An Overview of LCS Research from 2020 to 2021

David Pätzel
University of Augsburg, Germany
david.paetzel@uni-a.de

Michael Heider
University of Augsburg, Germany
michael.heider@uni-a.de

Alexander R. M. Wagner
University of Hohenheim, Germany
a.wagner@uni-hohenheim.de

ABSTRACT

The *International Workshop on Learning Classifier Systems* (IWLCS) is an annual workshop at the GECCO conference where new concepts and results regarding *learning classifier systems* (LCSs) are presented and discussed. One recurring part of the workshop agenda is a presentation that reviews and summarizes the advances made in the field over the last year; this is intended to provide an easy entry point to the most recent progress and achievements. The 2020 presentation was accompanied by a survey workshop paper, a practice which we hereby continue. We give an overview of all the LCS-related publications from 11 March 2020 to 10 March 2021. The 46 publications we review are grouped into seven overall topics: Formal theoretic advances, contributions to LCS-based multi-agent reinforcement learning, approaches to setting and adapting LCS hyperparameters, new LCS architectures and adaptations, LCS implementations, improvements to existing LCSs and applications of LCSs.

CCS CONCEPTS

• **Computing methodologies** → **Rule learning**; **Genetic algorithms**; • **General and reference** → *Surveys and overviews*;

KEYWORDS

Learning Classifier Systems, Survey

ACM Reference Format:
David Pätzel, Michael Heider, and Alexander R. M. Wagner. 2021. An Overview of LCS Research from 2020 to 2021. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3449726.3463173

1 INTRODUCTION

Since 2019, a recurring segment of the *International Workshop on Learning Classifier Systems* (IWLCS) is its organizers giving a presentation of an exhaustive overview of *learning classifier system* (LCS) research that was conducted over the past year. The 2020 presentation was accompanied by a survey workshop paper [31], a practice which we hereby continue. Our primary goal is to contribute to a better organized research community; showcasing the most recent developments of the field helps to connect LCS researchers but also serves people new to the field as they can more quickly assess the latest achievements, current challenges as well as links to other areas.

In the next section, we shortly present the methodology we used for identifying publications to be included in our overview. The remaining sections correspond to the seven major topics that the 46 contributions we found could be divided into and a concluding summary. We mention each publication only once, even if some fit into several sections; for each paper we choose the first section it fits into, in the order in which the sections appear.

2 METHODOLOGY

This survey is limited to contributions in English with *publication dates* on or after 11 March 2020 (one day after the end of the period of the previous survey) and before 10 March 2021. Note that some conferences may have published their proceedings within this interval despite in fact taking place at another date; since we only consider publication dates, these entries are covered by this survey as well. We intentionally *do not include military applications* of LCSs (of which we actually found one this year). Also, we leave out papers published on *arXiv* unless they are relevant for a follow-up paper which meets our criteria. Finally, we exclude publications that were already cited in the previous year's survey [31] even if they were eligible for the present survey as well (this may be the case, for example, if they have been re-published or moved from a 'pre-proof' to a 'published' status).

Our primary search tool was *Google Scholar*[1] with its time range feature as it seemed to consistently report more relevant publications than comparable tools, especially for more general search queries. We ultimately only used five different queries, two very general ones and one for each of the three major LCSs that we knew have been investigated in recent years; more specific terms did not yield relevant results that were not found by these as well (at least not in March 2021). The following lists each query with the number of Google Scholar result pages that we examined for papers meeting our requirements (numerator) and the total number of result pages (denominator); each Google Scholar result page displayed 10 publications and we set the time range filter to *Since 2020*.

- *learning "classifier system"* (34/>100)[2]
- *"evolutionary rule-based" learning* (3/3)
- *xcs classifier system* (14/14)
- *"extracs" classifier system* (1/1)
- *biohel* (2/2)

We also checked a number of LCS researchers (especially, but not exclusively, ones we know to be active) and the proceedings of the

[1] https://scholar.google.com
[2] We stopped when we noticed that the results of at least ten consecutive pages did not contain any LCS-related publications that had not yet been listed on the preceding pages (i. e. we estimated the probability of further relevant results to be very low).

three main venues that have attracted LCS researchers in the past: CEC, GECCO and *Evostar*. As we did not find any publications this way that we had not already found using Google Scholar, we are confident that our Google Scholar search was sufficiently exhaustive and there was no need to further refine the used search terms.

3 FORMAL THEORY

We only found a single contribution performing formal theoretic work regarding LCSs. Just like in last year's survey [31], we want to stress that although research effort in this direction has decreased over the years [30], there is a dire need for analysing LCSs more formally. Only with formal analysis will we be able to conclusively ascertain which classes of problems they are best suited for, which of the many design decisions were actually reasonable and, in the end, improve their performance substantially. One example for a step in that direction is the following publication.

Nakata and Browne [24] investigate more in-depth their previous formal results on setting XCS's hyperparameters β (learning rate), ϵ_0 (error tolerance) and θ_{sub} (subsumption threshold) optimally for binary classification problems. They discuss the issues arising from applying one of their central assumptions, namely, that a rule's true accuracy value can be determined at every iteration from previous experiences—which is not true in general due to sampling usually having to be considered random. However, they are able to derive an adjustment to their framework which takes this problem into account. They compare the theoretically derived ideal parameter settings (both with and without the adjustment) with commonly-used XCS parameter settings on a range of comparatively large-scale synthetic problems (70- and 135-bit binary multiplexers, the real-valued 37-bit multiplexer, a design verification toy problem as well as concatenated and aliased multiplexers). Configurations using the formally derived hyperparameter values perform best in all scenarios but the design verification problem where the (unadjusted) optimal configuration actually performs worst; however, the latter can be traced back to the above-mentioned problematic assumption and the adjusted optimal settings perform best by far.

4 MULTI-AGENT REINFORCEMENT LEARNING

This section showcases the two publications from last year that explicitly investigate general approaches to multi-agent reinforcement learning settings.

Chen et al. [7] present an XCS extension, X-OMQ(λ), for learning deterministic policies for zero-sum Markov Games (ZSMG) with fully competitive tasks and alternating turns. To speed up training, an eligibility trace mechanism is used: encountered states and actions are recorded and, instead of merely updating the last action set, X-OMQ(λ) also updates all classifiers matching that historical data. The authors provide a comprehensive study of their algorithm and compare it to several other state-of-the-art approaches that are typically used for solving ZSMG, that is, six Q-learning–based and five neural network–based ones. The tasks investigated are Littman's soccer, hunter prey and Hexcer with the respective RL-agents competing against random opponents or, in the case of Hexcer, both random opponents as well as an HAMMQ-based agent. From their detailed results, the authors conclude that X-OMQ(λ) performs similarly to the other approaches. However, they also stress the greater explainability over neural network–based algorithms as well as the possibility to transfer rules to similar scenarios and point out that the approach in its current form comes short of solving image-based or continuous action tasks.

