”Robust Optimal Classification Trees under Noisy Labels”: An Analysis and Experimental Reproduction

Efstratios Reppas*

***Abstract*—This report summarizes the main points and evaluates the experimental findings of the methodology introduced in ”Robust Optimal Classification Trees Under Noisy Labels,” which addresses the challenges posed by label noise in supervised classification tasks. The proposed approach integrates Support Vector Machines (SVM) with Optimal Classification Trees (OCT), leveraging margin-based splits and adaptive relabeling to enhance robustness and accuracy. By formulating the problem as a Mixed Integer Non Linear Programming (MINLP) model, the method ensures optimal tree construction while mitigating label noise effects. Experimental results on datasets from the UCI Machine Learning Repository demonstrate significant performance improvements over existing methods in terms of accuracy. This reproduction should validate the robustness of the proposed methodology and highlight its potential for real-world applications in noisy environments.**

***Index Terms*—Artificial Intelligence, Machine Learning, Supervised Learning, Support Vector Machines (SVMs), Decision Trees, Mixed Integer Non Linear Programming (MINLP)**

I. Introduction

SUPERVISED classification is a cornerstone of machine learning, offering tools for uncovering patterns in labeled data to predict outcomes for unseen samples. Traditional methods, such as Classification Trees (CART) and Support Vector Machines (SVM), have been widely adopted due to their simplicity and effectiveness. However, these methods often struggle in environments with noisy labels, a common challenge in real-world datasets. Label noise can significantly degrade model performance by introducing systematic errors into the decision boundary construction process.

Recent advances in optimization have provided opportunities to address these challenges. The integration of discrete optimization into machine learning methodologies has enabled the development of robust models capable of handling noise and achieving higher interpretability and accuracy. Among such developments, Optimal Classification Trees (OCT) have gained traction, offering an optimization-based alternative to heuristic tree-building methods like CART. Similarly, robust extensions to SVM have incorporated adaptive relabeling mechanisms to mitigate the impact of noisy labels.

This report focuses on reproducing and summarizing the methodology and results presented in the paper *”Robust Optimal Classification Trees Under Noisy Labels.”* The paper proposes a novel hybrid approach that combines the hierarchical interpretability of OCTs with the margin-maximization properties of SVMs, resulting in the Optimal Classification Tree with SVM Splits and Relabeling (OCTSVM). This method employs a Mixed Integer Nonlinear Programming (MINLP) formulation to construct decision trees optimized for noisy environments. The splits are determined by hyperplanes that maximize class separation margins, and noisy labels are addressed by allowing adaptive relabeling during tree construction.

Through a comprehensive experimental evaluation on benchmark datasets from the UCI Machine Learning Repository, the authors demonstrate that OCTSVM consistently outperforms baseline methods, including CART, OCT, and OCT-H, in terms of both accuracy and AUC. The model’s ability to adapt to noisy labels without compromising the structure or complexity of the classification tree positions it as a robust solution for real-world applications.

This report will summarize the theoretical foundation of OCTSVM, reproduce its experimental results, and provide insights into the implications of this methodology for classification under noisy label conditions. The remainder of this document is structured as follows: Section II lays the theoretical foundation and describes paradigms upon which OCTSVM is based. Section III will examine the specifics of this methodology, including its formulation and algorithmic contributions. Section IV outlines the experimental setup and results. Finally, Section V discusses the findings and their implications, followed by concluding remarks in Section VI.

November 29, 2024

II. Theoretical Background

A. Supervised Learning & Binary Classification

Supervised learning (see [6]) is a core concept in machine learning, involving the use of labeled data to train a model that maps inputs to outputs. The objective is to learn a function $f : R^d \to \mathcal{Y}$ from a given dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, where $x_i \in R^d$ represents the input features, and $y_i \in \mathcal{Y}$ represents the corresponding outputs or labels. This function is then used to predict outputs for new, unseen inputs.

The subset of supervised learning that the paper focuses on is binary classification. The label set $\mathcal{Y}$ is binary, typically $\{0, 1\}$ or $\{-1, 1\}$. The aim is to categorize inputs into one of two classes by finding a decision boundary that separates

*Efstratios Reppas is with the Department of Electrical and Computer Engineering, National Technical University of Athens, Zografou, Attiki, 157 72 Greece e-mail: el20002@mail.ntua.gr.

the classes. To achieve this, models optimize a chosen loss function, such as the log-loss or hinge loss, to minimize classification errors on the training data while ensuring generalization to new data.

