9}$$

The learning rate $\beta_\epsilon$ is smaller than the $\beta$ used to update the other classifier parameters. XCS$\mu$ updates the prediction error $\epsilon$ as follows instead of Equation (3).

$$cl.\epsilon \leftarrow cl.\epsilon + \beta(|P - cl.p| - cl.\mu - cl.\epsilon) \tag{10}$$

XCS$\mu$ estimates the prediction error $\epsilon$ by the classifier generalization by discounting the influence of the environmental noise.

The subsequent operation of the reinforcement component is the same as XCS.

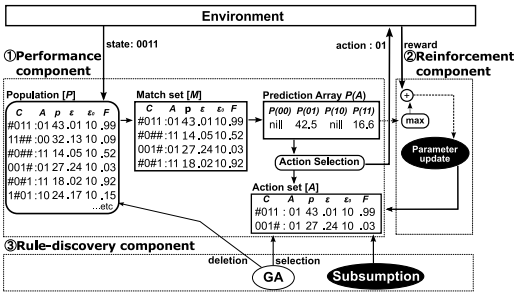



**Figure 3: Learning mechanism of XCS-CR**

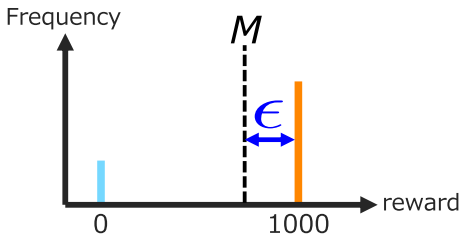



**Figure 4: $\epsilon$ of XCS-CR**

# 5 XCS BASED ON COLLECTIVE REWARD (XCS-CR)

## 5.1 Architecture of XCS-CR

The classifier of XCS-CR has the condition part, action part, prediction $p$, prediction error $\epsilon$, fitness $F$, and numerosity $n$ as same as the classifier of XCS. XCS-CR has some differences from XCS. XCS has one $\epsilon_0$ parameter that is shared by all classifiers, but in XCS-CR every classifier has its own $\epsilon_0$. XCS-CR classifiers have the mean of the acquired reward $M$ (see later) which is similar to $p$, but $M$ is more stable than $p$. In addition, an XCS-CR classifier has parameters to count the number of times it has received a particular reward $C_P$ and to count the number of times a particular estimated reward has been estimated $E_P$ and the mean of the acquired rewards $M$. The classifier has as many parameters $C_P$ and $E_P$ as there are possible reward values. In the multiplexer problem, since there are two possible reward values, 0 and 1000, the classifier has 4 counters $C_{P=0}$, $C_{P=1000}$, $E_{P=0}$, and $E_{P=1000}$.

## 5.2 Mechanism of XCS-CR

Most parts of the learning mechanism of XCS-CR are the same as the mechanism of XCS. The main differences from XCS are (a) the parameter updating in the reinforcement component and (b) the subsumption condition in the rule-discovery component. Figure 3 shows the learning mechanism of XCS-CR with the different parts that are blackened. In this section, the different parts are described in detail.

*5.2.1 Reinforcement Component.* XCS-CR does not judge the accuracy of the classifier directly using the reward acquired by the classifier. XCS-CR judges the accuracy of the classifier based on the estimated value of the acquired reward to be described later. A classifier that has an estimated reward that is always one is judged as an accurate classifier. A classifier with two or more estimated reward values is judged as an inaccurate classifier.

XCS-CR updates the classifiers in $[A]$ as follows. First, the classifier increments the count of the number of times each reward value has been acquired $C_P$. The $C_P$ to be incremented is the one whose value of $P$ is same the acquired reward $P$ (*i.e.*, $C_{P=0}$ for reward 0 and $C_{P=1000}$ for reward 1000).

$$cl.C_{P=P} \leftarrow cl.C_{P=P} + 1 \tag{11}$$

Second, the classifier updates the mean of its acquired reward $cl.M$. Third, the error $\epsilon$ of the classifier is updated as shown Figure 4. $\epsilon$ is calculated as the difference between the most frequently acquired reward $cl.mfr$ and the mean value of the acquired reward. The most frequently acquired reward $cl.mfr$ is the reward value for which $cl.C_P$ is the maximum.

$$cl.\epsilon \leftarrow cl.\epsilon + \beta(|cl.mfr - cl.M| - cl.\epsilon) \tag{12}$$

