

Cultivating Archipelago of Forests: Evolving Robust Decision Trees Through Island Coevolution

Adam Żychowski¹, Andrew Perrault², Jacek Mańdziuk^{1, 3, 4}

¹Faculty of Mathematics and Information Science, Warsaw University of Technology

²Department of Computer Science and Engineering, The Ohio State University

³Faculty of Computer Science, AGH University of Krakow

⁴Center of Excellence in Artificial Intelligence, AGH University of Krakow
adam.zychowski@pw.edu.pl, perrault.17@osu.edu, jacek.mandziuk@pw.edu.pl

Abstract

Decision trees are widely used in machine learning due to their simplicity and interpretability, but they often lack robustness to adversarial attacks and data perturbations. The paper proposes a novel island-based coevolutionary algorithm (ICoEvoRDF) for constructing robust decision tree ensembles. The algorithm operates on multiple islands, each containing populations of decision trees and adversarial perturbations. The populations on each island evolve independently, with periodic migration of top-performing decision trees between islands. This approach fosters diversity and enhances the exploration of the solution space, leading to more robust and accurate decision tree ensembles. ICoEvoRDF utilizes a popular game theory concept of mixed Nash equilibrium for ensemble weighting, which further leads to improvement in results. ICoEvoRDF is evaluated on 20 benchmark datasets, demonstrating its superior performance compared to state-of-the-art methods in optimizing both adversarial accuracy and minimax regret. The flexibility of ICoEvoRDF allows for the integration of decision trees from various existing methods, providing a unified framework for combining diverse solutions. Our approach offers a promising direction for developing robust and interpretable machine learning models.

1 Introduction

Decision trees (DTs) are a popular tool in the field of machine learning due to their inherent simplicity, interpretability, and efficiency. Their capacity to model complex decision boundaries in an understandable manner has made them a popular choice for both academic research and practical applications (Rokach and Maimon 2005). However, single DTs often struggle with issues such as overfitting, lack of robustness to noise, and limited generalization capabilities. To mitigate these problems, ensembles of DTs (Banfield et al. 2006), such as random forests, have been developed that aggregate multiple trees to improve overall performance.

Despite these advancements in constructing robust and accurate DT ensembles, several challenges remain. Traditional methods for creating robust decision trees (RDTs) and robust decision forests (RDFs) often rely on only one particular robustness metric, limiting the practical applicability of these methods in real-world scenarios. Balancing multiple

objectives, such as accuracy, robustness, and computational efficiency is often not adequately addressed. Existing methods also often struggle to maintain diversity within the ensemble, leading to reduced effectiveness.

We aim to develop better methods for building and combining decision tree ensembles (RDFs). Evolutionary algorithms (Michalewicz 2013), which mimic natural selection to solve complex problems, have demonstrated significant potential in optimizing complex, high-dimensional search spaces (Yu and Gen 2010), making them a suitable candidate for evolving decision tree ensembles. However, existing evolutionary approaches operate only with a single population and produce single decision trees (Żychowski, Perrault, and Mańdziuk 2024).

To address these limitations, we propose a novel island-based coevolutionary approach named ICoEvoRDF (Island-based CoEvolutionary Robust Decision Forests). The approach is inspired by island models (Tanese 1989; Skolicki and De Jong 2004) in evolutionary computations and divide a population into several isolated subpopulations, each evolving independently. Occasionally, individuals migrate between islands, introducing beneficial traits and enhancing overall diversity (Bull et al. 2007). This approach has been successful in various optimization problems, often performing better than traditional single-population methods (Skolicki and De Jong 2004; Skolicki 2005; Luong, Melab, and Talbi 2010).

In the context of DT ensembles, the use of island models can be particularly advantageous. By evolving multiple populations of RDTs in parallel, each with different initial conditions and evolutionary trajectories, we can generate a diverse set of trees that, when combined, form a more robust and versatile forest. Moreover, this approach may also be advantageous for creating single RDTs, as previous studies (Żychowski, Perrault, and Mańdziuk 2024) have demonstrated significant benefits from repeated runs of evolutionary methods.

Additionally, we employ the game-theoretic concept of Mixed Nash Equilibrium (MNE) to effectively weight the decision trees in the forest. Unlike uniform weighting methods, MNE optimizes tree contributions by reflecting the full set of perturbations encountered in adversarial settings. The island-based coevolutionary framework naturally generates diverse input data perturbations, which are crucial for cap-

turing the range of scenarios represented by the MNE. Integrating MNE into our coevolutionary framework enhances robustness and performance, demonstrating the power of combining game theory with coevolutionary methods.

Contributions

The main contributions of the work are:

- a novel island-based coevolutionary approach (ICoEvoRDF) for creating a mixture of Robust Decision Trees (RDTs), which utilizes a new game-theoretic approach based on Mixed Nash Equilibrium (Fudenberg and Kreps 1993) for weighting decision trees within a forest. The method is adaptable to optimizing various target metrics, such as adversarial accuracy and minimax regret. Furthermore, it is able to construct individual RDTs, that are superior to those constructed by evolutionary-based approaches with a single population;
- potential for ICoEvoRDF to improve results of other state-of-the-art (SOTA) methods by including their resulting decision trees in the initial population of one or more islands, thus allowing for the combination of multiple solutions into a single, synergetic framework;
- comprehensive analysis of how specific components of the proposed solution (e.g., number of islands, Nash-based voting, various initialization methods, population diversity) influence its performance and outcomes,
- experimental validation showing that ICoEvoRDF significantly outperforms existing methods for ensemble of trees as well as single DTs, with improvements of up to 0.010 in adversarial accuracy and 0.064 in minimax regret across 20 benchmark datasets.

2 Problem Definition

Let $X \subset \mathbb{R}^d$ be a d -dimensional feature space (inputs) and Y be the set of possible classes (outputs). A standard classification task involves finding a function (model) $h : X \rightarrow Y$, where $h(x_i) = y_i$ and y_i is the true class of x_i . The classification performance of h can be measured by its accuracy, defined as:

$$\text{acc}(h) = \frac{1}{|X|} \sum_{x_i \in X} I[h(x_i) = y_i], \quad (1)$$

where $I[h(x_i) = y_i]$ returns 1 if h predicts the true class of x_i and 0 otherwise.