To improve the capabilities of XCS in the context of other competitive multi-agent RL problems, Chen et al. [6] further introduce Bayes-XCS, a variant of XCS extended in several aspects:

- A library for the policies of each opponent which is used to assess offline the agent's available response policies with respect to the policies in the opponents' policy library by forming performance models using Bayesian policy reuse.
- The use of neural network–based opponent modelling from behavioural data collected during interaction.
- An online policy reuse part in which a response policy is selected based on multiple factores (e. g. opponent models and behaviours) during online interactions.

The evaluation in a soccer game on a discrete grid revealed that Bayes-XCS outperforms both examined state-of-the-art multi-agent RL algorithms (namely, Bayes-Pepper and Bayes-ToMoP) as Bayes-XCS is capable of accurately and efficiently detecting the opponent policy in this game and, thus, reuse the best response policy.

5 SETTING AND ADAPTING HYPERPARAMETERS

Evolutionary ML techniques often have a significant number of hyperparameters. In order to achieve optimal learning performance, some of these need to be configured carefully—which is not always possible in advance, especially in real-world problems. Also, having sensitive hyperparameters exposed reduces the overall usability of a method as non-expert users may be discouraged by default configurations not working that well or the prospect of having to perform more or less complicated analysis (e. g. hyperparameter studies) beforehand. This section presents four recent approaches to automatedly adjusting some of the more sensitive hyperparameters of XCS, XCSF and BioHEL, respectively.

Horiuchi and Nakata [15] build on the formal theoretic work of Nakata and Browne [24] (see Section 3) who derived optimal values for the XCS hyperparameters β (learning rate), ϵ_0 (error tolerance) and θ_{sub} (subsumption threshold) for Boolean classification problems. As stated earlier, the formal framework to compute those values relies on knowledge of an accuracy threshold value that correctly splits rules into being accurate or inaccurate; while this value can be determined for some synthetic problems, it is not known in general. Horiuchi and Nakata now formulate a scheme to estimate that value during learning which enables an adaptation mechanism for β, ϵ_0 and θ_{sub} that can be used on Boolean classification problems. They test their mechanism on several well-known Boolean problems and are able to show that the resulting XCS derivative is able to solve problems that XCS with commonly-used parameter settings cannot solve (e. g. the 11-bit class-imbalanced multiplexer, carry problems and majority-on). On very simple problems (e. g. the 6-bit multiplexer), however, their approach is inferior to the original XCS due to the optimal values not having much impact on performance and the additional time that the adaptation mechanism requires to converge.

Hansmeier et al. [13] aim at reducing the number of sensitive hyperparameters of XCSF and improving the system's performance in dynamic environments (i. e. cases where the function learned by XCSF changes over time). They introduce an adaptation mechanism for optimizing one of XCSF's most important hyperparameters: the highly problem dependent error tolerance ϵ_0, that is, the target bound on the approximation error. Different to the aforementioned approach by Horiuchi and Nakata [15] for binary classification with XCS, Hansmeier et al. rely on heuristics. The proposed online updates are based on the approximation error achieved on recent examples. The authors evaluate their approach on two dynamic test functions: the RMS function (which is suddenly scaled by a factor of four) and the oblique sine function (whose frequency is suddenly changed). Based on this analysis, the authors show that, in dynamic scenarios, their adaptive error threshold can be superior to static error thresholds and that it may, although the problem to be learned changes considerably, even perform similar to a close-to-optimal static threshold after a short adaptation time. In [12], the authors then extend their analysis to four additional scenarios and perform a parameter study regarding the hyperparameters of their adaptation updates. They conclude that even with suboptimal hyperparameters, the adaptation mechanism is "able to maintain an approximation error (...) close to what is achieved with well-suited static thresholds". Also, on the problems considered, performance is comparable to configurations with optimal static error thresholds (optimality determined empirically; the new approach led in the worst case only to a 11.5% higher approximation error) and significantly better than less suited error thresholds.

Like XCS(F), BioHEL has quite sensitive hyperparameters. Franco et al. [10] present an analysis method for binary problems with which the coverage breakpoint—one of BioHEL's most critical and sensitive hyperparameters—can be set at runtime without expensive preliminary testing. The employed heuristic combines observations from data and a sample of randomly initialised rules to group problems according to the structure of their k-DNF form. Besides a reduction in runtime, this approach benefits usability as users tend to opt for default configurations rather than performing expensive hyperparameter studies. The authors validate their approach on synthetic problems with and without noise as well as on a noisy real world binary protein structure prediction problem (i. e. the *contact number prediction*). For the synthetic data, the heuristic achieves sub-par results in cases of highly overlapping rules and for larger k with either too few or too many terms but is generally able to optimize the coverage breakpoint satisfactorily. For the contact number prediction, the highest accuracy models found by traditional hyperparameter search and the new approach were identical; however, computation time was reduced from 285 hours to 81 hours—a reduction of 71%. Based on this, the authors propose the heuristic be used for other LCSs like GAssist and XCS as well.

6 NEW LCS ARCHITECTURES AND ADAPTATIONS

We found seven contributions proposing some kind of new LCS architecture or extensions of existing LCSs to entirely new kinds of problems.

Liu et al. develop a new LCS for Boolean supervised learning problems which they name *Absumption and Subsumption based Learning Classifier System* (ASCS) [20]. Instead of attempting to learn what Butz and Kovacs termed the optimal solution [5], ASCS tries to compile what the authors call a *natural solution*, that is, the set containing all the *correct* and *unsubsumable* rules. Other than for optimal solutions, where there, in general, is more than one for a certain problem, there is exactly one natural solution and this solution provides human-understandable insights into the problem's properties. Learning natural solutions requires a more deterministic process than most currently-used LCSs have; ASCS thus only relies on absumption (removes over-general rules), subsumption (removes over-specific rules), informed mutation and compaction—this also results in less hyperparameters than, for example, XCS-based systems. The authors evaluate their system on a range of Boolean problems (multiplexers, carry, majority-on) of different sizes, showing that ASCS is capable of solving problems that XCS and UCS cannot and is more efficient in terms of computing time. On the problems considered, other than XCS and UCS, ASCS consistently finds the natural solution which can be directly translated into human-readable diagrams. However, further work is needed to adapt ASCS to real-world, noisy domains.

