Learning comprehensible and accurate hybrid trees

Rok Piltaver[1,2], Mitja Luštrek[1], Sašo Džeroski[2,3], Martin Gjoreski[1,2,], Matjaž Gams[1,2]*

[1]Department of Intelligent Systems – Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
[2]Jozef Stefan International Postgraduate School, Jamova cesta 39, 1000 Ljubljana, Slovenia
[3]Department of Knowledge Technologies – Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
[*]Corresponding author. Phone number: +386 30 417 622
(rok.piltaver, mitja.lustrek, saso.dzeroski, martin.gjoreski, matjaz.gams)@ijs.si

Keywords: hybrid tree; multi-objective learning; comprehensibility; accuracy; classification.

Statement of extensions: This paper is extended and improved version of the paper published in the Proceedings of European Conference on Artificial Intelligence 2014 entitled "Multi-objective Learning of Hybrid Classifiers" and a PhD dissertation "Constructing comprehensible and accurate classifiers using data mining algorithms" in 2016. The main improvements over the conference paper are the following: rewritten and extended sections 2, 3 and 4 (related work, hybrid tree model description and an in-depth formal explanation of the algorithm), and implementation of the algorithm in the Orange data-mining suite.

Abstract

Finding the best classifiers according to different criteria is often performed by a multi-objective machine learning algorithm. This study considers two criteria that are usually treated as the most important when deciding which classifier to apply in practice: comprehensibility and accuracy. A model that offers a broad range of trade-offs between the two criteria is introduced because they conflict; i.e., increasing one decreases the other. The choice of the model is motivated by the fact that domain experts often formalize decisions based on knowledge that can be represented by comprehensible rules and some tacit knowledge. This approach is mimicked by a hybrid tree that consists of comprehensible parts that originate from a regular classification tree and incomprehensible parts that originate from an accurate black-box classifier. An empirical evaluation on 23 UCI datasets shows that the hybrid trees provide trade-offs between the accuracy and comprehensibility that are not possible using traditional machine learning models. A corresponding hybrid-tree comprehensibility metric is also proposed. Furthermore, the paper presents a novel algorithm for learning MAchine LeArning Classifiers with HybrId TrEes (MALACHITE), and it proves that the algorithm finds a complete set of nondominated hybrid trees with regard to their accuracy and comprehensibility. The algorithm is shown to be faster than the well-known multi-objective evolutionary optimization algorithm NSGA-II for trees with moderate size, which is a prerequisite for comprehensibility. On the other hand, the MALACHITE algorithm can generate considerably larger hybrid-trees than a naïve exhaustive search algorithm in a reasonable amount of time. In addition, an interactive iterative data mining process based on the algorithm is proposed that enables inspection of the Pareto set of hybrid trees. In each iteration, the domain expert analyzes the current set of nondominated hybrid trees, infers domain relations, and sets the parameters for the next machine learning step accordingly.

1. Introduction

The recent General Data Protection Regulation ("GDPR") posed by the EU requires machine learning models that can explain the provided output. This requirement tasked the machine learning community to develop novel techniques that minimize the trade-off between the comprehensibility and the performance of the models. Some researchers even argue that a model's accuracy should be sacrificed and that comprehensible models should be preferred over black-box models for high stakes decisions (Rudin, 2019). In this paper, we present a novel method that combines black-box models and comprehensible models to bring about the best of both worlds, i.e., to create comprehensible and accurate models. Furthermore, the proposed method generates a set of classifiers in such a way that it supports the user in making the decision of how much accuracy she is willing to sacrifice to improve the comprehensibility, and vice versa

Our approach to generating combined black-box and comprehensible classifiers resembles the was human experts express their knowledge. In various complex domains, they can often explain only parts of their knowledge with simple rules. The remainder of their knowledge is tacit; as a result, while they can apply it, they cannot easily express it. The goal of this paper is thus to learn classifiers that provide good trade-offs between accuracy and comprehensibility, which are both equally important objectives of data mining. The hybrid tree is chosen as the model that enables trade-offs between the two objectives. The regular leaves of the hybrid tree correspond to comprehensible knowledge because they explain the classification of the instances that fall into the regular leaves. On the other hand, instances that fall into so-called black-box leaves are classified with a black-box classifier, which corresponds to the tacit knowledge. The black-box leaves do not offer a classification explanation; however, they increase the overall accuracy. The goal of the learning algorithm is therefore to find a hybrid tree that classifies the simple part of the domain with regular leaves while it uses black-box leaves to classify the complex part.

The first difficulty in learning accurate and comprehensible classifiers arises from the fact that the objectives must be considered at the same time and that they are conflicting (Jin and Sendhoff, 2008) —increasing one decreases the other and vice versa. The two objectives can be achieved with two general approaches (Jin and Sendhoff, 2008; Freitas, 2004). The traditional weighted-formula approach transforms the multi-objective problem into a single-objective problem. The second approach, which is becoming increasingly more popular over the past two decades, is the Pareto-based multi-objective approach, which enables treating all of the objectives simultaneously and independently. Freitas (2004) analyzes arguments for and against each of the two approaches: the Pareto-based approach is more complex; however, it is preferred because it avoids multiple runs of a single-objective optimization algorithm, it does not require an ad hoc specification of the optimization parameters (i.e., weights), and it provides an informative set of Pareto-optimal classifiers (Deb, 2008). Therefore, the latter approach is used to achieve the goal of this paper. Therefore, instead of focusing on the classification accuracy only, our approach returns a Pareto set of classifiers from the most comprehensible to the most accurate ones, supporting the user when deciding how much accuracy to sacrifice for comprehensibility and vice versa.

The second problem in achieving the goal of this paper is to choose an appropriate knowledge representation that offers a wide range of classifiers from the most comprehensible classifiers to the most accurate ones. The main purpose of this paper is not only to improve the classification accuracy but also to provide a good trade-off between the comprehensibility and the accuracy.

We describe the novel Multi-Objective Learning of Hybrid Classifiers algorithm (MALACHITE) (Piltaver, Luštrek, Zupančič, Džeroski and Gams, 2014). We prove that it finds the complete set of nondominated hybrid trees. Additionally, we provide an empirical evaluation of the algorithm and propose a data-mining process designed around the MALACHITE algorithm. The proposed interactive and iterative data-mining process shown in Figure 1 advocates using a domain expert in the loop to combine an expert's knowledge and the power of machine learning. This process enables the expert to make a well-informed decision when selecting a classifier with a good trade-off between accuracy and comprehensibility and to obtain additional insights into the classification domain.

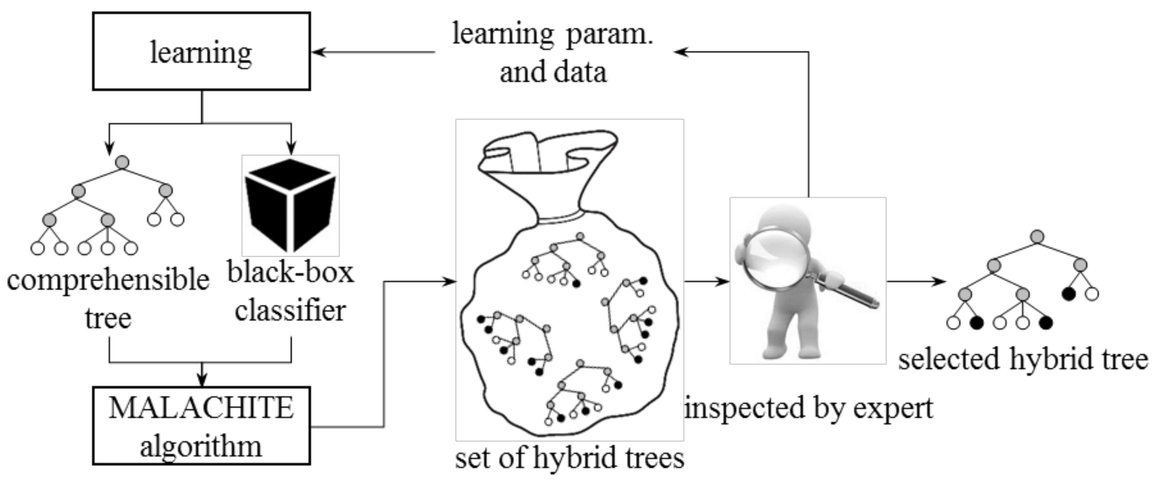

Figure 1: The proposed data-mining process uses multi-objective learning and a domain expert in a loop to combine a comprehensible tree and an accurate black-box classifier into a single hybrid tree.

The rest of the paper is organized as follows. A review of the related work is in Section 2. Section 3 describes the hybrid-tree classification model and defines an associated measure of comprehensibility. The pseudo code and explanation of the MALACHITE algorithm is given in Section 4. The algorithm's evaluation is presented in Section 5. First, the MALACHITE is shown to be significantly faster than a naïve exhaustive search, which is too slow to be used in practice (Section 5.1). Then, datasets from the UCI repository (Bache and Lichman, 2015; Witten, Frank and Hall, 2011) are described (Section 5.2) and used to show that MALACHITE outperforms the baseline algorithms (Section 5.3) and the well-known multi-objective optimization algorithm NSGA-II (Section 5.4). Limitations of the MALACHITE algorithm are discussed in Section 6, and the conclusions are given in Section 7.

