





Research Article

Semi-Supervised Predictive Clustering Trees for (Hierarchical) Multi-Label Classification

Jurica Levatić ^{1,2}, Michelangelo Ceci ^{1,3}, Dragi Kocev ^{1,2} and Sašo Džeroski ^{1,2}

¹Jožef Stefan Institute, Ljubljana, Slovenia

²Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

³Department of Computer Science, University of Bari, Bari, Italy

Correspondence should be addressed to Jurica Levatić; jurica.levatic@ijs.si

Received 6 October 2022; Revised 19 March 2024; Accepted 30 March 2024; Published 13 April 2024

Academic Editor: Vasudevan Rajamohan

Copyright © 2024 Jurica Levatić et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Semi-supervised learning (SSL) is a common approach to learning predictive models using not only labeled, but also unlabeled examples. While SSL for the simple tasks of classification and regression has received much attention from the research community, this is not the case for complex prediction tasks with structurally dependent variables, such as multi-label classification and hierarchical multi-label classification. These tasks may require additional information, possibly coming from the underlying distribution in the descriptive space provided by unlabeled examples, to better face the challenging task of simultaneously predicting multiple class labels. In this paper, we investigate this aspect and propose a (hierarchical) multi-label classification method based on semi-supervised learning of predictive clustering trees, which we also extend towards ensemble learning. Extensive experimental evaluation conducted on 24 datasets shows significant advantages of the proposed method and its extension with respect to their supervised counterparts. Moreover, the method preserves interpretability of classical tree-based models.

1. Introduction

Over the past decade, there has been growing interest for machine learning methods that can use both labeled and unlabeled data for learning a classification model. This interest is motivated by two important factors: (i) the high cost of assigning labels for large datasets and domains where labeling requires complex procedures and/or tedious manual effort and (ii) the opportunity to achieve greater predictive performance by better estimation of the distribution of data in the descriptive space, given the large amount of freely available unlabeled data. While the former factor is only of practical relevance, the latter stems from the theoretical observation that the underlying marginal data distribution $p(X)$ over the descriptive space X might contain information about the posterior distribution $p(Y | X)$ for the prediction of the Y values in the target space. The machine learning setting that takes into account both motivating factors is semi-supervised learning (SSL) [1]. It

accommodates the second factor by leveraging three (not independent) theoretical assumptions [2]: the *smoothness* assumption (if two samples x and x' are close in the input space, their labels y and y' should be the same); the *low-density* separation assumption (the decision boundary should not cut through high-density areas of the input space); and the *manifold* assumption (data points on the same low-dimensional manifold should have the same label).

Nowadays, many semi-supervised learning approaches are available that tackle the classification in multiple domains, including object recognition in images [3], human speech recognition [4], protein 3D structure prediction [5], IoT data analysis [6], and spam filtering [7]. However, only a few approaches are suited for the more complex tasks of multi-label classification (MLC) or hierarchical multi-label classification (HMLC), even though many applications (including the ones listed above) have an inherent complexity suitable for MLC and HMLC. Multi-label

classification is a predictive machine learning task where the examples can be labeled with more than one of the labels from a predefined set of labels C . In this case, the output variable y takes values in a subset of the label set \mathcal{Y} (i.e., $y \in 2^C$). Hierarchical multi-label classification is a particular case of MLC, where the output space is structured so that it accommodates dependencies between labels. In particular, labels are organized in a hierarchy: an example labeled with label c is also labeled with all parent/superlabels of c . MLC and HMLC problems are encountered in various domains, such as text categorization, image classification, object/scene classification, gene function prediction, and prediction of compound toxicity [8]. A common property for MLC and HMLC domains is that obtaining labeled examples is harder and more expensive compared to the classical (i.e., single-label) classification context. This contributes greatly to the need for developing SSL methods tailored for the MLC and HMLC tasks.

In the literature, only a few existing approaches tackle the problem of semi-supervised multi-label classification and hierarchical multi-label classification. Some examples include the work presented in [9–11] for the task of SSL MLC and that presented in [12] for SSL HMLC. However, all these methods adopt generative or optimization-based approaches, yielding complex and time-demanding learning processes, which produce noninterpretable models. On the contrary, in this paper, we propose an approach to SSL MLC/HMLC based on predictive clustering trees (PCTs) [13]. The advantage of predictive clustering trees is manifold: (1) the learning phase is time efficient; (2) the SSL models are interpretable for both MLC and HMLC tasks; (3) the SSL models can take both quantitative and categorical variables into account; (4) PCTs can be combined into ensembles, such as random forests, to further improve their predictive performance; and (5) the hierarchical structure of tree-based models can naturally model the hierarchical structure of the output space in the HMLC task.

In this paper, we propose a method for MLC and HMLC that works in the SSL setting. It defines a novel algorithm for learning predictive clustering trees by exploiting both the labeled and unlabeled data for MLC and HMLC tasks. In a nutshell, this is achieved by defining a new heuristic and prototype functions that take these specifics into account. Moreover, the proposed method has a parameter that balances the contribution of the descriptive and the target/label part of the data (i.e., controlling the degree of supervision in the model learning process). This mechanism safeguards against performance degradation, compared to learning only from the labeled data. Furthermore, we propose learning ensembles of the semi-supervised predictive clustering trees to further boost their predictive performance. The extensive experiments across 24 datasets from a variety of domains reveal that the proposed methods have better predictive performance compared to their supervised counterparts.

One of the explanations of the inner workings of the proposed methods is related to the interaction of semi-supervised learning with the label dependency of MLC and HMLC tasks. More specifically, we investigate whether the *smoothness* assumption (and, indirectly, since they are not

independent, the *low-density* and the *manifold* assumptions) holds in the MLC and HMLC contexts. Intuitively, better identification of the distribution of examples in the descriptive space (as performed in the semi-supervised learning setting) can lead to better exploitation of label dependency and/or label correlation in the output space, leading to improved predictive accuracy. To better explain this concept, let us consider the example reported in Figure 1. We can see that unlabeled examples provide useful information to better classify the examples in the classes “ a ” (“ a ” vs. not “ a ,” see the vertical dashed lines) and “ d ” (“ d ” vs. not “ d ,” see the horizontal dashed lines), especially in low-density regions. From Figure 1(b), we can also see that unlabeled examples reveal a higher correlation between the class labels “ a ” and “ e ” than between “ a ” and other class labels, such as “ d .” This is because “ a ” and “ e ” appear together in a region that is much denser than the region where “ a ” and “ d ” appear together. Such information, if exploited by the predictive model, can be used to better classify MLC examples.

In summary, the main contributions of this paper are as follows:

- (i) Novel semi-supervised methods based on predictive clustering trees and random forest ensembles able to deal with both MLC and HMLC tasks.
- (ii) A semi-supervised method able to produce interpretable MLC and HMLC models.
- (iii) A mechanism safeguarding the proposed method from the degradation of predictive performance.
- (iv) An extensive evaluation and analysis of the proposed method across 12 MLC and 12 HMLC datasets.

The rest of the paper is structured as follows. In Section 2, we briefly describe the work in the literature that is related to the present paper. In Section 4, we describe the proposed solution, while in Section 5, we evaluate its performance on publicly available datasets and discuss the results. Finally, in Section 7, we present the conclusions of this work and outline possible directions for future research.

2. Related Work and Motivations

SSL MLC is a relatively recent topic in machine learning and data mining. One of the most prominent works in this research area is [9], where the main idea is to combine large-margin MLC with unsupervised subspace learning. This is done by jointly solving two problems: (1) learning a subspace representation of the labeled and unlabeled inputs and (2) learning a large-margin supervised multi-label classifier on the labeled part of the data. The proposed algorithm works in a single optimization step, which results in a high-time complexity process. To alleviate the problem, the authors proposed a learning procedure which is based on sub-gradient search and coordinate descent.

In [10], the authors proposed an SSL MLC algorithm based on the optimization problem of estimating label concept compositions (label co-occurrence). Specifically, the algorithm derives a closed-form solution to this

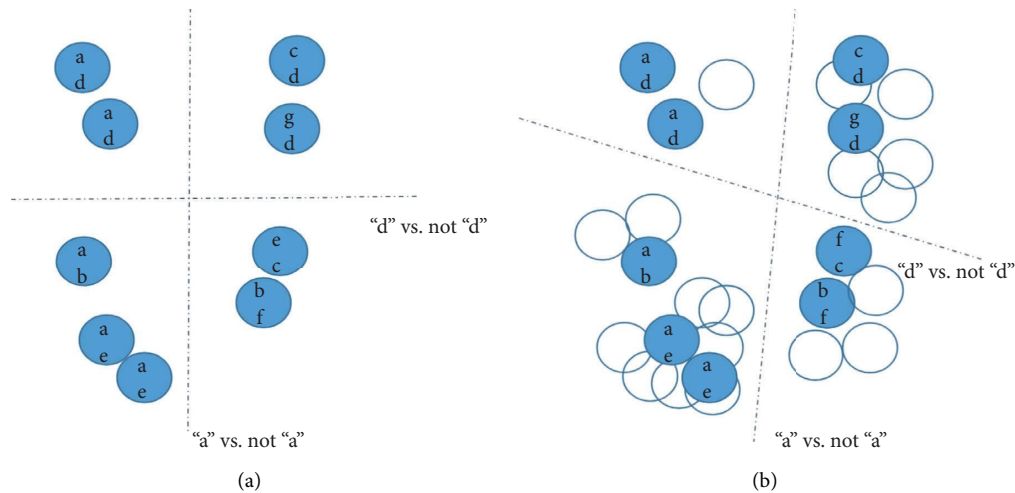


FIGURE 1: Semi-supervised learning in multi-label classification. Filled circles represent labeled examples, while empty circles represent unlabeled examples. Letters represent class labels. (a) Labeled examples only. (b) Labeled and unlabeled examples.

optimization problem and then assigns label sets to the unlabeled instances in a transductive setting.

In [11], the authors proposed a deep generative model to describe the label generation process for the SSL MLC task. For this purpose, the generative model incorporates latent variables to describe the labeled/unlabeled data as well as the labeling process. A sequential inference model is then used to approximate the model posterior and infer the ground truth labels. The same inference model is then used to predict the label of unlabeled instances.

More recently, in [14], the authors proposed a dual Relation Semi-Supervised Multi-Label Learning (DRML) approach which jointly explores the feature distribution and the label relation simultaneously. In this paper, a dual-classifier domain adaptation strategy is proposed to exploit both the feature distribution and the label relation between examples. Therefore, the optimization simultaneously takes into account instance-level relations across labeled and unlabeled samples in feature space and the relations across labels. This approach has been only applied in the image multi-label classification task.

In [15], the authors addressed the task of multi-label learning with incomplete labels, by combining the label imputation function and multi-label prediction function in a mutually beneficial manner. Specifically, the proposed method conducts automatic label imputation within a low-rank and sparse matrix recovery framework while simultaneously performing vector-valued multi-label learning and exploiting unlabeled data with vector-valued manifold regularization.

