Evolutionary Computation for the Automated Design of Category Functions for Fuzzy ART: An Initial Exploration

Islam Elnabarawy
Comp. Sci. Dept.
Missouri University of
Science and Technology
500 W. 15th St.
Rolla, Missouri 65409
ie3md@mst.edu

Daniel R. Tauritz
Comp. Sci. Dept.
Missouri University of
Science and Technology
500 W. 15th St.
Rolla, Missouri 65409
dtauritz@acm.org

Donald C. Wunsch
Electrical and Comp. Eng. Dept.
Missouri University of
Science and Technology
301 W. 16th St.
Rolla, Missouri 65409
dwunsch@mst.edu

ABSTRACT

Fuzzy Adaptive Resonance Theory (ART) is a classic unsupervised learning algorithm. Its performance on a particular clustering problem is sensitive to the suitability of the category function for said problem. However, classic Fuzzy ART employs a fixed category function and thus is unable to benefit from the potential to adjust its category function. This paper presents an exploration into employing evolutionary computation for the automated design of category functions to obtain significantly enhanced Fuzzy ART performance through tailoring to specific problem classes. We employ a genetic programming powered hyper-heuristic approach where the category functions are constructed from a set of primitives constituting those of the original Fuzzy ART category function as well as additional hand-selected primitives. Results are presented for a set of experiments on benchmark classification tasks from the UCI Machine Learning Repository, demonstrating that tailoring Fuzzy ART's category function can achieve statistically significant superior performance on the testing datasets in stratified 10-fold cross-validation procedures. We conclude with discussing the results and placing them in the context of being a first step towards automating the design of entirely new forms of ART.

CCS CONCEPTS

•Computing methodologies →Genetic programming;

KEYWORDS

evolutionary computing, genetic programming, hyper-heuristics, adaptive resonance theory, clustering, unsupervised learning, adjusted rand index

ACM Reference format:
Islam Elnabarawy, Daniel R. Tauritz, and Donald C. Wunsch. 2017. Evolutionary Computation for the Automated Design of
Category Functions for Fuzzy ART: An Initial Exploration. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3067695.3082056

DOI: http://dx.doi.org/10.1145/3067695.3082056

1 INTRODUCTION

Adaptive resonance theory (ART) [10] is a theory of how brains handle cognitive and information processing tasks. It has led to the introduction of a large number of neural network models that address different types of pattern recognition and prediction tasks [3, 4, 6, 7]. Many different ART architectures have been studied and applied successfully to supervised, unsupervised, and reinforcement learning problems; however, answering the question of which ART architecture is best suited to a given problem often needs some experimentation and domain expertise. Moreover, the most suitable ART architecture for a given problem may very well be one that is yet to be invented, even if it only requires a combination of ideas from two existing ART systems, or some modifications to a well-known one.

This work is the first step in a larger project aimed at building a system that uses evolutionary computing (EC) to evolve the architecture and functionality of ART neural networks that are well suited to a specific problem or class of problems. While both EC algorithms and ART neural networks have had many successes in solving a wide variety of computational intelligence problems, the use of EC algorithms to evolve ART architectures offers great promise but remains largely unexplored. The general problem of evolving ART architectures that perform well for a given problem or class of problems is a task that requires a lot of time and effort. The work presented here starts with one step along this direction, by introducing evolution into one ART neural network architecture, and showing the viability of evolving parts of it to achieve better performance on a given class of problems. The architecture chosen for this problem is Fuzzy ART [4], which is often used for unsupervised learning, also known as clustering. However, Fuzzy ART is a basic building block for many other ART-based architectures, such as the Fuzzy ARTMAP network [7] used for supervised learning and the Biclustering ARTMAP (BARTMAP) network [8, 27] used for biclustering. This paves the way to being capable of evolving such architectures as well.

Hyper-heuristics are meta-heuristic algorithms which search algorithm-space employing primitives typically derived from existing algorithms, automating the creation of custom algorithms. The highest possible level primitives are complete algorithms, while the lowest possible level are a Turing-complete set of primitives. The former translates into automated algorithm selection, while the latter results in an intractable search of complete algorithm space (which grows exponentially with the number of operations). In order to minimize the search space, the highest primitive level which is sufficient to represent the optimal custom algorithm is ideal. Hyper-heuristics tend to have a very high upfront computational cost as each fitness evaluation of the hyper-heuristic (the meta-level) requires executing the evolved algorithm which can itself be computationally expensive; however, this computational cost can be justified through a sufficient number of repeated uses of the evolved algorithm that the cost is amortized over.

In this study, we present a system that uses a type of EC algorithm called genetic programming (GP) [15] to evolve part of the functionality of the Fuzzy ART network. In particular, we use a strongly-typed GP [18] as a hyper-heuristic to evolve the category choice function of Fuzzy ART, and test the performance of the system using benchmark datasets from the machine learning literature.

The remainder of this paper is organized as follows. Section 2 provides the background on related research needed to understand the subsequent discussion, while Section 3 examines the contributions made in recent literature related to the subject of this research. Section 4 presents the details of our contribution, and Section 5 contains the experimental setup used to conduct the experiments. The results of those experiments are reported in Section 6, followed by a discussion of those results in Section 7. Finally, Section 8 provides the conclusion of the paper.

