University of Maribor

Faculty of Electrical Engineering
and Computer Science

FERI

Association Rule Mining as Knowledge Infusion into the Mechanics of an eXtended Classifier System

IEEE 28TH INTERNATIONAL CONFERENCE ON INTELLIGENT ENGINEERING SYSTEMS 2024 (INES 2024)

Authors:
Damijan Novak (damijan.novak@um.si), Domen Verber (domen.verber@um.si), Iztok Fister (iztok.fister@um.si), and Iztok Fister Jr. (iztok.fister1@um.si)

18.7.2024

Presentation agenda

1. Motivation and Purpose of the Article
2. Numerical Association Rule Mining (NARM)
3. eXtended Classifier System (XCS)
4. NARM-XCS Architecture
5. Experiment
6. Results
7. Conclusion

Motivation

- Humanity creates vast amounts of data every day.

- The data increases come on behalf of data sensors, the technological interconnections of people across the internet (e.g., social networks), due to the need to help us understand climate change better, LLMs, medical data, etc.

- Questions arise, such as how many computational resources are needed for processing, how soon they can be processed, or how interpretable the results are.

- Deep Neural Networks (although excellent!) are not the only possible solution to the problem. Simultaneously researching other Machine Learning (ML) methods is also highly relevant!

Purpose

- To position the work in the classical ML algorithms domain.
- The purpose of the algorithm presented in the article
  - Reusing knowledge.
  - Making the algorithms more adaptable to new data (e.g., in medicine).
  - Potentially enhancing the interpretability of the results of the process.

- The study, therefore, focuses on reusing Association Rule Mining results, which can be computationally expensive to produce.
- Utilizing Reinforcement Learning (RL) to learn associations between the environment input and the proper action output for any prioritized action (dataset attribute).
- Augmenting the adaptability of the system to accommodate the new environmental state when presented with new sensory input data.

Numerical Association Rule Mining

- Association Rule Mining (ARM) aims to discover the relations between attributes hidden in transaction databases.

- Most algorithms for mining association rules focus on datasets consisting of discrete attributes only.

- With the development in domains of Evolutionary Algorithms and Swarm Intelligence-based algorithms, universal algorithms have been developed for mining mixed types of attributes (i.e., discrete and numerical).

- These algorithms are known under the name Numerical Association Rule Mining (NARM).

Numerical Association Rule Mining (2)

- An association rule is defined as an implication:

$$X \Rightarrow Y$$

where $X \subset O$, $Y \subset O$, and $X \cap Y = \emptyset$,

for a set of features $O = \{O_1, \ldots, O_m\}$,

and a transaction database Db

(consisting of transactions Tr with each transaction containing a subset of features $Tr \subseteq O$)

- The quality of the association rules is typically evaluated using the following two measures:

$$supp(X \Rightarrow Y) = \frac{n(X \cup Y)}{N}$$

$$conf(X \Rightarrow Y) = \frac{n(X \cup Y)}{n(X)}$$

eXtended Classifier Systems (XCS)

- They are part of the larger Learning Classifier Systems (LCS) domain.
- They are a model-free system. The XCS doesn't attempt to comprehensively represent the relationships or dynamics within the data (i.e., no detailed environment model is constructed).
- Focus is on learning associations between inputs and outputs from the data.
- Learned associations (knowledge) are kept in a so-called population (a set of classifiers).

Classifier:

IF condition THEN action

- A condition also supports the concept of a 'don't care' (#) mechanism (for generalization purposes).