Binary classification problems are prevalent in numerous applications, including spam detection, medical diagnosis, and fraud detection. Commonly used algorithms for solving such problems include logistic regression, support vector machines (SVMs), and decision trees. These methods often involve balancing model complexity with the need to avoid overfitting, ensuring robust performance on unseen data.

B. Mathematical Programming

Mathematical programming (see [5]) is a branch of optimization that focuses on solving problems by selecting the best decision variables to optimize an objective function, often subject to a set of constraints. The general form of a mathematical programming problem can be expressed as:

$$\min_x f(x) \quad s.t. \quad g_i(x) \leq 0, \forall i, \quad h_j(x) = 0, \forall j \quad [1]$$

where $x \in R^n$ is called the decision variable,$f(x)$ is called the objective function to be minimized (or maximized, depending on the setting), $g_i(x)$ represents an inequality constraint, and $h_j(x)$ represents an equality constraint.

Non-linear optimization, which is discussed in this paper, deals with problems where the objective function or some of the constraints are non-linear. These problems are significantly more challenging than linear optimization due to the potential presence of multiple local minima and non-convexity, that increase the complexity of the algorithms solving this issues.

The decision variable x can be continuous, discrete, or a mix of both, depending on the nature of the problem. The authors of the paper make use of Mixed-Integer Programming (MIP), which is a specialized area of mathematical programming where some of the decision variables are constrained to take integer values, while others can be continuous. That complicates the situation even further, as the inclusion of integer variables makes these problems combinatorial in nature, often requiring sophisticated algorithms such as branch-and-bound, branch-and-cut, or heuristic methods for efficient solving.

However, it is worth noting that many algorithms for solving these problems exist, that have been implemented in various programming languages and work sufficiently well when the number of different variables and contraints is relatively low. We will be discussing more details about these solvers in Section IV.

C. Support Vector Machines

The topics above are both present in Support Vector Machines (SVMs), a widely used algorithm for supervised learning tasks, particularly binary classification (see [3]). The core idea of SVMs is to find the optimal hyperplane that maximally separates the data points of two classes in a high-dimensional feature space. Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, where $x_i \in R^d$ and $y_i \in \{-1, 1\}$, the objective is to determine a hyperplane in order for the classifier h to assign labels to its input, as defined by the equation:

$$h(x) = sign(w^\top x + b) \quad [2]$$

where $w \in R^d$ the weight vector, and $b \in R$ is the bias term.

The SVM formulation seeks to find that hyperplane that maximizes the distance between itself and the nearest data points (called support vectors) from either class. This is achieved by solving the following Non Linear Optimization problem:

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 s.t. \quad y_i(w^\top x_i + b) \geq 1, \forall i. \quad [3]$$

It is worth mentioning, although not essential for understanding the paper in question, that SVMs can be used to solve non-linearly separable classification problems as well. These are problems in which the classes cannot be separated by a hyperplane as denoted in equation [2], but need a more elaborate shape. That can be achieved through the use of slack variables in the objective function and kernels, that transform the problem in higher dimensions, where it can become linearly separable.

A variant of the main SVM paradigm that is extensively used in this paper is the Re-label Support Vector Machines (RE-SVM) as described in [4], which was designed to improve robustness in datasets with noisy labels. The key idea behind RE-SVM is to simultaneously construct the separating hyperplane like the regular SVM and also allow the relabeling of certain observations during the training process. This approach acknowledges that labels in the training data may not always be reliable and therefore relabeling can result in a more accurate classification rule.

The RE-SVM problem is formulated as follows:

$$\min_{w, w_0, e, \xi} \frac{1}{2}\|w\|^2 + c_1 \sum_{i=1}^n e_i + c_2 \sum_{i=1}^n \xi_i \quad [4]$$

subject to:

$$(1 - 2\xi_i) y_i (w^\top x_i + w) \geq 1 - e_i, \quad \forall i = 1, \ldots, n, \quad [5]$$

$$e_i \geq 0, \quad \xi_i \in \{0, 1\}, \quad \forall i = 1, \ldots, n. \quad [6]$$

In this formulation, w and w_0 are defined as above, e_i represents the misclassification error for the i-th observation and ξ_i is a binary variable that indicats whether the i-th observation is relabeled ($\xi_i = 1$) or not ($\xi_i = 0$). The c_1 and c_2 are cost parameters that trade off between misclassification errors and the cost of relabeling observations.