2 BACKGROUND

This section sheds some light on the background information referred to in the remainder of this paper. The Fuzzy ART unsupervised learning algorithm, which is one of the fundamental ART-based algorithms and the subject of evolution in this work, is briefly introduced below. The reader is referred to [4] for the full details of the algorithm.

2.1 Fuzzy ART algorithm

The Fuzzy ART network [4] is shown in Figure 1. It is composed of two layers: the input presentation layer, F_1, and the category representation layer, F_2. The connections between layers have weights w_{ij}, which are adjusted during the network's operation to fit each input sample into one of the F_2 categories, or clusters. Fuzzy ART's orienting subsystem is responsible for controlling this matching process, and sending a reset signal to the F_2 layer whenever a category match does not meet the vigilance criterion, ρ.

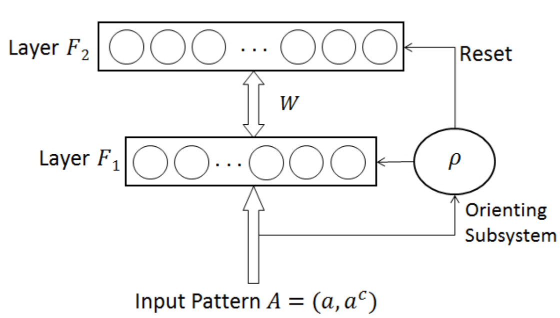

Figure 1: Architecture of the Fuzzy ART network

Before being presented to the F_1 layer, the input patterns need to be normalized and complement-coded. The normalization scales the input into the range $[0, 1]$ as follows:

$$X' = \frac{X - \min(X)}{\max(X) - \min(X)} \tag{1}$$

Let $\mathbf{a}$ denote an M-dimensional input vector. The complement coding process transforms the vector $\mathbf{a}$ into a $2M$-dimensional vector $\mathbf{A}$ such that

$$\mathbf{A} = (\mathbf{a}, \mathbf{a}^c) \tag{2}$$

where $\mathbf{a}^c$ is the complement of vector $\mathbf{a}$, computed as:

$$\mathbf{a}_i^c = 1 - \mathbf{a}_i \quad \forall i = 1, \ldots, M \tag{3}$$

This complement coding rule is intended to avoid category proliferation, and is a key characteristic of many ART neural network architectures [4].

The neurons in layer F_2 represent the prototypes of the formed categories. Initially, the Fuzzy ART network starts with a single neuron in F_2, and that neuron is in the uncommitted state. Layer F_1 has as many neurons as the number of elements in the complement-coded input vectors. Neurons are added to the F_2 layer as the learning progresses, while always maintaining one additional neuron that is in the uncommitted state.

When an input pattern is presented to the F_1 layer, the neurons in layer F_2 each compute a value for the category choice function

$$T_j = \frac{|\mathbf{A} \wedge w_j|}{\alpha + |w_j|} \tag{4}$$

where $\wedge$ represents the fuzzy AND operator, defined as

$$(\mathbf{A} \wedge w)_i \equiv \min(\mathbf{A}_i, w_i) \tag{5}$$

and where $|x|$ is the L^1 norm of x, defined as

$$|x| = \sum_i x_i \tag{6}$$

and $\alpha > 0$ is an algorithm parameter.

The winning neuron, determined using the winner-take-all rule,

$$T_J = \max_j \{T_j\} \tag{7}$$

is then activated, and its expectation is compared with the input pattern on layer F_1. The orienting subsystem then checks the match between the input pattern and the expectation, and compares it to a predetermined parameter $\rho \in [0, 1]$, known as the vigilance threshold, using the criterion

$$\rho \leq \frac{|\mathbf{A} \wedge w_J|}{|\mathbf{A}|} \tag{8}$$

If the match meets this vigilance criterion, then learning is allowed, and we apply the resonance procedure. The weights corresponding to the winning neuron are updated using the learning rule,

$$w_J(new) = \beta(\mathbf{A} \wedge w_J(old)) + (1 - \beta)w_J(old) \tag{9}$$

where the parameter $\beta \in [0, 1]$ denotes the learning rate. A rate of $\beta = 1$ represents what is known as fast learning [4].

However, if the match fails the vigilance criterion, the orienting subsystem sends a reset signal to layer F_2, which suppresses that winning neuron for the remainder of this input pattern presentation. The competition is re-run, and a new winning neuron is selected, until one of the neurons meets the vigilance criterion. If none of the committed neurons can meet this criterion, then the uncommitted neuron will be used, and its weights will be updated, making it a newly-committed neuron. When this happens, a new uncommitted neuron is added to layer F_2, to maintain the constant availability of uncommitted neurons.

This procedure is repeated for all of the input patterns, assigning each pattern to a cluster. Following that, the entire set of inputs are presented again, until input pattern category assignments no longer change between iterations.

2.2 Hyper-Heuristics

A hyper-heuristic is a meta-heuristic that searches program space to either select the most appropriate existing heuristic (called a selective hyper-heuristic) or generate an entirely new custom heuristic (called a generative hyper-heuristic) [2]. Hyper-heuristics most commonly employ GP to search a problem-specific space of algorithmic primitives [22]. GP is a field of evolutionary computation which uses a biologically inspired process to evolve a population of solutions to a given problem. In GP, the solutions being evolved take the form of programs or heuristics. One common method of representing these program solutions is through the use of parse trees [15].