Fourth, XCS-CR tries to correct for the effect of noise by estimating the correct action if there was no noise. For each action set $[A]$ XCS-CR calculates a Collective Reward $CR$ using the equation:

$$CR_{A=a} = \frac{\sum_{cl \in [M]|a} cl.M \times cl.exp}{\sum_{C \in [M]|a} C.exp} \tag{13}$$

XCS-CR believes the action with the highest CR is the correct action. The classifier of XCS-CR counts the number of estimations $E_P$ for each reward. However, like $C_P$, $E_P$ to be incremented is the one whose value of $P$ is same the estimated reward $P$.

$$cl.E_{P=P} \leftarrow cl.E_{P=P} + 1 \tag{14}$$

An example is shown in Figure 5. In the binary classification problem, this is a case when the input state is 0011. $[M]$ is divided into two based on the action of the classifier. On the left side is a set of the classifiers with action 0. On the right side is a set of the classifiers with action 1. Since the mean values of the acquired reward of the left classifiers are larger than those on the right side, $CR_{A=0}$ is larger than $CR_{A=1}$. $cl.E_{P=1000}$ is incremented if action is 0. (When the action is 0, XCS-CR estimate the acquiring reward is 1000.) $cl.E_{P=0}$ is incremented if action is 1. (When the action is 1, XCS-CR estimate the acquiring reward is 0.)

Next, XCS-CR updates $\epsilon_0$ of the classifier. The classifier whose estimated reward is always same, that is, whose $E_P$ is 0 for one reward value, is accurate. The more # symbols there are in the condition the more inputs a classifier matches. However, in order to prevent judgment by only some matched inputs, even if the number of experience exceeds $2^{number\ of\ \#} \times \theta_{RE}$, the classifiers that satisfy the above condition are targeted. The parameter $\theta_{RE}$ is constant. Let $\epsilon$ be $Max\epsilon$, the largest $\epsilon$ among the classifiers in $[A]$ satisfying the above conditions. XCS-CR updates $\epsilon_0$ as follows.

$$cl.\epsilon_0 \leftarrow cl.\epsilon_0 + \beta(Max\epsilon - cl.\epsilon_0) \tag{15}$$

Last, XCS-CR updates fitness $F$ of the classifier executing Equation (6), (7), and (8) same as XCS.

*5.2.2 Subsumption condition.* Since the classifiers with many # symbols subsume many classifiers, it is necessary to judge the accuracy of the classifiers carefully. XCS has the condition that the number of evaluation times ($cl.exp$) of the subsuming classifier is greater than $\theta_{sub}$. In XCS-CR this is changed to the condition that

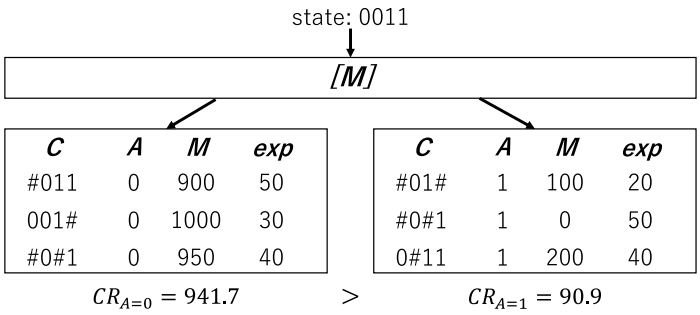



**Figure 5: Calculation of $CR_{A=a}$ of XCS-CR**

it is larger than $2^{number\ of\ \#} \times \theta_{RE}$ times. Since the values of $\theta_{sub}$ and $\theta_{RE}$ are assumed to be the same value, the subsumption condition of XCS-CR is more severe than the subsumption condition of XCS. The other condition ($\kappa(cl) = 1$) is the same as in XCS.

# 6 PROBLEM DESCRIPTION

## 6.1 Multiplexer problem

*6.1.1 Original problem.* This paper compares the results of XCS, XCS$\mu$, and XCS-CR in the $l$-Multiplexer ($l = 6$) problem that is a common benchmark problem of LCS [12]. In this problem, the first $k$ ($b_0 b_1 ... b_{k-1}$) bits of the input length $l = k + 2^k$ bits ($b_0 b_1 ... b_{l-1}$) convert into the decimal number $d$, and the $k + d$-th bit ($b_{k+d}$) is the correct answer. In the 6-Multiplexer problem where the input length is 6 (*i.e.*, $k = 2$), for example, $b_0 b_1$ ("11") convert into $d = 3$ when the input is given as "110010", and $b_{2+3} = b_5 = 0$ is the correct answer. In this problem, only the first $k$ bits ($b_0 b_1 ... b_{k-1}$) and the $(k+d)$-th bit ($b_{k+d}$) contribute to determining the action value, which means that the other bits can be any value because they do not contribute to determining the action value. XCS is expected to acquire the generalized classifiers by replacing the bits that do not affect the action with # as shown in the example below.