The semi-supervised multi-label learning task has also been investigated in the context of graph-structured data by incorporating the idea of label embedding to capture both network topology and higher-order multi-label correlations [16]. In this work, the label embedding is generated along with the node embedding based on the topological structure to serve as the prototype center for each class. Moreover, the similarity of the label embedding and node embedding is

used as a confidence vector to guide the label smoothing process, obtained by margin ranking optimization to learn the second-order relations between labels.

In [17], the authors derived an extension of the Manifold Regularization algorithm to multi-label classification in graph data. They then augmented the algorithm with a weighting strategy to allow differential influence on a model between instances having ground truth vs. induced labels. Therefore, the proposed approach includes three components: the graph construction, the manifold regularization with multiple labels, and the exploitation of a reliance weighting strategy.

All the previously mentioned works, although they tackle the SSL MLC problem, suffer from the common problem of not generating interpretable models. This is not the case with the method proposed in this paper, where the adoption of the PCT framework allows us to produce multi-label decision trees, which are directly interpretable and fast to learn (a preprint of this work has previously been published in [18]). Moreover, contrary to existing approaches, the approach we propose builds models by exploiting clustering. This allows us to take into account the smoothness assumption, both for the descriptive space and for the output space. Finally, the mentioned existing approaches cannot be directly used to impose limitations on the labels, and, therefore, cannot be directly used for the more complex task of HMLC.

As for the SSL HMLC task, the existing work in the literature is relatively limited. In [12], the authors extended the RAKEL system, initially developed for (supervised) MLC, to the SSL HMLC task, leading to three new methods, called HMC-SSBR, HMC-SSLP, and HMC-SSRAKEL. RAKEL is an ensemble-based wrapper method for solving MLC tasks using existing algorithms for multiclass classification. The idea is to build the ensemble by providing a small random subset of k labels (organized as a label powerset) to each base model, learned by a multiclass classifier. This approach is also used in HMC-SSBR, HMC-SSLP, and HMC-SSRAKEL, which, therefore, are not based on clustering and cannot

directly take into account the smoothness, the low-density, and the manifold assumptions.

In the more general context of semi-supervised structured output prediction, some approaches for *multitarget regression* also use predictive clustering trees. This is the case of the works in [19, 20], where the idea is to learn predictive clustering trees by using both labeled and unlabeled examples. The authors of [19] proposed a semi-supervised multitarget regression method based on the self-training approach with a random forest of predictive clustering trees. In self-training, a model is trained iteratively with its own most reliable predictions. The authors of [20] extended multitarget regression PCTs by adapting the heuristics used for the construction of the trees, in order to consider both labeled and unlabeled examples. Both methods, however, do not tackle the classification tasks.

3. Background: Predictive Clustering Trees

The predictive clustering trees (PCTs), presented in this paper for MLC and HMLC, are inspired by the work in [13]. In that work, the splits in the tree are evaluated by considering both descriptive and target attributes. The semi-supervised decision trees proposed here have similarities to the ones in [13], with multiple differences. First, Blockeel et al. [13] considered unlabeled examples only in tasks with primitive outputs, whereas we designed semi-supervised trees for structured outputs. Second, we established a parameter that allows varying degrees of supervision in the trees (i.e., how much the descriptive attributes influence the evaluation of the splits). In this way, we can build supervised, semi-supervised, or unsupervised trees, dictated by the demands of the specific dataset we are dealing with.

The PCT framework (PCTs are implemented in the CLUS system [21] available for download at <https://github.com/knowledge-technologies/clus>) treats a decision tree as a hierarchically organized set of clusters, where the topmost cluster contains all the data. This cluster is recursively divided into smaller clusters as one moves from the root to the leaves, generating PCTs. PCTs represent a generalization of default decision trees (e.g., C4.5 [22]) where the outputs are more complex structures than in conventional classification and regression tasks. Classical PCTs can predict several types of structured outputs, including nominal/real value tuples, class hierarchies, and short time series [8]. For each type, two functions must be defined: the prototype function and the variance function. The prototype function associates a class label to each leaf in the tree and it returns a representative structured value (i.e., a prototype). The variance function evaluates the homogeneity of a set of such structured values and is used to find the best splits while constructing the tree.

In this study, we propose semi-supervised PCTs and ensembles of semi-supervised PCTs, for the tasks of MLC and HMLC. Thus, in Sections 3.1 and 3.2, we present *supervised* PCTs for these tasks in more detail.

To build an ensemble model for predicting structured output, an appropriate type of PCTs is utilized as a base model. For example, to build an ensemble for the HMLC task, PCTs for HMLC are used as base models. An ensemble

predicts a new example by considering predictions of all the ensemble's base models. For regression tasks, predictions of the base models are averaged, while for classification tasks, various strategies can be used, such as the probability-based majority voting, which we used as suggested in [23]. According to this strategy, each base tree provides the probability of an example belonging to each of the possible classes. The class with the highest sum of probabilities, considering all of the base trees, is predicted.

3.1. PCTs for Multi-Label Classification. The variance function for learning PCTs for the MLC task computes the average of the *Gini* indices across all the target variables. For a set of examples E with target space Y , consisting of T nominal target variables Y_1, Y_2, \dots, Y_T , the variance function is defined as follows:

$$\text{Var}_f(E, Y) = \frac{1}{T} \cdot \sum_{i=1}^T \text{Gini}(E, Y_i), \quad (1)$$

where $\text{Gini}(E, Y_i)$ is the Gini score of the i^{th} target variable Y_i for a set of examples E . The Gini score of the i^{th} target variable is calculated as follows:

$$\text{Gini}(E, Y_i) = 1 - \sum_{j=1}^{C_i} p_j^2, \quad (2)$$

where C_i is the number of classes for the target variable Y_i (e.g., if Y_i is binary, then $C_i = 2$) and p_j is the *a priori* empirical probability of a class c_j (i.e., the relative frequency of instances in E that belong to the class c_j).

The sum of the entropies of class variables can also be used as a variance indicator, i.e., $\text{Var}_f(E, Y) = \sum_{i=1}^T \text{Entropy}(E, Y_i)$ (this was considered previously for MLC [24]). The CLUS framework includes other variance functions as well, such as reduced error, gain ratio, and the m -estimate.

The prototype function returns a vector denoting probabilities of an instance belonging to a given class for each target variable. To determine the predicted classes, the user can specify a threshold on probabilities, or the majority class (i.e., the most probable one) for each target can be calculated. In this study, we use the majority class.

3.2. PCTs for Hierarchical Multi-Label Classification. In HMLC, the target space Y is associated to a hierarchy of classes (C, \leq_h) , where $\forall c_l, c_j \in C: c_l \leq_h c_j \Leftrightarrow c_l$ is a super class of c_j . The set of labels of example e_i is represented as a binary vector L_i , whose j^{th} component is 1 if the example is labeled with the class c_j , 0 otherwise. The j^{th} component of the arithmetic mean of such vectors contains the relative frequency of examples of the set belonging to class c_j . Then, the variance indicator over a set of examples E is calculated as the average squared distance between each vector (L_i) and the set's mean class vector (\bar{L}):

$$\text{Var}_f(E, Y) = \frac{1}{|E|} \cdot \sum_{e_i \in E} d(L_i, \bar{L})^2. \quad (3)$$

In the HMLC context, the similarities at higher levels of the hierarchy are considered to be more important than the similarities at lower levels. The distance measure in the above formula (weighted Euclidean distance) is therefore defined as follows:

$$d(L_1, L_2) = \sqrt{\sum_{l=1}^{|C|} \omega(c_l) \cdot (L_{1,l} - L_{2,l})^2}, \quad (4)$$

where $L_{i,l}$ is the l^{th} component of the class vector L_i of an instance e_i , $|C|$ is the number of classes in the hierarchy (i.e., the size of the class vector), and the class weights $\omega(c)$ decrease with the depth of the class in the hierarchy. More precisely, $\omega(c) = \omega_0^{\text{depth}(c)}$, where $\text{depth}(c)$ denotes the depth of the class c in the hierarchy, and $0 < \omega_0 < 1$. Note that class weights can be calculated recursively, i.e., $\omega(c) = \omega_0 \cdot \omega(\text{par}(c))$, where $\text{par}(c)$ denotes the parent of class c . In this work, we use $\omega_0 = 0.75$, as recommended in [25].

The definition of $\omega(c)$ is general enough to represent classes that are organized as a directed acyclic graph (DAG). Generally, a DAG-like hierarchy can be interpreted in two ways: an example belonging to a class c , either (i) belongs to all superclasses of c , or (ii) belongs to one or more superclasses of c . In this work, we consider the former.

The variance indicator for tree-structured hierarchies uses the weighted Euclidean distance between the class vectors (as defined in equation (4)), where the weight of a class changes depending on its level in the hierarchy. Note that in DAG-shaped hierarchies, the classes do not have a unique level number. To resolve this issue, we follow the recommendation in [25]: the weight of a given class is calculated as an average of all the weights according to possible paths from the root to that class.

In classification trees, a leaf holds the majority class of its examples, which the tree predicts for examples arriving in that leaf. In the HMLC task, an example can have multiple classes, so the meaning *majority class* is not straightforward. The prediction, in this case, is a mean \bar{L} of the class vectors of the examples in the leaf. The i^{th} component of the vector \bar{L} can be considered as the probability that an example in the leaf belongs to class c_i . The final classification for an example that arrives in the leaf can be made using a threshold τ for the probabilities; if $\bar{L}_i > \tau$, then class c_i is predicted for the example. When making predictions, the parent-child relationships from the class hierarchy are preserved if the values for the thresholds τ are defined as follows: $\tau_i \leq \tau_j$ whenever $c_i \leq_h c_j$ (c_i is an ancestor of c_j). The selection of the threshold τ depends on the use scenario, e.g., trading off higher precision with lower recall. Here, we use a threshold-independent metric based on precision-recall curves to evaluate the predictive performance of the models.

4. Semi-Supervised PCT Learning for MLC and HMLC

4.1. Task Definition. Here, we formally define the semi-supervised learning tasks for the types of structured outputs considered in this study: predicting multiple targets and hierarchical multi-label classification.

4.1.1. Semi-Supervised Multi-Label Classification. In MLC, the task is to predict several binary values (i.e., labels) for each example. This is formalized as follows:

Given:

- (i) A descriptive (or input) space $X = X_1 \times X_2 \times \dots \times X_D$ spanned by D descriptive variables that consist of values of primitive data types (Boolean, nominal, or continuous).
- (ii) A target (or output) space $Y = Y_1 \times Y_2 \times \dots \times Y_T$ spanned by the T binary target variables.
- (iii) A set of labeled examples $E_l = \{(x_i, y_i) \mid x_i \in X, y_i \in Y, 1 \leq i \leq N_l\}$, where each example $e_i \in E_l$ is described according to both the descriptive space and the target space, and N_l is the number of labeled examples.
- (iv) A set of unlabeled examples $E_u = \{(x_i) \mid x_i \in X, 1 \leq i \leq N_u\}$, where each example $e_i \in E_u$ takes its values from the descriptive space only, and N_u denotes the number of unlabeled examples.
- (v) A quality metric q , e.g., which favours models with high predictive accuracy (or low predictive error).