2. Comparison with the related work

This paper is related to research from two areas: increasing the comprehensibility of machine learning models and constructing hybrid models that combine multiple types of classification models.

Some established machine-learning algorithms already use criteria other than the accuracy in the learning process. The RIPPER rule induction algorithm (Cohen, 1995), for example, achieves compactness of the learned rule set by using the minimum description length principle (Rissanen, 1978). Girosi, Jones and Poggio (1995) present an algorithm that includes comprehensibility criteria in the regularization of artificial neural networks. Jin (2000), and Gorzałczany and Rudziński (2012) applied the same idea to construct fuzzy rules with evolutionary optimization methods. Czajkowski and Kretowski (2019) combined decision trees and evolutionary algorithms to solve the problem of underfitting without inducing a substantial increase in the overall complexity of the tree. The common property of these algorithms is that they combine the accuracy and comprehensibility into a single objective function using a weighted-formula approach—to build a single classifier—which was shown to be inferior to the multi-objective approach (Freitas, 2004). In contrast, the MALACHITE algorithm uses a Pareto-based multi-objective approach to generate a set of classifiers that range from the most comprehensible to the most accurate. The set of classifiers provides additional information that enables the user to efficiently make a well-informed decision when she selects the final classifier. This approach is preferred to the weighted-formula approach in which the user uses an iterative search for the appropriate weights to find settings that correspond to her preferences.

A recent study presented a rule-based credit risk assessment approach for building decision rules using comprehensibility and accuracy as a multi-objective criteria (Soui, Gasmi, Smiti, and Ghédira, 2019). This work presented an extensive comparison of four evolutionary algorithms for building rule-based models for binary classification of high *vs*. low risk loans. The MALACHITE algorithm differs from this work because it is based on Hybrid trees and is a general algorithm for solving any classification problem.

The work of Kottathra and Attikiouzel (1996), who formulated the training of an artificial neural network as a bi-objective minimization problem with the mean square error and the number of neurons as conflicting objectives, is one of the earliest examples of multi-objective machine learning. They solved the problem with a branch-and-bound algorithm. Similarly, Shen, Han, Aberle, Bui and Hsu (2019) presented an interpretable convolutional neural network for lung nodule malignancy classification. The network outputs low-level semantic features, which reflect diagnostic features that are often reported by radiologists, and a high-level prediction. The network is trained with a combined loss function that combines both the low- and high-level tasks.

Other researchers have mostly used evolutionary optimization methods. For example, two studies constructed fuzzy rules (Ishibuchi, Nakashima, and Murata, 2001; Pulkkinen, 2009); Bock (2017) and later Obregon Obregon, Kim and Jung (2019) combined and simplified rules that were extracted from boosted decision trees. Jin, Sendhoff and Körner (2006) constructed artificial neural networks; Markowska-Kaczmar and Mularczyk (2006) extracted rules from neural networks; Tušar (2007) and later Blanco-Justicia and Domingo-Ferrer (2019) constructed classification trees; and Clark and Everson (2012) constructed relevance vector machines. The goal of the algorithms mentioned above is to extract a comprehensible classifier from an accurate but incomprehensible classifier. On the other hand, the MALACHITE algorithm starts from a comprehensible classifier and improves its accuracy by introducing black-box hybrid leaves. The advantage is that the algorithm starts from a comprehensible classifier that is easy to validate and can be constructed manually by a domain expert or automatically by a machine-learning algorithm. In addition, the MALACHITE algorithm is guaranteed to find the complete Pareto-optimal set, which is an improvement over the commonly used stochastic optimization algorithms, e.g., NSGA-II (Deb, 2002).

Most likely the best-known example of hybrid machine-learning models is the model tree, which uses linear regression functions in tree leaves (Quinlan, 1992). The resulting models are accurate and compact. In a similar way, the NBTrees place Naïve Bayes classifiers in the leaves of the classification tree (Kohavi, 1996), which retains the comprehensibility of both model types and often improves the accuracy. The VFDTc algorithm for mining data streams similarly combines Naïve Bayes with Hoeffding trees, improving the accuracy compared to regular Hoeffding trees (Gama, Rocha, and Medas, 2003). Hybrid SVM-based classification trees (Kumar and Gopal, 2010) use an SVM classifier to classify examples that are close to the class boundary and a classification tree for the examples farther away. This approach results in a significant speedup over SVM alone, without any compromise in accuracy. The MALACHITE algorithm is similar to the algorithm listed above in that it combines a classification-tree model with another model; however, its purpose is not (only) to improve the classification accuracy of the models or to speed up the learning. The goal of the MALACHITE algorithm when replacing regular leaves in the initial classification tree with hybrid leaves that invoke a black-box classifier is to increase the classification accuracy in the parts of the domain that cannot be classified accurately using a comprehensible model. Hence, the resulting hybrid trees mimic a human expert's knowledge: with some comprehensible and some incomprehensible parts. The hybrid trees are ideally suited to searching for trade-offs between the accuracy and the comprehensibility, because the comprehensibility can be adjusted to anywhere between none and complete.

In addition, our research introduces further distinctions from the related work. First, most papers that discuss the comprehensibility of classification trees define it as the complexity of the tree, which is not appropriate for hybrid trees. Therefore, we define the comprehensibility as the fraction of examples classified by the comprehensible leaves of the hybrid tree. The paper also discusses an interactive learning process in which a human expert in each iteration analyzes the set of learned hybrid trees and, based on the obtained insights, selects the inputs for the next run of the MALACHITE algorithm. The proposed process can, for example, replace the *build model* and *assess model* tasks in the *modeling phase* of the Cross-Industry Standard Process for Data Mining (CRISP-DM)(Chapman et al., 2000)—a widely used data-mining process model that encompasses the entire life-cycle of a data mining project. The analysis of the learned hybrid trees can also provide valuable conclusions about how to perform tasks such as *clean data*, *construct data*, and *integrate data* in the CRISP-DM *data preparation phase.* To the best of our knowledge, there is no standard data-mining process for multi-objective data mining; however, the differences compared to traditional data mining are minor. Therefore, the MALACHITE algorithm can be simply used in most data-mining process models, for example, the models compared in the survey by Kurgan and Musilek (2006).

3. The suggested knowledge representation and approach

This section defines the notion of a hybrid tree classification model, its comprehensibility and accuracy, as used in this study. The Pareto dominance relation, which is used to compare the quality of hybrid trees, and the nondominated set (i.e., Pareto set), which is the output of the MALACHITE algorithm, are defined for the case of hybrid trees.

The basic idea of the MALACHITE algorithm is to generate hybrid trees by replacing some leaves of a given classification tree with black-box leaves that invoke a given black-box classifier, i.e., that trade comprehensibility for accuracy. The MALACHITE algorithm uses a given learning set of data instances to compare the classification accuracy of the initial classification tree and the black-box classifier in each part of the domain and to decide which parts of the domain should be classified with each of the two classifiers.

Definition (HYBRID TREE): Given an initial classification tree and a black-box classifier, we can represent a hybrid tree T_i with a Boolean vector $(t_{i,1}, t_{i,2}, \ldots, t_{i,n}) \in \{0, 1\}^n$ with n components, where n is the number of leaves in the initial classification tree in which the black-box classifier achieves higher accuracy than the initial classification tree. The values 0 and 1 for each component of the vector denote the type of corresponding leaf: 0 for regular (i.e., as in initial tree) and 1 for black-box leaves.

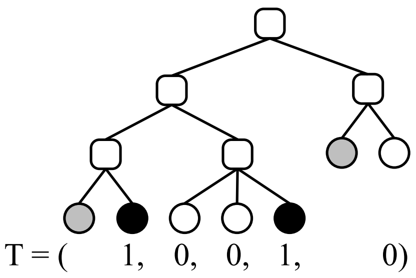

Figure 2: An example of a hybrid tree T = (1, 0, 0, 1, 0), where the black circles represent black-box leaves, the white circles represent regular leaves, and the gray circles represent leaves that will always remain regular because their accuracy is at least as good as the accuracy of the black-box classifier.

The initial classification tree is represented as $T_1 = (0, 0, \ldots, 0)$, while the hybrid-tree with the highest accuracy is represented as $T_{2^\wedge n} = (1, 1, \ldots, 1)$, where n is the number of leaves in which the black-box classifier achieves higher accuracy than the initial classification tree. It does not make sense to replace regular leaves with the black-box classifier if replacement does not improve the accuracy; therefore, such leaves are omitted from the vector representation.

The comprehensibility of the hybrid trees does not depend on only the structure of the tree but also on the number of examples that belong to the black-box leaves.

Definition (COMPREHENSIBILITY OF A HYBRID TREE): The comprehensibility $c_t \in [0, 1]$ of a hybrid tree t is defined by Equation 1 as the ratio between the sum of the number of examples N_i classified by a regular leaf i over all regular leaves and the number of all examples N used to evaluate the comprehensibility of the hybrid tree.

$$c_t = (\textstyle\sum_{i \notin \text{bb leaves}} N_i)/N \qquad (1)$$

The comprehensibility of a hybrid tree is therefore equal to the probability of classifying an instance with a comprehensible leaf. Therefore, the comprehensibility of the initial classification tree is 1, and the comprehensibility of the black-box classifier is 0.