(if) 11###0 (then) 0

These are called the optimal classifiers. There are 16 optimal classifiers that are the combination of the 3 bits if part and the 1 bit then part that obtain consistent rewards in the 6-Multiplexer. The 16 classifiers are called the optimal subset [$O$].

*6.1.2 Modified noisy problem.* As the same as the typical the multiplexer problem, the reward is set to 1000 for the correct answer and 0 for the incorrect answer. *Reward*s in this experiment are determined by the following Equations (refer Figure 1). When the answer is correct.

$$Reward = \begin{cases} 0 & \text{probability } \sigma_A \\ 1000 & \text{otherwise} \end{cases} \quad (16)$$

When the answer is incorrect.

$$Reward = \begin{cases} 1000 & \text{probability } \sigma_A \\ 0 & \text{otherwise} \end{cases} \quad (17)$$

# 7 EXPERIMENT

The experiment simulates the uncertain environments by adding action noise $\sigma_A$ and compares XCS, XCS$\mu$, and XCS-CR to verify the effectiveness of each approach.

This experiment consists of the learning phase and the evaluation phase, the same as in previous studies e.g., [1]. The two phases are executed alternately. In the learning phase, XCSs fully learn state-action space in the environment by selecting a random action. In the other phase, XCSs select the action that is expected to obtain the maximum reward. The learning performance of XCSs is determined on whether XCSs select the correct action to the input. Note that the rule-discovery component is not executed in the evaluation phase, *i.e.*, XCSs do not generate new classifiers [5].

## 7.1 Cases

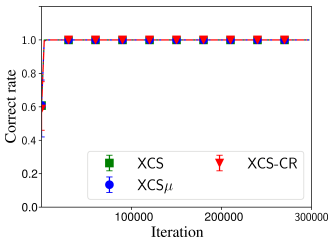
The magnitude of the action noise $\sigma_A$ is determined according to two cases.

- **Case A**: The action noise $\sigma_A$ with the uniform magnitude for all inputs is added to the rewards. The magnitude is set as $\sigma_A = 0, 0.1$, and $0.2$ in this case. Note that no noise is added when $\sigma_A = 0$.
- **Case B**: The magnitude of the $\sigma_A$ is randomly determined. The magnitude varies every trial, but does not change during the same trial. The magnitude is set as $\sup\{\sigma_A\} = 0.1, 0.2$ in this case. By comparing with the result of case A, we can see whether the learning performance of each method depends on the maximum value of the magnitude of the action noise $\sigma_A$ or on the average value of the magnitude of the action noise $\sigma_A$.

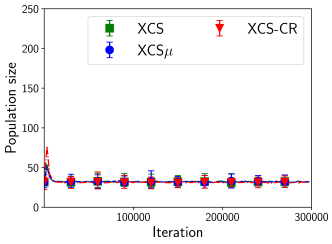
## 7.2 Evaluation criteria and parameters

The following evaluation criteria are employed in this experiment. All evaluations are averaged from 50 trials with different random seeds. Note that the performance and the population size are further averaged from 100 iterations as the moving average. Its window overlaps.

- **Performance**: This criterion evaluates the correct rate in the given problem, *i.e.*, the proportion of iterations in the moving average window on which the correct answer was given. A higher correct rate is better than a lower one. The term "performance" corresponds to the correct rate in this experiment.