To elaborate more on the specifics of this formulation, e_i, is defined through the first constraint as:

$$e_i = max\{0, 1 - y_i(\omega^\top x_i + \omega_0)\}, \ if \ \xi_i = 0,$$

$$e_i = max\{0, 1 + y_i(\omega^\top x_i + \omega_0)\}, \ if \ \xi_i = 1.$$

The definition above states that if the label is flipped for a specific datapoint, then the classification error's sign is reversed, signifying that change. Therefore, during the minimization of the objective fuction, the model is allowed to essential consider the opposite sign for a small number of observations, providing that this helps minimize the entire objective function. This allows RE-SVM to better handle noisy datasets by modifying incorrect labels when necessary.

Empirical results demonstrate that RE-SVM outperforms classical SVM and other robust SVM-based methodologies, particularly in datasets where label noise is prevalent, making it a promising approach for robust binary classification.

D. Classification Trees (CTs) and Optimal Classification Trees (OCTs)

Classification trees are another widely used classification method. This method organizes data hierarchically through a tree-like structure, where each node represents a classification rule that split the feature space in half. The tree is constructed by recursively partitioning the feature space into smaller, more homogenous subsets, ultimately assigning a label to each terminal leaf node.

Each such split in a decision tree is determined based on a criterion that aims to maximize the homogeneity of the resulting subsets. Traditional decision trees, such as those constructed using the CART (Classification and Regression Trees) algorithm, which is the algorithm discussed in the paper, rely on heuristic, greedy approaches to identify the splitting rule; at each node, the algorithm selects the split that minimizes impurity at a local level, without considering the global structure of the tree. While these methods are computationally efficient, they may lead to suboptimal trees with poor generalization performance or excessive complexity (overfitting). These trees usually consider a single feature in the splitting rule at each level, and the way they identify this is through heuristic functions called impurity measures. Common such measure include the **gini index** defined as:

$$G = 1 - \sum_{k=1}^{K} p_k^2 \quad [8]$$

where p_k is the proportion of observations belonging to class k. and **entropy**:

$$H = -\sum_{k=1}^{K} p_k \log(p_k). \quad [9]$$

The CART algorithm ([2]) actually makes use of the gini index, which is more computationaly efficient than entropy and yields similar performance, an advantage that has established it as a baseline Decision Tree algorithm.

In that context, the author is mainly concerned with the notion of Optimal Decision Trees (OCT) (see [7]). These trees address the limitations of traditional methods by formulating the tree construction as a global optimization problem. They are mainly focused on providing a tree that has some sense of optimality, as the tree is constructed to minimize an objective function that balances classification accuracy and tree complexity. This objective function can be written as:

$$\min \sum_{t \in \mathcal{L}} L_t + \alpha \sum_{t \in \mathcal{B}} d_t \quad [10]$$

where $\mathcal{L}$ is the set of leaf nodes, L_t is the mis-classification cost at leaf t, $\mathcal{B}$ is the set of branch nodes, d_t is a binary variable indicating whether a split occurs at node t and α is a regularization parameter that penalizes tree complexity. Of course, there are several constraints that define the meaning of these variables and also prohibit illogical values for the variables, however these will be further explored in the next Section.

The optimization problem involves binary decision variables to represent the splits and the assignment of observations to nodes. Compared to heuristic methods, OCT produces trees that are in fact globally optimal with respect to the given objective, capturing the underlying structure of the data in a more effective matter. However, that comes at a cost of computational efficiency, as solving the mathematical program as stated above is significantly more complex than the greedy approach of the heuristic algorithms.

Finally, OCT-Hs extend the method of the OCTs by using a hyperplane in each intermediate node as the separation rule, instead of using a single feature at each level which was the case for the previous models. This constitutes a generalized version of these algorithms, OCT can be thought of OCT-H but with weights equal to zero in every other feature except for a specific one in each level. This extends the geometry of the separation rules; while OCT and the other paradigms created hyperplanes that were parallel to the axis of the feature they examined in each level, OCT-H can have virtually unbounded such hyperplanes, in terms of orientation.