Define $\mathcal{N}_\varepsilon(x) = \{z : \|z - x\|_\infty \leq \varepsilon\}$ as the set of points within an L_∞ (L -infinity norm) ball centered at x with radius ε . The *adversarial accuracy* of a model h is the lowest accuracy over all possible perturbation sets constrained by \mathcal{N}_ε , formally:

$$\text{acc}_{\text{adv}}(h, \varepsilon) = \frac{1}{|X|} \sum_{x_i \in X} \min_{z_i \in \mathcal{N}_\varepsilon(x_i)} I[h(z_i) = y_i]. \quad (2)$$

If ε is large enough, it may not be possible to train M effectively, and the adversarial accuracy would be low. However, low adversarial accuracy does not indicate whether the reason is poor model robustness or impossibility of creating such a model due to the high value of ε .

The *max regret* (Savage 1951) of a model h is the maximum regret among all possible perturbations $z \in \mathcal{N}_\varepsilon$. Regret is defined as the difference between the best possible accuracy on a particular perturbation and the accuracy that h achieves:

$$\text{regret}(h, \{z_i\}) = \max_{h'} \text{acc}(h', \{z_i\}) - \text{acc}(h, \{z_i\}), \quad (3)$$

where $\text{acc}(h, \{z_i\})$ is the accuracy achieved by model h when the feature set $\{x_i\}$ is replaced with $\{z_i\}$. Max regret can be expressed as:

$$\text{mr}(h) = \max_{z_i \in \mathcal{N}_\varepsilon(x_i)} \text{regret}(h, \{z_i\}). \quad (4)$$

Max regret is defined as the maximum difference between the result of a given model and the result of the optimal model for any input data perturbation within a given range. Minimizing max regret is known as the minimax regret decision criterion. Minimax regret is a more comprehensive metric than adversarial accuracy because it takes into account the performance of a model on both clean (non-perturbed) and adversarial (perturbed) data, so it is less sensitive to the choice of ε .

The problem addressed in this paper is finding a decision forest (ensemble of decision trees) trained on X that for a given ε optimizes (maximizes for adversarial accuracy or minimizes for max regret) a given robustness metric. The final prediction from decision trees ensemble are obtained by the weighted voting.

We use *Mixed Nash Equilibrium* (MNE) for computing the weights of trees in an ensemble. An MNE $(\mathcal{T}, \mathcal{P})$ comprises a pair of mixed strategies: $\mathcal{T} = \{(T_1, p_{T_1}), \dots, (T_n, p_{T_n})\}$, and $\mathcal{P} = \{(P_1, p_{P_1}), \dots, (P_m, p_{P_m})\}$. In our case, those are a set of DTs (\mathcal{T}) and perturbations (\mathcal{P}), respectively, with associated probabilities. $(\mathcal{T}, \mathcal{P})$ fulfills the following conditions: $\forall_{\mathcal{T}' \neq \mathcal{T}}, \mathbb{E}_{\mathcal{T}, \mathcal{P}}[\xi(\mathcal{T}', \mathcal{P})] \preceq \mathbb{E}_{\mathcal{T}, \mathcal{P}}[\xi(\mathcal{T}, \mathcal{P})]$ and $\forall_{\mathcal{P}' \neq \mathcal{P}}, \mathbb{E}_{\mathcal{T}, \mathcal{P}}[-\xi(\mathcal{T}, \mathcal{P}')] \preceq \mathbb{E}_{\mathcal{T}, \mathcal{P}}[-\xi(\mathcal{T}, \mathcal{P})]$, where $\xi(\mathcal{T}, \mathcal{P})$ is some performance metric calculated for a “mixed” decision tree \mathcal{T} and “mixed” perturbation \mathcal{P} (accuracy for adversarial accuracy or regret for max regret).

Note that the game is zero-sum because in robust optimization the adversary aims to minimize the DT payoff. Thus, the minimax theorem holds (v. Neumann 1928), guaranteeing that the MNE maximizes the robustness metric, i.e., $\max_{\mathcal{T}'} \min_{\mathcal{P}'} \xi(\mathcal{T}', \mathcal{P}') = \mathbb{E}_{\mathcal{T}, \mathcal{P}}[\xi(\mathcal{T}, \mathcal{P})]$ if $(\mathcal{T}, \mathcal{P})$ is an MNE. In addition, for zero-sum games, MNEs can be computed in time polynomial in the strategy set size.

3 Related Work

The problem of finding RDTs is well-established in the literature, and several methods of solving it have been proposed. RIGDT-h (Chen et al. 2019) constructs robust decision trees based on the concept of adversarial Gini impurity (Breiman 2017), which is a modification of the classical Gini impurity adapted for perturbed input data. This method was further improved in the GROOT algorithm (Vos and Verwer 2021), which employs a greedy recursive splitting strategy similar to traditional decision trees and evaluates splits using the adversarial Gini impurity. Another algorithm, Fast Provably

Robust Decision Tree (FPRDT) (Guo et al. 2022), builds robust decision trees by directly minimizing the adversarial 0/1 loss through a greedy recursive approach, efficiently evaluating potential splits using sorted threshold lists. The algorithm ensures optimal splits only when beneficial, maintaining both robustness and efficiency.

Since finding RDTs is an optimization problem, some evolutionary-based methods were also proposed. Ranzato and Zanella (2021) introduced a genetic adversarial training algorithm (Meta-Silvae) to optimize DT stability. Co-evolutionary Algorithm for Robust Decision Trees (CoEvoRDT) (Żychowski, Perrault, and Mańdziuk 2024) constructs robust decision trees by maintaining two populations: one of decision trees and another of data perturbations, allowing the trees to adapt and learn from the perturbations. The populations alternately evolve using crossover, mutation, and selection operators until a stopping condition is met. The method emphasizes robustness by incorporating a game-theoretic approach to build a Hall of Fame (an archive that preserves the best solutions found during the algorithm’s run) with a Mixed Nash Equilibrium, enhancing decision tree robustness and population diversity. CoEvoRDT demonstrates strong performance in minimax regret and adversarial accuracy, making it one of the state-of-the-art algorithms.