Find: A function $f: X \rightarrow Y$ that maximizes q .

4.1.2. Semi-Supervised Hierarchical Multi-Label Classification. In HMLC, each example can have more than one class (multiple labels), and the classes are organized in a hierarchical structure, i.e., an example belonging to a class also belongs to all its superclasses. This is formalized as follows:

Given:

- (i) A descriptive (or input) space $X = X_1 \times X_2 \times \dots \times X_D$ spanned by D descriptive variables that consist of values of primitive data types (Boolean, nominal, or continuous).
- (ii) A target space Y , defined with a hierarchy of classes (C, \leq_h) , where C is a set of classes and \leq_h is a partial order among them, representing the superclass relationship, i.e., $\forall c_1, c_2 \in C, c_1 \leq_h c_2$ if and only if c_1 is a superclass of c_2 .
- (iii) A set of labeled examples $E_l = \{(x_i, Y_i) \mid x_i \in X, Y_i \subseteq C, 1 \leq i \leq N_l\}$, where each example $e_i \in E_l$ is a pair of a tuple x_i from the descriptive space and a set S_i from the target space, and each set satisfies the hierarchy constraint, i.e., $c \in S_i, c' \in C, c' \leq_h c \Rightarrow c' \in Y_i$, and N_l is the number of labeled examples.
- (iv) A set of unlabeled examples $E_u = \{(x_i) \mid x_i \in X, 1 \leq i \leq N_u\}$, where each example $e_i \in E_u$ takes its

values from the descriptive space only, and N_u is the number of unlabeled examples.

- (v) A quality metric q , e.g., which favours models with high predictive accuracy (or low predictive error).

Find: a function $f: X \rightarrow 2^C$ (where 2^C is the power set of C) such that f maximizes q and the predictions made by f satisfy the hierarchy constraint, i.e., $c \in f(x), c' \in 2^C, c' \leq_h c \Rightarrow c' \in f(x)$.

4.2. Tree Learning. The proposed semi-supervised algorithm (see Table 1) is based on the extension of the standard *top-down induction of decision trees* (TDIDT) algorithm used to build supervised PCTs [26]. An input to the TDIDT algorithm is a set of examples E . The heuristic (h) selects the best tests (t^*) based on the reduction of the variance resulting from partitioning (\mathcal{P}) the examples (BestTest function in Table 1). As the variance reduction is maximized, the homogeneity of the cluster is also maximized. If no suitable test is found, i.e., if none of the candidate tests results in a significant reduction of the variance or if there are fewer examples in a node than the specified limit, then a leaf is created and the prototype of the examples in that leaf is computed.

As an input, the SSL-PCT algorithm uses a set of labeled examples (E_l), a set of unlabeled examples (E_u), and a $w \in [0, 1]$ parameter. The w parameter is optimized using the procedure that relies on internal cross-validation.

The supervised TDIDT algorithm for PCTs is extended towards semi-supervised learning as follows. First, the input to the SSL algorithm dataset comprises both labeled and unlabeled examples: $E = E_l \cup E_u$, where E_l are labeled examples and E_u are unlabeled examples. Second, the variance function in the SSL algorithm considers both the target and the descriptive attributes in the evaluation of splits. It is calculated as a weighted sum of the variance over the target space Y and the variance over the descriptive space X :

$$\text{Var}_f(E, Y, X, w) = w \cdot \text{Var}_f(E, Y) + (1 - w) \cdot \text{Var}_f(E, X), \quad (5)$$

where $w \in [0, 1]$ is the parameter that controls the trade-off between the contribution of the target space and the descriptive space to the variance function. During the learning of semi-supervised regression trees, the w parameter is automatically optimized by an internal cross-validation procedure (OptimizeParamW function in Table 1).

This extension relies on the semi-supervised cluster assumption [1]: *if examples are in the same cluster, then they are likely to be of the same class*. We recall that the variance function of supervised PCTs uses only the target attributes (equations (1) and (3)). Consequently, (a) unlabeled examples cannot contribute to the tree construction (since only their descriptive attributes are known), and (b) the clusters produced by supervised PCTs are only homogeneous regarding the class label. Enforcing the similarity of examples in both the descriptive and the target space during the construction of SSL-PCTs results in clusters that are

TABLE 1: The proposed algorithm for learning of semi-supervised predictive clustering trees.

<i>Procedure</i> SSL-PCT	
<i>Input:</i> A dataset $E = E_l \cup E_u$, a w parameter	
<i>Output:</i> A predictive clustering tree	
(1):	$(t^*, h^*, \mathcal{P}^*) = \text{BestTest}(E, w)$
(2):	if $t^* \neq \text{none}$ then
(3):	for each $E_i \in \mathcal{P}^*$ do
(4):	tree _{i} = SSL-PCT(E_i, w)
(5):	return node($t^*, \cup_i \{\text{tree}_i\}$)
(6):	else
(7):	return leaf (Prototype(E))
<hr/>	
<i>Procedure</i> OptimizeParamW	
<i>Input:</i> A dataset $E = E_l + E_u$; a set of values W , $\forall w \in W, w \in [0, 1]$; a number of folds k	
<i>Output:</i> A w value	
(1):	for each $w \in W$ do
(2):	w _{opt} = arg max _{$w \in W$} CrossValidate(E, w, k)
(3):	return w _{opt}
<hr/>	
<i>Procedure</i> BestTest	
<i>Input:</i> A dataset E , a w parameter	
<i>Output:</i> The best test (t^*), its heuristic score (h^*), and the partition (\mathcal{P}^*) it induces on the dataset (E)	
(1):	$(t^*, h^*, \mathcal{P}^*) = (\text{none}, 0, \emptyset)$
(2):	for each possible test t do
(3):	$\mathcal{P} =$ partition induced by t on E
(4):	$h = \text{Var}_f(E, Y, X, w) - \sum_{E_i \in \mathcal{P}} E_i / E \text{Var}_f(E_i, Y, X, w)$
(5):	if $(h > h^*) \wedge \text{Acceptable}(t, \mathcal{P})$ then
(6):	$(t^*, h^*, \mathcal{P}^*) = (t, h, \mathcal{P})$
(7):	return $(t^*, h^*, \mathcal{P}^*)$
<hr/>	
<i>Procedure</i> CrossValidate	
<i>Input:</i> A dataset $E = E_l \cup E_u$, $w \in [0, 1]$, a number of folds k	
<i>Output:</i> An accuracy measure	
(1):	$E_l^{(1,2,\dots,k)}$ = partition E_l into k folds
(2):	for each $j \in \{1, 2, \dots, k\}$ do
(3):	tree _{j} = SSL-PCT ($E_l^{(1,2,\dots,k) \setminus j} \cup E_u, w$)
(4):	accuracy _{j} = evaluate (tree _{j} , E_l^j)
(5):	return $\sum_{j=1}^k \text{accuracy}_j / k$

homogeneous regarding both the descriptive and the target space. This allows us to exploit both labeled and unlabeled examples. Finally, following the cluster assumption, labeled and unlabeled examples that end up in the same leaf of the tree are likely to be of the same class.

Parameter w controls the magnitude of the contribution that unlabeled examples have on the learning of semi-supervised PCTs. In other words, parameter w enables learned models to range from fully supervised ($w = 1$) to completely unsupervised ($w = 0$). The control of the contribution of unlabeled examples enabled by the w parameter allows us to set the amount of supervision for different datasets appropriately. This aspect is discussed in more detail in the experimental analysis (Section 6.3).

The variance of a set of examples E on target space Y is calculated differently, depending on the type of structured output at hand:

$$\text{Var}_f(E, Y) = \begin{cases} \frac{\sum_{i=1}^T \text{Gini}(E, Y_i)}{T}, & \text{if } Y \text{ consists of } T \text{ binary variables,} \\ \frac{(1/|E|) \cdot \sum_{e_i \in E_i} d(L_i, \bar{L})^2}{\sum_{l=1}^{C_l} w(c_l)}, & \text{if } Y \text{ is a hierarchy of classes.} \end{cases} \quad (6)$$

Since the descriptive variables can be either numeric or nominal, the variance on the descriptive space of a set of examples E is computed as follows:

$$\text{Var}_f(E, X) = \frac{1}{D} \cdot \left(\sum_{X_i \text{ is numeric}} \text{Var}(E, X_i) + \sum_{X_j \text{ is nominal}} \text{Gini}(E, X_j) \right), \quad (7)$$

where D is the number of descriptive attributes and the variance or the Gini score of descriptive attributes is calculated following equations (8) and (9).

Let N be the number of examples (both labeled and unlabeled), and let K_i be the number of examples with nonmissing values of the i^{th} attribute Y_i . Then, the variance for the continuous attributes and the Gini index for the nominal attributes are calculated as follows, respectively:

$$\text{Var}(E, Y_i) = \frac{(N - 1/K_i - 1) \cdot \sum_{j=1}^{K_i} (y_{i,j})^2 - N \cdot ((1/K_i) \cdot \sum_{j=1}^{K_i} y_{i,j})^2}{N}, \quad (8)$$

$$\text{Gini}(E, Y_i) = 1 - \sum_{j=1}^{C_i} \left(\frac{|\{e : e \in E \wedge c_j \in \text{classes}(e)\}|}{K_i} \right)^2 = 1 - \sum_{j=1}^{C_i} \hat{p}_j, \quad (9)$$

where C_i is the number of class values of Y_i , and \hat{p}_j is the *a priori* probability of class value c_j , estimated by using only examples for which the value for variable Y_i is known. Note that for the HMLC task, the variance for the output space is calculated only on the labeled data (see equation (6)).

The variances of descriptive and target attributes are normalized, similarly to supervised PCTs, to ensure the equal contribution of attributes to the final variance. Normalization is performed by dividing the variance estimates of individual attributes in equations (8) and (9) (that consider the set of examples in the current node of the tree) with the variance of the corresponding attribute considering the entire training set.

During the semisupervised tree construction phase, two extreme cases can occur: (1) only unlabeled examples can end up in a leaf of the tree; therefore, the prototype function cannot be calculated, or (2) variance needs to be calculated for attributes where none of the examples (or only one) have nonmissing values (e.g., $K_i \leq 1$ in equation (8)). For the first extreme case, we calculate the prototype function of such a leaf by returning the prototype of its first parent node that contains labeled examples. In other words, from a leaf that contains only unlabeled examples, we move up the tree until we encounter a node containing labeled examples and we

return the prototype of such a node. The prototype is calculated using only labeled examples as described in Sections 3.1 and 3.2. Nodes with only unlabeled examples are not split further, while in leaf nodes containing labeled examples, we allow a minimum of 2 labeled examples. Both criteria can be considered as “stopping criteria,” to stop the tree construction phase. Note that these criteria are coherent with the stopping criteria implemented in supervised PCTs, where at least two labeled examples in a leaf node are required.