III. Model Formulation

It has been illustrated that OCTs provide an optimality aspect to the tree they construct, by minimizing the misclassification errors and the number of nodes used in the tree. OCT-Hs extend this notion by allowing the separation rule to be a hyperplane instead of a line parallel to the features' axis, by evaluating all features in each node instead of one at a time. however, there is no sense of optimality embedded in these models in terms of class separation. More specifically, the hyperplanes can have arbitrary distances from the classes they separate, as long as they have minimal missclassification errors in the training set. That could potentially hinder their generalization abilities, as the hyperplane could be very close to one of the classes (a form of overfitting). Also, there is no concern about noisy labels, and that means that if our dataset is noisy, the model, trying to minimize its errors, might end up expressing misguided classification rules.

The authors of [1] propose the OCTSVM model, a novel classification algorithm that is formulated as a Mixed Integer Non Linear Problem (MINLP), to ameliorate these issues. This model builds above the OCT-H paradigm, and attempts to include an additional, SVM-like optimality objective, by maximizing the margin between the separating hyperplane and the two classes. At the same time, they incorporate the relabelling idea from RE-SVMs to attribute additional robustness to the model.

It is also worth mentioning that now each node provides a complete classification for the input. That means that there is no need to make a distinction between the intermedoate and leaf nodes, as the separating rule in every node will suffice making a classification prediction. Therefore, this model requires on less level of depth in its tree to segment the observation space as many times as the other tree classifiers.

For the following subsections, a detailed explanation of the mathematical formulation of the model will be provided, in a constructive manner, meaning that the final formulation will be derived progressively.

A. SVM Objective Function

The first element to consider is the hyperplanes' weights. This is one of the main ways this model differentiates itself from the rest. More specifically, consider N observations in the training set, in each node $t \in 1, ..., T$ of the total that the tree can be comprised of, there exists a weight vector ω_t and its bias term ω_{t0} that creates a hyperplane $\omega_t^\top x + \omega_{t0} = 0$. If the input lies at the left of this hyperplane (negative sign) it will be propagated to the left child node, else it will be propagated to the right one, with their own separation rules. This hyperplane will attempt to maximize the margine $\frac{2}{\|\omega\|}$ between itself and the classes, or equivalently minimize its inverse. Therefore, similar to the SVM, we have the first constraint:

$$\frac{1}{2}\|\omega_t\|_2 \leq \delta \quad \forall t = 1, \ldots, T.$$

and δ will be minimized in the objective function, indirectly minimizing the left side of the inequality that we are interested in. This constraint provides our model's formulation with its non-linear characterization.

B. Relabeling

In order to include the relabeling capability to the model, we will have to include in the objective function the following, similar to RE-SVM:

$$c_1 \sum_{i=1}^{n} \sum_{t=1}^{T} e_{it} + c_2 \sum_{i=1}^{n} \sum_{t=1}^{T} \xi_{it}$$

The e_{it} and ξ_{it} are similarly defined as the misclassification error and binary indicator of label flipping for observation i respectively, but now there is one such variable in each node t, and that is why we have to sum them in that direction as well. The latter variable is also the first variable that constitutes our problem Mixed-Integer. The c_1 and c_2 parameters again play the role of balancing the two objectives.

Now, we have to use a constraint in order to define the aforementioned variables accordingly. That will be the following:

$$y_i(\omega_t x_i + \omega_{t0}) - 2y_i(\beta_t x_i + \beta_{it0}) \geq 1 - e_{it} - M(1 - z_{it}),$$
$$\forall i = 1, \ldots, N, t = 1, \ldots, T. \quad [12]$$

$$\beta_{itj} = \xi_{it}\omega_{tj}, \quad \forall i = 1, \ldots, N,\ t = 0, \ldots, T,\ j = 0, \ldots, p. \ [13]$$

b_t is an auxiliary variable vector, derived from b_{itj} (j represents the j-th feature of the observation, out of p in total), that merely becomes equal to the weights ω_t if the label flip indicator is 1. Also, $z_{it} \in \{0, 1\}$ is another binary value that indicates whether observation i is used in the separation rule of node t. According to that, we have that if

$$z_{it} = 0 \Rightarrow f(\omega, \xi, e) \geq -\infty \quad [14]$$

which always holds, so the constraint becomes irrelevant if the observation i isn't considered in node t. Now, considering that $z_{it} = 1$ we will have:

$$z_{it} = 1, \xi_{it} = 0 \Rightarrow e_{it} \geq 1 - y_i(w_t x_i + w_{t0}) \quad [15]$$

$$z_{it} = 1, \xi_{it} = 1 \Rightarrow e_{it} \geq 1 + y_i(w_t x_i + w_{t0}) \quad [16]$$

Which is precisely the constraint of RE-SVM. So that means that this contraint implements the RE-SVM for each level of the tree.