Tree ensembles were also explored in the context of finding robust classifiers as a natural subsequent step. Kantchevian, Tygar, and Joseph (2016) introduced a method to iteratively and recursively train robust decision trees using the L_0 metric based on the original and adversarial examples. Vos and Verwer (2021) proposed robust random forests based on the original idea of random forests (Breiman 2001). Andriushchenko and Hein (2019) use a robust boosting algorithm based on adversarial exponential loss. Provably Robust AdaBoost (PRAdaBoost) (Guo et al. 2022) method is built upon the AdaBoost algorithm (Freund, Schapire et al. 1996), a well-established method in machine learning that combines multiple “weak” learners to create a “strong” classifier. PRAdaBoost utilizes the FPRDT algorithm mentioned above as the base learner.

To the best of our knowledge, so far no other multi-population approaches have been proposed for optimizing RDTs or RDFs, and our method is the first successful attempt in this area.

4 ICoEvoRDF Algorithm

In this study, we introduce the Island-based CoEvolutionary algorithm for constructing Robust Decision Forests (ICoEvoRDF). This approach draws inspiration from the biological concept of speciation, where distinct populations evolve in isolated environments with limited gene flow.

ICoEvoRDF operates on a fixed set of islands, denoted as \mathcal{I} , whose cardinality $|\mathcal{I}|$ is a user-defined parameter. Each island $I \in \mathcal{I}$ maintains two coevolving populations: a decision tree population I^T and a perturbation population I^P , analogously to the CoEvoRDT algorithm detailed in the previous section. A pseudocode of the proposed algorithm is presented in the supplementary material (Żychowski, Perrault, and Mańdziuk 2025).

Islands initialization. We initialize each island independently. For the baseline implementation, the decision tree population is initialized with small random trees (with a maximum depth of 3). The potential advantages of alternative initialization methods, such as utilizing the results of other algorithms are discussed in Section 6.

The perturbation population is sampled uniformly from the space of all possible perturbations of the input data within a given radius ϵ . For input data selection, we propose to assign a unique training set to each island. It mirrors the approach of classic random forests and is achieved by sampling with replacement from X until each island’s training set contains $|X|$ instances. This approach permits the repetition of certain training instances across multiple islands.

Islands evolution. Each island evolves independently for a predetermined number of generations, denoted by n_g . This process entails coevolution between the T and perturbation populations, similarly to the CoEvoRDT algorithm. The DT and perturbation populations evolve alternately for n_g generations, each involving standard evolutionary operations: crossover, mutation, evaluation, and selection. The mutation operator randomly performs one of the three following actions: (i) replacing a subtree with a randomly generated one, (ii) changing the information in a randomly selected node (e.g., a new splitting value or operator), or (iii) pruning a randomly selected subtree. The crossover operator randomly selects one node in each of two individuals and exchanges the corresponding subtrees, generating two offspring added to the population. In the perturbation population, each pair (input instance, attribute) has a probability 0.5 of being perturbed, with a new feasible value assigned (according to the epsilon constraint). The crossover randomly mixes input instances from both individuals.

Evaluation is conducted against individuals from the opposing population (DTs are evaluated against perturbations and vice versa). Furthermore, a Hall of Fame (HoF) mechanism inspired by Nash equilibrium theory has been incorporated into ICoEvoRDF, which proved to be beneficial in CoEvoRDT method. HoF maintains a mixture of individuals from both populations which are used during the evaluation stage. The final stage involves binary tournament selection, which selects individuals for the next generation. In this process, individuals are randomly paired, and the one with higher fitness has a greater probability of being chosen. The size of the population is fixed, i.e., in each generation the same number of individuals are promoted.

Migration. Migration between islands is an important component of island-based evolutionary algorithms, differentiating them from $|\mathcal{I}|$ independent runs of a single island. If subpopulations evolve in complete isolation (without migration), they might converge prematurely to suboptimal solutions due to the lack of new genetic material (Gong et al. 2015). Migration introduces new genetic traits, reducing the risk of premature convergence and potentially leading to superior solutions through the sharing of beneficial traits.

For each island I , we define a set of neighboring islands $\eta(I)$ to which I transmits information about its discovered solutions. The connections between islands and their neigh-

bors constitute a graph referred to as the island topology. For instance, a ring topology forms a bidirectional cycle where each island possesses exactly two neighbors: $\forall I \in \mathcal{I} |\eta(I)| = 2$. Figure 1 provides an overview of the ICoEvoRDF algorithm and illustrates an example of a ring topology.

After every n_g generations of both the DT and perturbation populations, the top k_{top} fittest (i.e., highest fitness) decision trees from each neighboring island $I_n \in \eta(I)$ are migrated (copied) into the recipient island’s DT population:

$$I^T = I^T \cup \bigcup_{I_n \in \eta(I)} V_\xi^{k_{\text{top}}}(I_n^T), \quad (5)$$

where $V_\xi^{k_{\text{top}}}(I_n^T)$ is the k_{top} fittest decision trees from the DT population of the neighbor island I_n^T according to the optimized metric ξ . The k_{top} fittest perturbations are migrated in the same way.

Stop condition. This iterative process persists until at least one of the following two conditions is fulfilled: either a total of l_g generations have elapsed, or the fittest DT across all islands has not improved within the last l_c generations.

Upon termination, either a single, globally fittest DT is selected across all islands, or an ensemble of DTs from all islands are aggregated to construct a decision forest (DF).