The second extreme case can occur when the examples in a node are split in a way that only unlabeled examples go into a single branch of the tree. In such a case, a split needs to be evaluated with one of the branches containing only missing values for the target attribute(s); therefore, variance for such attributes cannot be calculated. Similarly, as in the first extreme, we handle this situation by using the variance of the parent node (for the attributes containing only missing values in the original node). Note that, since we do not split nodes with only unlabeled examples, the parent node is guaranteed to contain labeled examples.

4.2.1. Semi-Supervised PCTs with Feature Weighting. PCTs (and decision trees in general) are robust to irrelevant features since the learning algorithm chooses only the most

informative features when building (supervised) trees. Thus, irrelevant features will be ignored. However, in semi-supervised PCTs, this feature may be compromised, since the evaluation of the splits depends on both target and descriptive attributes. To deal with this issue, we propose feature-weighted SSL-PCTs.

Methods for feature weighting can be used to identify the most informative features by determining an importance score (weight), where a higher score denotes more informative features, while a lower score denotes less informative ones. The effectiveness of feature weighting with the importance scores was shown to help the k-nearest neighbors algorithm to deal with irrelevant features [27]. Similarly, we adapt the SSL-PCTs and use importance scores to assign weights to features.

More specifically, we use a feature ranking method based on a random forest of PCTs [8], to obtain the importance score σ_i for each descriptive attribute X_i . To calculate feature importance, this method uses the internal out-of-bag (OOB) error as an estimate of the noise in the descriptive space. The rationale is that if noise is introduced to a descriptive variable which is important, then the error of the model will increase (as measured by OOB error estimates).

The feature ranking is performed on the labeled examples E_l prior to building SSL-PCTs or SSL-RFs. The importance scores are then normalized as follows: $\sigma_i = \hat{\sigma}_i / \max(\hat{\sigma}_1, \hat{\sigma}_2, \dots, \hat{\sigma}_D)$. The function for the calculation of the variance of the descriptive attributes of SSL-PCTs is then adapted to include normalized feature importance scores σ_i as weights of the descriptive attribute X_i :

$$\begin{aligned} \text{Var}_f(E, X) = & \frac{1}{D} \cdot \sum_{X_j \text{ is numeric}} \sigma_i \cdot \text{Var}(E, X_i) \\ & + \frac{1}{D} \cdot \sum_{X_j \text{ is nominal}} \sigma_j \cdot \text{Gini}(E, X_j). \end{aligned} \quad (10)$$

This results in irrelevant features contributing less to the variance score. Henceforth, semi-supervised PCTs and random forests with feature weighting are denoted as SSL-PCT-FR and SSL-RF-FR, respectively.

4.3. Semi-Supervised Random Forests. SSL-PCTs can be easily extended to their random forest version [28]. This is done by using SSL-PCTs as the members of the random forest ensemble, instead of using classical supervised trees. The notable difference is, however, in the presence of both labeled and unlabeled examples in the bootstrap samples, which does not conform with the classical random forest algorithm [28]. That is, the trees can be built with only a small set of labeled examples and a large set of unlabeled examples, and thus bootstrap samples may end up containing only unlabeled examples. In order to overcome this problem, in the semi-supervised setting, we perform stratified bootstrap sampling where the proportions of labeled and unlabeled examples are preserved in each bootstrap sample. For example, if the training data contain 10% of

labeled and 90% of unlabeled examples, such a ratio is maintained in bootstrap samples. This is achieved by separately sampling labeled and unlabeled examples and later joining them to form a bootstrap sample for the random forest algorithm.

4.4. Computational Complexity. To assess the complexity of the algorithm for learning SSL-PCTs, we first introduce the computational complexity of learning supervised PCTs: sorting of D descriptive variables ($\mathcal{O}(DN \log N)$), used to determine the best split for T target variables ($\mathcal{O}(TDN)$), for N labeled training examples ($\mathcal{O}(N)$). If we assume that the expected depth of the tree is $\mathcal{O}(\log N)$ [29], the computational complexity of building a single PCT is $\mathcal{O}(DN \log^2 N) + \mathcal{O}(TND \log N) + \mathcal{O}(N \log N)$.

Now we discuss the changes introduced in SSL-PCTs. First, the number of training examples N in the semi-supervised case equals the combined number of unlabeled and labeled examples (i.e., $N = N_l + N_u$, instead of $N = N_l$). Second, SSL-PCTs use both D descriptive variables and T target variables to determine the best split; therefore, this step has the complexity of $\mathcal{O}((T+D)ND)$. Therefore, the computational complexity of building an SSL-PCT tree is $\mathcal{O}(DN \log^2 N) + \mathcal{O}((D+T)ND \log N) + \mathcal{O}(N \log N)$.

The computational complexity of random forests of semi-supervised PCTs is bounded by $k(\mathcal{O}(D'N' \log^2 N') + \mathcal{O}((T+D)N'D' \log N'))$, where N' is the number of bootstrap samples, D' is the number of features considered at each tree node, and k is the number of trees. The added computational complexity of feature ranking is that of randomly permuting the values of the out-of-bag samples ($N'' = N - N'$) and sorting the samples through the tree. Both operations are done for each descriptive attribute and their cost is $\mathcal{O}(DN'' + D \log N)$. This added computational cost is, however, negligible compared to the overall cost of building the random forest ensemble. Note that the number of examples in feature ranking is E_l because the feature weights are calculated considering only the labeled examples.

5. Experimental Design

In this section, we first describe the datasets used in the experimental evaluation. Next, we present the evaluation procedure, the specific parameter settings of the algorithms, and the performance measures.

5.1. Data Description. To evaluate the proposed methods, we use 24 datasets of the two structured output prediction tasks considered: MLC and HMLC. The datasets are from various domains and have different sizes and numbers of descriptive and target variables. The characteristics of the datasets are summarized in Tables 2 and 3 for the MLC and HMLC tasks, respectively.

5.2. Experimental Setup. We introduce semi-supervised PCTs (SSL-PCT) and their feature-weighted variant (SSL-PCT-FR). We compare these methods across different structured output prediction tasks with supervised PCT

TABLE 2: MLC datasets and their characteristics.

Dataset	Domain	N	D/C	\mathcal{L}	$\overline{\mathcal{L}}_L$
Bibtex [30]	Text	7395	1836/0	159	2.402
Birds [31]	Audio	645	2/258	19	1.014
Emotions [32]	Music	594	0/72	6	1.869
Corel5k [33]	Images	5000	499/0	374	3.522
Enron [34]	Text	1702	1001/0	53	3.378
Genbase [35]	Text	662	1186/0	27	1.252
Mediana [36]	Media	7953	21/58	5	1.205
Medical [37]	Text	978	1449/0	45	1.245
Scene [38]	Images	2407	0/294	6	1.074
SIGMEA real [39]	Ecology	817	0/4	2	0.726
Slovenian rivers [40]	Ecology	1060	0/16	14	5.073
Yeast [41]	Biology	2417	0/103	14	4.237

N is the number of examples, D/C is the number of descriptive variables (nominal/continuous), \mathcal{L} is the number of labels, and $\overline{\mathcal{L}}_L$ is the average number of labels per example.

TABLE 3: HMLC datasets and their characteristics.

Dataset	Domain	N	D/C	\mathcal{H}	$ \mathcal{H} $	\mathcal{H}_d	$\overline{\mathcal{L}}_L$
Danish farms [42]	Ecology	1944	132/5	Tree	72	3	7
Enron [34]	Text	1702	1001/0	Tree	53	2	3.38
Slovenian rivers [40]	Ecology	1060	0/16	Tree	724	4	25
ImCLEF07A [43]	Images	11006	0/80	Tree	96	3	1
ImCLEF07D [43]	Images	11006	0/80	Tree	46	3	1
Diatoms [44]	Images	3119	0/371	Tree	377	3	0.94
Cellcycle-GO [25]	Genomics	3766	0/77	DAG	4126	12	35.91
Church-GO [25]	Genomics	3764	1/26	DAG	4126	12	35.89
Derisi-GO [25]	Genomics	3733	0/63	DAG	4120	12	35.99
Eisen-GO [25]	Genomics	2425	0/79	DAG	3574	11.12	39.04
Expr-GO [25]	Genomics	3788	4/547	DAG	4132	12	35.87
Pheno-GO [25]	Genomics	1592	69/0	DAG	3128	12	36.43

N is the number of examples, D/C is the number of descriptive variables (nominal/continuous), \mathcal{H} is the type of the label hierarchy, $|\mathcal{H}|$ is the number of nodes in the hierarchy, \mathcal{H}_d is the maximal depth of the hierarchy, and $\overline{\mathcal{L}}_L$ is the average number of labels per example.

algorithms for MLC and HMLC, denoted as SL-PCT, in order to estimate the contribution of unlabeled data to the predictive performance of the methods under the same conditions. By such comparison, we can answer our main question: Are SSL-PCTs able to outperform supervised PCTs? In the experiments with single trees, we use the pruning procedure as implemented in M5 regression trees [22].

We also compare the predictive performance of semi-supervised random forests (SSL-RF) and their feature-weighted variant (SSL-RF-FR) to supervised random forests for structured output prediction (CLUS-RF). We use 100 unpruned trees to construct random forests. The number of features randomly selected at each node was set to $\lfloor \log_2(D) + 1 \rfloor$, where D is the total number of features [28].

To assess the influence of different proportions of labeled/unlabeled data for the semi-supervised method, we vary the number of labeled examples across the following set of values: {50, 100, 200, 350, 500}. The labeled examples are randomly sampled from the training set, while the rest of the

examples are used both as unlabeled examples and as testing data. We temporarily ignore their labels and use them in the semi-supervised methods as unlabeled training samples. The test set used to evaluate the models comprises the same examples and their original labels restored. The evaluation scenario is thus in the context of transductive learning. The supervised methods are trained using the selected labeled samples and evaluated on the same test set as semi-supervised methods. This is repeated 10 times using different random initialization, while the predictive performances are averaged over the 10 runs.

For each of the 10 runs, we optimize the parameter w (weight) by an internal 3-fold cross-validation procedure performed on the labeled portion of the training set. The semi-supervised methods also use the available unlabeled examples. The values of the parameter w vary from 0 to 1 with a step of 0.1.

The algorithms are evaluated by means of the area under the Precision-Recall curve (AUPRC). Since the considered tasks are MLC and HMLC, we use a variant of the

AUPRC—the area under the micro-averaged average Precision-Recall curve (AUPRC), as suggested in [25]. Specifically, the precision and recall values are computed as follows:

$$\begin{aligned}\overline{\text{Prec}} &= \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FP_i}, \\ \overline{\text{Rec}} &= \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FN_i},\end{aligned}\quad (11)$$

where i ranges over all the classes.

We statistically analyze the results following the recommendations of Demsar [45]. We use the nonparametric Wilcoxon paired signed-rank test [46] for the comparison of the predictive performance of the two methods over multiple datasets. We set the significance level to 0.05 in all the experiments.