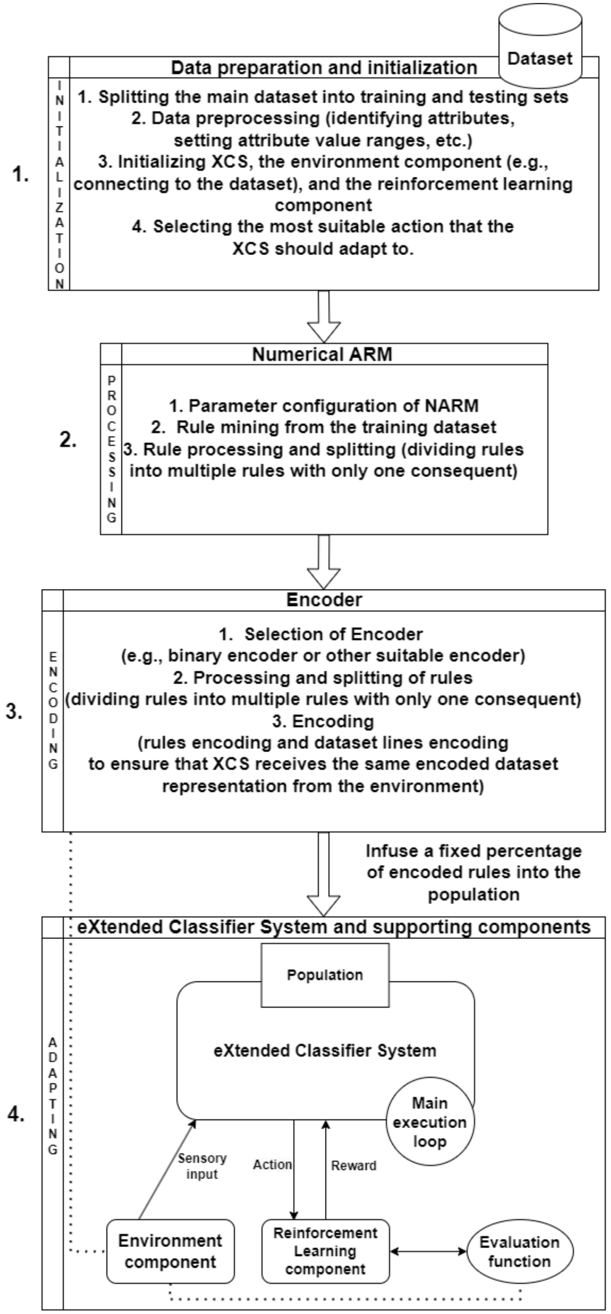
eXtended Classifier Systems (XCS) (2)

- It uses three sets (population, match set, and action set, all comprising classifiers) and operates within a main execution loop.
- The iterative behavior:
  1. The environment is observed, and the sensory input is received (In our case, the environment sensory input is one encoded line).
  2. The algorithm generates a match set by selecting classifiers that match the current input.
  3. Using this match set, XCS produces predictions for possible actions.
  4. The selection of an action based on these predictions.
- After the execution of the chosen action, the algorithm receives feedback from the environment, allowing it to adapt and evolve the classifiers in its population.

NARM-XCS architecture

- Four main steps are involved in the operation of the architecture:
  1. The data preparation and initialization.
  2. The generation and processing of NARM rules.
  3. Selecting and using an encoder on the rules and the dataset lines.
  4. Infusing the encoded rules into the XCS algorithm.