3 RELATED WORK

The match-based learning ART neural network was first introduced by Carpenter and Grossberg [6] as a neural architecture for unsupervised learning of categories based on binary input patterns. It was followed by the work in [3], which introduced a supervised learning neural architecture called ARTMAP, composed of two binary ART modules connected by an inter-ART module responsible for regulating the learning relationship between patterns and labels. The Fuzzy ART neural network was introduced in [4], to extend the functionality of the unsupervised learning ART networks to analog inputs instead of binary ones. Similarly, the Fuzzy ARTMAP architecture [7] allows the supervised learning ARTMAP networks to handle analog inputs, using two Fuzzy ART modules. Many extensions to that work have been studied since (e.g., [1, 24, 26]), with successful applications in supervised, unsupervised, semi-supervised, and reinforcement learning.

The application of EC to error-based-learning Artificial Neural Networks (ANNs) has seen extensive work, both for training the network weights as well as evolving its connectivity and architecture [19, 20, 25]. Even going back as early as 1999, the paper by Yao [25] provides a literature survey with over 300 references, examining the different combinations of EC and ANNs, including using EC to evolve network weights, architectures, learning rules, and input features. It shows how the combination of EC and ANNs can lead to significant improvements in performance over the use of only one of those systems. More recent examples of this type of neuro-evolutionary approach include the use of neuro-evolutionary hyper-heuristics for constraint satisfaction problems [20], and the evolution of higher-order neural network-based classifiers for medical data classification [19].

As for the use of EC to evolve match-based-learning ART neural networks, there are very few published attempts. In [13, 14], the authors use a genetic algorithm (GA) and multi-objective optimization to evolve the parameters and neuron weights in a set of ART-based networks used for supervised learning; namely Fuzzy ARTMAP [7], Ellipsoidal ARTMAP [1], and Gaussian ARTMAP [24]. However, the weights and network parameters are the subject of evolution in this case, rather than the ARTMAP network structure itself. The authors of [16] extend that idea further by using a multi-objective memetic algorithm to train populations of ARTMAP-based classifiers, that evolves many sub-populations of various degrees of complexity in parallel to find a Pareto front of non-dominated models. Similarly, in [23], the authors use a GA as part of the Fuzzy ARTMAP network to enhance the training process by evolving the network weights, but the architecture of the network itself remains constant.

4 METHODOLOGY

For the first stage of this work, we examined the different components of the Fuzzy ART architecture, and investigated how to represent and evolve each one of its components. The candidates were the category choice function (Eq. 4) responsible for deciding the membership of each input sample into one of the categories; the winner selection procedure (Eq. 7) to decide the winning category based on this membership; the match criterion component (Eq. 8) which decides whether the winner of this procedure should be accepted or a reset signal be activated; and the learning component (Eq. 9) that is responsible for adjusting the network weights after this process is concluded and a winning category is selected. The current Fuzzy ART algorithm has predefined functionality for each of those components, which is used as the baseline for our comparison.

The category choice function (Eq. 4) was chosen as the part of the functionality to be evolved for this study. To accomplish that, strongly-typed GP was used, where the category choice function was represented using a parse-tree format in which each operator has a defined type for each one of its inputs, and a defined type for its output. Strongly-typed GP checks those types for compatibility, and maintains the validity of the generated parse tree for each individual. The choice of strongly-typed GP stems from the different types of inputs that need to be represented for the category choice function, since it has a mixture of scalar and vector inputs, and some of the operators result in scalar outputs while others produce vector outputs.

The terminals (inputs) used in the parse tree represent the different arguments for the Fuzzy ART category choice function. The list of terminals is defined in Table 1 below:

Table 1: Listing of the terminals (inputs) of the parse tree representation.

Terminal	Description
A	Input pattern vector
W_j	Category weight vector
α	Algorithm parameter, where $\alpha > 0$

A set of primitives (operators) was chosen for this study based on the operations needed to represent the predefined category choice function (Eq. 4) in addition to a set of other category choice functions discussed in [5]. Each primitive has a defined type for its output as well as a defined number of arguments, each with a defined type. The list of primitives used in this study is shown in Table 2 below:

Table 2: Listing of the primitives (operators) of the parse tree representation.

Op	Description	Inputs	Output
+	Addition	2 scalars	scalar
−	Subtraction	2 scalars	scalar
∗	Multiplication	2 scalars	scalar
/	Division	2 scalars	scalar
∧	Fuzzy AND	2 vectors	vector
∨	Fuzzy OR	2 vectors	vector
$\vert . \vert$	L^1 norm	1 vector	scalar

During the operation of the GP, each individual represents one possible category choice function for Fuzzy ART. To evaluate the fitness of each individual in the population, we create a Fuzzy ART network that uses the individual's category choice function, and train the network using a classification dataset. We then use the Adjusted Rand-Index [12] cluster validity index to measure how well the clustering that this particular Fuzzy ART found matches the already-known clustering for the given dataset.

