# A Fast Interpretable Fuzzy Tree Learner

Javier Fumanal-Idocin
*School of Computer Science and Electronic Engineering*
*University of Essex*
Colchester, United Kingdom
j.fumanal-idocin@essex.ac.uk

Raquel Fernandez-Peralta
*Department of Mathematics*
*Slovak Academy of Sciences*
Bratislava, Slovakia
raquel.fernandez@mat.savba.sk

Javier Andreu-Perez
*School of Computer Science and Electronic Engineering*
*University of Essex*
Colchester, United Kingdom
j.andreu-perez@essex.ac.uk

***Abstract*—Fuzzy rule-based systems have been mostly used in interpretable decision-making because of their interpretable linguistic rules. However, interpretability requires both sensible linguistic partitions and small rule base sizes, which are not guaranteed by many existing fuzzy rule-mining algorithms. Evolutionary approaches can produce high-quality models but suffer from prohibitive computational costs, while neural-based methods, such as ANFIS, have problems retaining linguistic interpretations. In this work, we propose an adaptation of classical tree-based splitting algorithms from crisp rules to fuzzy trees, combining the computational efficiency of greedy algorithms with the interpretability advantages of fuzzy logic. This approach achieves interpretable linguistic partitions and substantially improves running time compared to evolutionary-based approaches while maintaining competitive predictive performance. Our experiments on tabular classification benchmarks prove that our method achieves comparable accuracy to state-of-the-art fuzzy classifiers, while incurring significantly lower computational costs and producing more interpretable rule bases with constrained complexity. Code will be available upon publication.**

***Index Terms*—Explainable Artificial Intelligence, Fuzzy Logic, Classification, Rule Mining**

## I. Introduction

Fuzzy rule-based systems have been popular for explainable artificial intelligence (XAI) as they rely on logic-based reasoning that humans easily understand [1]. However, this explainability is limited by the complexity of the system. For example, a system composed of hundreds of rules is much harder to understand than one with only a handful [2]. Therefore, besides the use of highly performant classifiers such as FURIA [3], there has also been particular research interest in a subset of fuzzy rule-based classifiers (FRBCs) that constrain the resulting rule base according to the end user's desired complexity [4].

Two main approaches have been proposed to address this challenge. Architecturally constrained methods, such as Adaptive Network-based Fuzzy Inference System (ANFIS) [5] and Fuzzy Rule Reasoner (FRR) [6], use neural architectures where the number of rules and conditions is pre-specified. Evolutionary optimization approaches [7] can incorporate complexity constraints into the optimization process, just as gradient-based methods enforce sparsity through loss functions [8]. However, this approach has limitations: constraints are often implemented as soft preferences that may still result in complex models, and even when successful, the computational cost typically exceeds that of standard classifiers, discouraging practitioners from using FRBCs.

The most popular method for performing crisp rule-based classification has been to use tree structures with Gini-conditioned splits. This has provided a computationally efficient and accurate way to perform rule learning, establishing CART [9] as a widely adopted classification algorithm. However, CART has significant limitations from an interpretability perspective. First, the number of leaf nodes can be quite high, making the overall decision structure difficult to understand. Second, CART uses hard splits that create arbitrary boundaries (e.g., "age $< 45.7$"), which do not reflect the gradual transitions often present in human reasoning. Finally, CART provides no mechanism for incorporating existing expert knowledge into the learning process, potentially producing rules that conflict with established domain knowledge or are difficult for practitioners to trust. This leaves room for opportunity for fuzzy systems, where conditions can be defined to take into account the intuitions of practitioners through meaningful linguistic partitions [10], and can potentially recover complex decision surfaces using fewer rules [11]. However, the task of finding optimal fuzzy partitions while keeping interpretability is not trivial. Fuzzy systems that rely on predefined fuzzy partitions are computationally efficient, but this strategy may produce suboptimal decision boundaries when the class distribution does not align well with these pre-existing semantics.