In their extended abstract, Verma et al. [46] report on the preliminary results of a *genetic programming* (GP) tree–based LCS called GP-LCS derived from ExSTraCS. Their project's goal is to develop a more flexible framework that allows for different knowledge representations (in particular, GP trees *and* rules) to coexist in an LCS population and automatedly chooses a well-suited mix of these for the problem at hand. As a first study, they compare GP-LCS to several GP-based learning methods on six regression and two classification datasets. They conclude that GP-LCS seems to be a competitive approach; one reason of which is conjectured to be the inherent ensemble nature of ExSTraCS. For future work, they plan to develop a well-behaving fitness function which allows for rules and GP trees to coevolve in a single population and then reintegrate ExSTraCS rules into their system.

An extension of UCS to multi-label classification tasks is presented by Nazmi et al. [25]. They investigate two fitness measures suitable in this context as well as two different ways for computing predictions. In a comparison with other multi-label classification methods, their system's average rank is better or competitive on most metrics; findings are supported by a sound statistical analysis.

With the *Supervised Rule-based Learning System* (SupRB-1), Heider et al. [14] present a new Pittsburgh-style LCS for supervised learning of multi-dimensional continuous decision problems. The system batch-learns an approximation of a continuous quality function and is then capable of predicting the quality of all possible choices for a given situation. In order to find an optimal choice for a given situation (the primary use case is finding optimal parametrizations of industrial machinery), analytical optimization is used. This aims at providing an ML-based simulation in contexts where traditional direct optimization on the real-world problem is too expensive due to high costs of individual function evaluations and actual physics-based simulations are unavailable or too inaccurate but data is gathered automatically with every production cycle. They propose usage in an additive manufacturing context and introduce an adjustable Gaussian mixture problem, which simulates

similarly complex fitness landscapes but allows to judge SupRB-1's optimization capabilities.

The work of Siddique et al. [39] builds on the observation that vertebrate brains are able to transfer knowledge attained on simple, small-scale problems to more complex, larger-scale tasks—which is an often-sought property of ML methods. In biology, this capability is often attributed to the lateralization and modularity of the brain and the resulting ability to consider inputs at different levels of abstraction. This inspired the authors to investigate a lateralized approach to ML; they use LCSs to do so, since these are suited due to their inherent niche-based nature. They discuss and evaluate three classes of problems:

- Boolean problems (parity and hierarchical multiplexer; serving as a proof of concept),
- navigation/RL (non-Markov maze environments), where they observe that the problem of aliasing states can be dealt with by the lateralization, and
- computer vision (cat vs. dog classification).

The merits of the approach for computer vision scenarios are elaborated on more in-depth in another publication by the same authors [38]: They are able to show that lateralization increases robustness against adversarial attacks by performing classification as follows: In the so-called *context phase*, multiple *deep neural networks* DNNs generate predictions either for the constituent level (i. e. the individual parts) or the holistic level (i. e. the big picture) of the given image. In case the predictions at the constituent level and the holistic level diverge in estimation, the system is in doubt about correctly classifying the given image and invokes the so-called *attention phase*. In this phase, multiple LCSs are used to generate either constituent-level predictions or holistic-level predictions based on different types of features computed for segmented versions of the given image. Finally, all predictions from both phases are analysed to determine the final prediction vote. The experimental results demonstrate that the lateralized system successfully exhibits robustness against adversarial attacks and outperforms state-of-the-art DNNs (such as ResNet and VGG) in classifying normal and adversarial images.

Another challenging task in machine vision is the classification of underwater images due to specific factors that affect visibility (e. g. low illumination and high noise). Motivated by this challenge, Irfan et al. [16] present a lifelong learning ML model, named CAXCF-LL, based on a *convolutional autoencoder* (CAE) and an XCSR variant called *XCSR with code fragment–based lifelong learning* (XCF-LL). The encoder of the CAE is used to reduce the dimensionality of the input images to an appropriate level for XCF-LL, which classifies the images based on the extracted features. After the learning phase, XCF-LL accumulates the learned knowledge through continuous learning as it stores beneficial knowledge from learned data sets in a *knowledge base* (KB) by extracting useful code fragments of all experienced and accurate rules present in the final population. The learnt knowledge stored in the KB is utilized to solve future problems in the same or related domains. In evaluation experiments comprising two underwater data sets and one standard data set, CAXCF-LL is compared to an implementation of CAXCF-LL without lifelong learning (i. e. reuse of previously learned knowledge) and state-of-the-art CNN algorithms (such as DenseNet and Xception). The obtained results reveal CAXCF-LL achieves superior accuracy in almost all the studied situations and learns the problems faster due to the reuse of knowledge.

7 NEW AND UP-TO-DATE LCS IMPLEMENTATIONS

Discussions among researchers at recent IWLCS workshops pointed towards the lack of an easy-to-use software as a contributing factor to the unfamiliarity to and poor adoption of LCS research within the broader field of artificial intelligence and ML. This led to Zhang and Urbanowicz [50] introducing *scikit-eLCS*, an implementation of a UCS derivative called *eLCS*. Scikit-eLCS is compatible with the well-known and widely used Python ML library *scikit-learn* which facilitates the application of LCS for a broader audience. As indicated by the evaluation results, scikit-eLCS is an easy-to-use yet effective supervised LCS and a potential starting point for LCS developement and research in Python.

Based on the discussion mentioned in the previous paragraph, we want to use this survey to also provide readers with a list of links to open source implementations of LCSs that 1) have received updates in the past two years and 2) are general enough to be applied to new data without major rewrites (i. e. are not isolated solutions created merely for investigating a certain fixed scenario). They are given in Table 1; aside from the name of each project and its supposed main maintainers (as far as we were able to determine them), we also list the LCS variants it implements as well as the programming language used.

8 FURTHER LCS IMPROVEMENTS AND EXTENSIONS

This section presents the remaining improvements and extensions to existing LCSs that have not already been referenced in one of the previous sections. We divided the 15 contributions we found into three groups based on whether they are connected to ACS2, to XCS or to neither of them.