Decision forest composition. For creation of a DT ensemble we perform a weighted voting among the fittest DT representatives from all islands. The simplest and most common approach is to assign equal contribution to each island representative. However, it may not be optimal, as the efficacy of the generated trees may vary, rendering equal contributions undesirable. Leveraging the fact that each island also generates a population of perturbations, we propose *Nash-based voting* (NV). This approach frames the scenario as a two-player game between DTs (represented by island representatives) and perturbations. The DT player, P_{DT} , chooses their strategy from a strategy set $\Pi_{P_{DT}} = \{V_\xi^1(I^T)\}_{I \in \mathcal{I}}$, while the perturbation player, P_P , chooses their strategy from a strategy set $\Pi_{P_P} = \bigcup_{I \in \mathcal{I}} I^P$. Then, a mixed Nash equilibrium is computed for this game. We use the Lemke-Howson algorithm (Lemke and Howson 1964), which has worst-case exponential time complexity, but is fast in practice (and has average linear time for random games (Codonotti, De Rossi, and Pagan 2008)). In the proposed Nash-based voting (NV) probabilities from the mixed equilibrium DT strategy \mathcal{T} are directly used for voting weights.

We also explored an alternative approach to extracting from each island for the final ensemble more DTs than only the fittest one. These experiments resulted in only marginal improvement (less than 0.1%), coupled with increased computational overhead and complexity of the final model.

5 Experimental Setup

Tested benchmarks. The proposed method was evaluated on 20 widely used classification benchmark problems with varying characteristics, including the number of instances (from 351 to 70000), features (4 – 3072), and perturbation coefficients ε (0.01 – 0.3). All selected datasets have been utilized in prior studies referenced in the Related Work section and are publicly available at <https://www.openml.org>.

The main parameters of the datasets (perturbation coefficient, number of instances, features, and classes) are presented in the supplementary material.

Parameterization. For the experiments conducted in this study, we employed the same set of parameters across all islands and their values followed recommendation from CoEvoRDT authors. Namely, decision tree population size $N_T = 200$, perturbation population size $N_P = 500$, number of consecutive generations for each population $n_p = 20$, number of best individuals from the DT population involved in the perturbations evaluation $N_{\text{top}} = 20$, crossover probability $p_c = 0.8$, mutation probability $p_m = 0.5$, selection pressure $p_s = 0.9$, elite size $e = 2$, HoF size $N_{\text{HoF}} = 200$. Generations without improvement limit l_c was set to 100 and generation limit $l_g = 1000$.

Preliminary evaluations were conducted to assess various migration topologies, such as ring, star, and circle configurations. The ring topology (depicted in Figure 1) yielded the most favorable results and was thus used for all experiments presented in the subsequent section. A more comprehensive discussion and detailed results for different topologies can be found in the supplementary material (Żychowski, Perrault, and Mańdziuk 2025). The number of generations between migrations, n_g , was set to 40.

The number of islands, $|\mathcal{I}|$, was fixed at 10 to maintain comparable computational time to the state-of-the-art (SOTA) method, PRAdaBoost. However, our experiments suggest that further increases in the DT population size (N_T) or the number of islands $|\mathcal{I}|$ may lead to enhanced performance. A detailed analysis of the relationship between population size/number of islands, computational time, and robustness can be found in the supplementary material (Żychowski, Perrault, and Mańdziuk 2025). This is also related to interpretability which is also important aspect in the context of DTs. While adding more DTs to the forest can further (slightly) enhance robustness, it also deteriorates the model’s overall interpretability.

All results for nondeterministic methods presented in the next section represent the average of 20 independent runs. We refer the reader to the supplementary material for detailed results about their variations (standard deviations). Statistical significance was checked according to the paired t -test with p -value ≤ 0.05 . All tests were executed on an Intel Xeon Silver 4116 processor operating at 2.10GHz. The CoEvoRDT source code is available at github.com/zychowskia/ICoEvoRDF. We employed the Nashpy Python library (Knight and Campbell 2018) implementation of the Lemke-Howson algorithm for computing the Mixed Nash Equilibrium. The CART method (Breiman 2017) was used to determine the reference tree for minimax regret (the tree with the highest accuracy for a given perturbation). The final results for minimax regret were computed based on 10^5 randomly generated perturbations (see supplementary material for reasoning) – the same for all tested methods. The adversarial accuracy results were calculated using the exact method based on Mixed Integer Linear Programming (Kantchelian, Tygar, and Joseph 2016), implemented by Chen et al. (2019).

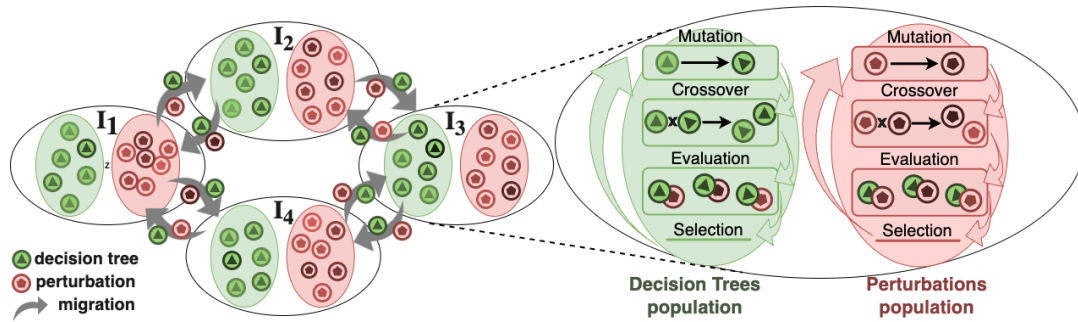


Figure 1: The ICoEvoRDF scheme. The illustration shows 4 islands with ring migration topology. Each island contains two populations: DTs and perturbations which are developed alternately using evolutionary operators.

6 Results and Discussion

Trees ensemble models. Tables 1 and 2 present the results for ensemble methods. GROOT, FPRDT, and CoEvoRDT construct decision forests based on their respective underlying DT creation algorithms, analogously to the standard random forest algorithm.

PRAdaBoost and CoEvoRDT boosting are boosting methods that combine multiple weak learners (FPRDT and CoEvoRDT, respectively) to create a stronger model. Each DT is trained sequentially, with an emphasis on rectifying errors made by previous trees. Misclassified instances are assigned greater weights to enhance overall accuracy. The final model is a weighted sum of all DTs.