Motivated by these limitations, this paper proposes a fuzzy tree learner algorithm with a greedy approach, designed for computational efficiency. The algorithm is designed to be competitive against straightforward genetic approaches while maintaining good interpretability by imposing strict constraints on the number of rules and conditions. We also propose a lightweight optimization framework that tunes fuzzy partition parameters to maximize class separability while maintaining computational tractability and interpretability guarantees. Unlike evolutionary approaches that require thousands of fitness evaluations, our method employs efficient search strategies to keep computational complexity low.

## II. Background and Related Work

### A. Complexity Accuracy Trade-off in rule-based learning

Rule-based learning methods need to balance model complexity and predictive accuracy [1]. As the number of rules

and conditions increases, the model can capture more specific patterns in the data, typically leading to improved classification performance. However, this comes at the cost of interpretability: a rule base with hundreds of rules becomes essentially a black box, defeating the primary purpose of using rule-based systems for explainable AI [12]. This trade-off is particularly critical in domains such as healthcare [13], finance [14], and legal decision-making, where stakeholders must not only trust the model's predictions but also understand the reasoning behind them [15]. Consequently, finding the optimal balance between accuracy and complexity remains a central challenge in rule-based machine learning [16].

### B. Different crisp and fuzzy rule-learning strategies

Classical crisp tree-based methods include CART [9] and C4.5 [17]. More recent crisp approaches leverage gradient-based learning to improve scalability [18]. In parallel, Bayesian crisp rule-learning methods have been proposed to control model complexity by minimizing the size of the induced rule base while providing theoretical guarantees [19].

For fuzzy systems, gradient-based approaches have likewise been widely adopted. The ANFIS [5] is perhaps the most well-known example, combining neural network learning capabilities with fuzzy reasoning. ANFIS employs a hybrid learning algorithm that uses backpropagation to tune membership function parameters in the premise part and least-squares estimation for the consequent parameters. However, this fine-tuning comes at the cost of interpretability: the learned membership functions often lose their linguistic meaning, with overlapping or irregular shapes. More recent advances in gradient-based fuzzy learning use classic Mamdani fuzzy inference [6] with a neural architecture that supports rule-learning without the curse of dimensionality and retains interpretability of linguistic partitions. Evolutionary approaches can simultaneously optimise rule structure, membership function parameters, and the number of rules [7]. However, the computational cost remains prohibitive for many applications, as evolutionary algorithms typically require thousands of fitness evaluations.

## III. Greedy Fuzzy Rule Tree Induction

We propose a novel fuzzy tree learning algorithm that efficiently constructs interpretable rule bases through greedy condition selection from pre-defined linguistic partitions. Unlike traditional decision tree algorithms that search for optimal split points in continuous feature spaces, our approach leverages existing fuzzy membership functions to build rules with natural linguistic interpretability (e.g., "IF 'Age' is *High* AND 'Income' is *Medium*..."). So, as long as the linguistic partitions are interpretable by the user, the resulting rules should be interpretable as well.

The key main characteristics of our approach compared to other fuzzy rule learners are:

1) Greedy Condition Search: At each node, the algorithm performs a heuristic search over all available fuzzy conditions, selecting the single condition that maximizes impurity reduction. This greedy strategy achieves computational efficiency comparable to classical tree learners, like CART.
2) Conjunctive Rule Building: Rules are constructed incrementally by adding conditions to existing nodes. Each path from root to leaf represents a complete fuzzy rule of the form: IF ($X_1$ is $A_1$) AND ($X_2$ is $A_2$) AND ... THEN Class. Internal nodes can also be considered if leaves do not fire significantly.
3) Multi-way Branching: Unlike binary tree algorithms that partition data into two subsets at each split, our approach uses natural multi-way splits using the given fuzzy partitions. A single node can spawn multiple children, each corresponding to a different linguistic value of the selected feature. We do not use negation conditions, as they significantly hinder rule interpretability.

### A. Fuzzy Split Quality Measure

To evaluate the quality of fuzzy conditions, we extend classical impurity metrics to handle fuzzy memberships. For a given rule $R$ and dataset $D$, each instance $x_i \in D$ has a membership degree $\mu_R(x_i) \in [0, 1]$ indicating the extent to which it satisfies $R$.