8.1 ACS2

As with other ML techniques, the non-determinism present in many real-world environments represents a learning challenge for the *Anticipatory Classifier System 2* (ACS2), especially in the face of *perceptual aliasing*. To enable ACS2 to learn an accurate internal representation of the environment in that case, Orhand et al. [29] devise PEPACS, an ACS2 variant based on adding *probability-enhanced predictions* (PEP) to classifiers whenever aliased states are detected. The problem of over-generalization due to PEPs is addressed through adjustments in the learning process, especially in the genetic algorithm, by transforming all PEPs in the offspring of PEP-enhanced classifiers to the corresponding symbol of the current state. PEPACS is evaluated on 16 different maze environments comprising aliased states. The obtained results show that PEPACS generates a complete and accurate environment representation even when perceptual aliasing is present.

In two other publications, the same authors pursue another, different, approach to deal with perceptual aliasing [27, 28]: They build *BACS*, another ACS2 variant that integrates behavioural sequences (i. e. sequences of actions) stored in so-called behavioural classifiers

Name	LCSs	Maintainer(s)	URL (GitHub)	Programming language/further comments
XCSF	XCSF	Preen	rpreen/xcsf	fast C backend with Python interface
scikit-eLCS	eLCS	Zhang and Urbanowicz	UrbsLab/scikit-eLCS	Python, scikit-learn interface
scikit-ExSTraCS	ExSTraCS	Zhang and Urbanowicz	UrbsLab/scikit-ExSTraCS	Python, scikit-learn interface
scikit-XCS	XCSR	Zhang and Urbanowicz	UrbsLab/scikit-xcs	Python, scikit-learn interface
Piecewise	XCS, XCSR	Bishop	jtbish/piecewise	Python
pyalcs	ACS, ACS2 YACS, MACS	Kozlowski	ParrotPrediction/pyalcs	Python
jGCS	GCS	Unold	ounold/jGCS	Java, documentation in Polish

Table 1: General open source LCS implementations that received updates in the past two years (in no particular order).

when aliased states are detected. An evaluation is conducted on 23 maze environments with different aliasing levels revealing that BACS performs similarly to ACS2 in the studied non-aliased environments and outperforms ACS2 in most of the studied aliased environments (14 out of 16).

To further enhance ACS2, Kozlowski and Unold [18] investigate exploration techniques for ACS2 and propose the application of the *optimistic initial quality* approach for exploration in ACS2, which is compared to the *epsilon-greedy* approach and the biased exploration techniques *action-delay* and *knowledge-array*. The authors conclude that the choice of the exploration technique affects the model learning performance, especially in the early phase of the learning process. In addition, they perform experiments in four real-valued environments (e. g. real multiplexer and cart pole), demonstrating ACS2 being applicable to real-valued environments through discretizing the input space. In [19], the same authors introduce the average reward criterion to ACS2, forming a modified variant called *average ACS2* (AACS2), to enhance the capabilites of ACS2 on sequential decision problems. To assess the performance of AACS2, the authors compare the modified variant to ACS2 and both a basic implementation of Q-learning and R-learning in three discretized multi-step environments: Corridor 20, Finite State Worlds 20 and Woods1. From the results they deduce that AACS2 can solve these multi-step problems as it is capable of generating a distinct payoff-landscape with uniformly spaced payoff levels.

8.2 XCS and its derivatives

We want to begin this section with a quick remark. In several of the papers we read, the term *XCS* was said to be an acronym for "extended classifier system". However, according to the very first papers on XCS [47, 48], this is *not* the case (and there also is not anything "extended" about XCS); instead, the term should probably rather be seen as a proper name or maybe as a recursive acronym for *XCS classifier system*.

In the past decades of LCS research, mechanisms such as *Experience Replay* (ER), one of the crucial factors for the success of Deep-Q-Networks, have barely attracted interest. To bridge this gap, Stein et al. [42] investigate a variant of XCS extended by ER, *XCS-ER*, which performs the learning process of XCS on previously encountered experiences (representing tuples comprising a state and the related previous state, action and reward). The performed evaluation experiments revealed on the one hand significant learning performance improvements through the increased sampling efficiency due to ER in single-step environments. On the other hand, the application of ER in XCS reduces the learning performance in multi-step environments, which is attributed to ER exacerbating the problem of over-generalization in XCS.

To improve covering in XCS and make it less reliant on a good choice of hyperparameters, Tadokoro et al. [43] introduce local covering for previously covered inputs (XCS-LCPCI). This approach utilizes neighbouring classifiers from the population for coverage by copying them and expanding their conditions' bounds until they also include the input. For previously unseen inputs, covering occurs as usual. They demonstrate applicability to XCS for binary problem domains on MNIST, the 20-bit multiplexer and the 11-bit class-imbalanced multiplexer and show that one less hyperparameter needs to be configured in contrast to traditional XCS.

One of the main strengths of LCSs are their inherent interpretability. However, the presence of redundant over-specific classifiers and inaccurate over-general classifiers obscures the important information contained in the generated model of the environment. Liu et al. [21] compare and evaluate in more detail three previously proposed approaches to enable human-discernable visualizations of an LCS model's underlying patterns: *feature importance maps* (FIM), *action-based feature importance maps* and *action-based feature's average value maps*. Their results highlight that these methods improve the interpretability of LCSs since they are able to precisely describe informative underlying patterns of a learned model. Additionally, FIMs are capable of reflecting the development of the classifiers' generalization levels during the learning process as well as of precisely identifying redundant attributes for hidden domains.

With *greedy niche mass compaction* (GNMC), Bishop and Gallagher [4] present a new compaction technique for XCSF in discrete RL settings. Evaluation of the technique takes place on both deterministic and stochastic maze-like *FrozenLake8x8* environments. Their XCSF baseline uses interval-based conditions, linear predictions and the XCSμ extension for handling uncertainty. To perform compaction, GNMC keeps classifiers greedily according to their quality (determined by a mass function) from each distinct action set. The number of classifiers that are kept depends on the distribution of the classifiers and an additionally introduced hyperparameter.

The authors stress that, in contrast to other compaction techniques, applying GNMC never results in all classifiers of an action set being removed. An evaluation shows a significant reduction in population size, without increasing function approximation error and only with a slight decrease in policy accuracy.

Nguyen et al. [26] present *mXOF*, an extension to *multi-task learning* for the XCS derivative XOF. mXOF consists of multiple XOF instances, one per task to be learned, which transfer features among each other if their respective tasks are related according to a novel dynamic task relatedness measure. The authors evaluate their system on groups of Boolean classification problems, both unrelated as well as related ones; as a baseline, multiple separate XOF instances without feature transfer are used. The results indicate that XOF's and mXOF's performance on groups of unrelated tasks is similar whereas for sets of related tasks, mXOF outperforms XOF. Besides that, a short comparison with several other standard ML methods on the UCI Zoo dataset shows mXOF's potential for being competitive.