In terms of adversarial accuracy, the proposed ICoEvoRDF algorithm achieved superior results on 17 benchmarks, for 9 of them with statistical significance. The most competitive non-evolutionary method, PRAdaBoost, yielded the highest score on two datasets. With respect to the max regret metric, the proposed approach outperformed all other methods across all benchmark datasets, in each case with statistical significance).

We also explored the combination of the strongest non-evolutionary method, FPRDT, with the proposed coevolutionary islanding approach. This was accomplished by initializing one island with the FPRDT algorithm (denoted by the suffix “+ FPRDT”), while all other islands were initialized with random DTs. For some datasets (particularly those where FPRDT performed well, such as the diabetes dataset) they further improved the results.

A comprehensive analysis of standard deviations (std dev) for each method is available in the supplementary material (Żychowski, Perrault, and Mańdziuk 2025). Notably, among the top-performing methods, PRAdaBoost exhibits an average std dev of 0.0085, while the ICoEvoRDF family of methods demonstrates a std dev of 0.0100.

Ablation study. To assess the influence of specific ICoEvoRDF components, we conducted an ablation study. We tested ICoEvoRDF with an alternative voting method—*equal voting (EV)*. Instead of calculating the mixed Nash equilibrium and using their probabilities as voting weights, EV assigns equal contribution to each island representative. This approach is common in classical random forest settings.

Another tested aspect of ICoEvoRDF was the input data selection. In the baseline version, a unique training set is assigned to each island by sampling with replacement. To evaluate the influence of this strategy, we tested an alternative approach: selecting the same input (SI) of all training examples as the training dataset X for each island. The results of these ablation studies are presented in Tables 1 and 2. Even the simplified island-based model with EV and SI settings (ICoEvoRDF $_{SI}^{EV}$) outperformed the baseline CoEvoRDT forest method. Further improvements were observed with the incorporation of Nash voting (ICoEvoRDF $_{SI}$) and distinct input sets for each island (ICoEvoRDF EV). The most significant improvement was achieved by combining NV with distinct input sets, i.e., ICoEvoRDF.

Additional experiments (presented later in this section) indicate that these improvements are correlated with increased diversity in the generated DT ensembles. Utilizing different inputs for each island enhances the diversity of the final ensemble, while Nash voting effectively assigns weights to these diverse DTs. This synergy between differentiated inputs and Nash voting leads to a significant improvement in performance.

Computation time. Due to space limits, a detailed computation time comparison is provided in the supplementary material (Żychowski, Perrault, and Mańdziuk 2025). Here, we only report the sum of the average computation times across all benchmarks and discuss the general conclusions.

Methods for creating single DTs (FPRDT and CoEvoRDT) exhibit significantly shorter average computation times (FPRDT: 246s, CoEvoRDT: 727s) compared to ensemble methods (PRAdaBoost: 6972s, CoEvoRDT boosting: 7606s, ICoEvoRDF: 6845s, ICoEvoRDF+FPRDT: 7091s). Computation times for all ensemble methods are of the same order of magnitude, ensuring a fair comparison across methods under a similar time constraint. The scalability and performance of ICoEvoRDF under increased computational budgets are presented in the supplementary material (Żychowski, Perrault, and Mańdziuk 2025).

The results presented above do not utilize parallelization, which could substantially reduce the computation time for non-boosting ensemble methods (CoEvoRDT forest, ICoEvoRDF, and ICoEvoRDT + FPRDT). Parallelization is straightforward for the CoEvoRDT forest, as each run of

| dataset | Random forests | GROOT forests | FPRDT forest | CoEvoRDT forest | PRAdaBoost | CoEvoRDT boosting | ICoEvoRDF _{SI} ^{EV} | ICoEvoRDF _{SI} | ICoEvoRDF ^{EV} | ICoEvoRDF | ICoEvoRDF + FPRDT |
|----------------|----------------|---------------|--------------|-----------------|--------------|-------------------|---------------------------------------|-------------------------|-------------------------|--------------|-------------------|
| ionos | 0.112 | 0.787 | 0.791 | 0.793 | 0.796 | 0.798 | 0.797 | 0.796 | 0.796 | 0.799 | 0.801 |
| breast | 0.217 | 0.884 | 0.873 | 0.885 | 0.879 | 0.899 | 0.891 | 0.894 | 0.896 | 0.900 | 0.900 |
| diabetes | 0.452 | 0.648 | 0.649 | 0.621 | 0.654 | 0.644 | 0.625 | 0.636 | 0.646 | 0.647 | 0.651 |
| bank | 0.509 | 0.641 | 0.658 | 0.661 | 0.668 | 0.669 | 0.667 | 0.670 | 0.664 | 0.673 | 0.672 |
| Japan3v4 | 0.519 | 0.658 | 0.669 | 0.679 | 0.682 | 0.684 | 0.684 | 0.688 | 0.684 | 0.688 | 0.690 |
| spam | 0.000 | 0.750 | 0.749 | 0.751 | 0.754 | 0.763 | 0.756 | 0.756 | 0.762 | 0.766 | 0.766 |
| GesDvP | 0.189 | 0.731 | 0.725 | 0.740 | 0.732 | 0.753 | 0.745 | 0.745 | 0.749 | 0.752 | 0.754 |
| har1v2 | 0.233 | 0.792 | 0.828 | 0.844 | 0.860 | 0.851 | 0.855 | 0.858 | 0.847 | 0.854 | 0.860 |
| wine | 0.091 | 0.633 | 0.681 | 0.688 | 0.690 | 0.708 | 0.691 | 0.691 | 0.707 | 0.708 | 0.708 |
| collision-det | 0.325 | 0.726 | 0.791 | 0.804 | 0.800 | 0.820 | 0.810 | 0.812 | 0.815 | 0.822 | 0.822 |
| mnist-1-5 | 0.000 | 0.925 | 0.964 | 0.964 | 0.969 | 0.975 | 0.969 | 0.972 | 0.968 | 0.976 | 0.976 |
| mnist-2-6 | 0.000 | 0.823 | 0.919 | 0.917 | 0.924 | 0.925 | 0.923 | 0.925 | 0.922 | 0.926 | 0.926 |
| mnist | 0.000 | 0.632 | 0.750 | 0.747 | 0.761 | 0.763 | 0.755 | 0.759 | 0.759 | 0.764 | 0.764 |
| F-mnist2v5 | 0.456 | 0.979 | 0.974 | 0.982 | 0.982 | 0.993 | 0.990 | 0.994 | 0.987 | 0.995 | 0.996 |
| F-mnist3v4 | 0.044 | 0.839 | 0.861 | 0.869 | 0.867 | 0.879 | 0.877 | 0.877 | 0.877 | 0.884 | 0.884 |
| F-mnist7v9 | 0.136 | 0.836 | 0.875 | 0.868 | 0.879 | 0.877 | 0.877 | 0.880 | 0.873 | 0.881 | 0.880 |
| F-mnist | 0.024 | 0.241 | 0.537 | 0.545 | 0.546 | 0.559 | 0.552 | 0.553 | 0.554 | 0.560 | 0.561 |
| cifar10:0v5 | 0.302 | 0.526 | 0.683 | 0.690 | 0.691 | 0.699 | 0.694 | 0.696 | 0.697 | 0.702 | 0.703 |
| cifar10:0v6 | 0.368 | 0.560 | 0.688 | 0.696 | 0.696 | 0.703 | 0.701 | 0.701 | 0.701 | 0.704 | 0.705 |
| cifar10:4v8 | 0.296 | 0.498 | 0.665 | 0.665 | 0.671 | 0.671 | 0.674 | 0.674 | 0.673 | 0.675 | 0.675 |
| AVERAGE | 0.214 | 0.705 | 0.767 | 0.771 | 0.775 | 0.782 | 0.777 | 0.779 | 0.779 | 0.784 | 0.785 |