The fuzzy Gini impurity for a node with rule $R$ is computed as [20]:

$$Gini_{fuz}(R) = 1 - \sum_{c \in Classes} \left( \frac{\sum_{i:y_i=c} \mu_R(x_i)}{\sum_i \mu_R(x_i)} \right)^2 \quad (1)$$

where $y_i$ is the class label of instance $x_i$. This formulation weights each instance's contribution to class proportions by its membership degree to the rule. The impurity gain when adding a new condition ($X_j$ is $A_j$) to rule $R$ is:

$$Gain(R, X_j, A_j) = Gini_{fuz}(R) - Gini_{fuz}(R \wedge (X_j \text{ is } A_j)) \quad (2)$$

### B. Algorithm Description

Algorithm 1 details the core tree induction procedure. The algorithm operates recursively, building the tree in a depth-first manner. At each node:

1) Termination Check (lines 4-6): If stopping criteria are met (e.g., minimum membership threshold, maximum depth, class purity), the node becomes a leaf and its predictions correspond to its confidence for each class.
2) Evaluate Existing Children (lines 9-15): For nodes with existing children, the algorithm considers if further searching in those children would improve the partition.
3) Search New Conditions (lines 16-27): The algorithm evaluates all valid fuzzy conditions from unused features. For each feature $F$ and each linguistic term $A$ in its partition, it computes the impurity gain of adding ($F$ is $A$) to the current rule. The condition $(F, A)$ is valid if feature $F$ has not been used in the path from root to current node, preventing contradictory conditions like $(Age \text{ is } High) \wedge (Age \text{ is } Low)$.
4) New Node Selection (lines 21-24): Among all candidate conditions (both from existing children and new

features), the algorithm selects the single best condition that maximizes impurity gain.

5) Expansion (lines 28-30): If the best gain exceeds threshold $\theta$, a new child node is created with the extended rule, and the procedure recurses. The threshold $\theta$ serves as a pruning mechanism, preventing splits that provide minimal improvement.

**Algorithm 1** Greedy Fuzzy Rule Tree Induction

```
1: procedure GrowTree(Node, Data, FuzzyPartitions)
2:     Rule ← Node.Rule                          ▷ Current conjunctive fuzzy rule
3:     Memberships ← ComputeMemberships(Data, Rule)
4:     if StoppingCriterion(Node, Data, Memberships) then
5:         Node.Class ← MajorityClass(Data, Memberships)
6:         return
7:     end if
8:     BestCondition ← ∅
9:     BestGain ← −∞
10:    for each child C in Node.Children do
11:        gain, condition ← EvaluateBestSplit(Rule, C, Data)
12:        if gain > BestGain then
13:            BestGain ← gain
14:            BestCondition ← condition
15:        end if
16:    end for
17:    for each feature F in Features do
18:        if F not used in path to Node then      ▷ Avoid contradictions
19:            for each linguistic term A in FuzzyPartitions[F] do
20:                NewRule ← Rule ∧ (F is A)
21:                Gain ← ImpurityGain(Data, Rule, NewRule)
22:                if Gain > BestGain then
23:                    BestGain ← Gain
24:                    BestCondition ← (F, A)
25:                end if
26:            end for
27:        end if
28:    end for
29:    if BestCondition ≠ ∅ and BestGain > θ then
30:        ChildNode ← CreateChildNode(BestCondition)
31:        GrowTree(ChildNode, Data, FuzzyPartitions)
32:    end if
33: end procedure
```

### C. Computational Complexity Analysis

The proposed greedy fuzzy tree learning algorithm achieves a time complexity of $\mathcal{O}(n \cdot m \cdot k \cdot d)$, where $n$ is the number of training instances, $m$ is the number of features, $k$ is the average number of linguistic terms per feature, and $d$ is the maximum tree depth. At each node in the tree, the algorithm evaluates all possible fuzzy conditions across $m$ features and their corresponding $k$ linguistic partitions, computing membership degrees and weighted Gini impurity for $n$ instances, resulting in $\mathcal{O}(n \cdot m \cdot k)$ operations per node. The greedy construction strategy makes one pass through the candidate splits without backtracking, limiting the total number of node evaluations to the tree depth $d$. This represents a substantial improvement over evolutionary approaches that typically require $\mathcal{O}(p \cdot g \cdot n \cdot m \cdot r)$ operations for $p$ population size, $g$ generations, and $r$ rules, often resulting in thousands of complete model evaluations.