Ramos et al. [32] integrate Hebbian learning (a learning method based on the plasticity of the connections of neurons) into the *generalized XCS* (GXCS), forming the variant GXCS-H by replacing the prediction array with a Hebbian weight vector and extending the classifier conditions with input-output pairs for the weight vector. The first results in comparison experiments with the standard GXCS in a woods environment and on the 6-multiplexer show that the addition of Hebbian learning to GXCS results in an improved overall learning performance. Furthermore, the use of a degradation variable leads to an increase in accurate classifiers as well as a reduction of over-specific classifiers.

8.3 Other LCSs and more general results

In LCSs, large amounts of computational resources are expended on *matching*, that is determining which classifiers' conditions are fulfilled by the current input. To reduce the computation time of this step, Rosenbauer et al. [34] propose two new generic approaches that—in contrast to previous proposals—do neither rely on specialised hardware nor on an implementation in a system-level programming language and should be suitable for several LCS representatives. The first approach relies on locking access to the match set, while the second distributes local match sets to threads and merges them afterwards. The authors measure a 58.33% to 67.74% decrease in runtime for matching and conclude that their first approach is better-suited if only a low number of cores is available while the latter outperforms on larger setups.

Motivated by the need for a robust and interpretable ML algorithm for classification tasks in the medical domain, Alaoui and Elberrichi [1] propose *NCGABIL*, an extension of the Pittsburgh-style LCS GABIL by the *neuronal communication algorithm* (NCA), which optimizes the present rule sets in each learning iteration. The authors compare NCGABIL to the original GABIL and multiple other classification algorithms present in WEKA (such as a conjunctive rule learner and a decision table/Naive Bayes hybrid classifier) on 16 well-known medical data sets from the UCI repository. The results indicate that the application of NCA leads to a considerable improvement of GABIL as NCGABIL achieves a superior predictive accuracy in 12 out of 16 data sets.

With the *weighted Grammar-based Classifier System* (wGCS), Unold et al. [45] introduce a new LCS for learning a weighted context-free grammar for some given data. wGCS builds upon previous work [44] on context-free grammars that did not yet include weights. Their experiments on three different synthetic context-free languages show competitiveness over standard methods for said task.

9 APPLICATIONS

Finally, a number of last year's contributions are about how LCSs can be applied to certain problems or how well they perform on them. We divided these into two groups based on whether they are directly associated to a real-world application or not (i. e. more theoretic).

9.1 Theoretic applications

Jordan et al. [17] propose LCSs as decision making modules for agent-based models that incorporate social influence and heterogeneous interconnected agents. Their case study is concerned with modelling pupils and their decisions of whether to engage with other pupils based on their expected marks. Each pupil agent uses a niche- and strength-based LCS to model the expected utility of its own education. In experiments, achieving good marks, bad marks and imitating peers were set to be the respective optimal strategy which the LCS was able to identify and follow. The authors conclude that LCSs may be an adequate approach to mimic human decision making in agent-based models; however, they also point out the restrictedness of their case study.

A major challenge in exploratory modelling methodology is understanding the data generated from the explored set of models. To provide improved explanations in the context of their *Strategy Learning System* framework for exploratory modelling, Rodriguez and Yilmaz [33] develop an XCS-based analytical tool which finds comprehensible rules and enables visualization of them using heat maps. The results of the performed experiment illustrate the increased generalization capabilities of the XCS-based tool when compared to directly visualizing the results of the explored set of models.

9.2 Real world applications

Automated testing plays a major role in the development and deployment of new products, both software and hardware. During recent years, due to paradigms such as test-driven development, for many products the number of tests defined has increased by so much that it has become infeasible to run all tests unless absolutely necessary. This led to the idea of prioritizing tests in such a manner that, given a fixed time budget, only the most important ones, according to some measure, are executed at any time. In [35], Rosenbauer et al. present their XCS-based solution to this problem and compare it with an earlier approach based on deep RL, showing that in six out of nine configurations (three reward functions, three data sets) their system is significantly superior. Besides this, they also investigate the effect of the inclusion of prioritized *experience replay* (ER), a technique well-known from its application to deep RL, into their system. This leads to a similar observation Stein et

al. made earlier [42], namely, that ER can actually worsen XCS's performance if the encountered states tend to be similar.

An improvement Rosenbauer et al. develop for their system uses XCSF with linear local models for approximating the state-action-value function [36]. This allows to assign a real-valued priority to each test instead of one of a finite set of ranks which results in the system outperforming basic XCS and being superior to the deep RL approach in eight of nine configurations considered. Later in the year, the same authors also investigate whether ER further improves XCSF's performance on the problem considered [37]; they conclude that this is indeed the case and that XCSF with ER performs equal or better than the deep RL–based approach.

In order to achieve the desired application goals in the area of conflict between adequate computing power and necessary performance constraints, *multiprocessor systems on chips* (MPSoCs) are increasingly dependent on adaptive resource management strategies. Maurer et al. [23] present a hierarchical, cross-layer hardware/software resource manager based on *learning classifier tables* (LCTs), which can be seen as a simplified version of LCSs. The resource manager comprises:

- A reflective monitoring control layer to monitor and, if necessary, control emergent behaviour by translating system requirements and application goals into objective functions for the LCTs.
- LCT-based leaf controllers that directly manipulate and enforce MPSoC building block operation parameters to explore and optimize potentially conflicting system requirements.
- An archive-based backup policy approach integrated within the leaf controllers providing the ability to adhere to constraints by resetting the system to valid configurations as it approaches constraints.

By evaluating their resource manager in a hardware implementation based on an FPGA board containing three SPARC-V8 cores, the authors show reflective supervision allows the hardware/software control layers of multicore processors to self-adapt under changing environmental or workload dynamics. In addition, archive-based backup policies manage potentially critical actions proactively, providing a trusted and effective means of complying with critical system constraints.

Smirnov et al. [40] develop an offline *design space exploration* (DSE) technique named *Learning Optimizer Constrained by ALtering conditions* (LOCAL) which is heavily inspired by LCSs. Offline DSE is the problem of pregenerating and optimizing a set of configuration alternatives to choose from at runtime in order to make a system adaptive that cannot perform expensive computations in order to choose a new configuration; examples for such systems are embedded systems. Smirnov et al.'s system generates a set of reconfiguration rules. They evaluate their approach on an adaptive many-core system tasked with dynamic application deployment and conclude that it outperforms a state-of-the-art approach to DSE in terms of performance and computational efficiency.

