Adopting Lexicase Selection for Michigan-Style Learning Classifier Systems with Continuous-Valued Inputs

Alexander R. M. Wagner
Dept. of Artificial Intelligence in Agricultural Engineering
& Computational Science Lab (CSL)
University of Hohenheim
Stuttgart, Germany
a.wagner@uni-hohenheim.de

Anthony Stein
Dept. of Artificial Intelligence in Agricultural Engineering
& Computational Science Lab (CSL)
University of Hohenheim
Stuttgart, Germany
anthony.stein@uni-hohenheim.de

ABSTRACT

In this work, two variants of lexicase selection are examined for their performance improvement potential in the context of XCS variants allowing for continuous-valued inputs. We furthermore propose a niche-specific mode of operation for lexicase selection when adopted for utilization in XCS, implemented by a dedicated *classifier experience storage*. To evaluate the impact of lexicase selection on XCS' classification and regression capabilities, our proposed modified variants are tested across various continuous-valued single-step tasks, including both typical toy problems and real-world datasets related to the agricultural domain as well as regression problems.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning approaches**;

KEYWORDS

LCS, XCS Classifier System, Parent Selection, Lexicase Selection

ACM Reference Format:
Alexander R. M. Wagner and Anthony Stein. 2021. Adopting Lexicase Selection for Michigan-Style Learning Classifier Systems with Continuous-Valued Inputs. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459533

1 MOTIVATION

The *XCS classifier system* (XCS) suffers from the problem of over-generalization, which is exacerbated in certain tasks, cf. [5]. To be appropriately solved, these tasks would demand a rule specialization pressure, the absence of which impedes the formation of rules with the required specialization level to capture small environmental niches. A crucial factor for evolving sufficiently specialized rules is the parent selection method used in the genetic algorithm.

A recent parent selection approach proposed by Helmuth et al. in [2] is *lexicase selection*. It resulted in more diverse populations and in improved problem-solving capabilities in *genetic programming* due to favoring the selection of *specialist individuals*. These specialists represent individuals capable of correctly predicting situations in which other individuals performing indeed superior in the error-aggregating fitness function would nevertheless fail. In the *sUpervised Classifier System*, Aenugu and Spector demonstrated an improved overall learning performance in terms of an increased classification accuracy and generalization capability due to lexicase selection [1]. Therefore, in our work, we build on their findings and investigate the effects of lexicase selection on XCS for continuous-valued inputs, XCSR [7] and XCSF [6], with the goal to realize an increased specialization pressure. Our overarching goal is to reach a more reliable specialization pressure in XCS to prevent detrimental effects due to over-generalization, as reported, e.g., in [5].

We propose adapted versions of two suitable lexicase selection variants in XCSR and XCSF, i.e., *batch-lexicase selection* (B-Lex) [1] in XCSR and ϵ*-lexicase selection* (ϵ-Lex) [3] in XCSF. Furthermore, we introduce a new niche-specific mode of operation for lexicase selection. To highlight the impact of B-Lex and ϵ-Lex on XCS' classification or regression capabilities, our proposed modified variants are tested across various continuous-valued single-step tasks, involving both typical toy problems and real-world datasets related to the agricultural domain as well as regression problems.

2 LEXICASE SELECTION IN XCSR AND XCSF

In this work, B-Lex and ϵ-Lex are adopted for utilization in XCSR and XCSF, respectively. The test cases used in lexicase selection resemble input/output-pairs and consist of previously experienced situations $\vec{x} \in \mathbb{R}^d$ and the associated target value, i.e., an action $a \in A$ or a payoff $p \in P$. The test cases, also referred to as experience, are stored in an *experience storage* (ES).

B-Lex and ϵ-Lex perform a non-elitist selection due to relaxing the strictness of the pass condition. Based on [1], B-Lex requires only minor modifications to be applicable in XCSR. B-Lex utilizes batches of test cases, which are defined as $tc := (\vec{x}, a)$ and are randomly drawn from the ES. The classifiers in the action set form the initial set of classifier candidates $[Cand]$, which are iteratively filtered using test case batches in the subsequent process. In addition to the implementation in [1], in our implementation, a classifier cl_i must also match at least one test case tc to be selected either for a further filter iteration or finally as a parent. In case no cl_i survived the current filter iteration, this filter iteration and the associated test case batch are discarded and a new filter iteration is initiated. We also modified B-Lex to perform *roulette-wheel selection* (RWheel) on the current $[Cand]$ once all stored test cases are exhausted.