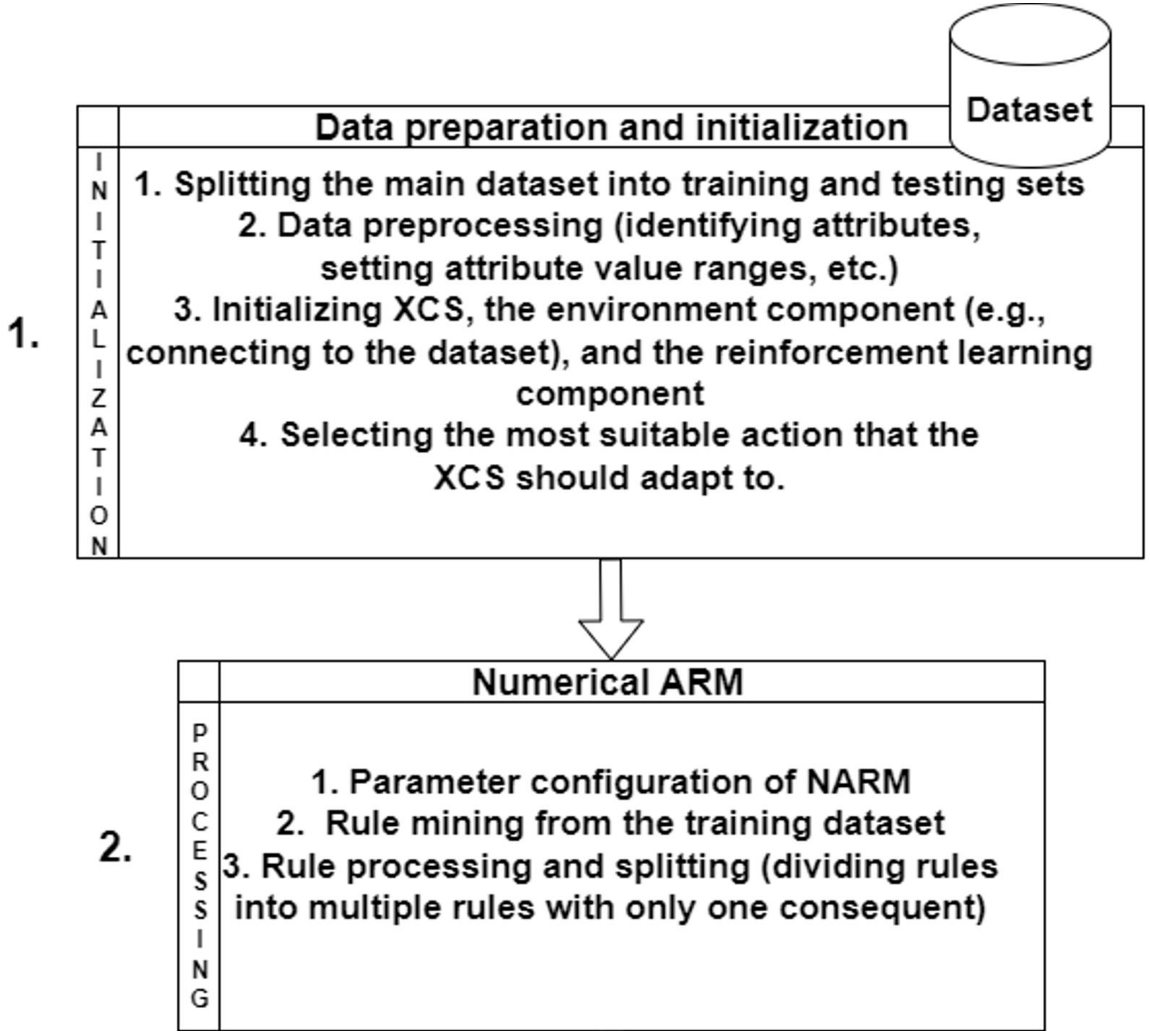
- The encoded rules are then evolved using the environment, RL components, and the evaluation function, which guides the search for the prioritized action.



NARM-XCS architecture

- A crucial aspect of the first two steps is that the dataset's attributes are used directly as actions by the XCS algorithm.

- When the NARM rules are integrated into the XCS population, the consequents of these rules become actions.



Example of the NARM rule:

0.766667 ['Attribute7_-43.0000_-8.3416']===>['Attribute4_558.8417_641.0848','Attribute5_-188.0000_19.5203'] 1 1