6. Results and Discussion

6.1. Predictive Performance of the Methods

6.1.1. Multi-Label Classification. Figure 2 presents the predictive performance (AUPRC) of semi-supervised (SSL-PCT, SSL-PTC-FR, SSL-RF, and SSL-RF-FR) and supervised methods (SL-PCT and CLUS-RF) on the 12 MLC datasets, with an increasing amount of labeled data.

We can clearly observe that semi-supervised PCTs are superior to SL-PCTs on most of the datasets. That is, on 8 out of 12 datasets, either SSL-PCTs or SSL-PCT-FRs (or both) dominate the performance of SL-PCTs by a good margin. On the other four datasets, namely, Corel5k, Emotions, Mediana, and SIGMEA real, the performance of supervised and semi-supervised PCTs is mostly the same as or similar to the performance of SL-PCTs.

Intuitively, the improvement of semi-supervised over supervised methods should diminish as the number of labeled examples increases, and eventually, semi-supervised and supervised methods are expected to converge to the same or similar performance. However, the “convergence point” changes from dataset to dataset. For instance, for Genbase, at 500 labeled examples, we already see this convergence. For the other datasets, the improvement of semi-supervised over supervised methods decreases less quickly.

The feature-weighted semi-supervised method (SSL-PCT-FR) and the non-feature-weighted one (SSL-PCT) have similar trends in predictive performance. However, on some datasets, there are notable differences. Namely, on Birds and Scene datasets, feature weighting is beneficial for the predictive performance of SSL-PCTs and even necessary for improvement over SL-PCTs on the Birds dataset with ≥ 350 labeled examples. On the other hand, feature weighting clearly damages the predictive performance of the SSL-PCT method on the Bibtext dataset. Thus, feature weighting helps in most cases, but the empirical results cannot support its use by default when building SSL-PCTs for MLC.

We next compare semi-supervised random forests (SSL-RF) with supervised random forests (CLUS-RF). From the results, we can observe that CLUS-RF improves over

CLUS-RF on several datasets: Bibtext, Corel5k, Genbase, Medical, SIGMEA real, and marginally on Emotions and Enron datasets. However, as compared to single trees, the improvements of the semi-supervised approach over the supervised are observed on fewer datasets and are smaller in magnitude. In other words, the improvement of SSL-PCTs over SL-PCTs does not guarantee the improvement of SSL-RF over CLUS-RF (e.g., Mediana and Yeast datasets), and vice versa, SSL-RF can improve over CLUS-RF even if SSL-PCTs does not improve over SL-PCTs (e.g., SSL-RF-FR on Emotions dataset for 200 and 350 labeled examples). As observed for the single trees, there is no clear advantage to using feature weighting when semi-supervised random forests are built, even though it is somewhat helpful on the Emotions and Enron datasets.

Feature-weighted PCTs and RFs could possibly be improved by considering a semi-supervised feature selection [47], instead of the supervised method based on random forests, as used here.

6.1.2. Hierarchical Multi-Label Classification. Figure 3 presents the learning curves in terms of the predictive performance (AUPRC) of semi-supervised (SSL-PCT, SSL-PTC-FR, SSL-RF, and SSL-RF-FR) and supervised methods (SL-PCT and CLUS-RF) on the 12 hierarchical multi-label classification datasets.

We observe different behaviours on the 6 functional genomics datasets and the 6 datasets from the other domains. That is, on all 6 datasets of the latter group, semi-supervised PCTs improve over SL-PCTs—albeit not necessarily always for all available amounts of labeled data. This is the case on the Enron and ImCLEF07A datasets, where both SSL-PCTs and SSL-PCT-FR dominate the performance of SL-PCTs, while it seems that on other datasets, at least 100 (Slovenian rivers and ImCLEF07D) or 500 (Danish farms) labeled examples are needed to improve over SL-PCTs.

On the other hand, semi-supervised PCTs are not so successful on functional genomics datasets. Analysis of the tree sizes (see Section 6.4) reveals an explanation for such results. That is, on all of the 6 functional genomics datasets, and for almost all different amounts of labeled data, both supervised and semi-supervised trees are composed of only one node. Note that these datasets have extremely large label hierarchies which are very sparsely populated. It seems that for such datasets the amount of labeled data we considered (i.e., up to 500 labeled examples) is not sufficient to build trees—neither supervised nor semi-supervised. In fact, semi-supervised trees have more than one node on Expr-GO (≥ 350 of labeled examples) and Eisen (for 500 labeled examples) datasets, and those are exactly the occasions where they improve over supervised trees. We thus hypothesize that for larger amounts of labeled data, SSL-PCTs could outperform supervised PCT also on functional genomics datasets.

In the HMLC task, the feature-weighted semi-supervised method (SSL-PCT-FR) and non-feature-weighted one (SSL-PCT) mostly have a very similar performance. Again, as for the MLC task, there is no clear benefit of feature weighting.

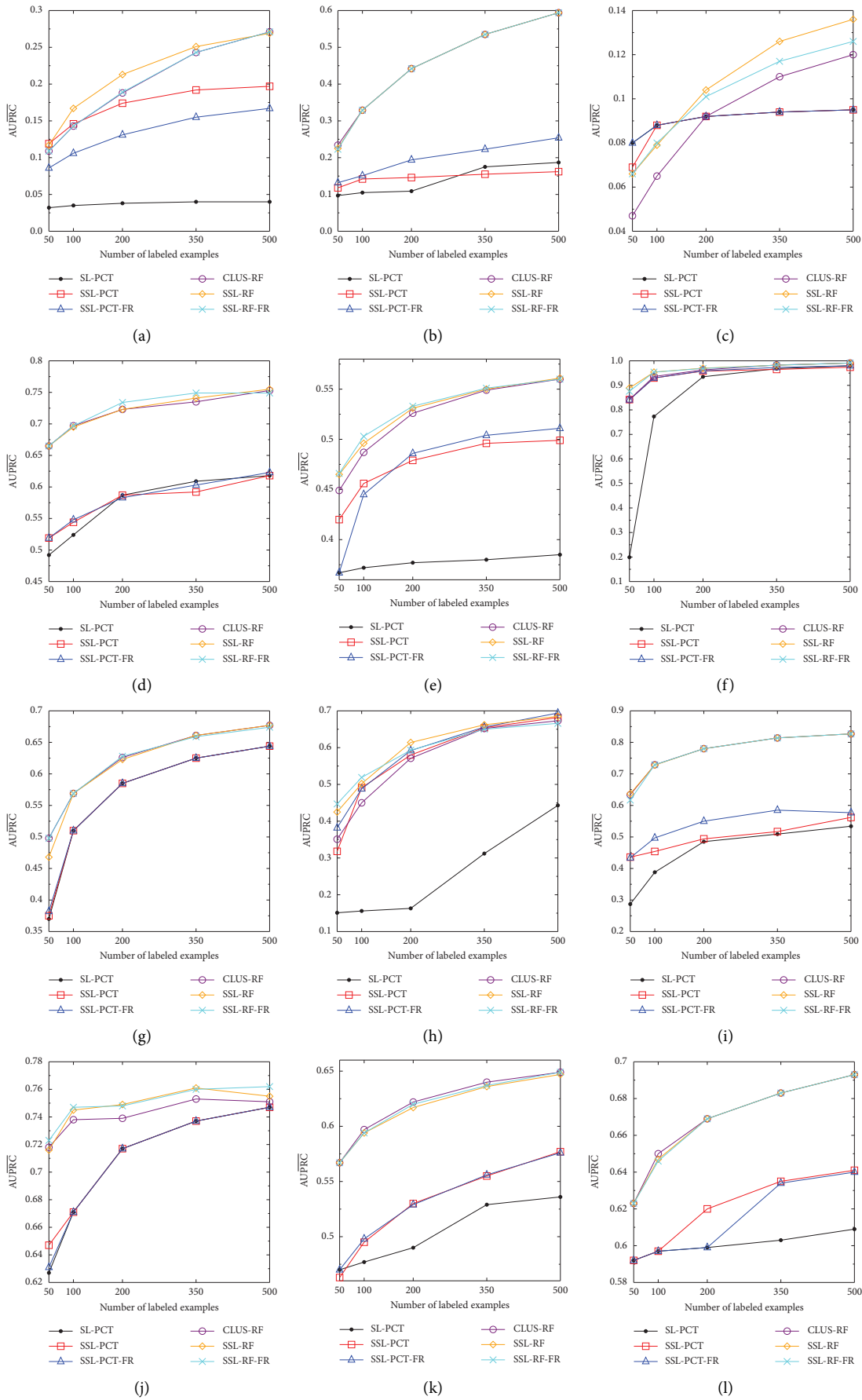


FIGURE 2: Predictive performance of the supervised and semi-supervised methods on the multi-label classification datasets. (a) Bibtex. (b) Birds. (c) Corel5k. (d) Emotions. (e) Enron. (f) Genbase. (g) Mediana. (h) Medica. (i) Scene. (j) SIGMEA real. (k) Slovenian rivers. (l) Yeast.