C. OCT constraint

In order to incorporate the minimal split objective as in the OCTs (the misclassification error has already been implemented as part of RE-SVM), we have to add the$c_3 \sum_{t=1}^{T} d_t$ in the objective function, where d_t is likewise the binary variable indicating whether node t performs a split - effectively if is used, and c_3 is its corresponding balancing parameter. In order to define d_t in our formulation, we will add the following constraints:

$$\|\omega_t\|_2 \leq M d_t \quad \forall t = 1, \ldots, T. \quad [17]$$

$$d_t \leq d_p(t) \ \forall t = 1, \ldots, T. \quad [18]$$

where M is a very big number (denoted as ∞ in the explanations, to express its intuition) and $p(t)$ denotes the parent of node t.

Equation [17] states that if $d_t = 0$, then it becomes $\|\omega_t\|_2 \leq 0 \Rightarrow \|\omega_t\|_2 = 0 \quad \forall t = 1, \ldots, T.$ and if $d_t = 1$ then $\|\omega_t\|_2 \leq \infty \quad \forall t = 1, \ldots, T.$ which always holds. Therefore, it zeroes-out the weights of the hyperplane of node t, if it is also to zero, effectively performing its function. Equation [18] merely states that if a node is excluded from spliting, then all its successors cannot be performing a split either, as effectively they are pruned from the tree.

Having defined the above, the final objective function to be minimized is:

$$\min(\delta + c_1 \sum_{i=1}^{n} \sum_{t=1}^{T} e_{it} + c_2 \sum_{i=1}^{n} \sum_{t=1}^{T} \xi_{it} + c_3 \sum_{t=1}^{T} d_t) \quad [19]$$

D. Sanity constraints

At this point, we have defined all the constraints and variables needed to perfrom the optimization task. However, we need to make sure that the values attributed to the variables are well-defined, and and retain the overall structure of the tree the model is trying to build. Therefore, a few sanity constraints need to be set.

We begin by specifying that each observation i can be used to train the model at exactly one node of each level of the tree. That can be done by constraining the z_{it} value:

$$\sum_{t \in u} z_{it} = 1 \quad \forall i = 1, \ldots, N, \forall u \in U \quad [20]$$

where U is the set of the levels of the tree. Since z_{it} is a binary value, it can only take the value 1 in a node per level, for each observation.

Then, we must prohibit an observation i, if not present in a node t. to be present in its successor nodes, in order to retain the tree's hierarchical structure. That is implemented by the following constraint:

$$z_{it} \leq z_{ip(t)} \;\; \forall i = 1, \ldots, N, t = 2, \ldots, T \quad [21]$$

That prohibits the binary indicator of the observation i belongs to node t, if t's parent doesn't use it to. Of course, that is recursively true for the rest of the nodes under the initial one. The equation holds for all t except the first, as it has no parent.

Lastly, we need to consider how observations are going to be inherited from node to node, meaning that whether an observation will be passed to the left node or the right node as it propagates through the tree. that will happen with the following set of constraints:

$$\omega_t x_i + \omega_{t0} \geq -M(1-\theta_{it}) \;\; \forall i = 1, \ldots, N, t = 1, \ldots, T. \quad [22]$$

$$\omega_t x_i + \omega_{t0} \leq M\theta_{it} \;\; \forall i = 1, \ldots, N, t = 1, \ldots, T. \quad [23]$$

where $\theta_{it} \in \{0, 1\}$ is a new binary constraint that indicates whether the observation is on the left (0) or on the right (1) of the separation plane. These constraints essentially express the following:

$$w_t x_i + w_{t0} \geq 0 \Rightarrow \theta_{it} = 1 \quad [24]$$

$$w_t x_i + w_{t0} \leq 0 \Rightarrow \theta_{it} = 0 \quad [25]$$

] correctly expressing the function of the θ variable. Now, having defined this auxiliary variable, we can model the inheritance of the observations from node to node:

$$z_{ip(t)} - z_{it} \leq \theta_{ip(t)} \;\; \forall i = 1, ..., N, t \in \tau_{bl} \quad [26]$$

$$z_{ip(t)} - z_{it} \leq 1 - \theta_{ip(t)} \;\; \forall i = 1, ..., N, t \in \tau_{br} \quad [27]$$