Definition (ACCURACY OF A HYBRID TREE): The classification accuracy $a_t \in [0, 1]$ of a hybrid tree t is defined by Equation 2 as the ratio between the number of correctly classified examples and the number of all examples N used to evaluate the accuracy of the hybrid tree. The number of correctly classified examples is summed over all of the leaves in the hybrid tree: $N_{j,\mathrm{t}}$ denotes the number of correctly classified examples in a regular leaf j, while $N_{j,\mathrm{bb}}$ denotes the number of correctly classified examples in a black-box leaf j.

$$a_t = (\textstyle\sum_{j \notin \text{bb leaves}} N_{j,\mathrm{t}} + \sum_{j \in \text{bb leaves}} N_{j,\mathrm{bb}}) \,/\, N \tag{2}$$

Multi-objective optimization uses the Pareto dominance relation ($\preccurlyeq$) to compare the quality of the solutions (Deb, 2008) and hence return a set of solutions with different incomparable vector-values of the objective functions, as opposed to single-objective optimization, which returns the solution(s) with the best value of a scalar objective function. The quality of a hybrid tree T_i is given by the pair, $(a_i, c_i) \in [0, 1] \times [0, 1]$, where c_i is the comprehensibility, and a_i is the accuracy of the hybrid tree T_i. Because the two objectives are conflicting, the MALACHITE algorithm uses a Pareto-based multi-objective approach that returns a set of nondominated classifiers (defined by Equation 4). They provide optimal trade-offs between the accuracy and the comprehensibility.

Definition (DOMINANCE RELATION ON HYBRID TREES): Equation 3 defines the relation "hybrid tree T_x dominates hybrid tree T_y", where the goal is to maximize the accuracy and comprehensibility: hybrid tree T_x dominates hybrid tree T_y if it is better than T_y in at least one objective and no worse in the other.

$$T_x \preccurlyeq T_y \Leftrightarrow (a_x > a_y \wedge c_x \geq c_y) \vee (a_x \geq a_y \wedge c_x > c_y) \tag{3}$$

Definition (NONDOMINATED SET): A hybrid tree T_x is said to be nondominated, i.e., optimal, in a set of hybrid trees T if no hybrid tree that dominates T_x exists in T, as stated by Equation 4.

$$T_x \in nondom(T) \Leftrightarrow T_x \in T,\ \nexists T_y \in T\colon T_y \preccurlyeq T_x \tag{4}$$

A set of hybrid trees $P' = nondom(P) \subseteq P$ is the nondominated subset of the set of hybrid trees P if it contains exactly the hybrid trees that are not dominated by any member of the set P. The Pareto set (i.e., globally Pareto-optimal set) is the nondominated set of the entire solution space (Deb, 2008).

4. The MALACHITE algorithm

4.1. Preliminaries

The MALACHITE algorithm takes an initial comprehensible tree and replaces some of its leaves with a black-box classifier to produce hybrid trees that have higher classification accuracy (at the cost of lower comprehensibility) than the initial classification tree. The goal of the MALACHITE algorithm is to find the Pareto set of hybrid trees, from which the user selects one or more of the hybrid trees. Considering hybrid trees that are not in the Pareto set is not required, because it is guaranteed that all of them are worse in both accuracy and comprehensibility than at least one hybrid tree in the Pareto set. Analyzing the set of nondominated hybrid trees enables the user to extract additional knowledge about the domain and the classification task, which facilitates a well informed decision when choosing a single classifier from the Pareto set.

Any algorithm that learns hybrid trees performs a search in the space of hybrid trees, i.e., the Cartesian product $\{0, 1\} \times \{0, 1\} \times \ldots \times \{0, 1\} = \{0, 1\}^n$, which contains 2^n hybrid trees, where n is the number of leaves. The result of a learning algorithm is therefore a set of Boolean vectors that correspond to a set of hybrid trees, and a set of vectors with two components that represent the accuracy and comprehensibility of the hybrid trees.

A naïve algorithm that simply calculates the quality of all 2^n possible hybrid trees and selects the nondominated trees faces a combinatorial explosion in the search space and is therefore limited to small initial classification trees (up to ~20 leaves). To enable using considerably larger initial classification trees, the MALACHITE algorithm calculates the qualities of the hybrid trees using a dynamic programming approach. Therefore, the computing quality of a hybrid tree has a constant time complexity instead of the O(n) complexity of the naïve algorithm. Furthermore, MALACHITE avoids generating dominated hybrid trees, thus limiting the search space, i.e., the number of hybrid trees that must be generated and evaluated. As a result, MALACHITE performs an exhaustive search over a pruned space of all possible combinations of the initial classification tree and the black-box classifier. In contrast with stochastic evolutionary optimization algorithms, which are most often used in multi-objective learning, MALACHITE is an exact algorithm. In other words, the output of the MALACHITE algorithm is the complete Pareto set of hybrid trees and not only an approximation of the Pareto set. Nevertheless, the MALACHITE algorithm is computationally more efficient than the well-known multi-objective optimization algorithm NSGA-II when used with initial classification trees of modest size—a prerequisite for comprehensibility.

4.2. The algorithm

The MALACHITE algorithm (pseudo code given in Algorithm 1) starts by identifying the subsets of the learning examples that belong to each leaf of the initial tree (lines 1–4). The number of correctly classified instances (from the provided learning set) that belong to each leaf l is computed using the initial tree ($N_{j,\mathrm{t}}$ in line 5) and the black-box classifier ($N_{j,\mathrm{bb}}$ in line 6). These values are used to compute the relative difference in the accuracy $\delta_{l,a}$ (Equation 5) and comprehensibility $\delta_{l,c}$ (Equation 6) that is introduced by replacing leaf l with a black-box leaf (lines 8–15).

Definition (Relative Difference in Accuracy and Comprehensibility**)**: The relative difference in the accuracy $\delta_{l,a}$ and comprehensibility $\delta_{l,c}$ between a hybrid tree and its variant obtained by replacing leaf l with a black-box leaf are defined by Equations 5 and 6, respectively:

$$\delta_{l,a} = (N_{l,bb} - N_{l,t})/N \tag{5}$$

$$\delta_{l,c} = -N_l/N < 0 \tag{6}$$

The relative differences $\delta_{l,a}$ and $\delta_{l,c}$ depend only on the leaf l and not on the other leaves. The relative difference in the comprehensibility $\delta_{l,c}$ is always negative because replacing a regular leaf with a black-box leaf decreases the comprehensibility. On the other hand, the relative difference in the accuracy $\delta_{l,a}$ can be positive, zero or negative. However, only the leaves l with $\delta_{l,a} > 0$ are included in the further analysis since replacing a regular leaf with a black-box leaf that does not increase the accuracy would only decrease the comprehensibility.

Definition (Dominance Relation on Leaves): Equation 7 defines the following relation: leaf l dominates leaf k according to two objectives, that the relative difference in the comprehensibility $\delta_{l,c}$ and accuracy $\delta_{l,a}$ are both objectives that must be maximized:

$$l \preccurlyeq k \Leftrightarrow (\delta_{l,a} > \delta_{k,a} \wedge \delta_{l,c} \geq \delta_{k,c}) \vee (\delta_{l,a} \geq \delta_{k,a} \wedge \delta_{l,c} > \delta_{k,c}) \tag{7}$$

Equation 7 states that the better the leaves are, the more they increase the accuracy and, at the same time, the worse the leaves are, the more they decrease the comprehensibility. The quality of the leaves therefore depends on the relative difference in the accuracy and comprehensibility and is compared using the Pareto dominance relation.

In the second step of the MALACHITE algorithm, nondominated sorting of leaves is performed, i.e., the solutions are ranked according to the Pareto dominance (line 16, Figure 4). This step partitions the set of leaves that are represented in the objective space in such a way that the leaves with rank i are nondominated by the leaves with ranks $j \leq i$, and thus, they dominate all of the leaves with ranks $k > i$. The leaves with the same Pareto dominance rank are sorted according to increasing the relative difference in the accuracy $\delta_{l,a}$ (the same as sorting according to the decreasing relative difference in the comprehensibility $\delta_{l,c}$) and assigned ascending integer indices according to the described sorting (line

16, Figure 4). In addition, the set of leaves D_l that are directly dominated by a leaf l (Equation 8) is calculated for each leaf l (lines 17–19). This step concludes the initialization phase of the algorithm.

Definition (DIRECTLY DOMINATED LEAVES): The set of leaves D_l that are directly dominated by the leaf l is the set of nondominated leaves in the set of leaves that are dominated by the leaf l as defined by Equation 8 and shown by an example in Figure 3.

$$D_l = nondom\{k: l \preccurlyeq k\} \tag{8}$$

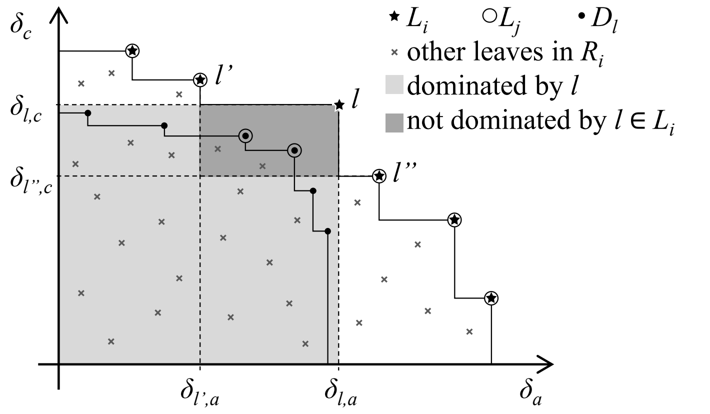

Figure 3: The set of nondominated regular leaves L_j (encircled) contains all of the leaves from L_i (stars), except for the leaf l and the leaves that are directly dominated by l (black circles) but are not dominated by leaves from $L_i \setminus \{l\}$; those are leaves $k \in D_l$, and thus, $\delta_{k,a} > \delta_{l',a}$ and $\delta_{k,c} > \delta_{l'',c}$ (dark background).