A challenging task in the field of swarm robotics is the creation of a *mobile ad-hoc network* (MANET), that is, establishing connectivity between mobile network devices with minimal operator interaction in network restricted areas. In their paper, Smith et al. [41] address this challenge with a neuroevolution- and an LCS-based algorithm that are both designed to generate the required behaviour within a swarm to establish a MANET. In evaluation experiments with virtual swarms, the LCS-based algorithm outperforms the neuroevolution-based algorithm by up to 61% with regard to data transmission efficiency and also provides improved swarm performance and reliability in both known and unknown environments.

Guevara and Santos [11] develop a hybrid system combining UCS, Bayesian networks and GANN-C for the assessment of the correct execution of rehab exercises by patients with hip-surgery. The authors evaluate several algorithms (besides the aforementioned ones as well as C4.5 and FURIA) on classifying the correct execution of rehab exercises based on motion capture data annotated by a physiotherapist. Then, the algorithm performing best is selected for each limb, yielding a hybrid approach (e. g. UCS is used for the right leg whereas a Bayesian network is used for the left leg) which achieves an accuracy of over 98%.

Yazdani [49] applies XCS to measurements of the force needed by robot manipulators to displace potential breast cancer tissue. The author evaluates his approach by training and testing XCS on a simulated dataset (400 simulation instances for training and 120 for testing) and reveals that XCS is, in this scenario, capable of both detecting breast cancer and determining the size and location of the cancer tissue.

The use of wind turbines for power generation has led to an increase in the variability and uncertainty of power supply as well as frequency control and transient stability issues in power grids, which poses significant challenges for grid operators. To address this, Alipour et al. [2] introduce a hybrid classifier combining XCSR with an *adaptive neuro-fuzzy inference system* (ANFIS) to predict wind speed ranges for arbitrary time intervals. Their approach essentially solves a binary classification task ('Is the wind speed range suitable for energy production or not?') using multiple meteorological features (such as temperature and solar radiation). After having completed XCSR's learning process, the classifier population is used to generate two ANFIS models (one for each class) that serve as the final model in the following. The authors compare their method to the original XCSR model and several other classifiers included in MATLAB and demonstrate its superior performance and accuracy for both short- and long-term horizons.

Alsharafat [3] conducts a study on utilizing the cuckoo algorithm in combination with XCS to select features for intrusion detection systems—however, the exact details are slightly obscure. The author's evaluation results hint towards their method being competitive.

Malhotra and Jain [22] compare 16 search-based techniques regarding their performance on predicting software defects in 13 object oriented software projects. Among the 8 methods considered that the authors classify as "statistically good predictors" are UCS, BioHEL as well as other LCSs.

A decision support system for classifying ECG signals is developd by Zouri et al. [51]. They employ an LCS to improve existing rules and—if needed—discover new rules that are stored in a Semantic Web Rule Language (SWRL) repository. The use of SWRL is meant to facilitate integration into common systems. Discovering new rules

is either triggered by expert users or whenever no rules matching the current input are found.

Ferjani et al. [8, 9] present a multi-agent system for sleep analysis. Each learning agent encapsulates an XCS and provides predictions for sleep stage classification. Those predictions are then aggregated through majority voting and, if needed, selected against a static knowledge background. In their experiments, the authors utilize four agents each operating on a distinct physiological signal. They report difficulties in configuring XCS's hyperparameters for the used data although the results are competitive to literature while needing less labeled data.

10 SUMMARY AND CONCLUSIONS

This paper gave an overview of all the LCS-related publications since 11 March 2020, that is, since the submission of the previous such survey to the IWLCS 2020. We clustered the contributions we found into seven overall topics.

Formal theory has only seen a single work in the last year; however, based on that and earlier work, there has been created a practical, well-performing online hyperparameter adaptation method. We encourage researchers to contribute further formal theory in order to, in the long term, develop more such formally backed improvements to LCSs.

Besides the just mentioned one, there were also several other promising approaches to setting and adapting the hyperparameters of some of the most popular LCSs. Especially, since some of these approaches may be utilized in other LCSs than just the ones they were originally developed for, we expect the number or at least the sensitivity of LCS hyperparameters to be reduced in the upcoming years.

We listed recently updated implementations and noticed an increased awareness to the benefits of compatibility with quasi-standards such as scikit-learn.

Just like in recent years, both ACS2 and XCS, received the most attention regarding different kinds of improvements and extensions.

Finally, Sections 6 and 9 showed that new LCS architectures for solving different kinds of problems are still being developed as well as that LCSs have been applied to a variety of theoretical and practical problems; this shows, once more, the versatility of the LCS paradigm.

ACKNOWLEDGMENTS

This work is partially supported by the German Federal Ministry for Economic Affairs and Energy (BMWi).

REFERENCES

[1] Abdiya Alaoui and Zakaria Elberrichi. 2020. Neuronal Communication Genetic Algorithm-Based Inductive Learning. *Journal of Information Technology Research (JITR)* 13, 2 (2020), 141–154. https://doi.org/10.4018/JITR.2020040109

[2] Mohammadali Alipour, Jamshid Aghaei, Mohammadali Norouzi, Sattar Hashemi, and Matti Lehtonen. 2021. A Novel Cooperative Fuzzy Classifier for Predicting the Permissible Wind Speed Range in Wind Farms. *Iranian Journal of Science and Technology, Transactions of Electrical Engineering* 45, 1 (01 Mar 2021), 29–45. https://doi.org/10.1007/s40998-020-00347-z

[3] Wafa Alsharafat. 2020. The Cuckoo Feature Filtration Method for Intrusion Detection (Cuckoo-ID). *International Journal of Advanced Computer Science and Applications (IJACSA)* 11, 5 (2020).

[4] Jordan T. Bishop and Marcus Gallagher. 2020. Optimality-Based Analysis of XCSF Compaction in Discrete Reinforcement Learning. In *Parallel Problem Solving from Nature – PPSN XVI*, Thomas Bäck, Mike Preuss, André Deutz, Hao Wang, Carola Doerr, Michael Emmerich, and Heike Trautmann (Eds.). Springer International Publishing, Cham, 471–484.

[5] Martin V. Butz and Stewart W. Wilson. 2001. An Algorithmic Description of XCS. In *Advances in Learning Classifier Systems, Third International Workshop, IWLCS 2000*, Pier Luca Lanzi, Wolfgang Stolzmann, and Stewart W. Wilson (Eds.). Springer, 253–272.