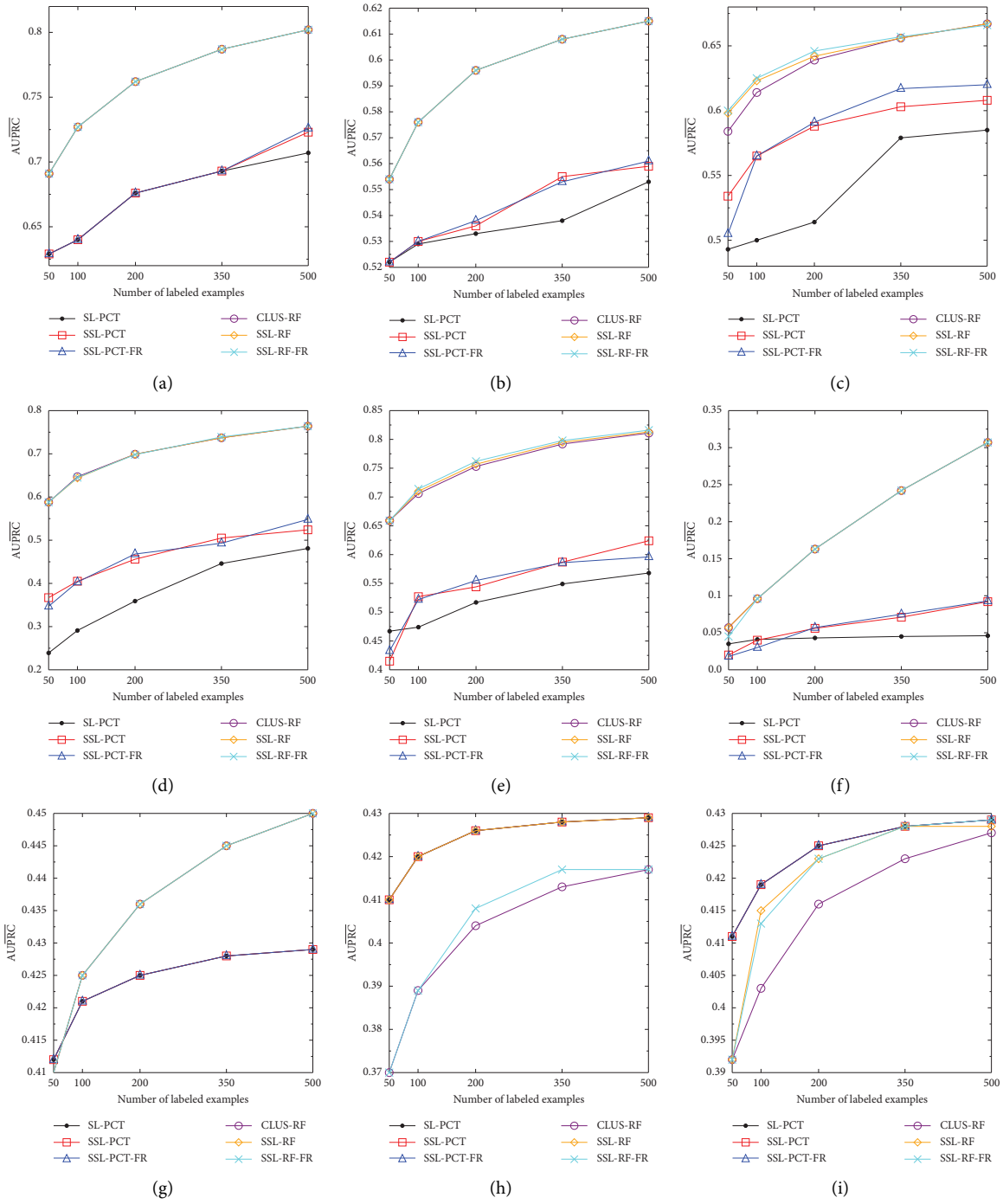


FIGURE 3: Continued.

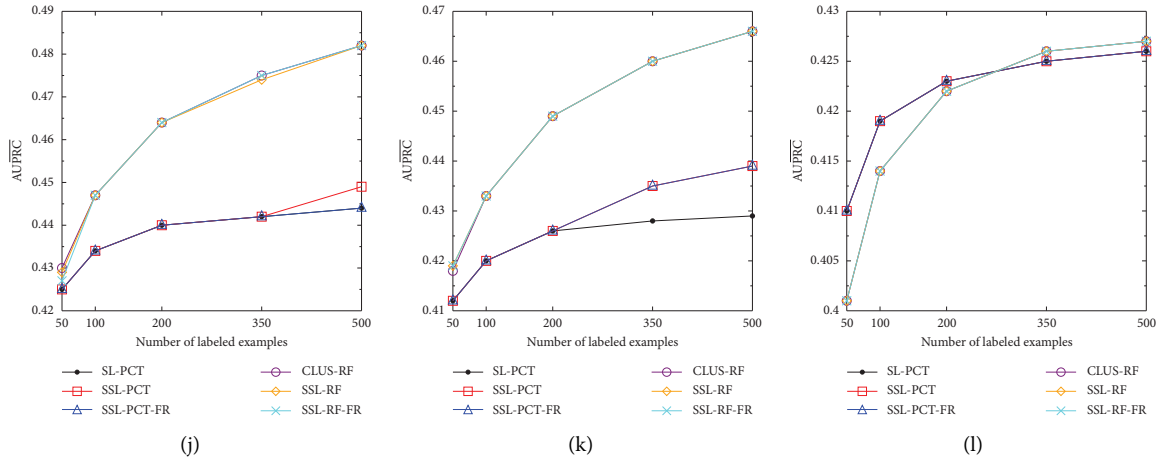


FIGURE 3: Predictive performance of the supervised and semi-supervised methods on the hierarchical multi-label classification datasets. (a) Danish farms. (b) Slovenian rivers. (c) Enron. (d) ImCLEF07A. (e) ImCLEF07D. (f) Diatoms. (g) Cellcycle-GO. (h) Church-GO. (i) Derisi-GO. (j) Eisen-GO. (k) Expr-GO. (l) Pheno-GO.

Finally, the semi-supervised random forests (SSL-RF and SSL-RF-FR) outperform supervised random forests (CLUS-RF) on some datasets, namely, in the initial part of the learning curve for the Enron dataset, and for the Church-GO and Derisi-GO datasets, meaning that unlabeled data improve the predictive performance of random forests of PCTs for HMLC. On the remaining datasets, it seems that unlabeled data are not beneficial for the performance of random forests of PCTs for HMLC.

6.2. Statistical Analysis of Predictive Performance. The results of the statistical analysis (Table 4) show that SSL-PCTs and SSL-PCT-FR are statistically significantly better than the SL-PCTs for most of the different amounts of labeled data, considered for both structured output prediction tasks. More specifically, for the HMLC task, usually, at least 200 labeled examples are needed to achieve statistical significance. In the MLC task, on the other hand, SSL-PCT achieves statistically significantly better results than SL-PCT up to 200 labeled examples. In this task, the feature-weighted SSL-PCTs are more successful: statistically, they significantly outperform SL-PCT across all different amounts of labeled examples.

Considering the feature-weighted and non-feature-weighted semi-supervised methods (both single trees and ensembles), there is no statistically significant difference between them in most cases, except at the HMLC task for 200 labeled examples where, statistically, SSL-PCT-FR significantly outperforms SSL-PCT.

As discussed previously, semi-supervised random forests improve over supervised ones in fewer cases as compared to single trees. A statistically significant improvement over CLUS-RF is observed only for the MLC task with 200 labeled examples and the HMLC task with 350 labeled examples. However, in none of the cases, did the proposed semi-supervised methods perform statistically significantly worse than their supervised counterparts.

The statistical test is applied to the predictive performances (AUPRC) of the supervised and semi-supervised single trees (SL-PCT, SSL-PCT, and SSL-PCT-FR) on the datasets considered in this study: 12 for multi-label classification and 12 for hierarchical multi-label classification. In bold we report significant P values (< 0.05). In a comparison of the two algorithms, i.e., Algorithm 1 vs. Algorithm 2, the “-” sign indicates that the sum of ranks where the first algorithm outperformed the second is higher than the sum of ranks where the second algorithm outperformed the first. The “+” sign indicates the opposite.

6.3. Influence of the Amount of Supervision. As previously mentioned, the amount of supervision in the SSL-PCTs is controlled by the w parameter, where $w = 0$ results in unsupervised PCTs, $0 < w < 1$ in semi-supervised PCTs, and $w = 1$ in supervised PCTs. This ability to tune the degree of supervision in SSL-PCTs for the predictive problem at hand is of great practical importance. That is, semi-supervised methods can, in general, degrade the performance of their supervised counterparts [48–51]. In this respect, some studies noted that the success of semi-supervised methods is domain-dependent [52]. How to choose a suitable SSL method for the dataset at hand is an unresolved issue; therefore, even if the primary task of SSL methods is to achieve improved performance in comparison to supervised methods, it is also a high priority to make semi-supervised methods safe, i.e., to make sure that they do not perform worse than their fully supervised counterparts.

In SSL-PCTs, such a safety mechanism is provided by the w parameter. Theoretically, given the optimal value of w , SSL-PCTs and SSL-RF would always perform at least as well as their supervised counterparts both for MLC and HMLC. The reason is that SL-PCTs and CLUS-RF are special cases of SSL-PCT and SSL-RF when $w = 1$. In practice, however, the w parameter is chosen via internal cross-validation on labeled examples in the training set. Thus, it is possible to select w suboptimal for the test set considered.

TABLE 4: P values of the Wilcoxon signed-rank test.

Methods	Number of labeled examples					
	50	100	200	350	500	
<i>Multi-label classification</i>						
SL-PCT vs. SSL-PCT	0.012 (+)	0.008 (+)	0.008 (+)	0.117 (+)	0.071 (+)	
SL-PCT vs. SSL-PCT-FR	0.008 (+)	0.008 (+)	0.023 (+)	0.012 (+)	0.008 (+)	
SSL-PCT vs. SSL-PCT-FR	0.969 (+)	0.666 (+)	0.556 (+)	0.078 (+)	0.182 (+)	
CLUS-RF vs. SSL-RF	0.209 (+)	0.126 (+)	0.078 (+)	0.092 (+)	0.182 (+)	
CLUS-RF vs. SSL-RF-FR	0.17 (+)	0.117 (+)	0.013 (+)	0.638 (+)	0.695 (-)	
SSL-RF vs. SSL-RF-FR	0.937 (+)	0.209 (+)	0.754 (-)	0.327 (-)	0.388 (-)	
<i>Hierarchical multi-label classification</i>						
SL-PCT vs. SSL-PCT	0.937 (+)	0.147 (+)	0.034 (+)	0.025 (+)	0.008 (+)	
SL-PCT vs. SSL-PCT-FR	1 (-)	0.158 (+)	0.034 (+)	0.025 (+)	0.01 (+)	
SSL-PCT vs. SSL-PCT-FR	0.388 (-)	0.136 (-)	0.034 (+)	0.695 (-)	0.347 (+)	
CLUS-RF vs. SSL-RF	0.367 (+)	0.136 (+)	0.099 (+)	0.347 (+)	0.136 (+)	
CLUS-RF vs. SSL-RF-FR	1 (-)	0.347 (+)	0.136 (+)	0.034 (+)	0.48 (+)	
SSL-RF vs. SSL-RF-FR	0.367 (-)	0.666 (+)	1 (-)	0.239 (+)	0.969 (-)	

Our empirical evaluation showed that SSL-PCT and SSL-RF rarely degrade the performance of SL-PCT and CLUS-RF (Figures 2 and 3). Across all the experiments we performed, SSL-PCTs outperformed their supervised counterpart (SL-PCT) in 52% of the experiments, performed worse in 9% of the experiments, and performed equally in 39% of the experiments. Moreover, the occasional degradation of the predictive performance was small compared to the improvement of SSL-PCT over SL-PCT. For example, the average relative improvement of SSL-PCTs over SL-PCTs (across all the experiments) was 40%, while the average degradation was 7%.

Figure 4 clearly shows the role of parameter w on the predictive performance for 4 datasets with different types of structured output. The Emotions dataset (Figure 4(a)) requires no supervision because $w = 0$ provides better predictive performance of the SSL-PCT method, whereas the Genbase dataset (Figure 4(b)) requires small amount of supervision (i.e., w close to 0) for SSL-PCT and high amount of supervision (i.e., w close to 1) for SSL-RF. For the HMLC dataset, Danish farms (Figure 4(c)), more supervision (i.e., higher w) provides better predictive performance of SSL-PCT. However, for up to 500 labeled examples, the SSL-PCT method is unable to improve its supervised counterpart; therefore, $w = 1$ is selected to prevent performance degradation. For the other HMLC dataset, ImCLEF07A (Figure 4(d)), on the other hand, the performance drops for high levels of supervision (i.e., $w > 0.5$).