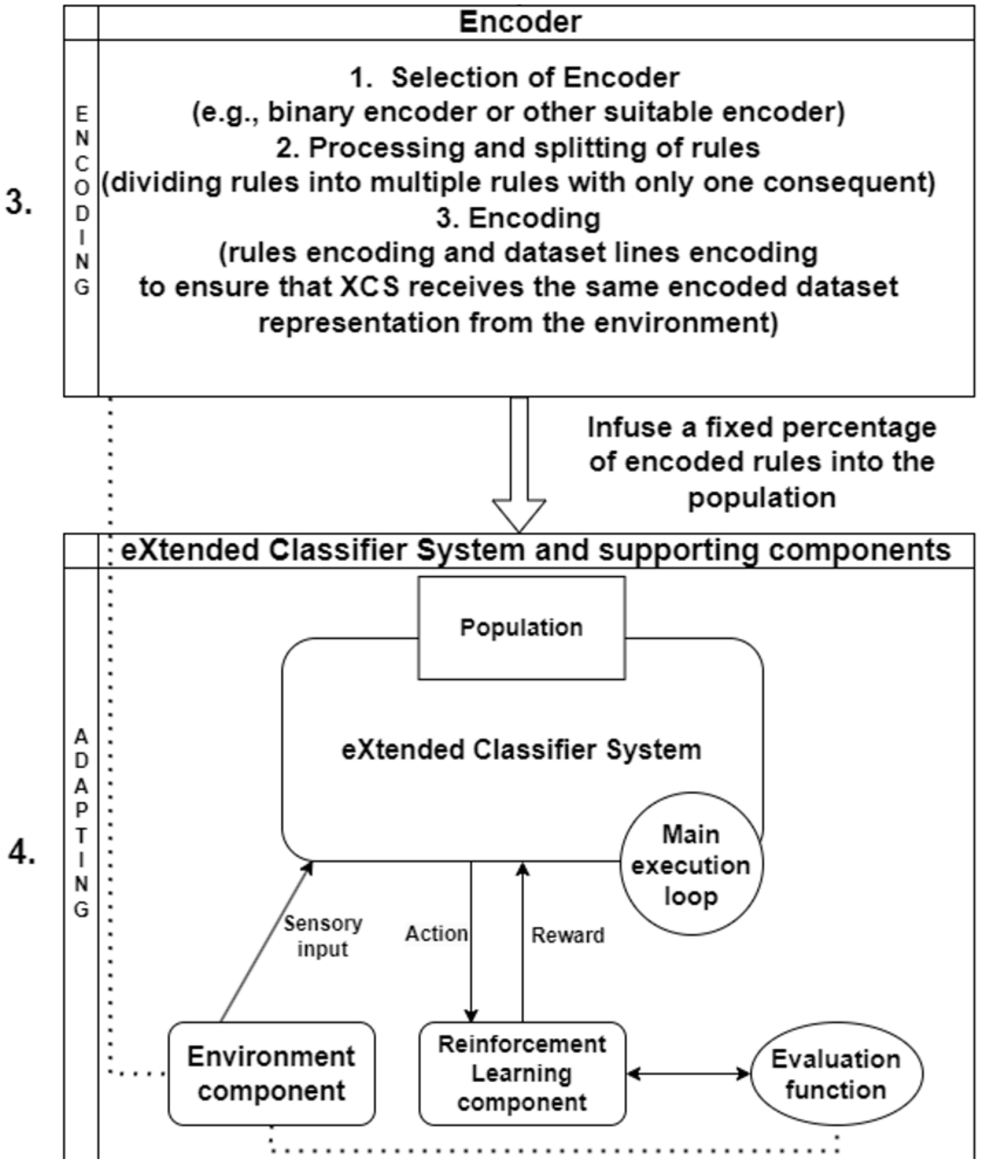
Result after the rule processing:

['Attribute7_-43.0000_-8.3416']===>['Attribute4_558.8417_641.0848']

['Attribute7_-43.0000_-8.3416']===>['Attribute5_-188.0000_19.5203']

NARM-XCS architecture

- A basic binary encoder was used in experiments.
- The binary encoder encodes numerical data into a set of binary values by calculating the bin number for each attribute value.
- Then the number is converted to a binary string with the appropriate number of bits, and each bit is added to an encoded list.
- A fixed percentage of encoded rules are infused into the XCS population.



Experiment: Main settings

- A preliminary investigation into an infusion of the rules in the population of the XCS algorithm to test if the learning performance of the XCS algorithm was improved.
- The study also aimed to assess the outcomes of the training section, and how stable the reward received was during the subsequent testing phase of the experiment.
- The experiment was run on an interval of 0 to 100 percent of encoded rules used in the XCS population, with a step of 10 percent.
- The training and testing runs were done five times for each step percentage, and the reward average was calculated.
- Only one pass of the dataset was allowed for each run.
- NARM rule generation was done only once.

Experiment: Rewards

- The reward was calculated by a basic evaluation function. It provides a positive reward of 0.2 if the XCS selected the action per the prioritized action (attribute), and a negative reward of -0.1 otherwise.
- In a subsequent experiment, a positive reward of 1.0 was added (i.e., enhancing the evaluation function) when the encoded XCS selected an action and its action value matched the corresponding encoded attribute and its value from the testing dataset.
- This subsequent experiment aimed to better understand how NARM-XCS behavior changed, and how its adaptation was affected.
- Important note: The XCS doesn’t include mechanisms to change the values of actions during the main execution loop.

Experiment: Other settings

- Two classification datasets from the UCI ML repository:
  - Wine dataset, which holds 178 instances and has 13 features, and the feature type is comprised of integer and real values.
  - Statlog (Shuttle) dataset, which holds 58,000 instances, has eight features and comprises integer values.
- The training and testing samples in the ratio of 80:20.