[6] Hao Chen, Jian Huang, Quan Liu, Chang Wang, and Hanqiang Deng. 2020. Detecting and Tracing Multi-Strategic Agents with Opponent Modelling and Bayesian Policy Reuse. In *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. 1098–1103. https://doi.org/10.1109/ICIEA48937.2020.9248178

[7] Hao Chen, Chang Wang, Jian Huang, Jiangtao Kong, and Hanqiang Deng. 2020. XCS with opponent modelling for concurrent reinforcement learners. *Neurocomputing* 399 (2020), 449–466. https://doi.org/10.1016/j.neucom.2020.02.118

[8] Rahma Ferjani, Lilia Rejeb, and Lamjed Ben Said. 2020. Cooperative Reinforcement Multi-Agent Learning System for Sleep Stages Classification. In *2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA)*. 1–8. https://doi.org/10.1109/OCTA49274.2020.9151700

[9] Rahma Ferjani, Lilia Rejeb, and Lamjed Ben Said. 2020. Unsupervised Sleep Stages Classification Based on Physiological Signals. In *Advances in Practical Applications of Agents, Multi-Agent Systems, and Trustworthiness. The PAAMS Collection*, Yves Demazeau, Tom Holvoet, Juan M. Corchado, and Stefania Costantini (Eds.). Springer International Publishing, Cham, 134–145.

[10] Maria A. Franco, Natalio Krasnogor, and Jaume Bacardit. 2020. Automatic Tuning of Rule-Based Evolutionary Machine Learning via Problem Structure Identification. *IEEE Computational Intelligence Magazine* 15, 3 (2020), 28–46. https://doi.org/10.1109/MCI.2020.2998232

[11] César Guevara and Matilde Santos. 2020. Intelligent models for movement detection and physical evolution of patients with hip surgery. *Logic Journal of the IGPL* (09 2020). https://doi.org/10.1093/jigpal/jzaa032 arXiv:https://academic.oup.com/jigpal/advance-article-pdf/doi/10.1093/jigpal/jzaa032/33786547/jzaa032.pdf jzaa032.

[12] Tim Hansmeier, Paul Kaufmann, and Marco Platzner. 2020. An Adaption Mechanism for the Error Threshold of XCSF. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1756–1764. https://doi.org/10.1145/3377929.3398106

[13] Tim Hansmeier, Paul Kaufmann, and Marco Platzner. 2020. Enabling XCSF to Cope with Dynamic Environments via an Adaptive Error Threshold. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 125–126. https://doi.org/10.1145/3377929.3389968

[14] Michael Heider, David Pätzel, and Jörg Hähner. 2020. Towards a Pittsburgh-Style LCS for Learning Manufacturing Machinery Parametrizations. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 127–128. https://doi.org/10.1145/3377929.3389963

[15] Motoki Horiuchi and Masaya Nakata. 2020. Self-Adaptation of XCS Learning Parameters Based on Learning Theory. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 342–349. https://doi.org/10.1145/3377930.3389814

[16] Muhammad Irfan, Zheng Jiangbin, Muhammad Iqbal, and Muhammad Hassan Arif. 2021. A novel lifelong learning model based on cross domain knowledge extraction and transfer to classify underwater images. *Information Sciences* 552 (2021), 80–101. https://doi.org/10.1016/j.ins.2020.11.048

[17] Tobias Jordan, Philippe de Wilde, and Fernando Buarque de Lima Neto. 2020. Decision making for two learning agents acting like human agents : A proof of concept for the application of a Learning Classifier Systems. In *2020 IEEE Congress on Evolutionary Computation (CEC)*. 1–8. https://doi.org/10.1109/CEC48606.2020.9185570

[18] Norbert Kozlowski and Olgierd Unold. 2020. Investigating Exploration Techniques for ACS in Discretized Real-Valued Environments. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1765–1773. https://doi.org/10.1145/3377929.3398079

[19] Norbert Kozłowski and Olgierd Unold. 2021. Anticipatory Classifier System with Average Reward Criterion in Discretized Multi-Step Environments. *Applied Sciences* 11, 3 (2021). https://doi.org/10.3390/app11031098

[20] Yi Liu, Will N. Browne, and Bing Xue. 2020. Absumption and Subsumption Based Learning Classifier Systems. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 368–376. https://doi.org/10.1145/3377930.3389813

[21] Yi Liu, Will N. Browne, and Bing Xue. 2021. Visualizations for rule-based machine learning. *Natural Computing* (29 Jan 2021). https://doi.org/10.1007/s11047-020-09840-0

[22] Ruchika Malhotra and Juhi Jain. 2021. Predicting Software Defects for Object-Oriented Software Using Search-based Techniques. *International Journal of Software Engineering and Knowledge Engineering* 31, 02 (2021), 193–215. https://doi.org/10.1142/S0218194021500054 arXiv:https://doi.org/10.1142/S0218194021500054

[23] Florian Maurer, Bryan Donyanavard, Amir M. Rahmani, Nikil Dutt, and Andreas Herkersdorf. 2020. Emergent Control of MPSoC Operation by a Hierarchical Supervisor / Reinforcement Learning Approach. In *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*. 1562–1567.

[24] Masaya Nakata and Will N. Browne. 2021. Learning Optimality Theory for Accuracy-Based Learning Classifier Systems. *IEEE Transactions on Evolutionary Computation* 25, 1 (2021), 61–74. https://doi.org/10.1109/TEVC.2020.2994314

[25] Shabnam Nazmi, Xuyang Yan, Abdollah Homaifar, and Emily Doucette. 2020. Evolving multi-label classification rules by exploiting high-order label correlations. *Neurocomputing* 417 (2020), 176–186. https://doi.org/10.1016/j.neucom.2020.07.055

[26] Trung B. Nguyen, Will N. Browne, and Mengjie Zhang. 2020. Relatedness Measures to Aid the Transfer of Building Blocks among Multiple Tasks. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 377–385. https://doi.org/10.1145/3377930.3390169

[27] Romain Orhand, Anne Jeannin-Girardon, Pierre Parrend, and Pierre Collet. 2020. BACS: A Thorough Study of Using Behavioral Sequences in ACS2. In *Parallel Problem Solving from Nature – PPSN XVI*, Thomas Bäck, Mike Preuss, André Deutz, Hao Wang, Carola Doerr, Michael Emmerich, and Heike Trautmann (Eds.). Springer International Publishing, Cham, 524–538.