The third step of the MALACHITE algorithm uses an iterative search for the hybrid trees (lines 27–41) that avoids calculating the comprehensibility and accuracy of the hybrid tree as the sum over all of the leaves (Equations 1 and 2), which would require $2 \cdot 2^n(n-1)$ additions. The accuracy a_i and the comprehensibility c_i of a hybrid tree $T_i = (t_{i,1}, t_{i,2}, \ldots, t_{i,n})$ are used instead to compute the accuracy a_j and the comprehensibility c_j of a hybrid tree $T_j = (t_{j,1}, t_{j,2}, \ldots, t_{j,n})$, where $t_{j,l} = 1$ and $t_{j,k} = t_{i,k}$ for each leaf $k \neq l$ (the trees differ only in the fact that leaf l is a regular leaf in T_i while it is a black-box leaf in T_j). This approach requires a single addition operation for each objective according to the Equation 9 (lines 33–34). Only the accuracy a_1 of the initial hybrid tree T_1 must be computed, as the sum over all the leaves using Equation 2 (line 23), whereas its comprehensibility c_1 is 1 by Equation 1 (line 24). Therefore, the number of required additions is reduced from $2 \cdot 2^n \cdot (n-1)$ to $2 \cdot ((2^n - 1) + (n-1))$.

$$(a_j, c_j) = (a_i + \delta_{l,a}, c_i + \delta_{l,c}) \tag{9}$$

The iterative search must therefore proceed in the order of the increasing number of replaced (i.e., black-box) leaves. The algorithm starts with the initial classification tree $T_1 = (0, 0, \ldots, 0)$, i.e., the only hybrid tree with 0 black-box leaves (lines 20–26). In each iteration, the algorithm uses the set of hybrid trees *tmp* generated in the previous iteration to generate the set of hybrid trees *tmp'* with one more black-box leaf: from a hybrid tree $T_i \in tmp$, it generates a set of hybrid trees by replacing, one at a time, the regular leaves in T_i with a black-box leaf (lines 30–37). The algorithm does not replace each regular leaf in T_i; it replaces only the nondominated regular leaves (named L_i)—the first optimization. This approach avoids generating most of the dominated hybrid trees (i.e., it narrows the search space), and this avoidance is the most important improvement over the naïve algorithm. To prevent generating a hybrid tree more than once only, leaves with indices greater than the highest index of a black-box leaf in T_i are replaced (line 30)—the second optimization. To prevent conflicts between the two optimizations, the leaf indices must be assigned according to the Pareto sorting (as shown in Figure 4). Despite the optimizations, the algorithm is still guaranteed to find the complete Pareto set of hybrids. The iteration loop stops when all of the leaves are replaced with black-box leaves.

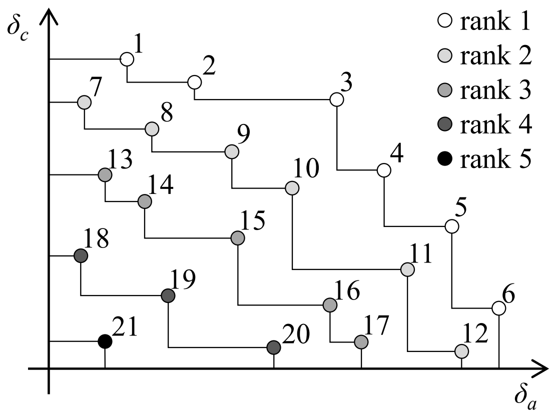

Figure 4: An example of Pareto sorting of leaves (rank 1–5) according to the maximal relative difference in accuracy δ_a and comprehensibility δ_c. The numbers represent indices that are assigned to the leaves.

In addition to the accuracy and comprehensibility of a hybrid tree T_j, the algorithm must also maintain the set of nondominated leaves L_j for each generated hybrid tree. The set of nondominated leaves for the initial classification tree T_1 is the set of nondominated leaves among all of the leaves, i.e., the set of leaves with Pareto dominance rank 1 (line 25). The set of nondominated regular leaves L_j in the tree T_j generated from the hybrid tree T_i is calculated using Equation 10. L_j contains all of the nondominated leaves in the tree T_i (denoted as L_i) except for leaf l (replaced when generating T_j from T_i) and a subset of leaves from D_l (the leaves that are directly dominated by the leaf l) that are not dominated by any leaf from $L_i \setminus \{l\}$. In other words, the leaves that should be replaced in the newly generated hybrid tree T_j are the leaves that were considered for replacement in the original tree T_i (i.e., the leaves in the set L_i) except for the replaced leaf l plus leaves that are dominated by leaf l but are not dominated by the other leaves in L_i. Equation 10 enables efficient computation of the nondominated set of regular leaves L_j by comparing only the qualities of some leaves and not of all of the remaining regular leaves, which would be required by a naïve approach.

$$L_j = nondom(L_i \setminus l \cup D_l) = L_i \setminus \{l\} \cup \{k\colon k \in D_l \wedge \nexists m \in L_i \setminus \{l\}\colon m \preccurlyeq k\} \qquad (10)$$

In the final step (line 42), the algorithm selects the nondominated hybrid trees from the set of all generated hybrid trees. The output of the MALACHITE algorithm is the Pareto set of hybrid trees and their corresponding objective values (the accuracy and comprehensibility of the hybrid trees), which are used to plot the Pareto front. The user analyzes the Pareto front as discussed in Piltaver, Luštrek and Gams, (2014) and finally selects the most appropriate hybrid tree.

For the efficient implementation of the MALACHITE algorithm, the following subproblems must be solved using appropriate algorithms. First, the Pareto dominance between a set of leaves or a set of hybrid trees is computed using the Two-dimensional Skyline Operator algorithm (Borzsonyi, Kossmann and Stocker, 2001), which decreases the time complexity $O(n^2)$ of a naïve implementation (Deb, 2008) to $O(n(\log n))$—the lowest possible time complexity (Kung, Luccio and Preparata, 1975). Second, the Pareto sorting of leaves is performed with the nondominated sorting algorithm proposed by Deb (2002), and this proposed algorithm has a time complexity $O(n^2)$. Using more elaborate algorithms such as the Better Nondominated Sorting Algorithm (Shi, Chen, and Shi, 2005) theoretically provides time complexity improvement for a constant factor but is not important for the MALACHITE algorithm, which addresses only two objectives and requires sorting a relatively small set only once. Third, the algorithm that calculates the set of directly dominated leaves for each leaf in the initial tree should take advantage of the Pareto sorting of the leaves. Only the leaves with a Pareto dominance rank higher than the rank of a considered leaf l can be dominated by it. Furthermore, the leaves with the same Pareto dominance rank are sorted according to the relative difference in accuracy δ_a, and therefore, only one objective must be compared to determine that a leaf with a higher Pareto dominance rank is dominated by leaf l. Finally, the sorting also enables calculating the set of nondominated leaves in each generated hybrid tree using a dynamic programming algorithm described in Piltaver's thesis (Piltaver, 2016) with corresponding theoretical proofs.

Algorithm 1: MALACHITE (find nondominated hybrid trees)

```
Input: learning_set, initial_tree, black_box
1:  for each (instance $i \in$ learning_set)
2:      $N$++;
3:      $l$ = find_corresponding_leaf($i$, initial_tree);
4:      $N_l$++;
5:      if (initial_tree.classify($i$) == $i$.class) $N_{l,t}$++;
6:      if (black_box.classify($i$) == $i$.class) $N_{l,bb}$++;
7:  end for;
8:  leaves = {};
9:  for each (leaf $l$ in initial_tree)
10:     if ($N_{l,bb} > N_{l,t}$)
11:         leaves = leaves ∪ {$l$};
12:         $\delta_{l,c} = -N_l/N$;
13:         $\delta_{l,a} = (N_{l,bb} - N_{l,t})/N$;
14:     end if;
15: end for;
16: leaves = sort_by_accuracy(pareto_sort(leaves));
17: for each ($l \in$ leaves)
18:     $D_l$ = directly_dominated($l$, leaves)
19: end for;
20: h_trees = {};               // set of processed hybrid trees
21: $T_1 = (0, 0, \ldots, 0)$;          // the initial_tree is a hybrid tree with no black-box leaves
22: $m_1 = -1$;                 // max index of black-box leaf in hybrid tree
23: $a_1 = \Sigma_l(N_{l,t})$;          // accuracy of the initial_tree
24: $c_1 = 1$;                  // comprehensibility of the initial_tree
25: $L_1$ = non_dom(leaves);  // non_dominated leaves in the initial_tree
26: tmp = {$T_1$};              // set of unprocessed hybrid trees
27: while (tmp ≠ {});
28:     tmp' = {};
29:     for (each $T_i$ in tmp)
30:         for each ($l \in L_i$) if ($l > m_i$)
31:             $T_j = T_i$; $t_{j,l} = 1$; // replace leaf $l$ with black-box to obtain $T_j$
32:             $m_j = l$;
33:             $a_j = a_i + \delta_{l,a}$;
34:             $c_j = c_i + \delta_{l,c}$;
35:             $L_j$ = non_dom($L_i \setminus \{1\} \cup D_l$); // non-dom. leaves in $L_j$
36:             tmp' = tmp' ∪ {$T_j$};
37:         end if; end for;
38:         h_trees = h_trees ∪ {$T_i$};
39:     end for;
40:     tmp = tmp';
41: end while;
42: h_trees = non-dom(h_trees);
43: return h_trees;
Output: h_trees—set of hybrid trees
```