Since ϵ-Lex [3] has not been applied to an LCS before, we propose an adapted implementation for XCSF. In ϵ-Lex, a test case tc is defined as $tc := (\vec{x}, p)$. The selection of a classifier cl_i is based on the absolute deviation of the classifier prediction $cl_i.p$ to payoff

$tc.p$ of the currently considered test case tc. Subsequent to the random selection of a test case tc, a prefiltered set $[Pre]$ is generated, containing $cl_i \in [Cand]$ satisfying $tc.\vec{x} \in cl_i.C$. In case $|[Pre]| < 2$, the current tc is discarded and a new iteration is initiated. Otherwise, for each $cl_i \in [Pre]$, the absolute error $|tc.p - cl_i.p|$ with respect to tc is determined, denoted as $e_{tc}(cl_i)$. Vector $\vec{e}_{tc}$ comprises each error $e_{tc}(cl_i)$ for tc. Next, two values are determined: (1) e^*_{tc}, the smallest or *best error* achieved for tc. (2) The *median absolute deviation* $\lambda(\vec{e}_{tc})$ of $\vec{e}_{tc}$. The filtered set $[F]$ is formed by selecting each $cl_i \in [Pre]$ satisfying lexicase selection's ϵ pass condition $e_{tc}(cl_i) < e^*_{tc} + \lambda(\vec{e}_{tc})$. The $cl_i \in [F]$ form $[Cand]$ for the next filter iteration. In case a single cl_i survived, it is returned as parent. Analogous to B-Lex, in case no cl_i survived, the current filter iteration and the associated tc are discarded and a new filter iteration is initiated. Also, RWheel is applied on the last $[Cand]$ once all stored test cases are exhausted.

Moreover, we propose a dedicated *classifier experience storage* (CES) introducing a local storage $cl_i.es$ within the cl_i, enabling a niche-specific mode of operation for lexicase selection. In each iteration of XCSR or XCSF, a new test case is added to $cl_i.es$ for each $cl_i \in [M]$. For parent selection using one of the lexicase selection variants, $[Cases] := \bigcup_{cl_i \in [A]} cl_i.es$, representing a duplicate-free set. This assures that at least one cl_i matches the current tc. CES provides more suitable test cases for selection, avoiding the elimination of every $cl_i \in [A]$ as potential parent. In this work, CES is applied in both B-Lex and ϵ-Lex.

3 EVALUATION

Table 1: The table shows the min/median/max deviations of lexicase selection compared to tournament selection w. r. t. the mean values obtained in the different evaluation experiment series. Positive/negative values represent increases/reductions of a metric.

XCSR on Toy Problems (B-Lex)	Min	Median	Max
Reward / Accuracy	+1.73%	+9.94%	+30.05%
System Error	-7.90%	-20.91%	-35.83%
Population Size	+1.30%	+5.64%	+11.43%
XCSR on Real World Data (B-Lex)	Min	Median	Max
Reward / Accuracy	+0.13%	+0.72%	+7.71%
System Error	-4.28%	-7.73%	-20.20%
Population Size	+0.49%	+21.82%	+45.67%
XCSF for Regression (ϵ-Lex)	Min	Median	Max
System Error	-5.34%	-13.21%	-28.40%
Population Size	+11.85%	+13.45%	+19.36%

We demonstrate the effect of the adapted lexicase selection variants on XCS for continuous-valued problem domains by conducting a series of experiments for B-Lex used in XCSR on classification tasks and for ϵ-Lex used in XCSF on regression tasks. The classification tasks on the one hand consist of four well-known toy problems, i.e., the *Real k-multiplexer problem* [7] with 11 dimensions, the *Checkerboard problems* [7] CBP(3,6) and CBP(3,8) with 6 and 8 divisions in each of the 3 input dimensions, respectively, and the *Mario* pixel art, as introduced by Stein et al. [4]. On the other hand, they include four available real-world datasets related to several aspects in the agricultural domain, i.e., the *Horse Colic*, the *Iris* and the *Soybean Disease* datasets from the UCI repository as well as the *Paddy Leaf* dataset from Kaggle. The regression tasks consist of four test functions, already used for evaluation of XCSF in [6], i.e.,