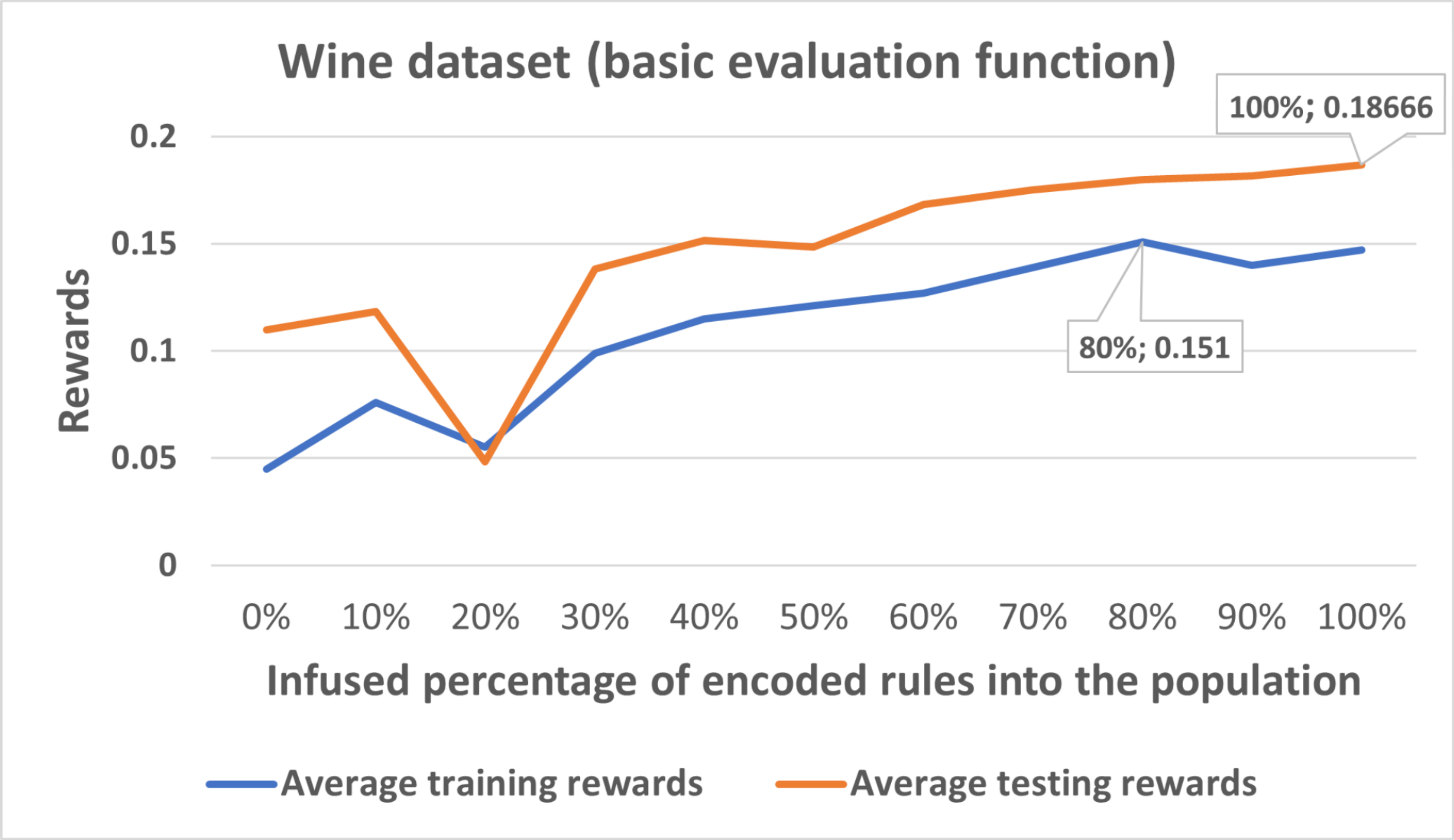
Parameters	Values
Algorithm	Differential Evolution
NP	100
nFes	1000
F	0.5
CR	0.9

TABLE I
uARMSolver hyperparameters

Parameters	Values
N (population size)	500
alfa	0.1
beta	0.01
gamma	0.71
delta	0.1
θ_{GA}	25
ε_0	10
θ_{del}	20
ν	5
χ	0.5
μ	0.01
θ_{sub}	20
$P\#$ (probability of using #)	0.33
doGASubsumption	true
doActionSetSubsumption	true