## IV. Fuzzy Partition Optimization

### A. Partition Representation

Each linguistic term is represented by a trapezoidal membership function $\mu(x; a, b, c, d)$, defined by four parameters where $a \leq b \leq c \leq d$:

$$\mu(x; a, b, c, d) = \begin{cases} 0 & \text{if } x < a, \\ \frac{x-a}{b-a} & \text{if } a \leq x < b, \\ 1 & \text{if } b \leq x \leq c, \\ \frac{d-x}{d-c} & \text{if } c < x \leq d, \\ 0 & \text{if } x > d. \end{cases} \tag{3}$$

The key challenge in optimizing these parameters is ensuring that validity constraints are maintained throughout optimization: each trapezoid must satisfy $a \leq b \leq c \leq d$, and the trapezoids must maintain interpretable ordering (*Low* $<$ *Medium* $<$ *High*).

To enable unconstrained optimization while guaranteeing validity by construction, we developed a novel interleaved encoding scheme. Rather than directly optimizing the trapezoid parameters, we encode them as incremental differences in a carefully chosen order. The encoding interleaves parameters from adjacent trapezoids, ensuring that touching points between consecutive linguistic terms are encoded together.

By representing each position (except the first) as the positive difference from the previous accumulated value, this encoding ensures monotonicity, validity constraints, interpretable ordering, and full domain coverage by construction, eliminating the need for constraint handling during optimization. The decoding process accumulates these increments, extracts parameters in the interleaved order, and normalizes to the feature domain.

*1) Optimization Objective:* We optimize partitions to maximize the *separability index*, which measures how well each fuzzy partition concentrates samples of the same class:

$$SI = \sum_{v=1}^{V} \sum_{c=1}^{C} \frac{\left[\sum_{i=1}^{N} \mu_v(x_i) \cdot \mathbb{I}(y_i = c)\right]^2}{\sum_{i=1}^{N} \mu_v(x_i)}, \tag{4}$$

where $v$ indexes linguistic terms, $c$ indexes class labels, $\mu_v(x_i)$ is the membership degree of sample $i$ to term $v$, and $\mathbb{I}(y_i = c)$ is the indicator function for class membership. Higher values indicate better class separation in the fuzzy space, as partitions that concentrate same-class samples together yield larger numerators while diverse class mixing yields smaller values.

### B. Search Strategy and Integration with Tree Learning

We optimize one parameter at a time cyclically, using progressively smaller step sizes: $\{0.10, 0.05, 0.02\} \times (x_{\max} - x_{\min})$. At each iteration, the algorithm tests both directions ($\pm$) for each of the three critical parameters and accepts improvements. The algorithm converges when no improvement is found with the smallest step size.

The optimized partitions are generated once during the training phase using only the training data within each cross-validation fold. The tree learning procedure then uses these

TABLE I: Characteristics of datasets used in our experiments.

| Dataset | Instances | Features | Classes | Domain |
|---|---|---|---|---|
| Appendicitis | 106 | 7 | 2 | Medical |
| Australian | 690 | 14 | 2 | Financial |
| Dermatology | 366 | 34 | 6 | Medical |
| Hepatitis | 155 | 19 | 2 | Medical |
| Pima | 768 | 8 | 2 | Medical |
| Ring | 7,400 | 20 | 2 | Synthetic |
| Saheart | 462 | 9 | 2 | Medical |
| Spambase | 4,601 | 57 | 2 | Text/Email |
| Wine | 178 | 13 | 3 | Chemical |
| Zoo | 101 | 16 | 7 | Biology |

optimized partitions in place of the default quantile-based partitions, with all other algorithmic components remaining unchanged.