Table 1: Averaged adversarial accuracies for ensemble forests methods. The best results are bolded.

the CoEvoRDT algorithm is independent and can be executed concurrently. However, migration in islanding methods complicates parallelization, although it can be facilitated by incorporating shared memory. Each island can store its k_{top} individuals from the last generation in this shared memory, and instead of a synchronous migration phase where each island sends its k_{top} individuals to neighbors, islands can retrieve the individuals needing migration from the shared memory. We have verified that this approach does not impact the results and reduces computation time by a factor of 8-10. A detailed discussion of this optimization and its results can be found in the supplementary material. For clarity of the ICoEvoRDT description, we have opted to omit the details of this optimization from the main body of the paper. Such parallelization cannot be implemented for boosting methods (PRAdaBoost and CoEvoRDT boosting), which is an additional advantage of the proposed island-based approach.

Diversity analysis. We hypothesize that the performance differences between different variants of the proposed algorithm are related to the diversity of DTs generated by the algorithm. To quantify the diversity within a set of DTs, we measure the fraction of perturbed input instances for which the predictions of each pair of trees in the set differ.

Formally, let $\mathcal{T} = \{T_1, T_2, \dots, T_k\}$ represent the ensemble of decision trees, and let $X' = \{x'_1, x'_2, \dots, x'_m\}$ denote the set of perturbed input instances. The diversity between two trees T_i and T_j is defined as: $\text{div}(T_i, T_j) = \frac{1}{m} \sum_{k=1}^m I[T_i(x'_k) \neq T_j(x'_k)]$.

The *average diversity* of the ensemble is then given by:

$$\text{avg_div}(\mathcal{T}) = \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^n \text{div}(T_i, T_j) \quad (6)$$

and the *maximum diversity* is defined as:

$$\text{max_div}(\mathcal{T}) = \max_{1 \leq i < j \leq n} \text{div}(T_i, T_j) \quad (7)$$

We conducted a diversity analysis for two variants: *external* and *internal* diversity. External diversity was calculated using the decision trees that comprise the final ensemble, while internal diversity was assessed on the DTs within each island population. The results of the diversity analysis are presented in Table 3. Given that the voting method is independent of diversity and does not influence its value, Nash voting and equal voting were considered jointly.

Our first observation is that internal is significantly higher than external diversity. This difference arises from the dynamic nature of the DT population evaluation, as each population potentially contains weaker DTs resulting from the exploratory nature of mutation operations, leading to higher internal diversity. Secondly, internal diversity appears to be consistent across different methods, while external diversity varies. The primary proposed method, ICoEvoRDF, achieves the highest average and maximum external diversity across all datasets when compared to the method without migration (N ICoEvoRDF) and the method using the same training set for each island (ICoEvoRDF_{SI}). This suggests that the introduction of migration and varying inputs leads to a more diverse ensemble of DTs. Furthermore, incorporating FPRDT as an initialization for one island (ICoEvoRDF+FPRDT) further enhances diversity. These findings suggest a positive correlation between the diversity of the generated ensemble and the method’s performance. A more diverse ensemble is likely to be more robust against adversarial attacks and to generalize better to unseen data.

Single decision trees. While ICoEvoRDF is designed for the construction of random decision forests, it can also be utilized to produce a single DT by selecting the most robust tree across all islands. This method will be referred to as ICoEvoRDT. It resembles the N CoEvoRDT method proposed in (Żychowski, Perrault, and Mańdziuk 2024), which involves multiple independent runs of CoEvoRDT followed by the selection of the best DT. However, the island model