4.3. The MALACHITE data-mining process

The MALACHITE algorithm is used within the data-mining process described in this subsection and is presented in Figure 1. The data-mining process is iterative and includes a domain expert in the loop to combine the expert's knowledge and the power of machine learning. Each iteration starts by choosing an initial classification tree and a black-box classifier as the inputs to the MALACHITE algorithm; iteratively improving the inputs to the MALACHITE algorithm improves the quality of the produced hybrid trees. The input classification tree can be built according to the expert's knowledge or trained using a machine-learning algorithm; in the latter case, an additional training set is used, i.e., not the training set used as an input to the MALACHITE algorithm. The classification tree must be comprehensible to the expert and as accurate as possible—at least in some of its leaves. An existing accurate classifier can be used as the black-box classifier or it can be trained on the same dataset as the classification tree.

Second, the MALACHITE algorithm is executed, and the resulting Pareto set of hybrid trees is analyzed. If the expert is satisfied with the accuracy and comprehensibility of some of the hybrid trees, the data mining process is finished. Otherwise, the next iteration of the process begins by selecting new inputs to the MALACHITE algorithm. If the difference in accuracy between the most accurate hybrid tree and the initial classification tree is too small, a more accurate black box classifier needs to be learned. The insights obtained by analyzing the hybrid trees often enable training a more accurate black-box classifier. To do so, an alternative learning algorithm, new attributes, or additional learning instances with less noise are added. In addition, an alternative learning algorithm can be used or the parameters of the same learning algorithm can be tuned. If a more accurate black-box classifier cannot be learned, the process is stopped. In this case, the initial classification tree (or a very comprehensible hybrid tree) is chosen to be the final classifier.

On the other hand, the expert might not be satisfied with the hybrid trees because of their low comprehensibility. This problem is solved by training a more accurate classification tree as an input for the next execution of the MALACHITE algorithm. The analysis of the Pareto set of hybrid trees determines which subtrees should be improved. To do so, the relative quality of each leaf in the initial tree is first computed based on the number of hybrid trees from the Pareto set in which the leaf is replaced with the black-box classifier. The leaves that are never replaced with the black-box classifier and leaves with high relative quality, i.e., the leaves that are infrequently replaced with black-box leaves, do not need improvement. The leaves with low relative quality, on the other hand, are the leaves that should be improved first. Failing to improve the accuracy by replacing such a leaf with a shallow subtree shows that the corresponding part of the domain cannot be classified accurately with a classification tree, and therefore, a black-box classifier should be used instead, or additional attributes must be added to the training dataset. Note that the relative quality of the leaves is complementary information to the classification accuracy of the leaves (Piltaver, 2014). If hybrid trees are not comprehensible enough due to poorly chosen attributes, another approach to solving the problem of low comprehensibility is to replace some of the attributes with more comprehensible attributes in the hope that the accuracy in the next iteration of the data-mining process will not suffer much as a result of this step.

The MALACHITE algorithm for learning Hybrid Trees was implemented as an add-on for the Orange data-mining suite (Demšar, Zupan, Leban and Curk 2004) and is publicly available. The technical details of the implementation are described by Novak, Piltaver and Gams (2017).

5. MALACHITE evaluation

To evaluate the MALACHITE algorithm, three sets of experiments described in the following subsections were performed. First, the run times of the naïve and the MALACHITE algorithm are compared to evaluate the benefit of the MALACHITE algorithm. Second, the Pareto-front of the hybrid trees is compared with the baseline classifiers on 23 datasets to verify that the hybrid trees provide trade-offs between the accuracy and comprehensibility that are not achievable using traditional machine-learning algorithms. Third, the run time of the efficient algorithm is compared with the run time of a well-known evolutionary multi-objective optimization algorithm and evaluated as to how close to the Pareto front the multi-objective optimization algorithm converges to in a reasonable amount of time.

5.1. Run time of the suggested algorithms

The naïve algorithm (i.e., exhaustive search) and MALACHITE algorithm both have exponential time complexities $O(2^n)$ in the worst case, where n is the number of leaves in the initial classification tree that is considered for replacement with black-box leaves—each leaf can be replaced with the black-box or not. Since the problem is obviously at least as difficult as the NP-hard 0/1 knapsack problem, no algorithm that solves the problem in polynomial time is expected to exist.

On the other hand, the MALACHITE algorithm avoids generating some of the hybrid trees and is therefore faster: the run times can be approximated with 2^{2n} for an exhaustive search algorithm and $2^{0.4n}$ for the MALACHITE algorithm. The average run-time of the exhaustive search algorithm implemented in Java™ for a tree with 18 leaves is approximately 18 minutes (3.2 min for 17 leaves), while it is only 0.8 ms for the MALACHITE algorithm (< 0.5 s for 40 leaves) on a 3 GHz Intel® Core™ 2 Duo computer. This finding shows that exhaustive search is not usable for initial trees with 20 or more leaves.

Since the goal is to find comprehensible classifiers, the initial tree must have a reasonable size. Most people are not capable of maintaining more than 7 ± 2 conditions in their short-term memory (Miller, 1956); therefore, only (binary) trees with approximately 2^5 (or at most 2^7) leaves are considered to be comprehensible. Experiments show that in this setting, the MALACHITE algorithm finds the complete Pareto set of hybrid trees within seconds.

5.2. Testing datasets

The evaluation of the MALACHITE algorithm is based on 23 datasets from the UCI repository (Bache and Lichman, 2015). A subset of 49 datasets available in ARFF format and with more than 300 instances was considered because it had to be split into three parts with at least 100 instances each: the first was used to train the initial tree and the black-box classifier, the second was used to find the Pareto set of hybrid trees and predict their accuracy and comprehensibility, and the third was used to evaluate the set of hybrid trees, the initial classification tree and the black-box classifier. The set of datasets was finally reduced to the 23 datasets for which the difference in the classification accuracy between the black-box classifier and a tree with approximately 20 leaves was at least 10 %. There is no point in learning hybrid trees if the difference in the classification accuracy between the initial tree and the most accurate (black-box) classifier is small. The following algorithms were used to build the black-box classifiers: SVM Platt, 1999), kNN (Aha and Kibler, 1991), aNN (Rumelhart, Hinton, and Williams, 1986), logistic regression (Sumner, Frank and Hall, 2005) and Naïve Bayes (John, and Langley, 1995), which were all implemented in Weka (Witten, Frank and Hall, 2011) and used with the default algorithm parameters. Among them, the classifier with the highest classification accuracy computed using 10-fold cross-validation was chosen as the black-box classifier for each dataset. Finally, 40 experiments with the suggested algorithm were conducted: one using a small initial tree (~20 leaves) for each of the 23 datasets and another using a larger initial tree (~40 leaves) for the 17 datasets that allowed building a larger tree. The initial classification trees were learned using the J48 algorithm, which is the Weka implementation of the C4.5 algorithm (Quinlan, 1993).

5.3. Comparison with the baseline classifiers

MALACHITE is a multi-objective learning algorithm, and therefore, it must be evaluated by comparing the learned set of hybrid trees (not only a single hybrid tree) with the set of baseline classifiers according to multiple objectives. The baseline classifiers used in the evaluation are the following: the initial classification tree as a completely comprehensible classifier and the black-box classifier as an incomprehensible classifier with maximal classification accuracy.

The hypervolume measure (Zitzler, Thiele, Laumanns, Fonseca and da Fonseca, 2003) is used to compare the two sets of classifiers because it is a widely used measure for the comparison of multi-objective optimization algorithms. It measures the volume of the dominated objective space between a reference point and the attainment surface—the envelope that marks all of the classifiers, which is certain to be dominated by the set of classifiers returned by the algorithm (Deb, 2008). Since two objectives are considered by the MALACHITE algorithm, the hypervolume is in fact the area under the attainment surface represented by a line in Figure 5a and b. The reference point was set to (−0.2, −0.2) following the typically used rule-of-thumb (Beume, Naujoks and Emmerich, 2007; Knowles, 2005), advocating a space that is slightly larger than the actual objective space: in this way, the accuracy of a classifier with a comprehensibility of 0 contributes to the hypervolume, whereas it would not influence the hypervolume if the reference point was set to (0,0).