## V. Experimental Setup

### A. Datasets and evaluation

We selected 10 benchmark classification datasets from the UCI Machine Learning Repository [21], chosen to represent diverse application domains and exhibit varying characteristics in terms of size and dimensionality. Table I summarizes the properties of each dataset. Results are reported using 5-fold cross-validation, and all features are normalized by subtracting their mean and dividing by the standard deviation. Results are reported using the classification accuracy metric. To measure complexity in the resulting rule bases we use the average rule size, the number of rules per classifier, and the total number of conditions in the rule base.

### B. Baseline Methods

We compared our approach against five established rule-learning methods representing different algorithmic paradigms:

*1) Crisp Rule-Based Classifiers:*

- Classification and Regression Trees [9]: the foundational decision tree algorithm that uses Gini impurity for split selection. We used the scikit-learn [22] implementation with cost complexity pruning parameters tried at $\{0.0, 0.001, 0.003\}$.
- C4.5 [17]: An extension of ID3 that employs information gain ratio for split selection and includes pessimistic pruning. Implemented via scikit-learn with maximum depths of $\{5, 10\}$.
- RIPPER [23]: Repeated Incremental Pruning to Produce Error Reduction, a separate-and-conquer rule learning algorithm that constructs ordered rule lists. We used the `wittgenstein` Python library implementation with default parameters.

*2) Fuzzy Rule-Based Classifiers:*

- FRR [6]: Fuzzy Rule Reasoner, a recent gradient-based approach that learns Mamdani fuzzy rule bases while preserving linguistic interpretability through a neural architecture that avoids the curse of dimensionality and non-differentiability issues.

TABLE II: Classification accuracy (%) across all datasets using 5-fold cross-validation. FGRT* uses optimized separability-based partitions.

| Dataset | FGRT | FGRT* | FRR | CART | C4.5 | RIPPER | FGA |
|---|---|---|---|---|---|---|---|
| Appendicitis | 80.22 | 81.17 | 86.36 | 72.55 | 74.50 | 84.03 | 90.91 |
| Australian | 86.09 | 85.65 | 83.91 | 81.16 | 83.04 | 83.04 | 85.51 |
| Dermatology | 68.45 | 66.76 | 80.05 | 86.87 | 86.87 | 84.13 | 50.00 |
| Hepatitis | 83.75 | 85.00 | 90.00 | 83.75 | 90.00 | 66.15 | 81.25 |
| Pima | 65.10 | 65.10 | 72.59 | 69.15 | 69.41 | 70.08 | 74.68 |
| Ring | 59.86 | 83.70 | 84.41 | 88.15 | 88.82 | 60.55 | 68.58 |
| Saheart | 65.37 | 65.37 | 70.96 | 60.37 | 84.74 | 64.05 | 66.67 |
| Spambase | 88.71 | 90.12 | 78.78 | 90.80 | 92.23 | 87.33 | 88.48 |
| Wine | 94.41 | 96.62 | 97.22 | 89.87 | 92.08 | 86.20 | 91.67 |
| Zoo | 77.24 | 77.24 | 66.85 | 95.05 | 94.05 | 83.75 | 71.43 |
| **Mean** | 76.92 | 79.67 | 81.11 | 81.77 | 83.57 | 76.93 | 76.91 |

- Genetic Fuzzy Classifier (FGA) [7]: An evolutionary algorithm-based fuzzy classifier, implemented using the Ex-Fuzzy library.

### C. Fuzzy Partition Configuration

To ensure fair comparison across all fuzzy methods, we employed consistent fuzzy partitions throughout the experiments, following the quantile-based approach described in the FRR methodology [6]. This ensures that differences in performance come from the rule-learning strategy rather than fuzzy partition choices. For that, each continuous feature was partitioned using three trapezoidal membership functions with linguistic labels *Low*, *Medium*, and *High*.

The membership functions are defined using quantile-based parameters:

- Low: Trapezoid $(Q_0, Q_0, Q_1, Q_2)$
- Medium: Trapezoid $(Q_1, (Q_1+Q_2)/2, (Q_2+Q_3)/2, Q_3)$
- High: Trapezoid $(Q_2, Q_3, Q_4, Q_4)$

where $Q_i$ represents the $i$-th quantile at percentages $\{0, 20, 40, 60, 80, 100\}$ of the training data.