Here, the τ_{bl} and τ_{br} sets contain the nodes that are on the left side of their parent and on the right respectively. That can be implemented through coding in by indexing the former with even numbers and the latter with odd. What these constraints express, regarding the constraint [26] (equation [27] is exactly the same, but because the θ must now be the inverse, the formulation is different) is that if $\theta_{ip(t)} = 0, z_{ip(t)} = 1 \Rightarrow z_{it} = 1$, so it forces the left succeeding node to utilize the observation.

The final Mixed Integer Non Linear Program is the following:

$$\min \; \delta + c_1 \sum_{i=1}^{n} \sum_{t=1}^{T} e_{it} + c_2 \sum_{i=1}^{n} \sum_{t=1}^{T} \xi_{it} + c_3 \sum_{t=1}^{T} d_t$$

$$\begin{aligned}
&s.t. \quad \tfrac{1}{2}\|\omega_t\|_2 \leq \delta, \quad \forall t = 1, \ldots, T, \\
&y_i(\omega_t' x_i + \omega_{t0}) - 2y_i(\beta_t' x_i + \beta_{it0}) \geq 1 - e_{it} - M(1 - z_{it}), \\
&\forall i = 1, \ldots, N, \; t = 1, \ldots, T, \\
&\beta_{itj} = \xi_{it}\omega_{tj}, \quad \forall i = 1, \ldots, N, \; t = 0, \ldots, T, \; j = 0, \ldots, p, \\
&\|\omega_t\|_2 \leq M d_t, \quad \forall t = 1, \ldots, T, \\
&d_t \leq d_p(t), \quad \forall t = 1, \ldots, T, \\
&\textstyle\sum_{t \in u} z_{it} = 1, \quad \forall i = 1, \ldots, N, \; u \in U, \\
&z_{it} = z_{ip(t)}, \quad \forall i = 1, \ldots, N, \; t = 2, \ldots, T, \\
&\omega_t' x_i + \omega_{t0} \geq -M(1 - \theta_{it}), \quad \forall i = 1, \ldots, N, \; t = 1, \ldots, T, \\
&\omega_t' x_i + \omega_{t0} \leq M\theta_{it}, \quad \forall i = 1, \ldots, N, \; t = 1, \ldots, T, \\
&z_{ip(t)} - z_{it} \leq \theta_{ip(t)}, \quad \forall i = 1, \ldots, N, \; t \in \tau_{bl}, \\
&z_{ip(t)} - z_{it} \leq 1 - \theta_{ip(t)}, \quad \forall i = 1, \ldots, N, \; t \in \tau_{br}, \\
&e_{it} \in R^+, \quad \xi_{it} \in \{0, 1\}, \quad \beta_{it0} \in R, \\
&\theta_{it} \in \{0, 1\}, \quad \forall i = 1, \ldots, N, \; t = 1, \ldots, T, \\
&\omega_t \in R^p, \quad \omega_t \in R, \quad d_t \in \{0, 1\}, \quad \forall t = 1, \ldots, T.
\end{aligned}$$

IV. Experimental Reproduction

Having formulated the OCTSVM, we can now proceed in reproducing the experimental results from [1].

In this work, the authors of [1] tested their model alongside CART, OCT, OCT-H which have already been described. They tested these models on 8 different UCI Machine Learning Repository datasets ([8]). As they aimed to test their model's robustness in noisy datasets, they also applied different amounts of noise (label flips) in these datasets, to a certain percentage of the total values each time. They compared the models in terms of average accuracy, using cross-validation.