The GP used in this experiment maintains a population size of 100 individuals, initially generated at random using an equal mix of balanced and unbalanced parse trees. The GP generates 20 new individuals at each iteration, by selecting individuals at random from the population and choosing to either do one-point tree crossover with 95% probability, or mutation with 5% probability, on each of the chosen individuals, to generate new ones. Then, the best 100 individuals of the population (based on fitness function values) are allowed to survive to the following generation. Mutation is performed by selecting a point in the parse tree at random, with uniform probability, and then replacing the subtree at that point by a new randomly-generated one. Randomly generated trees and subtrees are limited to a maximum height of 2, whereas the individuals are limited to a maximum height of 17. Each GP was allowed to run 500 generations, keeping track of the globally best performing individuals after each iteration. The parameter values for the algorithm were chosen based on ad-hoc experimentation and commonly used values.

The GP implementation relied on the use of the Distributed Evolutionary Algorithms in Python (DEAP) library [9]. DEAP is a flexible and modular EC framework that contains a set of commonly-used EC functions and algorithms, allowing for rapidly building EC systems while maintaining flexibility and room for customization. It is also compatible with parallelization mechanisms, which allowed us to use multiprocessing to improve the running time of our implementation. Additionally, functionality from the Scikit-learn machine learning library [21] was used to calculate the adjusted rand-index used in evaluating the fitness of individuals.

5 EXPERIMENTAL SETUP

The goal of the experiments in this study is to evaluate the performance of the evolved Fuzzy ART category choice functions against the predefined one. To accomplish this, we tested them using 3 classification benchmark datasets from the UCI Machine Learning Repository [17]. The details of the datasets are shown in Table 3. The selected datasets were chosen as classic benchmarks that can prove the basic viability of this line of exploration, before attempting to apply it to larger and more complex datasets. The datasets were preprocessed using Weka [11] to normalize their attribute values to the range $[0, 1]$, which is the necessary range for Fuzzy ART.

Table 3: Attributes of the classification benchmark datasets used in the experiment

Dataset	Attributes	Instances	Classes	Data Type
Glass	10	214	7	Real
Iris	4	150	3	Real
Wine	13	178	3	Integer, Real

To show evidence of the statistical accuracy of the results, a stratified 10-fold cross-validation procedure was used. Each dataset was split into 10 smaller sets (folds) of approximately equal size, and approximately the same class distribution as the full dataset. Then, a holdout procedure was used, where one of the 10 folds was taken out of the dataset, and the remaining folds were used for training. The fold that was taken out of the dataset was then used as a testing set to evaluate the performance of the system on previously-unseen data. This process was repeated 10 times, once with each one of the folds being used as a testing set, and the remaining folds being used as the training set. Both the training and testing set performance was measured for each run, then averaged across the different runs. This averaging helps guard against having one really good or really bad run, and instead provides an average performance measure with different portions of the dataset. The generation of the cross-validation folds was done using Weka [11].

The Python code used for running the experiment is publicly available at: https://github.com/islamelnabarawy/EvolvingART

For the purposes of this experiment, a basic parameter tuning procedure was done using the original Fuzzy ART category choice function by employing an evolutionary algorithm similar to the one used in the experiments. The code used for this parameter tuning procedure can be found within the code package referenced above. The Fuzzy ART parameters found to perform best for each dataset are shown in Table 4.

Table 4: Fuzzy ART parameters used in the experiment for each of the datasets

	ρ	α	β
Glass	0.65605238	0.947480219	0.907550853
Iris	0.485963005	0.918750107	0.982337104
Wine	0.375031876	0.861897945	0.979738595

A strongly-typed GP was run using the DEAP framework, and the top performing individual from each training run was subsequently used to measure both the testing set performance by running on each cross-validation fold 100 times using a different order of input presentations and averaging the results. Then, a two-tailed t-student statistical test was done with an alpha of 0.05 to measure the statistical significance of the improvements gained by using the evolved category choice functions in comparison with the original one.

6 EXPERIMENTAL RESULTS

Tables 5, 6, and 7 show the mean and standard deviation of the Adjusted Rand-Index performance results for the Glass, Iris, and Wine datasets, respectively, for the best generated category choice functions on each training fold. Evaluation was done by averaging the performance of each category choice function over each of the testing folds, and repeating this procedure for a hundred times, then calculating the mean and standard deviation. The comparison brings to light a consistent improvement over the baseline by several of the generated category choice functions for each of the datasets, which is confirmed to be statistically-significant by the results of the two-tailed t-student statistical test.

Figures 2, 3, and 4 show parse tree representations of the best performing generated category-choice function for each dataset. The figures show large and complex functions, although some of them can be manually simplified to reduce the height or span of the parse tree. It is interesting to note that the parse tree for the Glass dataset is wide but not very deep, where the Iris dataset parse tree has deeper nodes but smaller spanning subtrees, and the Wine dataset parse tree is very deep. This goes to show that the characteristics of the best performing category choice function differ according to the characteristics of the dataset being clustered.

Table 5: Adjusted Rand-Index results for the Glass dataset, averaged over 100 presentations of each of the cross-validation folds, for both the baseline and evolved functions. Two of the evolved category choice functions (highlighted in boldface) achieved statistically-significant improvement over the baseline.