## VI. Experimental Results

### A. Classification Performance

Table II presents the classification accuracy results comparing our proposed Fuzzy Greedy Rule Tree (FGRT) with established baseline methods across 10 benchmark datasets using 5-fold cross-validation. FGRT with default quantile partitions achieves 76.92% mean accuracy, which is competitive with FGA (76.91%) and matches RIPPER (76.93%), both established rule-learning algorithms. While FGRT does not reach the accuracy levels of CART (81.77%) or C4.5 (83.57%), this 4.85 percentage point gap is expected given that FGRT operates under stricter interpretability constraints—using predefined linguistic partitions rather than optimizing split points, and maintaining interpretable fuzzy rules rather than creating potentially hundreds of crisp leaf nodes.

Notably, FGRT achieves the highest accuracy among all methods on australian (86.09%) and outperforms crisp learners on wine (94.41%), demonstrating that fuzzy partitions can effectively capture decision boundaries. Compared to FRR (81.11%), FGRT's lower accuracy represents a reasonable

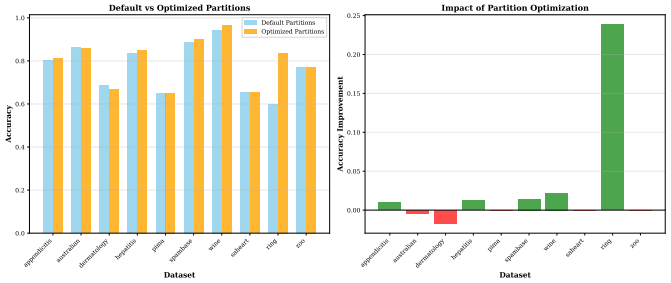


Fig. 1: Performance improvements using pre-configured or optimized fuzzy sets. Left: Accuracy comparison showing default (blue) versus optimized (orange) partitions. Right: absolute accuracy improvements.

TABLE III: Average model complexity metrics across all 10 datasets. Lower values indicate simpler models.

| Metric | FGRT | FRR | CART | C4.5 | RIPPER | FGA |
|---|---|---|---|---|---|---|
| Num. of Rules | 10.40 | 13.77 | 39.75 | 131.92 | 16.04 | 7.12 |
| Conditions/Rule | 2.28 | 1.94 | 5.75 | 8.10 | 1.96 | 2.23 |
| Rule base Size | 23.71 | 26.71 | 228.56 | 1068.55 | 31.43 | 15.87 |

trade-off for its single-pass greedy construction that completes in seconds on CPU hardware.

Compared to FRR, which achieves 81.11% mean accuracy through gradient-based optimization, FGRT's 4.19 percentage point lower accuracy represents a reasonable trade-off considering the algorithmic differences: FRR requires multiple training epochs, careful hyperparameter tuning, GPU acceleration, and backpropagation through fuzzy inference layers, while FGRT uses a single-pass greedy construction that completes in seconds on standard CPU hardware.

### B. Impact of Partition Optimization

We use our proposed fuzzy partition optimization to investigate whether fuzzy partition quality limits FGRT's performance. We denote this variant as FGRT*. Figure 1 visualizes the impact of this optimization across all datasets, comparing default quantile-based partitions with optimized separability-based partitions.

As shown in Table II, FGRT* with optimized partitions achieves 79.67% mean accuracy, representing a +2.75 percentage point improvement over the default configuration. This optimization improved accuracy on 5 out of 10 datasets, with three datasets (*pima*, *saheart*, *zoo*) showing no change, indicating that default quantile partitions were already well-aligned with their class structures. Two datasets (dermatology, australian) show slight decreases, suggesting minor overfitting when default partitions were already near-optimal.

### C. Model Complexity

Beyond predictive accuracy, interpretability requires models with manageable structural complexity. Table III presents complexity metrics averaged across all 10 datasets, quantifying the size and structure of the learned rule bases.