Dataset	% Flips	CART		OCT		OCTSVM	
		Paper	Ours	Paper	Ours	Paper	Ours
Australian	0	82.99	**85.51**	85.44	83.48	**86.34**	84.5
	20	74.85	83.47	83.45	**85.5**	**84.55**	78
	30	66.87	**82.03**	79.15	70.72	**80.24**	72.08
	40	56.93	**76.67**	71.34	68.68	**73.89**	55.94
	Avg.	70.41	**81.92**	79.48	77.1	**81.26**	72.63
BreastCancer	0	92.22	94.56	93.18	94.71	**96.25**	**95.57**
	20	90.47	93	90.92	88.13	**91.57**	**95.28**
	30	83.29	93.42	**90.87**	89.56	87.98	**92.41**
	40	77.75	87.69	**86.50**	86.82	81.92	**92.27**
	Avg.	85.93	92.17	**90.37**	89.81	89.43	**93.88**
Heart	0	73.66	77.37	73.88	77.01	**84.13**	**84.84**
	20	72.98	67.78	71.63	71.09	**82.61**	**81.1**
	30	68.22	65.57	70.18	68.87	**80.51**	**79.98**
	40	62.36	66.62	64.52	63.77	**76.19**	**68.54**
	Avg.	69.31	69.34	70.05	70.16	**81.26**	**78.62**
Ionosphere	0	83.08	**89.45**	81.80	82.62	**85.51**	89.16
	20	75.65	**81.2**	79.59	78.91	**80.51**	74.95
	30	70.02	73.53	75.20	**76.08**	**77.79**	50.12
	40	60.17	64.1	70.65	**75.81**	**75.71**	43.29
	Avg.	72.23	77.07	76.81	**78.36**	**79.88**	64.38
MONK's	0	60.67	78.01	59.03	79.63	**61.26**	**85.18**
	20	57.11	75	59.59	75.46	**60.48**	**78.7**
	30	57.19	76.15	57.87	**77.55**	**60.09**	73.14
	40	54.01	64.81	55.25	**75**	**60.12**	74.07
	Avg.	57.25	73.49	57.94	76.91	**60.49**	**77.77**
Sonar	0	65.36	71.63	66.90	69.23	**74.76**	**79.33**
	20	57.41	60.58	60.38	**67.3**	**69.92**	65.38
	30	54.64	59.13	58.97	**62.98**	**67.30**	62.02
	40	55.16	55.77	57.77	**62.01**	**63.84**	56.25
	Avg.	58.14	61.78	61.01	65.38	**68.96**	**65.75**
Wholesale	0	**90.28**	**91.59**	90.01	91.14	89.75	88.41
	20	83.79	88.86	87.84	**90.23**	82.06	87.72
	30	78.54	84.09	**84.58**	84.77	78.13	**85.91**
	40	69.52	67.72	**75.79**	76.14	71.72	**85.23**
	Avg.	80.53	83.07	**84.56**	85.57	80.42	**86.81**
Total Avg.		69.92	76.98	73.71	**79.14**	**77.02**	77.12

TABLE I
Accuracy (%) comparison of the models between [1] and our experiments (**bold** is best).

More specifically, the authors examined the Australian, BreastCancer, Heart, Ionosphere, MONK's, Parkinson, Sonar and Wholesale datasets - more specific details are discussed in [1]. For each dataset, they applied three different noise levels by flipping 20%, 30% and 40% of the training dataset's labels. For each such instance, they measured the average accuracy across a 4-fold cross-validation scheme. They performed this process for each model they tested using different model parameter values, displaying the one that achieved the highest accuracy in each case. The search space for these parameters was the set $\{10^i : i \in [-5, 5]\}$ for the single parameter in the case of CART, OCT and OCT-H, while they used the same search space for the c_1 and c_2 parameters of OCTSVM, and a space of $\{10^i : i \in [-2, 2]\}$. Also, the authors used a max depth of 3 for the CART, OCT and OCT-H, which translates to a max depth of 2 for OCTSVM, as discussed in the previous sections. For the runtime of the mathematical programming models, they set a time limit of 30 seconds per experiment.

Due to the scope of this project and the limited time given to perform the experiments, a few amendments to the authors' approach were made. First of all, the OCT-H model was excluded from the comparison, mainly due its highly time-consuming training. For the same reason, the search space for the OCTSVM hyperparameters was reduced to $\{10^i : i \in [-3, 1]\}$ and $\{10^i : i \in [-2, 1]\}$ respectively for each setup. The reasoning behind the specific values was that in the few experiments we ran with the complete setup, the excluded values appeared to be ineffective. One final remark is that the Parkinson dataset was omitted from the experiment list, as we encountered issues when processing this specific dataset.