Function	Avg	Stdev	*p*-value
Baseline	0.148816	0.030336	-
1	0.135092	0.028208	0.001160538
2	0.136748	0.028502	0.004350454
3	0.136049	0.027728	0.00228423
4	**0.156821**	**0.025455**	**0.045653767**
5	0.111471	0.037342	5.51869E-13
6	0.135092	0.028208	0.001160538
7	0.136049	0.027728	0.00228423
8	0.135092	0.028208	0.001160538
9	**0.176210**	**0.023265**	**1.84168E-11**
10	0.135092	0.028208	0.001160538

Table 6: Adjusted Rand-Index results for the Iris dataset, averaged over 100 presentations of each of the cross-validation folds, for both the baseline and evolved functions. Six of the evolved category choice functions (highlighted in boldface) achieved statistically-significant improvement over the baseline.

Function	Avg	Stdev	*p*-value
Baseline	0.526977	0.042583	-
1	**0.541392**	**0.030967**	**0.007026**
2	**0.541845**	**0.034582**	**0.007608**
3	0.535355	0.035196	0.132933
4	**0.544562**	**0.043101**	**0.004312**
5	**0.575092**	**0.043337**	**2.14E-13**
6	0.416604	0.067836	1.6E-30
7	0.417733	0.067386	2.73E-30
8	0.529835	0.040364	0.628563
9	**0.612434**	**0.044360**	**7.03E-31**
10	**0.552093**	**0.038352**	**2.08E-05**

Table 7: Adjusted Rand-Index results for the Wine dataset, averaged over 100 presentations of each of the cross-validation folds, for both the baseline and evolved functions. All but two of the evolved category choice functions (highlighted in boldface) achieved statistically-significant improvement over the baseline.

Function	Avg	Stdev	*p*-value
Baseline	0.133080	0.051846	-
1	**0.291386**	**0.062147**	**7.49065E-48**
2	0.084326	0.043279	1.35326E-11
3	0.023232	0.030128	3.17418E-44
4	**0.239885**	**0.064688**	**8.74047E-28**
5	**0.464734**	**0.062062**	**2.47148E-98**
6	**0.299598**	**0.063700**	**6.58629E-50**
7	**0.405670**	**0.063923**	**2.59204E-82**
8	**0.263583**	**0.062939**	**2.68425E-37**
9	**0.185648**	**0.060647**	**4.70702E-10**
10	**0.381861**	**0.070059**	**9.38528E-72**

7 DISCUSSION

The results in Section 6 show great promise towards achieving superior performance using Fuzzy ART networks by using GP to evolve the network architecture for a certain problem or class of problems. The GP generated category choice functions that can outperform the original category choice function of Fuzzy ART for the specified datasets, and could still achieve generalization by producing superior results on the testing datasets.

Table 7 shows that 8 out of 10 of the evolved category choice functions outperformed the baseline on the average testing set Adjusted Rand-Index performance for the Wine dataset. These results indicate that such improvement is possible, which is the goal this study attempted to ascertain.

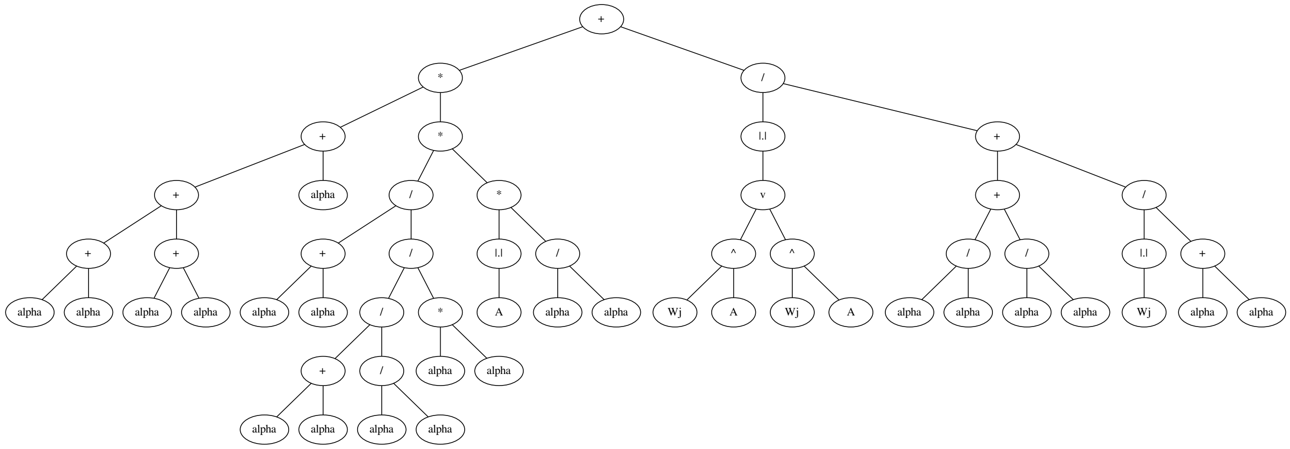

Figure 2: A parse tree of the best performing category-choice function for the Glass dataset.