| dataset | Random forests | GROOT forests | FPRDT forest | CoEvoRDT forest | PRAdaBoost | CoEvoRDT boosting | ICoEvoRDF ^{EV} _{SI} | ICoEvoRDF _{SI} | ICoEvoRDF ^{EV} | ICoEvoRDF | ICoEvoRDF + FPRDT |
|----------------|----------------|---------------|--------------|-----------------|------------|-------------------|---------------------------------------|-------------------------|-------------------------|--------------|-------------------|
| ionos | 0.094 | 0.088 | 0.061 | 0.052 | 0.060 | 0.045 | 0.046 | 0.046 | 0.048 | 0.044 | 0.044 |
| breast | 0.103 | 0.097 | 0.055 | 0.047 | 0.055 | 0.035 | 0.047 | 0.044 | 0.039 | 0.034 | 0.034 |
| diabetes | 0.202 | 0.194 | 0.111 | 0.092 | 0.113 | 0.027 | 0.076 | 0.045 | 0.029 | 0.026 | 0.026 |
| bank | 0.186 | 0.177 | 0.086 | 0.075 | 0.086 | 0.050 | 0.051 | 0.050 | 0.061 | 0.046 | 0.046 |
| Japan3v4 | 0.107 | 0.106 | 0.063 | 0.060 | 0.062 | 0.027 | 0.028 | 0.025 | 0.032 | 0.025 | 0.025 |
| spam | 0.097 | 0.095 | 0.071 | 0.067 | 0.071 | 0.046 | 0.063 | 0.061 | 0.054 | 0.044 | 0.044 |
| GesDvP | 0.152 | 0.143 | 0.127 | 0.111 | 0.122 | 0.077 | 0.110 | 0.108 | 0.090 | 0.073 | 0.073 |
| har1v2 | 0.105 | 0.100 | 0.066 | 0.063 | 0.061 | 0.020 | 0.020 | 0.018 | 0.023 | 0.019 | 0.019 |
| wine | 0.140 | 0.139 | 0.106 | 0.089 | 0.102 | 0.064 | 0.084 | 0.083 | 0.067 | 0.063 | 0.063 |
| collision-det | 0.142 | 0.137 | 0.088 | 0.059 | 0.084 | 0.032 | 0.041 | 0.041 | 0.036 | 0.030 | 0.031 |
| mnist-1-5 | 0.249 | 0.234 | 0.066 | 0.053 | 0.067 | 0.046 | 0.052 | 0.047 | 0.054 | 0.044 | 0.044 |
| mnist-2-6 | 0.268 | 0.253 | 0.066 | 0.054 | 0.063 | 0.047 | 0.048 | 0.046 | 0.053 | 0.045 | 0.045 |
| mnist | 0.395 | 0.381 | 0.118 | 0.109 | 0.112 | 0.065 | 0.081 | 0.077 | 0.076 | 0.061 | 0.062 |
| F-mnist2v5 | 0.273 | 0.247 | 0.230 | 0.188 | 0.234 | 0.165 | 0.173 | 0.165 | 0.189 | 0.156 | 0.156 |
| F-mnist3v4 | 0.290 | 0.269 | 0.225 | 0.196 | 0.220 | 0.146 | 0.176 | 0.167 | 0.179 | 0.135 | 0.136 |
| F-mnist7v9 | 0.283 | 0.280 | 0.234 | 0.205 | 0.226 | 0.156 | 0.168 | 0.144 | 0.193 | 0.144 | 0.145 |
| F-mnist | 0.427 | 0.401 | 0.283 | 0.235 | 0.259 | 0.121 | 0.171 | 0.153 | 0.153 | 0.110 | 0.109 |
| cifar10:0v5 | 0.419 | 0.380 | 0.305 | 0.231 | 0.287 | 0.137 | 0.173 | 0.166 | 0.156 | 0.130 | 0.129 |
| cifar10:0v6 | 0.403 | 0.372 | 0.329 | 0.278 | 0.328 | 0.204 | 0.214 | 0.212 | 0.222 | 0.198 | 0.198 |
| cifar10:4v8 | 0.408 | 0.392 | 0.326 | 0.275 | 0.310 | 0.208 | 0.211 | 0.203 | 0.215 | 0.205 | 0.205 |
| AVERAGE | 0.237 | 0.224 | 0.151 | 0.127 | 0.146 | 0.086 | 0.102 | 0.095 | 0.098 | 0.082 | 0.082 |

Table 2: Averaged max regret for ensemble forests methods. The best results are bolded.

| diversity | N CoEvoRDT | ICoEvoRDF _{SI} | ICoEvoRDF | ICoEvoRDF + FPRDT |
|--------------|--------------|-------------------------|-----------|-------------------|
| external avg | 0.037 | 0.038 | 0.045 | 0.054 |
| external max | 0.041 | 0.042 | 0.050 | 0.061 |
| internal avg | 0.107 | 0.110 | 0.111 | 0.109 |
| internal max | 0.200 | 0.211 | 0.209 | 0.210 |

Table 3: Diversity averaged over all benchmark datasets.

described in this paper introduces the crucial element of migration between islands, each running an independent version of CoEvoRDT. Our experiments indicate that this migration mechanism significantly improves results, outperforming both the baseline coevolutionary method CoEvoRDT and its simple rerun counterpart (N CoEvoRDT), where the number of repetitions (N) is equivalent to the number of islands in ICoEvoRDT. This underscores the importance of the individual migration concept detailed in the ICoEvoRDF algorithm description section.

For all benchmarks, the incorporation of islanding with individual migration yielded superior outcomes (averaged over all benchmarks) – adversarial accuracy (AA): 0.779, max regret (MR): 0.113, compared to the baseline method CoEvoRDT (AA: 0.767, MR: 0.131) and all other competitors: GROOT (AA: 0.718, MR: 0.158), FPRDT (AA: 0.765, MR: 0.155), N CoEvoRDT (AA: 0.776, MR: 0.122). Detailed results are presented in the supplementary material (Żychowski, Perrault, and Mańdziuk 2025).

7 Conclusions

In this paper, we introduced ICoEvoRDF, a novel island-based coevolutionary algorithm for constructing robust decision forests. Our approach leverages the strengths of multiple independently evolving populations of decision trees and perturbations, with periodic migration of top-performing individuals between islands. This strategy fosters diversity and

promotes the exploration of a wider range of potential solutions, leading to more robust decision tree ensembles.

Our experimental results on 20 datasets demonstrate the effectiveness of ICoEvoRDF in optimizing both adversarial accuracy and minimax regret metrics. The algorithm consistently outperforms state-of-the-art methods, showcasing its ability to generate highly robust decision trees and forests. The flexibility of ICoEvoRDF allows for the integration of decision trees from various existing methods, offering a unified framework for combining diverse solutions.