The comparison between the results obtained in our experiments and the experiments of [1] is listed in Table 1. This further confirms the main observation from [1]: The OCTSVM often achieved the superior performance in terms of accuracy among the models. In fact, that is true in 4 out of 7 datasets when no noise is applied and in 5 out of 7 datasets taking the average across all noise levels. In fact, it exhibits a remarkable 10.75% improvement in accuracy over the second best model in the Sonar dataset when no noise is applied and over 12% in the average accuracy in the Heart dataset. However, this analysis showed a discrepancy with [1] in that the other models often appear to be even more capable than OCTSVM in high noise levels, although they were not necessarily intentionally designed to handle such cases. Actually, it seems that the exceptional results of the OCTSVM in low noise levels attributed it with its dominant average performance, as it fails to achieve the greatest accuracy in high noise levels (30% & 40%) in 4 out of 7 datasets. An observation that should be highlighted is also its breakdown in performance in high noise levels in the Australian and Ionosphere datasets, a factor that probably resulted in OCTSVM not achieving the highest average accuracy across all datasets, in spite of it having the greatest average accuracy in the majority of datasets. However, this performance drop in noisy datasets could possibly be attributed to the limited search space for the model's parameters, as they would be more crucial when datasets exhibit high levels of noise. Another point worth emphasizing is the fact that OCTSVM used only the first level of its tree out of the two in many cases, essentially reducing itself to a RE-SVM. That perhaps puts the effectiveness of the additional complexity of the tree structure in question, as that might seem redundant. The last observation that we will highlight is that CART exhibits greater performance in our analysis in contrast to [1] - larger than 10% in certain cases. This resulted in the total average of CART being very close with the accuracy of the other models, within a 3% difference regarding OCT which had the best overall performance and less than 0.2% regarding OCTSVM. Given the significantly faster runtime of CART - notably it performed all of the experiments within a couple of seconds, whereas the other models required hours if not days to execute them - the applicability of these mathematical programming models is rather questionable.

V. Conclusions

This report has summarized and reproduced the findings from the paper "Robust Optimal Classification Trees Under Noisy Labels" [1] to evaluate the proposed OCTSVM model. The results confirm that OCTSVM generally achieves superior performance in low noise levels, demonstrating a marked improvement in accuracy compared to baseline models like CART, OCT, and OCT-H. Notably, OCTSVM exhibited up to a 12% accuracy improvement in certain datasets, affirming its robustness and utility in handling label noise in supervised classification tasks.

However, the analysis also highlighted certain issues. Firstly, the model's performance diminished under high noise levels in specific datasets, such as Australian and Ionosphere. Although these results might have been caused by certain discussed limitations of our reproduction, they might suggest a limited capacity of the model to adapt in high-noise scenarios. Furthermore, observations indicate that OCTSVM often reduced its complexity to resemble a RE-SVM, raising questions about the necessity of its complex tree structure.

Another key finding was the unexpectedly competitive performance of CART in this reproduction. In combination with its runtime efficiency, it poses important and potentially insurmountable challenges to the broader applicability of optimization-based models like OCTSVM in real-world settings where computational resources and time might be limited.

In conclusion, while OCTSVM demonstrates strong potential for classification tasks in noisy environments, its reliance on extensive computational resources and possibly its susceptibility to parameter tuning raise a demand for further exploration. Future work should investigate more efficient parameter tuning strategies and adaptive mechanisms to enhance robustness under diverse noise conditions, along with examining the scalability of such models for larger datasets and real-time applications.

References

[1] V. Blanco, A. Japón, and J. Puerto, "Robust optimal classification trees under noisy labels," arXiv:2012.08560 [cs.LG], Dec. 2020.

[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, 1st ed. Belmont, CA, USA: Wadsworth, 1984.

[3] V. Vapnik, *The Nature of Statistical Learning Theory*, 1st ed. New York, NY, USA: Springer, 1995.

[4] V. Blanco, A. Japón, and J. Puerto, "A mathematical programming approach to SVM-based classification with label noise," *Computers & Industrial Engineering*, vol. 172, p. 108611, 2022.

[5] H. A. Taha, *Operations Research: An Introduction*, 10th ed. Upper Saddle River, NJ, USA: Pearson, 2017.

[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.

[7] D. Bertsimas and J. Dunn, "Optimal classification trees," *Machine Learning*, vol. 106, no. 7, pp. 1039–1082, 2017.

[8] Dua, D., and Graff, C. UCI machine learning repository, 2017

PLACE
PHOTO
HERE

Efstratios Reppas Efstratios Reppas is a senior student in Electrical and Computer Engineering in the National Technical Univesity of Athens.