In conclusion, our results show that the optimal value of w depends on the dataset and on the different amounts of labeled data, as exemplified in Figure 4. This confirms our initial intuition that a different amount of supervision is suitable for different datasets. Therefore, it is difficult to provide a general recommendation for the value of w , and it is advisable to optimize this parameter by internal cross-validation for each dataset, as it is done in our study.

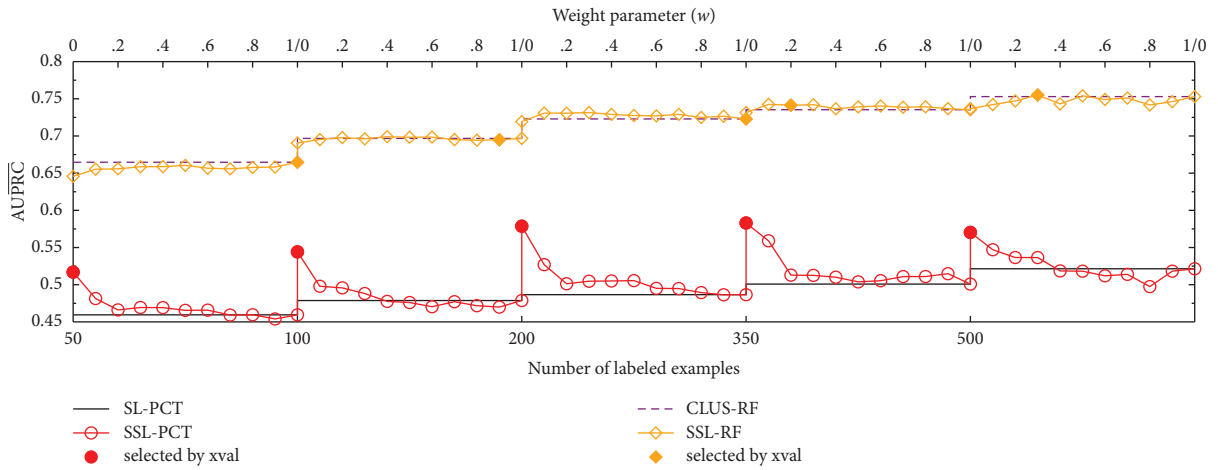
6.4. Interpretability of the Models. Interpretability of the predictive models is often a desirable property of machine learning algorithms. Since the models produced by the

SSL-PCTs are in the form of a decision tree, they are readily interpretable. To the best of our knowledge, in the literature, no other semi-supervised method for MLC and HMLC produces interpretable models.

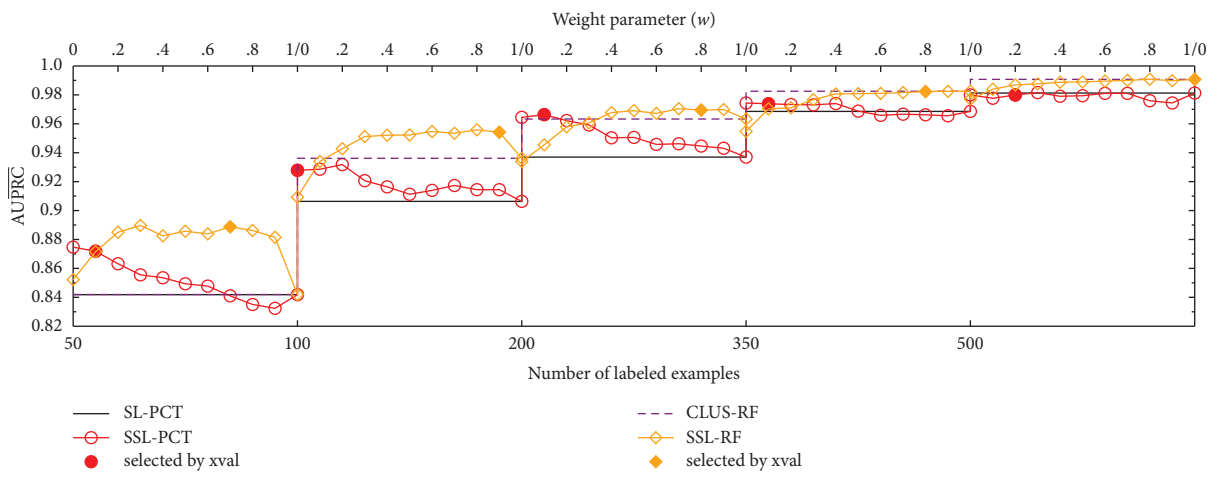
The degree of interpretability of the tree-based models is typically expressed in terms of their size. A large tree can be more difficult to interpret, and vice versa, a small tree can be easier to interpret. The tree size is often a trade-off between accuracy and interpretability. Small trees are easy to interpret but due to their simplicity may fail to capture interactions in the data and therefore provide a satisfactory accuracy. On the other hand, larger trees may mitigate such issues, but at the cost of lower interpretability. Note that increased size does not necessarily mean improved predictive power of tree models, due to possible overfitting. In general, it is not easy to identify (*a priori*) the best size of a tree, in order to balance between overfitting and underfitting.

In Table 5, we compare tree sizes of supervised and semi-supervised PCTs. We observe that, on average, the semi-supervised trees are somewhat larger than the supervised trees. This is intuitive since semi-supervised algorithms use much more data to grow the trees, i.e., both labeled and unlabeled examples. If we focus on individual datasets, we can observe that the size of both the supervised and semi-supervised trees is mainly in the range of a few tens of nodes. This is still a reasonable size for manual inspection. However, there are a few exceptions. Semi-supervised trees are sometimes, with a few hundred nodes, much larger than the corresponding supervised trees. In particular, this can be observed in the following datasets: Mediana (≥ 350 labeled), Danish farms (500 labeled), ImCLEF07A, and ImCLEF07D (≥ 200 labeled). These cases, generally characterized by a large number of classes, can be infeasible for analysis.

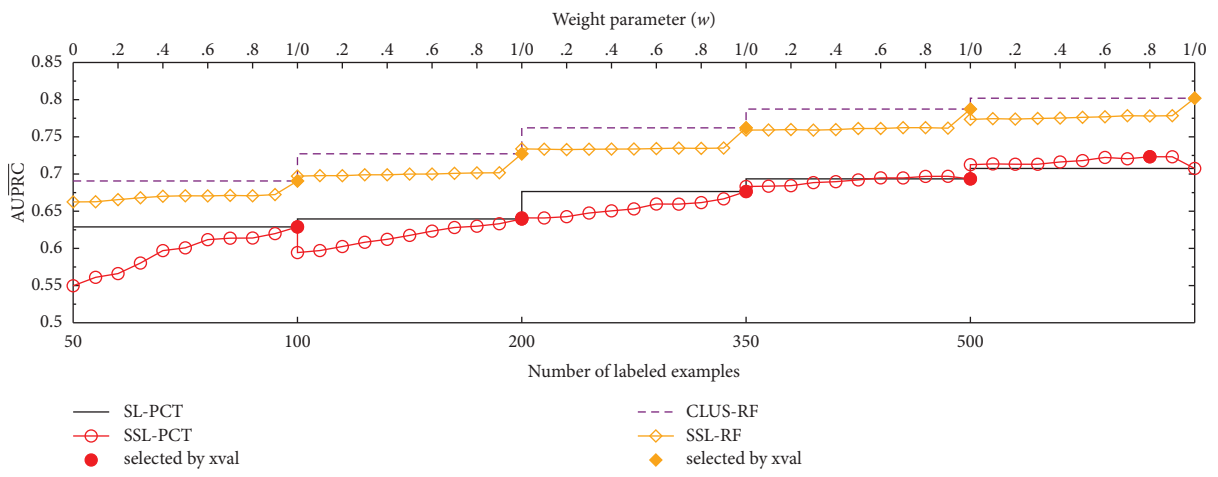
To exemplify the interpretability and to highlight the possible differences between SL-PCTs and SSL-PCTs, we provide an example of supervised and semi-supervised predictive clustering trees obtained for the Emotions dataset with 100 labeled examples (Figure 5) where the task is to



(a)



(b)



(c)

FIGURE 4: Continued.

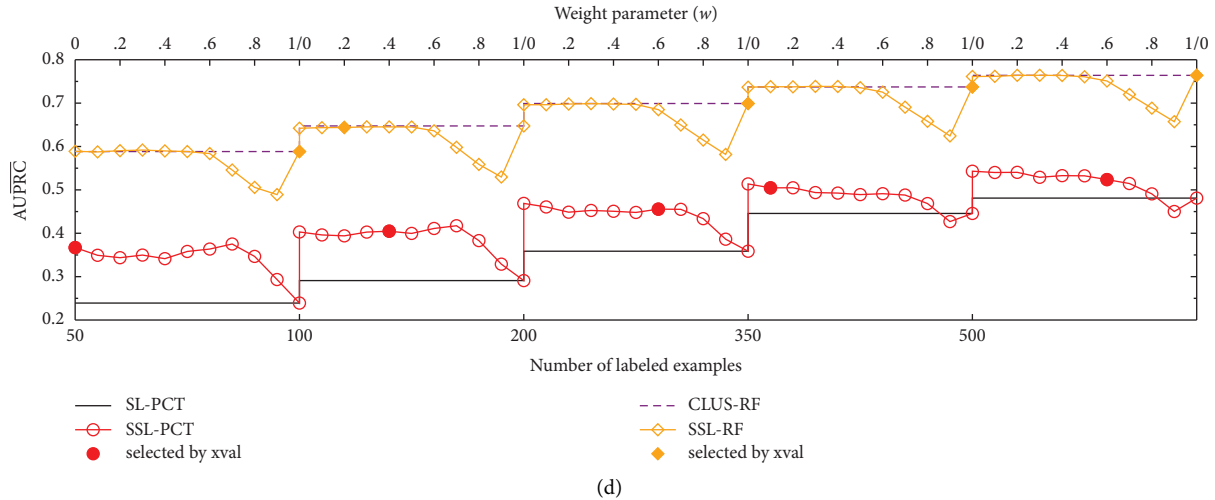


FIGURE 4: Influence of parameter w on SSL-PCT (red line) and SSL-RF (orange line) methods. The results refer to 4 datasets with different types of structured outputs: (a) Emotions (MLC), (b) Genbase (MLC), (c) Danish farms (HMLC), and (d) ImCLEF07A (HMLC). The w values selected by the internal cross-validation algorithm and used in the experiments are marked with colored dots.

TABLE 5: Model sizes expressed as the number of nodes in trees.