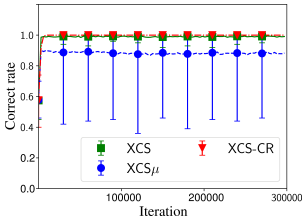



(a) Correct rate

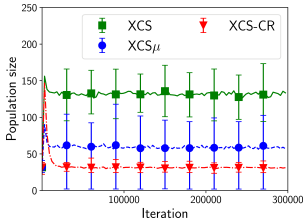



(b) Population size

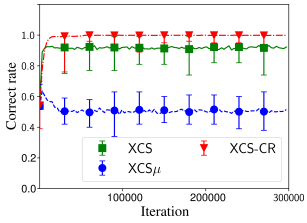
**Figure 6: Case A ($\sigma_A = 0$)**



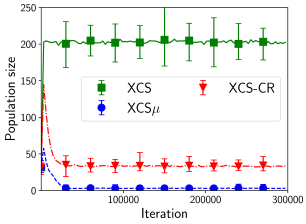

(a-1) Cor. rate ($\sigma_A = 0.1$)




(a-2) Pop. size ($\sigma_A = 0.1$)

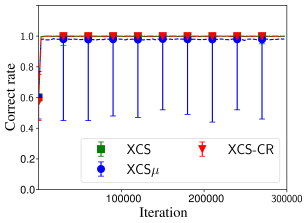



(b-1) Cor. rate ($\sigma_A = 0.2$)

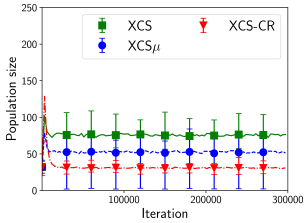



(b-2) Pop. size ($\sigma_A = 0.2$)

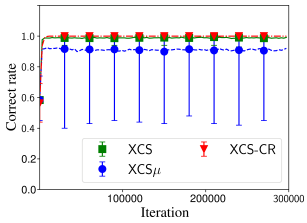
**Figure 7: Case A ($\sigma = 0.1, 0.2$)**



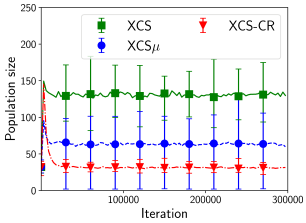

(a-1) Cor. rate ($\sup\{\sigma_A\} = 0.1$)




(a-2) Pop. size ($\sup\{\sigma_A\} = 0.1$)




(b-1) Cor. rate ($\sup\{\sigma_A\} = 0.2$)




(b-2) Pop. size ($\sup\{\sigma_A\} = 0.2$)

**Figure 8: Case B ($\sup\{\sigma_A\} = 0.1, 0.2$)**

- **Population size**: This criterion evaluates the number of the classifiers in $[P]$ in each iteration. The smaller population size is better than the larger one because the necessary memory size becomes small.
- **Total number of the acquired optimal subset** $[O]$: The 6-multiplexer problem has 16 optimal classifiers and we use two measures of how successfully they are learned. The first measure is the percentage of 50 trials in which all 16 optimal classifiers are in $[P]$ at the end of the trial. XCS-type algorithms tend to assign higher fitness to optimal classifiers than other classifiers in multiplexer problems. The second measure is the percentage of 50 trials in which the 16 optimal classifiers are the fittest 16 classifiers in $[P]$ at the end of the trial. We call the second measure "top 16".

The correct rate and the population size of the end of the learning are significantly verified by Wilcoxon signed rank test. The significance level is 1% in this paper.

XCS was implemented based on [2]. XCS$\mu$ and XCS-CR were implemented based on the XCS by adding the difference. For the parameters setting, the following values are employed in the experiments, which are the mostly standard ones in the conventional XCS [1]: For the common parameters of XCS, XCS$\mu$, and XCS-CR, $N = 400, \epsilon_0 = 10, \mu = 0.04, P_\# = 0.35, P_{explr} = 1.0, \chi = 0.8, \nu = 5, \theta_{GA} = 25, \theta_{del} = 20, \theta_{sub} = 20$. XCS-CR specific parameter $\theta_{RE}$ is set to 20. Each trial ran for 300,000 exploit steps.

## 7.3 Results

The results in the two case are shown in Figure 6, 7, and 8. The horizontal axis indicates the number of the iterations, while the vertical axis indicates the correct rate or the population size. The square mark, the circle mark, and the triangle mark represent XCS, XCS$\mu$, and XCS-CR respectively. The upper and the lower of the error bar indicate the "maximum" and "minimum" values of the correct rate or the population size in 50 trials, respectively.