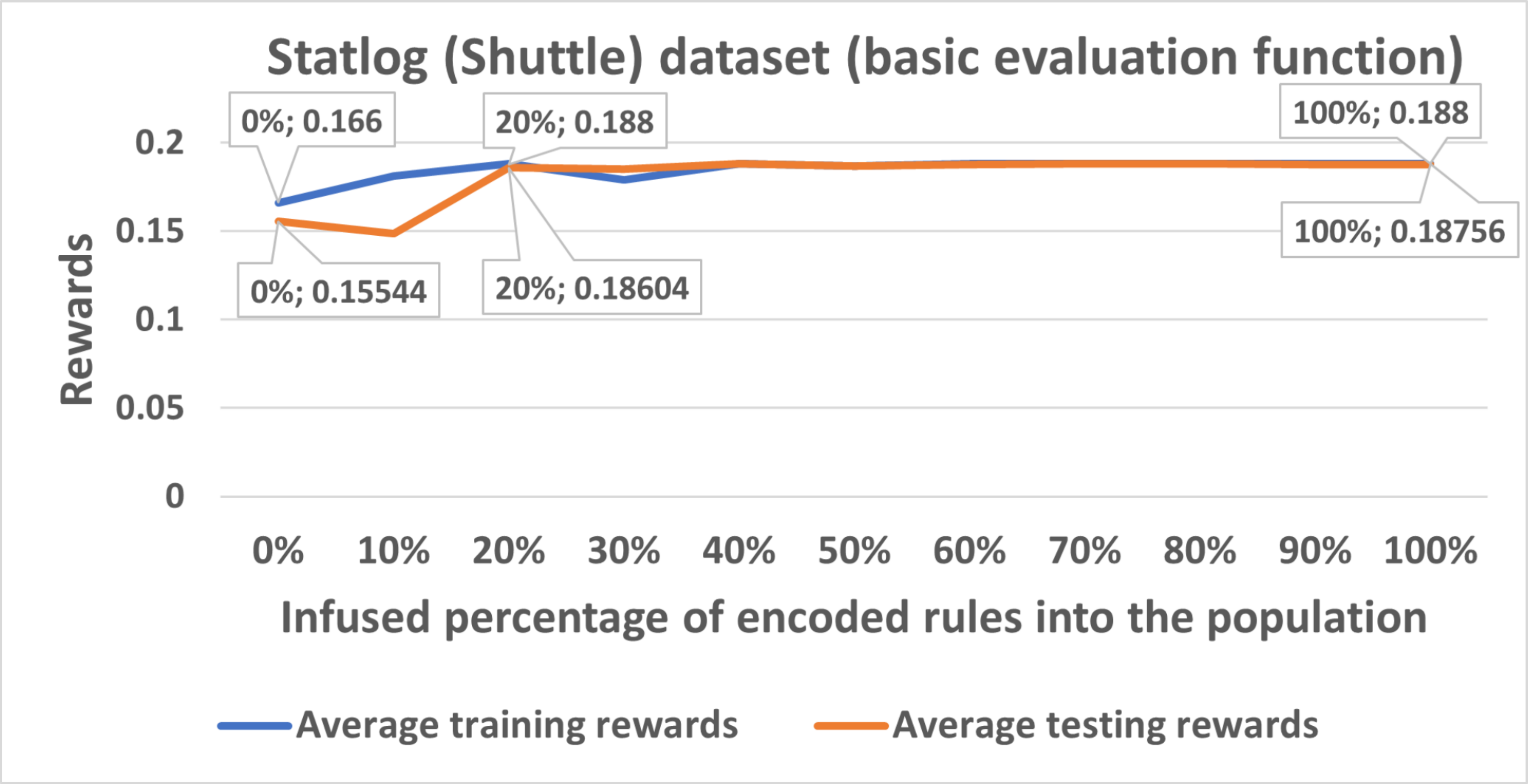
TABLE II
XCS Hyperparameters

Results

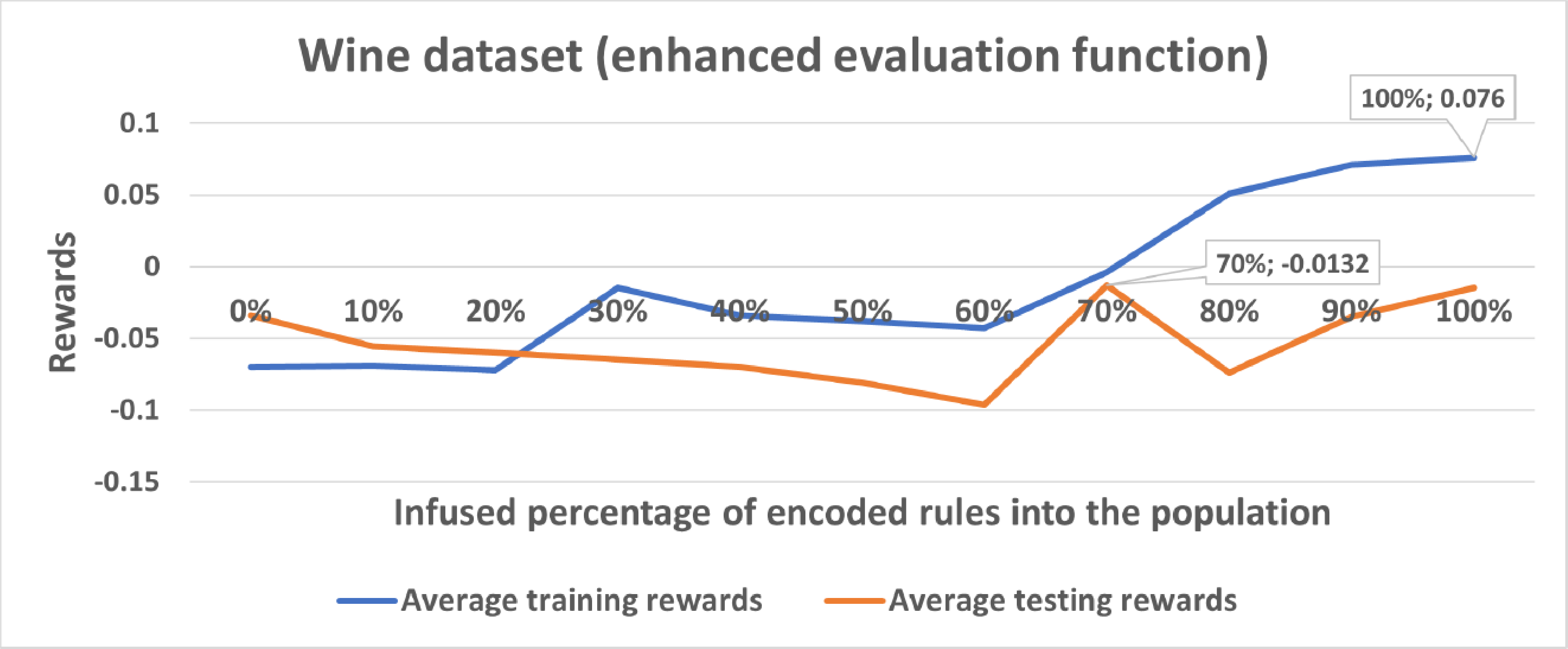


Wine dataset (basic evaluation function)
100%; 0.18666
80%; 0.151
Rewards
0.2
0.15
0.1
0.05
0
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Infused percentage of encoded rules into the population
Average training rewards
Average testing rewards

Results (2)