A notable aspect of our work is the trade-off between model interpretability and robustness. ICoEvoRDF can generate more robust ensemble models, although their complexity and the associated weighting may reduce interpretability. Conversely, it can produce simpler single DTs that are less robust but easier to interpret. The balance between robustness and interpretability can be adjusted by the number of islands, allowing to control which aspect is prioritized.

To the best of our knowledge, this is the first study where mixed Nash equilibrium has been combined with an island-based evolutionary algorithm. Our results demonstrate that this synergy between coevolutionary methods and game theory is highly effective. We believe that it holds significant application potential also in other domains, worth further investigation. Future work can also focus on extending the application of island-based coevolutionary algorithms to other related tasks in machine learning. One promising direction is investigating the use of ICoEvoRDF to ensure fairness in machine learning models. By incorporating fairness metrics into the objective function and evolving decision trees that minimize bias, we can potentially develop fairer and more equitable decision-making systems. Another direction of work can be using a similar setting for analysis of models properties (e.g., explainability) or constructing other machine learning models for which the target metric is not differentiable or it is challenging to compute an exact value (like robustness metrics presented in this paper).

Acknowledgments

Jacek Mańdziuk gratefully acknowledges the funding support from the program ‘Excellence initiative - research university’ for the AGH University of Krakow, as well as the ARTIQ project: UMO- 2021/01/2/ST6/00004 and ARTIQ/0004/2021.

Adam Żychowski was funded by the Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) program.

References

- Andriushchenko, M.; and Hein, M. 2019. Provably robust boosted decision stumps and trees against adversarial attacks. *Advances in neural information processing systems*, 32.
- Banfield, R. E.; Hall, L. O.; Bowyer, K. W.; and Kegelmeyer, W. P. 2006. A comparison of decision tree ensemble creation techniques. *IEEE transactions on pattern analysis and machine intelligence*, 29(1): 173–180.
- Breiman, L. 2001. Random forests. *Machine learning*, 45: 5–32.
- Breiman, L. 2017. *Classification and regression trees*. Routledge.
- Bull, L.; Studley, M.; Bagnall, A.; and Whittle, I. 2007. Learning classifier system ensembles with rule-sharing. *IEEE transactions on evolutionary computation*, 11(4): 496–502.
- Chen, H.; Zhang, H.; Boning, D.; and Hsieh, C.-J. 2019. Robust decision trees against adversarial examples. In *International Conference on Machine Learning*, 1122–1131. PMLR.
- Codenotti, B.; De Rossi, S.; and Pagan, M. 2008. An experimental analysis of lemke-howson algorithm. *arXiv preprint arXiv:0811.3247*.
- Freund, Y.; Schapire, R. E.; et al. 1996. Experiments with a new boosting algorithm. In *icml*, volume 96, 148–156. Citeseer.
- Fudenberg, D.; and Kreps, D. M. 1993. Learning mixed equilibria. *Games and economic behavior*, 5(3): 320–367.
- Gong, Y.-J.; Chen, W.-N.; Zhan, Z.-H.; Zhang, J.; Li, Y.; Zhang, Q.; and Li, J.-J. 2015. Distributed evolutionary algorithms and their models: A survey of the state-of-the-art. *Applied Soft Computing*, 34: 286–300.
- Guo, J.-Q.; Teng, M.-Z.; Gao, W.; and Zhou, Z.-H. 2022. Fast Provably Robust Decision Trees and Boosting. In *International Conference on Machine Learning*, 8127–8144. PMLR.
- Kantchelian, A.; Tygar, J. D.; and Joseph, A. 2016. Evasion and hardening of tree ensemble classifiers. In *International Conference on Machine Learning*, 2387–2396. PMLR.
- Knight, V.; and Campbell, J. 2018. Nashpy: A Python library for the computation of Nash equilibria. *Journal of Open Source Software*, 3(30): 904.
- Lemke, C. E.; and Howson, J. T., Jr. 1964. Equilibrium points of bimatrix games. *Journal of the Society for Industrial and Applied Mathematics*, 12(2): 413–423.
- Luong, T. V.; Melab, N.; and Talbi, E.-G. 2010. GPU-based island model for evolutionary algorithms. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, 1089–1096.
- Michalewicz, Z. 2013. *Genetic algorithms + data structures = evolution programs*. Springer Science & Business Media.
- Ranzato, F.; and Zanella, M. 2021. Genetic adversarial training of decision trees. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 358–367.
- Rokach, L.; and Maimon, O. 2005. Decision trees. *Data mining and knowledge discovery handbook*, 165–192.
- Savage, L. J. 1951. The theory of statistical decision. *Journal of the American Statistical Association*, 46(253): 55–67.
- Skolicki, Z. 2005. An analysis of island models in evolutionary computation. In *Proceedings of the 7th annual workshop on Genetic and evolutionary computation*, 386–389.
- Skolicki, Z.; and De Jong, K. 2004. Improving evolutionary algorithms with multi-representation island models. In *International conference on parallel problem solving from nature*, 420–429. Springer.
- Tanese, R. 1989. *Distributed genetic algorithms for function optimization*. University of Michigan.
- v. Neumann, J. 1928. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1): 295–320.
- Vos, D.; and Verwer, S. 2021. Efficient training of robust decision trees against adversarial examples. In *International Conference on Machine Learning*, 10586–10595. PMLR.
- Yu, X.; and Gen, M. 2010. *Introduction to evolutionary algorithms*. Springer Science & Business Media.
- Żychowski, A.; Perrault, A.; and Mańdziuk, J. 2024. Co-evolutionary Algorithm for Building Robust Decision Trees under Minimax Regret. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(19): 21869–21877.
- Żychowski, A.; Perrault, A.; and Mańdziuk, J. 2025. Cultivating Archipelago of Forests: Evolving Robust Decision Trees through Island Coevolution (extended version). *arXiv:2412.13762*.