Figure 6 shows the correct rate and the population size when the noise is not added to the rewards (Case A ($\sigma = 0$)). The correct rate of XCS, XCS$\mu$, and XCS-CR reached 1. There was no significant difference. The population size was about 30 and there was also no significant difference in XCS, XCS$\mu$, and XCS-CR. Table 1 shows

**Table 1: Case A ($\sigma = 0$)**

| Method | Acquisition $[O]$ | Acquisition $[O]$ (top 16) |
|---|---|---|
| XCS | 48 (96%) | 47 (94%) |
| XCS$\mu$ | 47 (94%) | 47 (94%) |
| XCS-CR | 50 (100%) | 50 (100%) |

**Table 2: Case A**

| Method | Acquisition $[O]$ | Acquisition $[O]$ (top 16) |
|---|---|---|
| XCS ($\sigma_A = 0.1$) | 17 (34%) | 0 (0%) |
| XCS$\mu$ ($\sigma_A = 0.1$) | 13 (26%) | 0 (0%) |
| XCS-CR ($\sigma_A = 0.1$) | 50 (100%) | 48 (96%) |
| XCS ($\sigma_A = 0.2$) | 2 (4%) | 0 (0%) |
| XCS$\mu$ ($\sigma_A = 0.2$) | 0 (0%) | 0 (0%) |
| XCS-CR ($\sigma_A = 0.2$) | 49 (98%) | 45 (90%) |

**Table 3: Case B**

| Method | Acquisition $[O]$ | Acquisition $[O]$ (top 16) |
|---|---|---|
| XCS ($\sup\{\sigma_A\} = 0.1$) | 42 (84%) | 0 (0%) |
| XCS$\mu$ ($\sup\{\sigma_A\} = 0.1$) | 35 (70%) | 2 (4%) |
| XCS-CR ($\sup\{\sigma_A\} = 0.1$) | 50 (100%) | 49 (98%) |
| XCS ($\sup\{\sigma_A\} = 0.2$) | 20 (40%) | 0 (0%) |
| XCS$\mu$ ($\sup\{\sigma_A\} = 0.2$) | 9 (18%) | 0 (0%) |
| XCS-CR ($\sup\{\sigma_A\} = 0.2$) | 50 (100%) | 50 (100%) |

the acquisition percentage of the optimal classifier subset $[O]$. XCS and XCS$\mu$ could not acquire $[O]$ in a few trials, while XCS-CR could acquire $[O]$ in the top 16 in $[P]$ for all trials.

Figure 7 shows the correct rate and the population size with the action noise $\sigma_A$ (Case A ($\sigma = 0.1$ and 0.2)). From this figure, the correct rate of XCS-CR was higher than the rate of XCS and XCS$\mu$ ($p < 0.01$). In three methods, the correct rates got worse as the magnitude of the action noise $\sigma_A$ increases. However, the correct rate of XCS-CR in most trials is 1. The population size is stable only in XCS-CR regardless of the magnitude of noise $\sigma_A$. The population size of XCS increased. As the magnitude of the noise $\sigma_A$ increased, the number of trials where the population size became 2 increased in XCS$\mu$. (In these trials XCS$\mu$ overgeneralizes and the population converges on two fully-general classifiers.) The population sizes of the three methods were significantly different from each other ($p < 0.01$). Table 2 shows the acquisition percentage of the optimal classifier subset $[O]$. XCS-CR could not acquire $[O]$ in the top 16 in $[P]$ in a few trials while XCS and XCS$\mu$ could not acquire $[O]$ in most trials.

Figure 8 shows the correct rate and the population size with the action noise $\sigma_A$ (Case B ($\sup\{\sigma_A\} = 0.1$ and 0.2)). From this figure, the correct rate of XCS-CR was higher than the rate of XCS and XCS$\mu$. There was no significant difference when the magnitude of the noise $\sigma_A$ was 0.1, but there was a significant difference when the magnitude of the noise $\sigma_A$ was 0.2 ($p < 0.01$). In XCS and XCS$\mu$, the correct rates got worse as the magnitude of the action noise $\sigma_A$ increases. However, the correct rate of XCS-CR reached to 1 in all trials. The population size is stable only in XCS-CR regardless of the magnitude of noise $\sigma_A$. The population size of XCS and XCS$\mu$ increased. The population sizes of the three methods