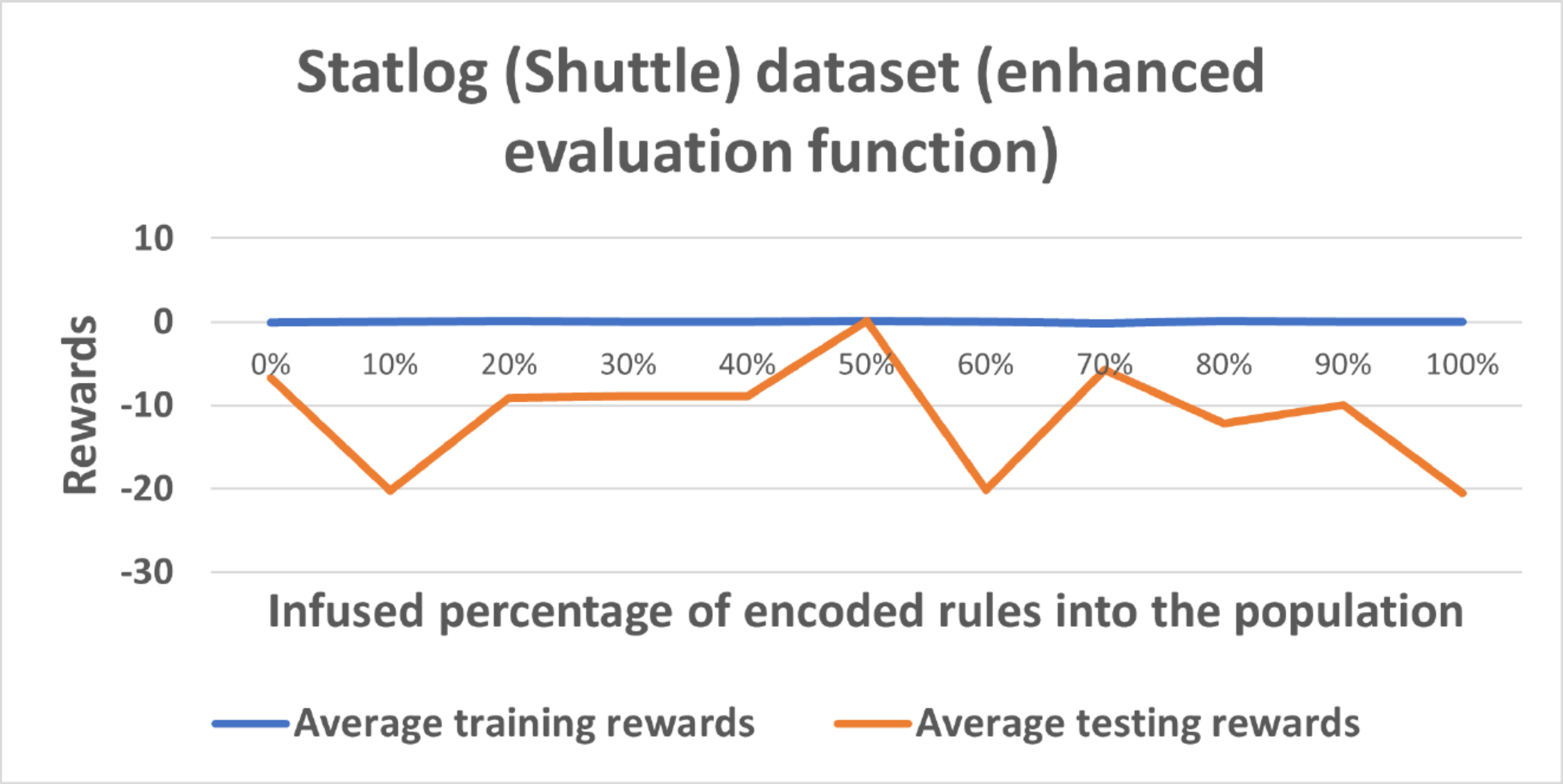

Statlog (Shuttle) dataset (basic evaluation function)
0%; 0.166
20%; 0.188
100%; 0.188
0%; 0.15544
20%; 0.18604
100%; 0.18756
Rewards
0.2
0.15
0.1
0.05
0
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Infused percentage of encoded rules into the population
Average training rewards
Average testing rewards

Results (3)



Wine dataset (enhanced evaluation function)
100%; 0.076
70%; -0.0132
0.1
0.05
0
-0.05
-0.1
-0.15
Rewards
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Infused percentage of encoded rules into the population
Average training rewards
Average testing rewards

Results (4)



Statlog (Shuttle) dataset (enhanced evaluation function)
10
0
-10
-20
-30
Rewards
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Infused percentage of encoded rules into the population
Average training rewards
Average testing rewards

Conclusion

The following contributions were achieved with this work:

- Bridging the gap between previously generated NARM rules and the proper action-taking of XCS in the environment (e.g., priming for rare events).
- Fast adaptation of the rules with new sensory input data (i.e., can be used to adapt to the specific sub-domains).
- Future work:
  - The first steps taken with our work were not only to choose a proper action, but also, in the future, the proper value for that action (e.g., connecting small subsets of data (rules) to exceptional behavior (actions)).
  - Advanced encoders, filtering the rules based on their fitness thresholds, improved evaluation functions, explainable AI (i.e., explainability of classifier population results), etc.

Thank you for your time. Please join the discussion.