FGRT produces models with an average of 10.40 rules and 2.28 conditions per rule, yielding a total rule base size (rules × conditions/rule) of 23.71. This complexity is comparable to FRR (26.71) and RIPPER (31.43), and substantially smaller than CART (228.56) and especially C4.5 (1068.55). While FGA produces the smallest rule bases (15.87), FGRT's slightly larger models achieve competitive or superior accuracy on most datasets (7 out of 10 datasets where FGRT* matches or exceeds FGA performance).

### D. Hyperparameter study

Figure 2 presents an ablation study examining the sensitivity of FGRT to its three main hyperparameters across all datasets. On the left, it illustrates the evolution of FGRT complexity and performance. Performance clearly improves as the number of rules increases from 5 to 15. Beyond this point, however, we observe diminishing returns, with performance even decreasing at 30 rules. In the center, coverage threshold exhibits minimal impact on accuracy, with performance remaining stable across the tested range. Finally, the minimum improvement threshold shows relatively flat performance across values from 0 to 0.05, indicating that FGRT is robust to this parameter.

### E. Running times

Figure 3 shows training time scaling with dataset size (10 features, 100–5,000 samples). FGRT demonstrates near-constant training times ($0.47 \pm 0.29$ to $1.08 \pm 0.01$ seconds), growing only $2.3\times$ over a $50\times$ increase in samples. Genetic optimization requires 2–3 orders of magnitude longer ($6.09 \pm 0.05$ to $20.62 \pm 0.12$ seconds), being approximately $19\times$ slower at 5,000 samples while achieving comparable accuracy. CART and C4.5 are faster due to highly optimized implementations in mature libraries, but sacrifice the interpretability advantages of linguistic fuzzy rules.

## VII. Conclusions

In this work, we propose a greedy method for training fuzzy rule-based classifiers using linguistic partitions and fuzzy Gini impurity reduction. We introduce a lightweight partition optimization framework that tunes membership functions to maximize class separability, using a novel encoding scheme that guarantees validity by construction. Our approach achieves computational efficiency, interpretability preservation, and adaptive partition improvement.

Experiments on 10 benchmark datasets show that our proposed method, FGRT, achieves 76.92% mean accuracy with pre-established fuzzy partitions, competitive with genetic search and with established methods like RIPPER (76.93%). With optimized partitions, accuracy improves to an average 79.67%, approaching the FRR (81.11%) while maintaining low computational complexity. FGRT also produces models with only 10.40 rules on average—approximately 4-13× fewer than CART (39.75) and C4.5 (131.92), which makes it a compelling alternative for practitioners.

For future research, we plan to explore joint optimization across non-independent features and alternative optimization

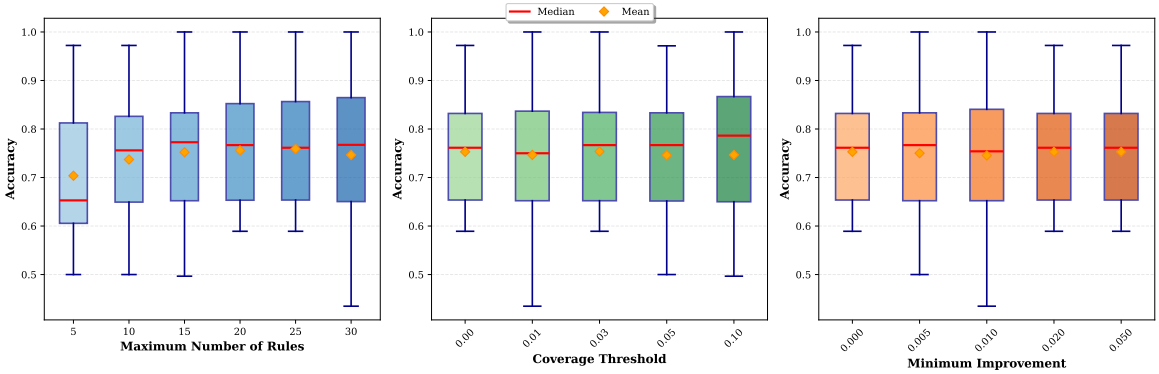


Fig. 2: Averaged performance for all datasets using different configuration hyperparameters.