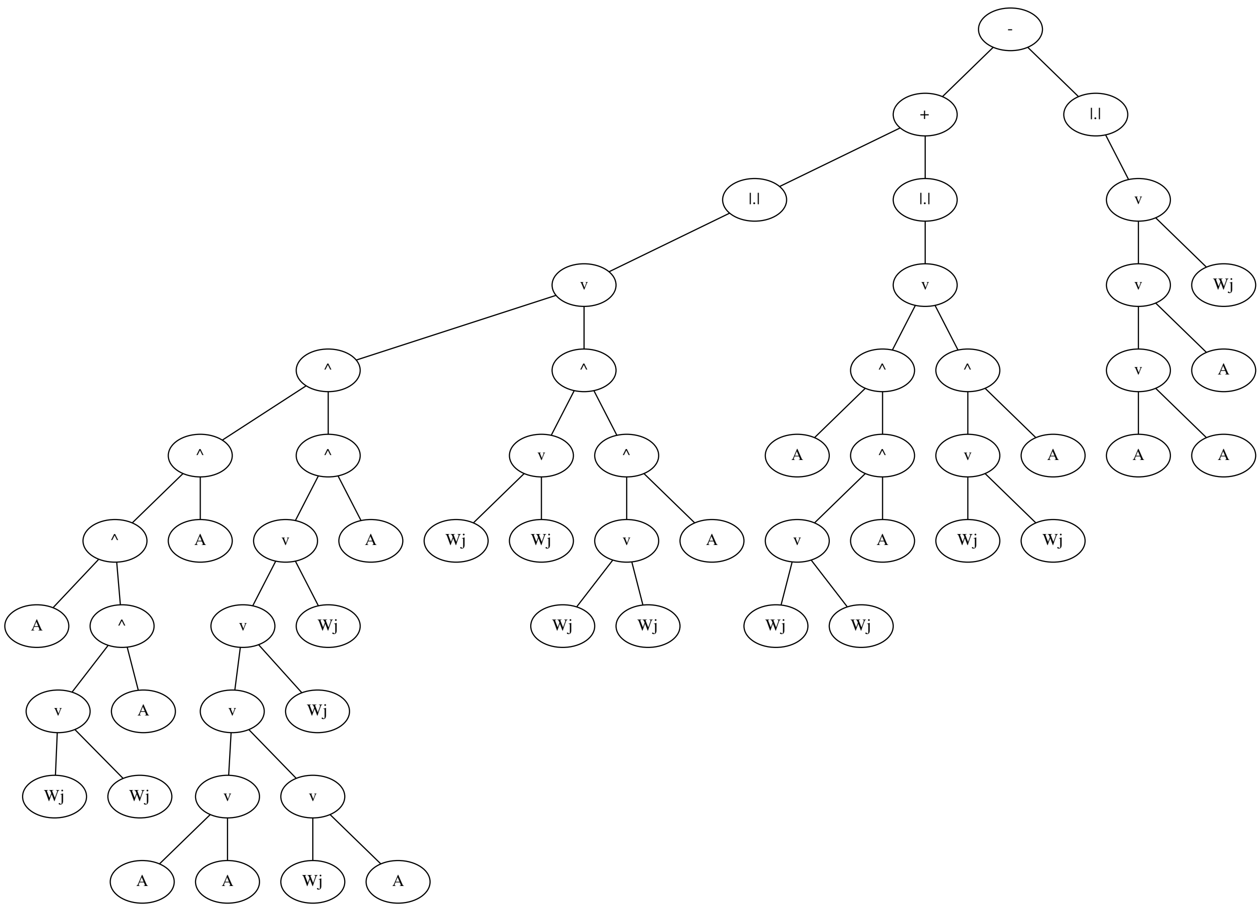

Figure 3: A parse tree of the best performing category-choice function for the Iris dataset.