We observe the difference in the hypervolumes between the baseline and the MALACHITE algorithm. The difference is small if the hybrid trees produced by the MALACHITE algorithm lie close to the attainment surface of the two baseline classifiers. In other words, if the Pareto set returned by the MALACHITE algorithm is composed of hybrid trees with high comprehensibility and only slightly higher accuracy then the initial tree, and hybrid trees with low comprehensibility and approximately the same accuracy as the black-box classifier—this result is not a good result. On the other hand, a large difference in the hypervolume means that some hybrid trees produced by the MALACHITE algorithm lie farther away from the baseline attainment surface (i.e., the initial classification tree and the black-box classifier), and therefore, they provide trade-offs between the comprehensibility and the accuracy, which cannot be matched using traditional machine learning models.

Figure 5 shows the predicted (on the training set) and validated (on the separate test set) results of the MALACHITE algorithm and the results of the baseline in the objective space for two datasets. The predicted attainment surface (i.e., the Pareto front) obtained by the MALACHITE algorithm is shown with a red line, while the validated surface is shown in green. The circles represent the accuracy and comprehensibility of the hybrid trees from the Pareto set computed on the training (red) and test dataset (green). The triangles represent the quality of the two baseline classifiers: (1) the initial classification tree with a comprehensibility equal to 1 and an accuracy calculated on the test set; and (2) the black-box classifier with a comprehensibility equal to 0 and an accuracy calculated on the test set.

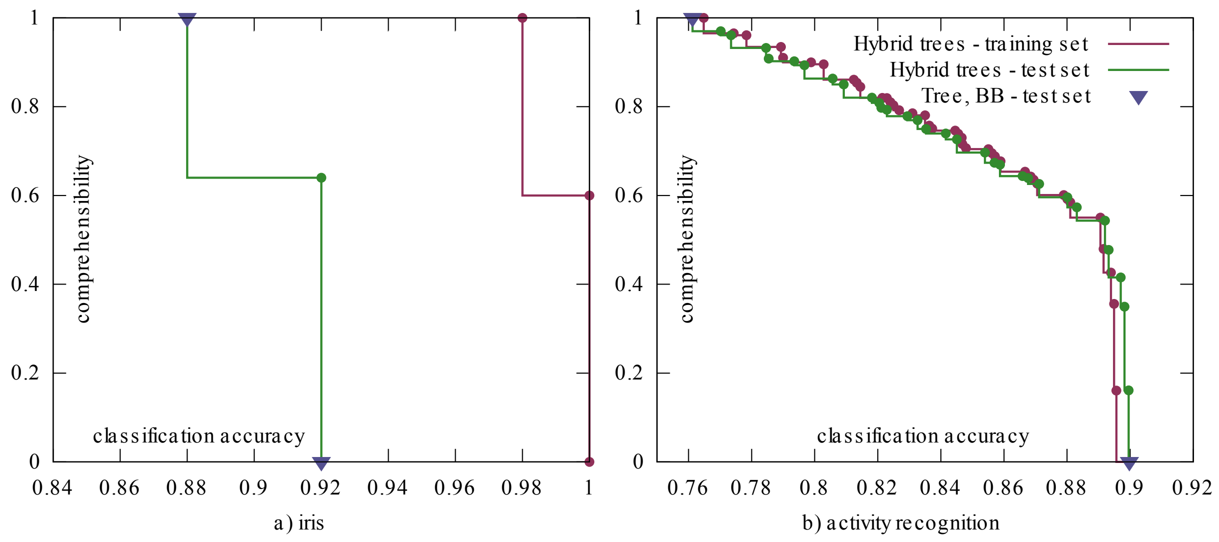

Figure 5: Examples of two Pareto fronts for the iris and activity-recognition datasets. The circles represent the accuracy and comprehensibility of the learned hybrid trees, the lines represent the Pareto-fronts (green refers to the data obtained on the test dataset, red on the training set), and the triangles refer to the two baseline classifiers.

Table 1 shows the experimental results from seven experiments: three with the highest difference in hypervolume, one with an average result, and three with the lowest difference out of the 40 experiments (i.e., 23 datasets: 17 with a small and large initial classification tree, and others with only a small initial tree). For example, for the first three datasets (left to right), the baseline black-box classifier achieved an accuracy of 100 % (BB accuracy). The baseline comprehensible classifier (with corresponding tree accuracy) achieved an accuracy below 50 %. The baseline hypervolume is close to 0.8, whereas the MALACHITE's hypervolume is above 1. The last two rows represent the absolute and the relative hypervolume difference between MALACHITE and the baseline. In general, it can be seen that the MALACHITE offers better improvements for the domains for which the BB accuracy is significantly higher than the tree accuracy.

Table 1. MALACHITE evaluation and comparison with baseline (from left to right, three best, one average, and worst domains).

	Dataset						
	mfeat-pixel	letter	vowel	mfeat-zernike	cylinder-bands	balance-scale	flags
# instances	2000	20000	990	2000	540	625	194
# classes	10	26	11	10	2	3	8
# attributes	241	17	14	48	38	5	30
BB accuracy	100	100	100	89,06	90,00	92,82	100
Tree accuracy	42,72	39,66	44,55	83,81	66,66	88,04	86,20
Baseline hypervolume	0.833	0.815	0.772	1.031	1.002	1.201	0.896
MALACHITE hypervolume	1.138	1.108	1.003	1.154	1.028	1.229	0.914
Absolute hypervolume difference	0.305	0.293	0.231	0.123	0.026	0.029	0.017
Relative hypervolume difference [%]	36.6	36	29.9	11.9	2.6	2.3	2

Figure 6 shows the relative difference in the hypervolume obtained with MALACHITE compared to baseline for all of the 40 experiments: the best improvement is 36.6 %, the average improvement is 11.17 %, and the worst is 1.9 %. This finding shows that the hybrid trees generated using the MALACHITE algorithm indeed provide a trade-off between the accuracy and comprehensibility that is not possible using traditional machine learning algorithms. Therefore, hybrid trees provide an important advantage in applications where the classifier comprehensibility is important, but completely comprehensible classifiers do not achieve the desired accuracy.

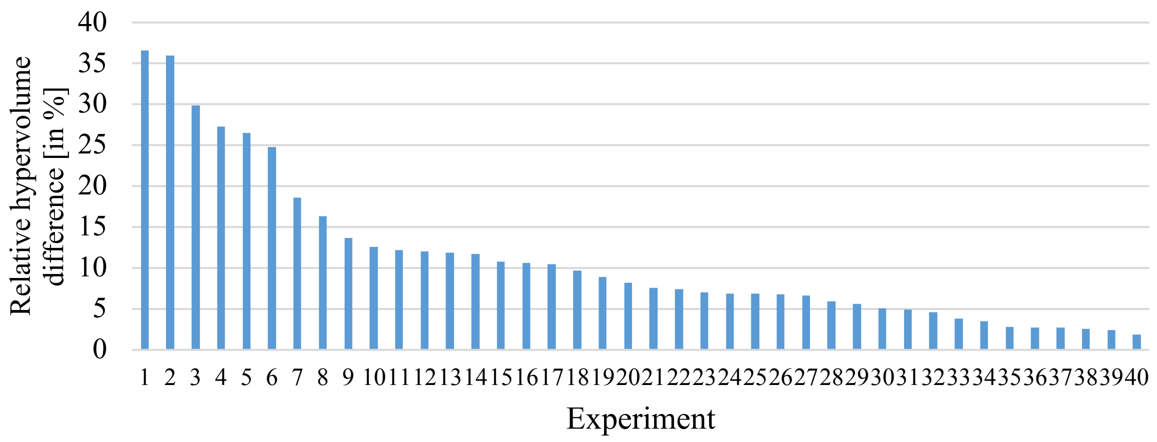

Figure 6. The improvement of MALACHITE vs. baseline measured as the relative difference in the hypervolume [in %] for the 40 experiments.

The difference in the accuracy between a comprehensible classification tree and the most accurate black-box classifier was less than 10 % in 26 out of 40 tested datasets. In such cases, there is not much room for a compromise between the accuracy and comprehensibility. Nevertheless, using the proposed data-mining process with the MALACHITE algorithm still provides valuable insights. Even if the user decides to use the initial classification tree or the black-box classifier instead of the generated hybrid trees, the user is informed about the difference in accuracy between the two classifiers, the available alternatives (i.e., the hybrid trees), and the relative qualities of the leaves in the initial classification tree, which are important for the validation of the tree.

5.4. Comparison with the NSGA-II multi-objective optimization algorithm

The MALACHITE algorithm was compared with the well-known multi-objective optimization algorithm NSGA-II (Deb, 2002). NSGA-II is a stochastic evolutionary algorithm that converges toward the Pareto set of hybrid trees. The parameters of the NSGA-II algorithm were set as follows: a two-point crossover was used as the crossover technique, and a bit flip was the mutation operator, with the probability of a mutation equal to 1/*number of leaves* for each bit. Three sizes of populations were set: 6.3 × *number of leaves* − 15 (medium population), half that number (small population) and twice that number (large population). The linear formula approximates the number of hybrid trees in the Pareto set in relation to the number of leaves considered for replacement with black-box leaves. In a general setting, setting an appropriate size of the population is not trivial because the number of hybrid trees in the Pareto front is not known in advance. The stopping criterion was defined by setting the NSGA-II execution time limit to 10, 50 and 100 times longer than the average run-time (over 10 runs) of the MALACHITE algorithm. The number of generations was not limited explicitly; however, it depended on the run-time limit and the size of the population: a longer run-time corresponds to a higher number of generations, while a larger population size causes the number of generations to decrease for a constant run-time. The DEAP framework (Distributed Evolutionary Algorithms in Python)[1] implementation of NSGA-II was used.