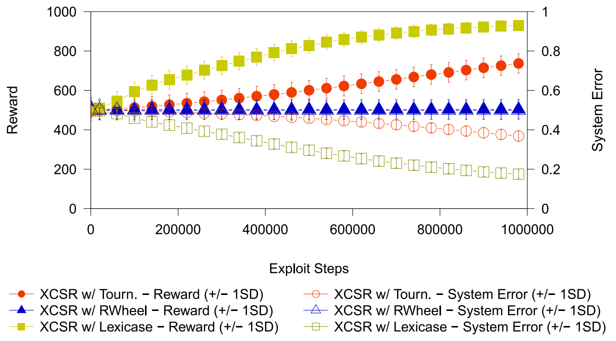

Figure 1: Exemplary results of XCSR in CBP(3,8).

the *Eggholder* function, the *Sine-in-Sine* function, the 3-dimensional *Cross* function and the 5-dimensional *Styblinski-Tang*.

As shown in Table 1, the use of B-Lex results in improved accuracy and error metrics in the studied toy problems and real-world datasets. Figure 1 shows exemplary learning curve plots for CBP(3,8), demonstrating a remarkable performance improvement in this task by B-Lex compared to XCSR with tournament selection and XCSR with RWheel. In the regression tasks, significant reductions in system error level are achieved when using ϵ-Lex. Both B-Lex and ϵ-Lex cause an increase in population size, presumably due to an increased selection of more specialized classifiers. These classifiers are less able to subsume child classifiers, possibly leading to increased formation of transient rules, an aspect we will more thoroughly examine in the future.

4 CONCLUSION

We demonstrated the application of the recent parent selection technique *lexicase selection* in XCS for continuous-valued problem domains by adapting two variants, namely *batch-lexicase selection* (B-Lex) and *ϵ-lexicase selection* (ϵ-Lex). In addition, we proposed a dedicated *classifier experience storage* for the adoption of lexicase selection, providing a niche-specific mode of operation. We evaluated B-Lex in XCSR for classification tasks comprising toy problems and real-world datasets. ϵ-Lex was assessed in XCSF for regression tasks based on well-known benchmark functions from the domain of global optimization. The reported results of the conducted empirical studies revealed the application of B-Lex and ϵ-Lex results in significant improvements regarding the overall learning performance on the investigated problems.

REFERENCES

[1] S. Aenugu and L. Spector. 2019. Lexicase Selection in Learning Classifier Systems. In *Proc. of GECCO '19*. ACM, New York, NY, USA, 356–364.
[2] T. Helmuth, L. Spector, and J. Matheson. 2015. Solving Uncompromising Problems With Lexicase Selection. *IEEE T Evolut Comput* 19, 5 (2015), 630–643.
[3] W. La Cava, L. Spector, and K. Danai. 2016. ϵ-Lexicase Selection for Regression. In *Proc. of GECCO '16*. ACM, New York, NY, USA, 741–748.
[4] A. Stein, R. Maier, and J. Hähner. 2017. Toward Curious Learning Classifier Systems: Combining XCS with Active Learning Concepts. In *Proc. of GECCO '17 Companion*. ACM, New York, NY, USA, 1349–1356.
[5] A. Stein, R. Maier, L. Rosenbauer, and J. Hähner. 2020. XCS Classifier System with Experience Replay. In *Proc. of GECCO '20*. ACM, New York, NY, USA, 404–413.
[6] A. Stein, S. Menssen, and J. Hähner. 2018. What about Interpolation? A Radial Basis Function Approach to Classifier Prediction Modeling in XCSF. In *Proc. of GECCO '18*. ACM, New York, NY, USA, 537–544.
[7] C. Stone and L. Bull. 2003. For Real! XCS with Continuous-Valued Inputs. *Evol Comput* 11, 3 (Sep. 2003), 299–336.