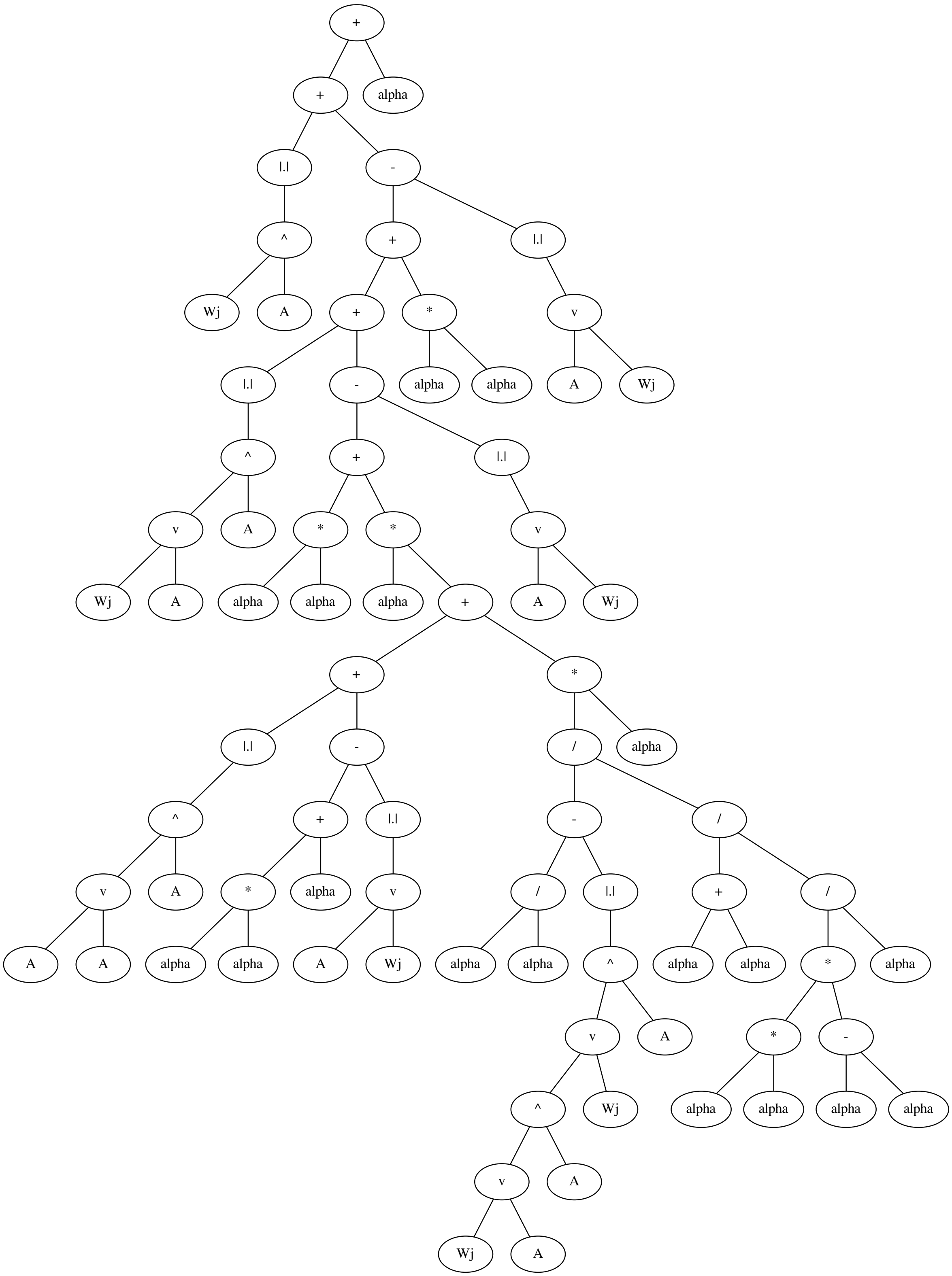

Figure 4: A parse tree of the best performing category-choice function for the Wine dataset.

Those results are interesting and warrant further investigation to extract insights about what makes some of those functions outperform the baseline for those particular datasets. Since the category choice function is essentially a type of distance metric that measures the distance between a given input pattern and a cluster, this can lead to some valuable insights for which distance measure works well for the datasets used in this study.

8 CONCLUSION

In this study, we examined the use of GP to evolve part of the functionality of the Fuzzy ART unsupervised learning neural network, to achieve better performance for specific classes of problems. The individuals in the GP each represent a variation on the category choice function of Fuzzy ART, and were used in a Fuzzy ART network, with the fitness of each individual evaluated as the Adjusted Rand-Index value calculated for a classification dataset. The experiment was conducted using a 10-fold cross-validation procedure, and the testing set results were calculated using the best performing individual from each GP run and compared with the baseline. The experiment resulted in superior, statistically-significant testing set Adjusted Rand-Index performance for the evolved category choice functions over the baseline for the chosen benchmark datasets. These results show the potential for using evolutionary computing to improve the performance of Fuzzy ART for specific domains or applications, which can lead to even further improvements in other architectures that are based on Fuzzy ART, such as Fuzzy ARTMAP or BARTMAP. The next steps in our exploration involve applying this approach to larger datasets and comparing the results to more challenging benchmarks, as well as adding a parse tree simplification routine to the process to help the interpretability of the generated expressions. Additionally, we also intend to apply this approach to more complex ART-based architectures, and test it against benchmark problems in both unsupervised and supervised learning modalities.

ACKNOWLEDGMENT

Support from the Missouri University of Science and Technology Intelligent Systems Center and the M.K. Finley Missouri Endowment is gratefully acknowledged.

REFERENCES

[1] G. C. Anagnostopoulos and M. Georgiopoulos. 2001. Ellipsoid ART and ARTMAP for incremental clustering and classification. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN '01)*, Vol. 2. 1221–1226. DOI: http://dx.doi.org/10.1109/IJCNN.2001.939535

[2] Edmund K Burke, Michel Gendreau, Matthew Hyde, Gabriela Ochoa, Graham Kendall, Ender Özcan, and Rong Qu. 2013. Hyper-heuristics : A Survey of the State of the Art. *Journal of the Operational Research Society* (2013), 1695–1724. DOI: http://dx.doi.org/10.1057/jors.2013.71

[3] GA Carpenter, S Grossberg, and JH Reynolds. 1991. ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural networks* (1991).

[4] GA Carpenter, S Grossberg, and DB Rosen. 1991. Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural networks* 4, 6 (1991), 759–771.