Table 2 shows the average relative difference in the hypervolume expressed as a % between the MALACHITE algorithm and the average over five runs of the NSGA-II algorithm. The table shows the average over six domains. Table 2 includes the results for two sizes of the initial classification trees, the three run-time multipliers, and the three sizes of the population used by NSGA-II. The results show that the MALACHITE algorithm outperforms NSGA-II when searching for trees of comprehensible size. Table 2 shows that the difference in the hypervolume decreases with decreasing population size (i.e., increasing the number of generations) for a short run-time limit or for a large search space (i.e., an initial tree with many leaves). In this case, NSGA-II is too slow to converge to the Pareto set. Furthermore,

[1] https://code.google.com/p/deap/].

Table 2 shows that the difference in the hypervolume increases with decreasing size of the population for a small search space (i.e., few leaves in the initial classification tree) and a sufficiently long run-time. In the latter case, NSGA-II has enough time to converge to the Pareto set, but the size of the population is too small to include all of the hybrid trees from the Pareto set. The low difference in the hypervolume (0.02 and 0.03) for a large population is caused by the stochastic nature of the NSGA-II: even if NSGA-II has enough time to converge to the Pareto set and a large enough population, its stochastic nature prevents it from finding the complete Pareto set of hybrid trees in each run. The results confirm that the MALACHITE algorithm is computationally more efficient then NSGA-II (for reasonably sized initial trees). In addition, it is preferred because it does not require any parameter setting and is guaranteed to find the complete Pareto set.

Table 2: Relative difference in the % of hypervolume (MALACHITE vs. NSGA-II) for three run-times and population sizes of NSGA-II, and two sizes of initial tree

Population / Run-time	**Small tree** (avg. 12.8 leaves)			**Large tree** (avg. 22 leaves)		
	Large	Medium	Small	Large	Medium	Small
t(MALACHITE) × 10	3.8	1.8	1.4	5.5	4.9	2.7
t(MALACHITE) × 50	0.2	0.1	0.2	2.8	0.6	0.4
t(MALACHITE) × 100	0.02	0.03	0.2	1.5	0.3	0.1

6. Discussion

One issue in the application of the MALACHITE algorithm could be a possible high number of hybrid trees, which does not allow the user to compare all of them when choosing the best one. Figure 7 shows that the number of hybrid trees and hence the density of the Pareto front depends on the number of leaves in the initial tree that are considered for replacement with the black-box classifier. The number of hybrid trees in the Pareto set also depends on the number of leaves with the same Pareto dominance rank, which is not known in advance—this aspect explains the outliers in Figure 7. In general, the number of Pareto-optimal hybrid trees is manageable, especially considering that not all of them are necessary to analyze (Piltaver, Luštrek and Gams, 2014).

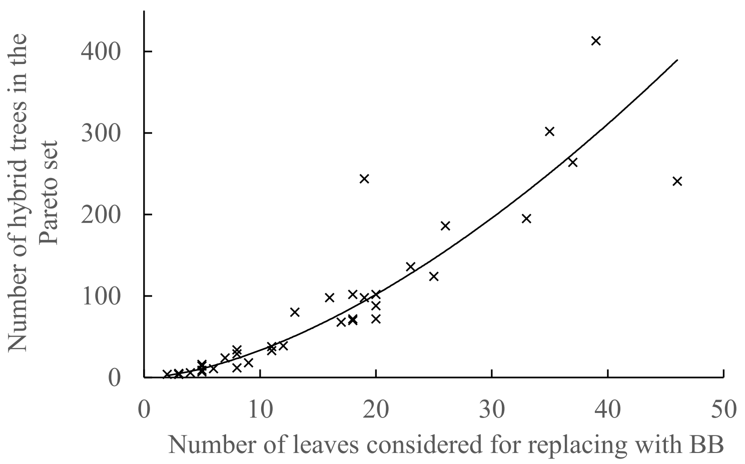

Figure 7: Number of hybrid trees in a Pareto set depends on the number of leaves in the initial tree that are considered for replacement with black-box leaves (results obtained from 40 initial trees built on 23 UCI datasets).

Another issue could be caused by errors in the estimated accuracy and comprehensibility of the learned hybrid trees, which could mislead the user when choosing a hybrid tree from the Pareto front (Figure 5), where the hybrid trees' performance on the training dataset is plotted. This problem is a typical problem for any machine-learning algorithm that requires hyperparameter tuning. For a reliable estimation of the comprehensibility and accuracy, overfitting on the training data must be avoided. In our future work, we plan to improve the MALACHITE algorithm using internal *n*-fold cross-validation. Preliminary results show that this approach could enable a more reliable performance on small datasets by limiting the errors of the predicted comprehensibility and accuracy of the hybrid trees.

Another potential issue is using the algorithm with an insufficient amount of training data, which does not enable accurate estimation of the classification accuracy of the initial classification tree and the black-box classifier on each subset of the training data that corresponds to instances that belong to a leaf in the original classification tree.

In a typical case, there were no problems with replacing reliable transparent nodes with less reliable black-box classifiers, since it is an essential part of the system when the user observes various versions of the trees and chooses the trees that do not have anomalies. Furthermore, in practical use, anomalies were quite rare and were easily detectable by the users.

The MALACHITE algorithm is specialized for classification tasks. One direction for future work could be to develop a similar approach for regression tasks. For example, regression trees can provide understandable regression models (Cappelli, D'Urso, and Di Iorio, 2013; Cappelli, D'Urso, and Di Iorio, 2015; Cappelli, Di Iorio, Maddaloni, and D'Urso, 2019). Those models can be combined with more powerful black-box regression models based on deep learning (e.g., Long short-term memory – LSTMs) to maximize the performance and comprehensibility.

7. Conclusions

This paper bridges the gap between comprehensible classifiers that often achieve lower accuracy (e.g., classification trees and rules) and more accurate but incomprehensible black-box classifiers (e.g., ANN, SVM, ensembles). Most classification algorithms focus on the accuracy, or some other performance measure, and neglect comprehensibility or treat it as secondary. Our approach treats both as equally important, or rather—their importance is chosen by the user. Most classification algorithms return a single classifier in one run and force the user to (blindly) choose algorithm parameters in order to tune the trade-off between the comprehensibility and accuracy in the next runs. Our approach, on the other hand, in one run returns a Pareto set of classifiers from the most comprehensible to the most accurate ones, supporting the user when deciding how much accuracy to sacrifice for comprehensibility and vice versa. At the same time, the user can compare several versions of trees at once to detect the differences and infer domain relations.

This paper provides a framework that includes the hybrid tree as a classifier model, the corresponding comprehensibility measure, the MALACHITE learning algorithm and the corresponding data-mining process. The hybrid trees provide trade-offs between the accuracy and comprehensibility that cannot be achieved using a comprehensible classifier or a black-box classifier alone. The proposed MALACHITE algorithm returns the set of nondominated hybrid trees with some regular comprehensible leaves and some more accurate black-box leaves. This set enables visual inspection and enables comparisons between optimal trees, inferring properties of the domain and finally choosing the most promising hybrid tree according to the user's preferences.

The dynamic programming approach and search space pruning are used to achieve low run-times for reasonably sized trees. The proposed algorithm is also faster than the genetic multi-objective optimization algorithm NSGA-II, guarantees finding the complete set of nondominated hybrid trees, and does not require setting the optimization parameters. This approach enables using the algorithm within the proposed multi-objective data-mining process, which helps the user to choose the appropriate algorithm inputs, obtains insights into the classification problem and finally selects a hybrid tree with a good accuracy/comprehensibility trade-off. Several properties of the algorithm and the constructed classifiers are presented in Piltaver's thesis (Piltaver, 2016) with corresponding theoretical proofs.

To improve the usability of the suggested multi-objective learning framework, several interesting topics for further research should be investigated. Examples are automatic selection of the initial classification tree (used as an algorithm input), graphical tools based on clustering methods to support the user while selecting a hybrid tree from the set of learned hybrid trees, and improving the accuracy of the estimated quality of the hybrid trees for small datasets.

Acknowledgements: Part of this work was funded by the Slovenian Research Agency (ARRS).

8. References

Aha, D., & Kibler D. (1991). Instance-based learning algorithms. *Machine Learning* 6:37–66.

Bache K., & Lichman M. (2015). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Beume, N., Naujoks, B., & Emmerich, M. (2007). SMS-EMOA: Multiobjective Selection Based on Dominated Hypervolume. *European Journal of Operational Research* 181 (3): 1653–1669.

Blanco-Justicia, A., & Domingo-Ferrer, J., Machine Learning Explainability Through Comprehensible Decision Trees. InInternational Cross-Domain Conference for Machine Learning and Knowledge Extraction 2019 Aug 26 (pp. 15-26). Springer, Cham.

Borzsonyi, S., Kossmann, D., & Stocker, K., (2001). The Skyline Operator, In: Proceedings of 17th International Conference on Data Engineering. IEEE Computer Society, Heidelberg, Germany, pp. 421–430.