Fig. 3: Comparison of running times between different rule mining approaches, in synthetic datasets from 100 to 5000 samples.

objectives to incorporate complexity penalties in the FGRT construction.

## VIII. Acknowledgement

This research and Javier Fumanal-Idocin were supported by EU Horizon Europe under the Marie Skłodowska-Curie COFUND grant No 101081327 YUFE4Postdocs. Raquel Fernandez-Peralta is funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I03-03-V04- 00557.

## References

[1] C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova, and C. Zhong, "Interpretable machine learning: Fundamental principles and 10 grand challenges," *Statistic Surveys*, vol. 16, pp. 1–85, 2022.

[2] M. J. Gacto, R. Alcalá, and F. Herrera, "Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures," *Information Sciences*, vol. 181, no. 20, pp. 4340–4360, 2011.

[3] J. Hühn and E. Hüllermeier, "Furia: an algorithm for unordered fuzzy rule induction," *Data Mining and Knowledge Discovery*, vol. 19, pp. 293–319, 2009.

[4] J. Fumanal-Idocin, J. Andreu-Perez, O. Cord, H. Hagras, H. Bustince, *et al.*, "Artxai: Explainable artificial intelligence curates deep representation learning for artistic images using fuzzy techniques," *IEEE Transactions on Fuzzy Systems*, 2023.

[5] J.-S. Jang, "Anfis: adaptive-network-based fuzzy inference system," *IEEE transactions on systems, man, and cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[6] J. Fumanal-Idocin, R. Fernandez-Peralta, and J. Andreu-Perez, "Compact rule-based classifier learning via gradient descent," *arXiv preprint arXiv:2502.01375*, 2025.

[7] J. Fumanal Idocin and J. Andreu-Perez, "Ex-fuzzy: A library for symbolic explainable ai through fuzzy logic programming," *Neurocomputing*, 2024.

[8] H. McTavish, C. Zhong, R. Achermann, I. Karimalis, J. Chen, C. Rudin, and M. Seltzer, "Fast sparse decision tree optimization via reference ensembles," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, pp. 9604–9613, 2022.

[9] L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Chapman and Hall/CRC, 2017.

[10] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning," *Information sciences*, vol. 8, no. 3, pp. 199–249, 1975.

[11] R. Fernandez-Peralta, J. Fumanal-Idocin, and J. Andreu-Perez, "Crisp complexity of fuzzy classifiers," in *2025 IEEE International Conference on Fuzzy Systems (FUZZ)*, pp. 1–6, 2025.

[12] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature machine intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[13] M. Chua, D. Kim, J. Choi, N. G. Lee, V. Deshpande, J. Schwab, M. H. Lev, R. G. Gonzalez, M. S. Gee, and S. Do, "Tackling prediction uncertainty in machine learning for healthcare," *Nature Biomedical Engineering*, vol. 7, no. 6, pp. 711–718, 2023.

[14] G. Erion, J. D. Janizek, C. Hudelson, R. B. Utarnachitt, A. M. McCoy, M. R. Sayre, N. J. White, and S.-I. Lee, "A cost-aware framework for the development of ai models for healthcare applications," *Nature Biomedical Engineering*, vol. 6, no. 12, pp. 1384–1398, 2022.

[15] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.," *Queue*, vol. 16, no. 3, pp. 31–57, 2018.

[16] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

[17] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[18] Z. Wang, W. Zhang, N. Liu, and J. Wang, "Learning interpretable rules for scalable data representation and classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[19] T. Wang, C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl, and P. MacNeille, "A bayesian framework for learning rule sets for interpretable classification," *Journal of Machine Learning Research*, vol. 18, no. 70, pp. 1–37, 2017.

[20] B. Belhadj, F. Kaabi, and M. Bouanani, "Fuzzy version of gini's index," *Social Indicators Research*, vol. 157, no. 3, pp. 1079–1087, 2021.

[21] M. Kelly, R. Longjohn, and K. Nottingham, "The uci machine learning repository."

[22] 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, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[23] W. W. Cohen *et al.*, "Fast effective rule induction," in *Proceedings of the twelfth international conference on machine learning*, pp. 115–123, 1995.