[28] Romain Orhand, Anne Jeannin-Girardon, Pierre Parrend, and Pierre Collet. 2020. BACS: Integrating Behavioral Sequences to ACS2. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 147–148. https://doi.org/10.1145/3377929.3390002

[29] Romain Orhand, Anne Jeannin-Girardon, Pierre Parrend, and Pierre Collet. 2020. PEPACS: Integrating Probability-Enhanced Predictions to ACS2. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1774–1781. https://doi.org/10.1145/3377929.3398121

[30] David Pätzel, Anthony Stein, and Jörg Hähner. 2019. A Survey of Formal Theoretical Advances Regarding XCS. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO '19)*. Association for Computing Machinery, New York, NY, USA, 1295–1302. https://doi.org/10.1145/3319619.3326848

[31] David Pätzel, Anthony Stein, and Masaya Nakata. 2020. An Overview of LCS Research from IWLCS 2019 to 2020. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1782–1788. https://doi.org/10.1145/3377929.3398105

[32] Marco Ramos, Vianney Muñoz-Jiménez, and Félix F. Ramos. 2020. Learning Clasiffier Systems with Hebbian Learning for Autonomus Behaviors. In *Pattern Recognition*, Karina Mariela Figueroa Mora, Juan Anzurez Marín, Jaime Cerda, Jesús Ariel Carrasco-Ochoa, José Francisco Martínez-Trinidad, and José Arturo Olvera-López (Eds.). Springer International Publishing, Cham, 328–339.

[33] Brodderick Rodriguez and Levent Yilmaz. 2020. Learning Rule-Based Explanatory Models from Exploratory Multi-Simulation for Decision-Support Under Uncertainty. In *2020 Winter Simulation Conference (WSC)*. 2293–2304. https://doi.org/10.1109/WSC48552.2020.9383858

[34] Lukas Rosenbauer, Anthony Stein, and Jörg Hähner. 2020. Generic Approaches for Parallel Rule Matching in Learning Classifier Systems. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1789–1797. https://doi.org/10.1145/3377929.3398102

[35] Lukas Rosenbauer, Anthony Stein, Roland Maier, David Pätzel, and Jörg Hähner. 2020. XCS as a Reinforcement Learning Approach to Automatic Test Case Prioritization. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1798–1806. https://doi.org/10.1145/3377929.3398128

[36] Lukas Rosenbauer., Anthony Stein., David Pätzel., and Jörg Hähner. 2020. XCSF for Automatic Test Case Prioritization. In *Proceedings of the 12th International Joint Conference on Computational Intelligence - Volume 1: ECTA*. INSTICC, SciTePress, 49–58. https://doi.org/10.5220/0010105700490058

[37] Lukas Rosenbauer, Anthony Stein, David Pätzel, and Jörg Hähner. 2020. XCSF with Experience Replay for Automatic Test Case Prioritization. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. 1307–1314. https://doi.org/10.1109/SSCI47803.2020.9308379

[38] Abubakar Siddique, Will N. Browne, and Gina M. Grimshaw. 2020. Lateralized Learning for Robustness against Adversarial Attacks in a Visual Classification System. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 395–403. https://doi.org/10.1145/3377930.3390164

[39] Abubakar Siddique, Will N. Browne, and Gina M. Grimshaw. 2020. Learning Classifier Systems: Appreciating the Lateralized Approach. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1807–1815. https://doi.org/10.1145/3377929.3398101

[40] Fedor Smirnov, Behnaz Pourmohseni, and Jügen Teich. 2020. Using Learning Classifier Systems for the DSE of Adaptive Embedded Systems. In *2020 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. 957–962. https://doi.org/10.23919/DATE48585.2020.9116383

[41] Phillip Smith, Asad Khan, Aldeida Aleti, Vincent C. S. Lee, and Robert Hunjet. 2020. Data Communication Assistance via Swarm Robotic: A Behaviour Creation Comparison. In *Artificial Intelligence. IJCAI 2019 International Workshops*, Amal El Fallah Seghrouchni and David Sarne (Eds.). Springer International Publishing, Cham, 111–126.

[42] Anthony Stein, Roland Maier, Lukas Rosenbauer, and Jörg Hähner. 2020. XCS Classifier System with Experience Replay. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 404–413. https://doi.org/10.1145/3377930.3390249

[43] Masakazu Tadokoro, Satoshi Hasegawa, Takato Tatsumi, Hiroyuki Sato, and Keike Takadama. 2020. Local Covering: Adaptive Rule Generation Method Using Existing Rules for XCS. In *2020 IEEE Congress on Evolutionary Computation (CEC)*. 1–8. https://doi.org/10.1109/CEC48606.2020.9185669

[44] Olgierd Unold. 2005. Context-free grammar induction with grammar-based classifier system. *Archives of Control Sciences* Vol. 15, no. 4 (2005), 681–690.

[45] Olgierd Unold, Mateusz Gabor, and Wojciech Wieczorek. 2020. Unsupervised Statistical Learning of Context-free Grammar. In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence - Volume 1: NLPinAI,*. INSTICC, SciTePress, 431–438. https://doi.org/10.5220/0009383604310438

[46] Siddharth Verma, Piyush Borole, and Ryan Urbanowicz. 2020. Evolving Genetic Programming Trees in a Rule-Based Learning Framework. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 233–234. https://doi.org/10.1145/3377929.3390071

[47] Stewart W. Wilson. 1995. Classifier Fitness Based on Accuracy. *Evolutionary Computation* 3, 2 (1995), 149–175.

[48] Stewart W. Wilson. 1996. Generalization in XCS. In *Machine Learning, Proceedings of the 13th International Conference (ICML '96)*, Lorenza Saitta (Ed.). Morgan Kaufmann, 3–11.

[49] Navid Moshtaghi Yazdani. 2020. Diagnosing Soft Tissue Sub-Surface Masses Using the XCS Classification System. *Frontiers in Health Informatics* 9, 1 (2020), 49. https://doi.org/10.30699/fhi.v9i1.239

[50] Robert F. Zhang and Ryan J. Urbanowicz. 2020. A Scikit-Learn Compatible Learning Classifier System. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion (GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 1816–1823. https://doi.org/10.1145/3377929.3398097

[51] Muthana Zouri, Nicoleta Zouri, and Alex Ferworn. 2020. ECG Knowledge Discovery Based on Ontologies and Rules Learning for the Support of Personalized Medical Decision Making. In *2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 0701–0706. https://doi.org/10.1109/IEMCON51383.2020.9284951