Cappelli, C., D'Urso, P., & Di Iorio, F. (2013). Change point analysis of imprecise time series. Fuzzy Sets and Systems, 225, 23-38.

Cappelli, C., D'Urso, P., & Di Iorio, F. (2015). Regime change analysis of interval-valued time series with an application to PM10. Chemometrics and Intelligent Laboratory Systems, 146, 337-346.

Cappelli, C., Di Iorio, F., Maddaloni, A., & D'Urso, P. (2019). Atheoretical Regression Trees for classifying risky financial institutions. Annals of Operations Research, 1-21.

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer C., & Wirth, R. (2000). CRISPDM 1.0 step-by-step data mining guide. SPSS Inc.

Clark, A. R. J., & Everson, R.M. (2012). Multi-objective learning of Relevance Vector Machine classifiers with multi-resolution kernels. *Pattern Recognition*: 45(9): 3535–3543.

Cohen, W.W. (1995). Fast Effective Rule Induction, In: Swarm and Evolutionary Computation. Morgan Kaufmann, San Francisco, pp. 115–123.

Czajkowski, M., & Kretowski, M. (2019). Decision tree underfitting in mining of gene expression data. An evolutionary multi-test tree approach. Expert Systems with Applications. Dec; 15;137:392-404.

De Bock, K. W. (2017). The best of two worlds: Balancing model strength and comprehensibility in business failure prediction using spline-rule ensembles. Expert Systems with Applications, Jul; 90:23-39.

Deb, K. (2002). A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE transactions on evolutionary computation* 6(2): 182–197.

Deb. K, (2008). Multi-Objective Optimization Using Evolutionary Algorithms, John Wiley & Sons Inc., New York, NY, USA.

Demšar, J., Zupan, B., Leban, G., & Curk, T. (2004), Orange: From experimental machine learning to interactive data mining. Knowledge discovery in databases: PKDD 2004, pages 537-539, 2004.

Freitas, A. A. (2004). A Critical Review of Multi-objective Optimization in Data Mining: A Position Paper. *SIGKDD Explorations Newsletter* 6-2: 77–86.

Gama, J., Rocha, R., & Medas, P. (2003). Accurate decision trees for mining high-speed data streams. In: Getoor, L., Senator, T. E., Domingos, P. et al. (eds). Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, Washington, D.C., USA, pp. 523–528.

Girosi, F., Jones, M., & Poggio, T. (1995). Regularization Theory and Neural Networks Architectures. *Neural Computation* 7: 219–269.

Gorzałczany, M. B., & Rudziński, F. (2012). Accuracy vs. Interpretability of Fuzzy Rule-Based Classifiers — An Evolutionary Approach, In: Rutkowski, L., Korytkowski, M., Scherer, R. et al. (eds). Swarm and Evolutionary Computation. Springer, Berlin, pp. 222–230.

Ishibuchi, H., Nakashima, T., & Murata, T. (2001). Three-Objective Genetics-Based Machine Learning for Linguistic Rule Extraction. *Information Sciences*: 136(1–4): 109–133.

Jin, Y. (2000). Fuzzy Modelling of High-Dimensional Systems: Complexity Reduction and Interpretability Improvement. *IEEE Transactions on Fuzzy Systems* 8(2): 212–221.

Jin, Y., Sendhoff, B. & Körner, E. (2006). Simultaneous Generation of Accurate and Interpretable Neural Network Classifiers, In: Jin Y (ed) Multi-Objective Machine Learning. Springer, Berlin, pp. 291–312.

Jin, Y., & Sendhoff, B. (2008). Pareto-Based Multiobjective Machine Learning: An Overview and Case Studies. *IEEE transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 38(3): 397–415.

John, G. H., & Langley, P. (1995). Estimating Continuous Distributions in Bayesian Classifiers. In: Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers Inc., Montreal, Quebec, Canada, pp. 338–345.

Knowles, J. (2005). ParEGO: A Hybrid Algorithm with On-Line Landscape Approximation for Expensive Multiobjective Optimization Problems. *IEEE Transactions on Evolutionary Computation* 10 (1): 50–66.

Kohavi, R. (1996). Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid, In: Simoudis E, Han J and Fayyad U (eds) Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. AAAI Press, Menlo Park, pp. 202–207.

Kottathra, K., & Attikiouzel, Y. (1996). A Novel Multicriteria Optimization Algorithm for the Structure Determination of Multilayer Feedforward Neural Networks. *Network and Computer Applications* 19-2: 135–147.

Kumar, M. A., & Gopal, M. (2010). A Hybrid SVM Based Decision Tree. *Pattern Recognition* 43(12): 3977–3987.

Kung, H. T., Luccio, F., & Preparata, F. P. (1975). On finding the maxima of a set of vectors. *Journal of the ACM* 22(4): 469–476.

Kurgan, L. A., & Musilek P. (2006). A survey of Knowledge Discovery and Data Mining process models. *Knowledge Engineering Review* 21(1): 1-24.

Markowska-Kaczmar, U., & Mularczyk, K. (2006). GA-based Pareto Optimization for Rule Extraction from Neural Networks, In: Jin Y (ed) Multi-Objective Machine Learning. Springer, Berlin, 291–312.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review* 63(2): 81–97.

Novak, B., Piltaver, R., & Gams, M. User-friendly Multi-objective Learning of Accurate and Comprehensible Hybrid-trees. In: Slovenian Conference on Artificial Intelligence : proceedings of the 20th International Multiconference Information Society - IS 2017, 9th-13th October, 2017, Ljubljana, Slovenia : volume A. pp. 39-42.

Obregon, J., Kim, A., & Jung, J. Y. (2019) RuleCOSI: Combination and simplification of production rules from boosted decision trees for imbalanced classification. Expert Systems with Applications. Jul 15; 126:64-82.

Piltaver, R., Luštrek, M., & Gams, M. (2014). Multi-objective Learning of Accurate and Comprehensible Classifiers—a Case Study, In: Endriss U, Leite J (eds) Proceedings of the 7th European Starting AI Researcher Symposium. IOS Press, Prague, Czech Republic, pp. 220–229.

Piltaver, R., Luštrek, M., Zupančič, J., Džeroski, S., & Gams, M. (2014) Multi-objective Learning of Hybrid Classifiers, In: Schaub T, Friedrich G, O'Sullivan B (eds) ECAI 2014: 21st European Conference on Artificial Intelligence. IOS Press, Prague, Czech Republic, pp. 717–722.

Piltaver, R. (2016). Constructing Comprehensible and Accurate Classifiers Using Data Mining Algorithms: Doctoral Dissertation (Doctoral dissertation, R. Piltaver).

Platt, J. (1999). Fast Training of Support Vector Machines using Sequential Minimal Optimization, In: Schöelkopf B, Burges C, Smola AJ (eds) Advances in Kernel Methods—Support Vector Learning. MIT Press, Cambridge, MA, USA, pp. 185–208.

Pulkkinen, P. (2009). Multiobjective Genetic Fuzzy System for Obtaining Compact and Accurate Fuzzy Classifiers with Transparent Fuzzy Partitions, In: Proceedings of the 2009 International Conference on Machine Learning and Applications. IEEE Computer Society, Washington, DC, USA, pp. 89–94.

Quinlan. J. R. (1992). Learning with Continuous Classes, In: Proceedings of the 5th Australian Joint Conference on Artificial Intelligence. World Scientific, pp. 343–348.

Quinlan. R. (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco, CA, USA.

Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence 1, no. 5:206-215, 2019.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica* 14(5): 465–471.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning Internal Representations by Error Propagation, In: Rumelhart DE, McClelland JL, PDP Research Group Co. (eds). Parallel distributed processing: Explorations in the microstructure of cognition, Vol. 1. MIT Press, Cambridge, MA, USA, pp. 318–362.

Shen, S., Han, S. X., Aberle, D. R., Bui, A. A., & Hsu, W. (2019). An interpretable deep hierarchical semantic convolutional neural network for lung nodule malignancy classification. Expert Systems with Applications, 128:84-95.

Shi, C., Chen, M., & Shi, Z. (2005). A Fast Nondominated Sorting Algorithm, In: Proceedings of the International Conference on Neural Networks and Brain 2005, vol. 3. IEEE, pp. 1605–1610.

Soui, M., Gasmi, I., Smiti, S., & Ghédira, K. (2019). Rule-based credit risk assessment model using multi-objective evolutionary algorithms. Expert Systems with Applications. 126:144-57.

Sumner, M., Frank, E., & Hall, M. (2005). Speeding up Logistic Model Tree Induction, In: Jorge A, Torgo L, Brazdil P et al. (eds). Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases. Springer-Verlag, Porto, Portugal, pp. 675–683

Tušar, T. (2007). Optimizing accuracy and size of decision trees, In: Zajc B and Trost A (eds). Proceedings of the 16th International Electrotechnical and Computer Science Conference, vol. B. IEEE Region 8, Portorož, Slovenia, pp. 81–84

Witten, I. H., Frank, E., & Hall, M. A. (2011). Data Mining: Practical Machine Learning Tools and Techniques, Third Edition. Morgan Kaufmann, Burlington, MA, USA

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & da Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation* 7(2): 117–132