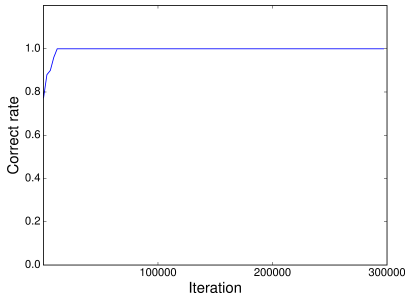



**Figure 9: XCS-CR performance with $\sigma_A = 0.2$ on a good trial**

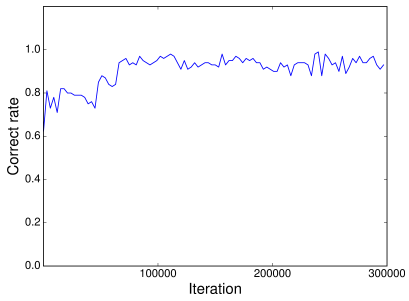



**Figure 10: XCS-CR performance with $\sigma_A = 0.2$ on a bad trial**

were significantly different from each other ($p < 0.01$). Table 3 shows the acquisition percentage of the optimal classifier subset $[O]$. XCS-CR could acquire $[O]$ in the top 16 in $[P]$ in most trials, while XCS and XCS$\mu$ could not acquire $[O]$ in most trials.

XCS-CR could generalize the classifiers properly even if the magnitude of the added noise $\sigma_A$ uneven by input and can acquire the optimal classifier subset $[O]$.

## 8 DISCUSSION

### 8.1 XCS-CR estimation of reward

XCS-CR acquired the optimal classifier subset $[O]$ in most trials. XCS-CR is more effective than XCS and XCS$\mu$ in this action noise environment. This section describes the estimated reward based on the collective reward (Equation (13)) which is the most important mechanism in XCS-CR.

In the initial stage of the learning where $\epsilon_0$ is set to the initial value, all classifiers are judged as inaccurate classifiers. Since the value of $\kappa$ increases as $\epsilon$ decreases from equation 6, the classifiers with less variance of the acquired reward are likely to remain in $[P]$. There are many classifiers that are not over-generalized in $[P]$. XCS-CR can estimate acquisition reward correctly for each action to equation 13. After the value of $\epsilon_0$ is stable, the accurate classifiers and inaccurate classifiers are correctly judged. Since the accurate classifiers remain in $[P]$ and the inaccurate classifiers are deleted, XCS-CR can estimate acquisition reward correctly for each action as in the initial stage of the learning.

Figures 9 and 10 each show a single trial (random seed) of XCS-CR in Case A ($\sigma_A = 0.2$). In Figure 9, the correct rate was reached to 1 from the early stage of learning and it was stable. In the latter stage of the learning, since $[P]$ was consisted only of the optimal

classifier subset [$O$] and newly created classifiers by GA, the estimation of acquisition reward was stable. In contrast, Figure 10 shows a trial where the correct rate did not reach to 1. One of the cause is that overgeneralized classifiers (*e.g.* the condition part is "#00###") were judged as accurate. The overgeneralized classifiers were generated at the beginning of learning, were judged as accurate at the time the estimation of acquisition reward mechanism was unstable and satisfied the subsumption condition of XCS-CR. The overgeneralized classifiers subsumed the accurate classifiers (*e.g.* the condition part is "000###") and the overgeneralized classifiers became the classifier with the highest experience at the input to be matched. Since the classifier with a large number of experience is early judged as accurate from Equation (13), once it is judged to be accurate, even an overgeneralized classifier can easily remain in [$P$].

### 8.2 Relationship between learning performance and magnitude of the noise $\sigma_A$

Comparing Case A and Case B, the learning performance (the performance, the population size and the acquisition percentage of the optimal classifier subset [$O$]) of Case A ($\sigma_A = 0.1$) and Case B ($\sup\{\sigma_A\} = 0.2$) were similar. In Case B, since the uniform noise is applied to the magnitude of the noise $\sigma_A$, the average of the magnitude of the noise $\sigma_A$ was 0.1. The learning performance of XCS, XCS$\mu$, and XCS-CR are strongly influenced by the average value of the magnitude of the noise $\sigma_A$ rather than the maximum value of the magnitude of the noise $\sigma_A$.

### 8.3 Learning performance of XCS$\mu$

From Figures 7 and 8, XCS$\mu$ can generalize classifiers better than XCS, but there are trials in XCS$\mu$ where classifiers are overgeneralized (*e.g.* the condition part is "######"). The cause of the overgeneralization is the smallness of parameter $\theta_{sub}$. This paper set $\theta_{sub}$ to a standard value of 20 in the 6-multiplexer problem. XCS$\mu$ uses the minimum prediction error $\overline{\mu}$ for calculating prediction error $\epsilon$. It is probably enough that the acquired rewards were biased to one side even if the classifier is inaccurate until the classifier evaluated $\theta_{sub}$ times after being generated. The overgeneralized classifier was judged as accurate and subsumed all other classifiers. The "######" condition part classifiers were generated for each action and the population size was two, because there are two possible actions.