[5] Gail A. Carpenter and Marin N. Gjaja. 1994. Fuzzy ART Choice Functions. *Proceedings of the World Congress on Neural Networks (WCNN'94)* 1993, 12 (1994).

[6] Gail A. Carpenter and Stephen Grossberg. 1987. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing* 37, 1 (jan 1987), 54–115. DOI: http://dx.doi.org/10.1016/S0734-189X(87)80014-2

[7] Gail A. Carpenter, Stephen Grossberg, Natalya Markuzon, John H. Reynolds, and David B. Rosen. 1992. Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks* 3, 5 (sep 1992), 698–713.

[8] I Elnabarawy, D C Wunsch, and A M Abdelbar. 2016. Biclustering ARTMAP collaborative filtering recommender system. In *2016 International Joint Conference on Neural Networks (IJCNN)*. 2986–2991. DOI: http://dx.doi.org/10.1109/IJCNN.2016.7727578

[9] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. 2012. DEAP: Evolutionary Algorithms Made Easy. *Journal of Machine Learning Research* 13 (jul 2012), 2171–2175.

[10] Stephen Grossberg. 1980. How does a brain build a cognitive code? *Psychological Review* 87, 1 (jan 1980), 1–51.

[11] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter* 11, 1 (2009), 10–18.

[12] Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification* 2, 1 (1985), 193–218.

[13] A. Kaylani, M. Georgiopoulos, M. Mollaghasemi, and G.C. Anagnostopoulos. 2009. AG-ART: An adaptive approach to evolving ART architectures. *Neurocomputing* 72, 10-12 (jun 2009), 2079–2092. DOI: http://dx.doi.org/10.1016/j.neucom.2008.09.016

[14] A Kaylani, M Georgiopoulos, M Mollaghasemi, G C Anagnostopoulos, and C Sentelle. 2010. An adaptive multiobjective approach to evolving ART architectures. *IEEE Transactions on Neural Networks* 21, 4 (apr 2010), 529–50. DOI: http://dx.doi.org/10.1109/TNN.2009.2037813

[15] John R Koza. 1992. *Genetic programming: on the programming of computers by means of natural selection.* Vol. 1. MIT press.

[16] Rong Li, Timothy R. Mersch, Oriana X. Wen, Assem Kaylani, and Georgios C. Ana. 2010. Multi-objective memetic evolution of ART-based classifiers. In *Proceedings of the IEEE Congress on Evolutionary Computation.* 1–8. DOI: http://dx.doi.org/10.1109/CEC.2010.5586385

[17] M Lichman. 2013. UCI Machine Learning Repository. (2013). http://archive.ics.uci.edu/ml

[18] David J Montana. 1995. Strongly Typed Genetic Programming. *Evol. Comput.* 3, 2 (jun 1995), 199–230. DOI: http://dx.doi.org/10.1162/evco.1995.3.2.199

[19] Sanjib Kumar Nayak, Sarat Chandra Nayak, and H. S. Behera. 2016. Evolving low complex higher order neural network based classifiers for medical data classification. In *2nd International Conference on Computational Intelligence in Data Mining, ICCIDM 2015 (Advances in Intelligent Systems and Computing)*, Himansu Sekhar Behera and Durga Prasad Mohapatra (Eds.), Vol. 411. Springer India, New Delhi, 415–425. DOI: http://dx.doi.org/10.1007/978-81-322-2731-1

[20] José Carlos Ortiz-Bayliss, Hugo Terashima-Marín, and Santiago Enrique Conant-Pablos. 2015. A Neuro-evolutionary Hyper-heuristic Approach for Constraint Satisfaction Problems. *Cognitive Computation* (dec 2015), 1–13. DOI: http://dx.doi.org/10.1007/s12559-015-9368-2

[21] F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[22] Travis Bueter Sean Harris and Daniel R. Tauritz. 2015. A Comparison of Genetic Programming Variants for Hyper-Heuristics. In *Proceedings of the 17th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '15)*. 1043–1050.

[23] Shing Chiang Tan, Junzo Watada, Zuwairie Ibrahim, and Marzuki Khalid. 2015. Evolutionary Fuzzy ARTMAP neural networks for classification of semiconductor defects. *IEEE Transactions on Neural Networks and Learning Systems* 26, 5 (2015), 933–50. DOI: http://dx.doi.org/10.1109/TNNLS.2014.2329097

[24] James R Williamson. 1996. Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps. *Neural networks* 9, 5 (1996), 881–897.

[25] Xin Yao. 1999. Evolving artificial neural networks. *Proc. IEEE* 87, 9 (sep 1999), 1423–1447. DOI: http://dx.doi.org/10.1109/5.784219 arXiv:1108.1530

[26] Rui Xu, Georgios C. Anagnostopoulos, and Donald C. Wunsch. 2007. Multiclass Cancer Classification Using Semisupervised Ellipsoid ARTMAP and Particle Swarm Optimization with Gene Expression Data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 4, 1 (jan 2007), 65–77. DOI: http://dx.doi.org/10.1109/TCBB.2007.1009

[27] Rui Xu and Donald C Wunsch II. 2011. BARTMAP: A viable structure for biclustering. *Neural Networks* 24, 7 (2011), 709–716. DOI: http://dx.doi.org/10.1016/j.neunet.2011.03.020