## 9 CONCLUSIONS

This paper proposes a new XCS, XCS-CR (XCS based on Collective Reward) that can acquire the optimal classifier subset [$O$] in noisy action environments.

XCS-CR does not judge the accuracy of the classifier directly from the reward acquired by the classifier. XCS-CR judges the accuracy of the classifier based on the estimated value of the acquired reward that is determined by the collective reward. The collective reward value is calculated, for each action, from the classifiers that match the current input. The estimated reward value is determined from the magnitude relation of the collective rewards. XCS-CR can accurately judge the accuracy of the classifier without being affected by the bias of acquisition reward due to a small number of evaluations by using the estimated reward. This paper shows that XCS-CR can generalized the classifier properly even if the magnitude of the added noise $\sigma_A$ is unknown and uneven by input and can acquire the optimal classifier subset [$O$] through the experiments. However, as shown in Table 2, XCS-CR failed to acquire the optimal classifier subset [$O$] in some trials.

XCS-CR has the parameter $\theta_{RE}$ related to determining of accuracy criterion $\epsilon_0$ and subsumption condition. If $\theta_{RE}$ is too small, some inaccurate classifiers are judged as accurate, and conversely if $\theta_{RE}$ is too large, rule generalization does not occur. As the number of # symbol included in the condition part of the optimum classifier increases, it is necessary to be careful with setting $\theta_{RE}$.

What should be noted here is that the above results have only been shown from the noisy 6-multiplexer problem. Such important directions must be pursued in the near future in addition to the following future research: (1) improvement of the estimation acquiring reward mechanism; (2) adaptation to the multi-class classification problem; and (3) adaptation to environments with vast solution space with alternative reward noise.

## ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Number JP17J03593.

## REFERENCES

[1] M. V. Butz, T. Kovacs, P. L. Lanzi, and S. W. Wilson. 2004. Toward a Theory of Generalization and Learning in XCS. *Evolutionary Computation, IEEE Transactions on* 8, 1 (2004), 28–46.
[2] M. V. Butz and S. W. Wilson. 2002. An algorithmic description of XCS. *Soft Computing* 6, 3-4 (2002), 144–153.
[3] D. E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning* (1st ed.). Addison-Wesley Longman Publishing Co., Inc.
[4] J. H. Holland. 1986. Escaping Brittleness: The Possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems. *Machine learning* (1986), 593–623.
[5] P. L. Lanzi. 1999. An Analysis of Generalization in the XCS Classifier System. *Evolutionary Computation Journal* 7, 2 (1999), 125–149.
[6] P. L. Lanzi and M. Colombetti. 1999. An Extension to the XCS Classifier System for Stochastic Environments. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*. 353–360.
[7] P. L. Lanzi and S. W. Wilson. 2000. Toward Optimal Classifier System Performance in Non-Markov Environments. *Evol. Comput.* 8, 4 (Dec. 2000), 393–418.
[8] R. S. Sutton. 1988. Learning to Predict by the Methods of Temporal Differences. *Machine Learning* 3, 1 (1988), 9–44.
[9] H. Takagi. 2001. Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. *Proc. IEEE* 89, 9 (2001), 1275–1296.
[10] T. Tatsumi, T. Komine, M. Nakata, H. Sato, T. Kovacs, and K. Takadama. 2016. Variance-based Learning Classifier System without Convergence of Reward Estimation. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion (GECCO '16 Companion)*. ACM, 67–68.
[11] A. Webb, E. Hart, P. Ross, and A. Lawson. 2003. *Controlling a Simulated Khepera with an XCS Classifier System with Memory.* Springer Berlin Heidelberg, Berlin, Heidelberg, 885–892.
[12] S. W. Wilson. 1995. Classifier Fitness Based on Accuracy. *Evol. Comput.* 3, 2 (June 1995), 149–175.
[13] S. W. Wilson. 2000. *Get Real! XCS with Continuous-Valued Inputs.* Springer Berlin Heidelberg, 209–219.