Dataset	Number of labeled examples									
	50		100		200		350		500	
	SL-PCT	SSL-PCT	SL-PCT	SSL-PCT	SL-PCT	SSL-PCT	SL-PCT	SSL-PCT	SL-PCT	SSL-PCT
<i>Multi-label classification</i>										
Bibtex	1	21	1	21.4	1	19.4	1	19.4	1	19
Birds	1	15.8	1	20.8	1.2	15.2	2.6	13	4	12.8
Corel5k	1	33.6	1	1	1	1	1	1	1	1
Emotions	3	18.8	5	19	7.4	7.4	11.8	19.2	14.8	14.8
Enron	1	17.6	1	19.2	1	21.4	1	21.2	1.2	20.2
Genbase	2.6	22.8	21.4	23	31.2	26.6	37.2	36.8	43	41.4
Mediana	1.4	32.8	4	4	7.8	7.8	10.2	10.2	12.4	12.4
Medical	1	15.4	1	41.8	1	43.4	3.2	63.6	13.6	63.2
Scene	7.8	25.8	12.4	28.4	19.8	28.4	31.8	29	36.6	37.2
SIGMEA real	3	35.2	3.2	3.2	4.6	4.6	6.6	6.6	8.4	8.4
Slovenian rivers	1	39.2	1	64	1.4	67.6	3	70.8	3.8	55.8
Yeast	1	1	1	1	1	25	1.2	25	2.2	25
Average	2.1	23.3	4.4	20.6	6.5	22.3	9.2	26.3	11.8	25.9
<i>Hierarchical multi-label classification</i>										
Danish farms	1	1	1.4	1.4	3.2	3.2	6	6	8.8	257.6
Slovenian rivers	1	19.8	1	18.2	1	47.4	1.4	52.4	3	50.8
Enron	1	11.6	1	13.8	1.6	15.8	5.6	16.6	6.6	17.2
ImCLEF07A	1	42.8	2	86.6	6.6	133.4	12.8	150.4	19.4	177.4
ImCLEF07D	1	47.4	1.8	215	4.4	295.2	7	159.2	15	172.2
Diatoms	1	49.6	1	56.2	1	57.4	1	72.4	1	89.8
Cellcycle-GO	1	1	1	1	1	1	1	1	1	1
Church-GO	1	1	1	1	1	1	1	1	1	1
Derisi-GO	1	1	1	1	1	1	1	1	1	1
Eisen-GO	1	1	1	1	1	1	1	1	1	27.4
Expr-GO	1	1	1	1	1	1	1	9.2	1	9.2
Pheno-GO	1	1	1	1	1	1	1	1	1	1
Average	1.0	14.9	1.2	33.1	2.0	46.5	3.3	39.3	5.0	67.1

predict an emotion evoked by music on the basis of features such as Mel Frequency Cepstral Coefficients (MFCCs) that describe timbre or Rolloff describing a frequency response below or above a certain limit. We can observe that

unlabeled examples enabled the semi-supervised algorithm to build a larger and, in this case, more accurate tree than the supervised one (note the predictive performance in Figure 2). Next, we can observe that the most important features

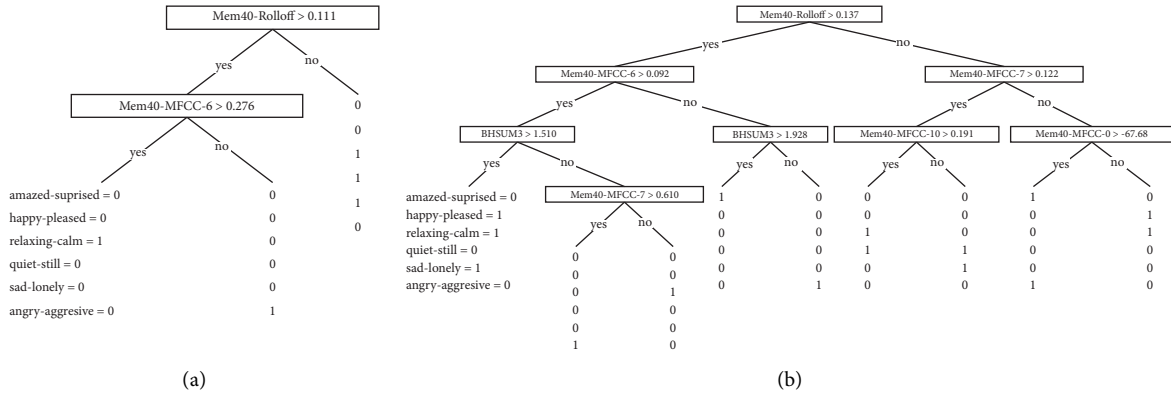


FIGURE 5: Supervised and semi-supervised predictive clustering trees obtained for the Emotions dataset with 100 labeled examples. (a) Supervised PCT. (b) Semi-supervised PCT.

(i.e., the ones at the top of the tree) are the same in both trees; however, the splitting points are different implying that unlabeled examples can help semi-supervised trees to refine the splits.

A closer analysis of the results is shown in Figure 6, where it is possible to evaluate the influence of parameter w on the tree size. The analysis reveals that unsupervised trees ($w = 0$) are much bigger than semi-supervised ($0 < w < 1$) or supervised ($w = 1$) trees. Unsupervised trees do not rely on the output space at all; therefore, it is understandable that, in the presence of a very large amount of unlabeled data, big trees are grown. We recall that the W parameter is optimized for predictive performance, but by increasing the value of w (i.e., increasing the degree of supervision), a trade-off between tree size and model performance can be achieved.

6.5. Training times. In Table 6, we present the training times of supervised and semi-supervised algorithms. For simplicity, we present times for experiments with 500 labeled examples, since conclusions for other amounts of labeled data are similar. We can observe that semi-supervised PCTs and random forests can take considerably more time to train the model than their supervised counterparts, which is expected since they use more data (i.e., additional unlabeled examples) and they also calculate the heuristic score to determine the best splits across all descriptive and target attributes, as opposed to the supervised algorithms that use only the target attributes. The increased learning time is hence the most pronounced on datasets with many attributes, such as Expr-GO and Enron datasets for the HMLC task and Bibtex and Medical datasets for the MLC task. Note that in some cases, the learning times between supervised and semi-supervised algorithms are the same. This is because in such cases $w = 1$ was chosen, i.e., the semi-supervised model is equal to the supervised one. Note that in Table 6, the time used to optimize the w parameter is not included.

The training times are in seconds, obtained for experiments with 500 labeled examples.

6.6. The Influence of Unlabeled Data. SSL-PCTs differ from supervised PCTs in two aspects: (i) they use both the descriptive attributes and target variables for the candidate split evaluation and (ii) they use unlabeled examples in the training process. We have shown that SSL-PCTs have highly competitive predictive performance with respect to supervised PCTs, but we can still question the source of this improvement. Is this improvement due to the combination of (i) and (ii)? Or is (i) sufficient to yield improvements over supervised PCTs? To answer this question, we compare SSL-PCTs with the supervised modification of PCTs which use both the descriptive attributes and target variables for split evaluation in the same way as SSL-PCTs, but does not use unlabeled data (henceforth, this variant will be denoted as SL-PCT^{D+T}). By using this modification, we can evaluate the effect of the unlabeled examples on the predictive performance, since both SSL-PCTs and SL-PCT^{D+T} are trained using the same algorithms—the only difference being in the usage of unlabeled data. In these experiments, we optimize parameter w for SL-PCT^{D+T} via internal 3-fold cross-validation, analogously to SSL-PCTs.

Considering all the datasets and the various percentages of labeled data, the SL-PCT^{D+T} algorithms perform better than the SL-PCT in 36% of the cases, the same in 54% of the cases, and worse in 11% of the cases. We recall that the corresponding figures for the SSL-PCTs algorithm are 52%, 39%, and 9%. Thus, even without the help of unlabeled data, the SSL-PCTs proposed in this work can improve over SL-PCTs, but they have a better chance to do so if they are supplied with unlabeled data. The following result shows that the unlabeled data are indeed the principal component for the success of the SSL-PCTs: the average relative improvement of SL-PCT^{D+T} over SL-PCT is a mere 4%, while for SSL-PCT, this figure is 40% (considering only the cases where SL-PCT^{D+T} and SSL-PCTs improve over SL-PCTs, respectively). This observation, i.e., the importance of unlabeled data, is in line with the findings of Ženko [53], where a rule learning process that considers both the descriptive and target spaces is adopted. The results reported in [53]

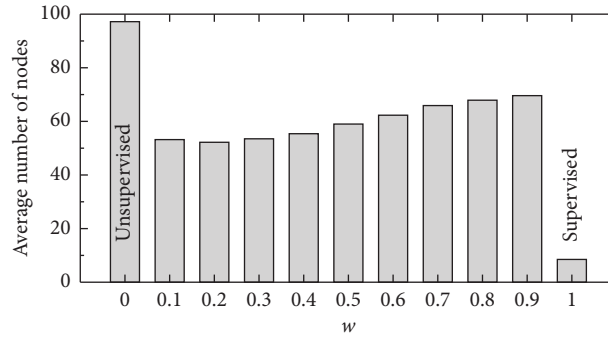


FIGURE 6: Average tree size per value of parameter w across all datasets and amounts of labeled data.

TABLE 6: Training times for SL-PCTS and SSL-PCTS.

Dataset	SL-PCT	SSL-PCT	CLUS-RF	SSL-RF
<i>Hierarchical multi-label classification</i>				
Danish farms	0.3	3.8	1.4	1.4
Slovenian rivers	0.7	1.6	14.7	14.7
Enron	1	291.2	2.1	2.1
ImCLEF07A	0.4	20.1	2.4	2.4
ImCLEF07D	0.4	8.4	1.8	47.1
Diatoms	7.2	100.6	11.8	11.8
Cellcycle-GO	18.5	18.5	109.9	109.9
Church-GO	2.3	2.3	49.4	5
Derisi-GO	16.8	16.8	117.1	2500.4
Eisen-GO	12.1	196.8	157.3	157.3
Expr-GO	120.4	1031.6	267.3	267.3
Pheno-GO	1.2	1.2	4.8	4.8
Average	15.1	141.1	61.7	260.4
<i>Multi-label classification</i>				
Bibtex	8.1	790.2	12.9	686.4
Birds	2.1	16.2	4.3	4.3
Corel5k	16.1	16.1	99.7	1314
Emotions	0.4	0.4	1.5	7.7
Enron	1.7	61.8	2.7	77.8
Genbase	0.1	0.5	0.4	0.4
Mediana	0.2	0.2	1.2	1.2
Medical	1	51.8	1.3	35.1
Scene	1.1	84.6	2.2	2.2
SIGMEA real	0.1	0.1	0.5	1
Slovenian rivers	0.2	0.4	1.4	3.8
Yeast	1	12	3.4	3.4
Average	2.7	86.2	11.0	178.1

show that including the descriptive space in the heuristic was not beneficial for the predictive performance of predictive clustering rules. However, the study was performed in a supervised learning context, i.e., unlabeled examples were not used.

Finally, Figure 7 allows a detailed evaluation of the improvement/degradation of SL-PCT^{D+T} over SL-PCT and of SL-PCT^{D+T} over SSL-PCT. As stated previously, the

SSL-PCT method outperforms SL-PCT more often than SL-PCT^{D+T} (this happens when the points are above the diagonal). Furthermore, SSL-PCT yields much larger improvements over SL-PCT than SL-PCT^{D+T} (for most of the points in the figure, the improvement along the y -axis is much larger than the improvement along the x -axis). However, there is some complementarity between the two methods. That is, SL-PCT^{D+T} sometimes improves over

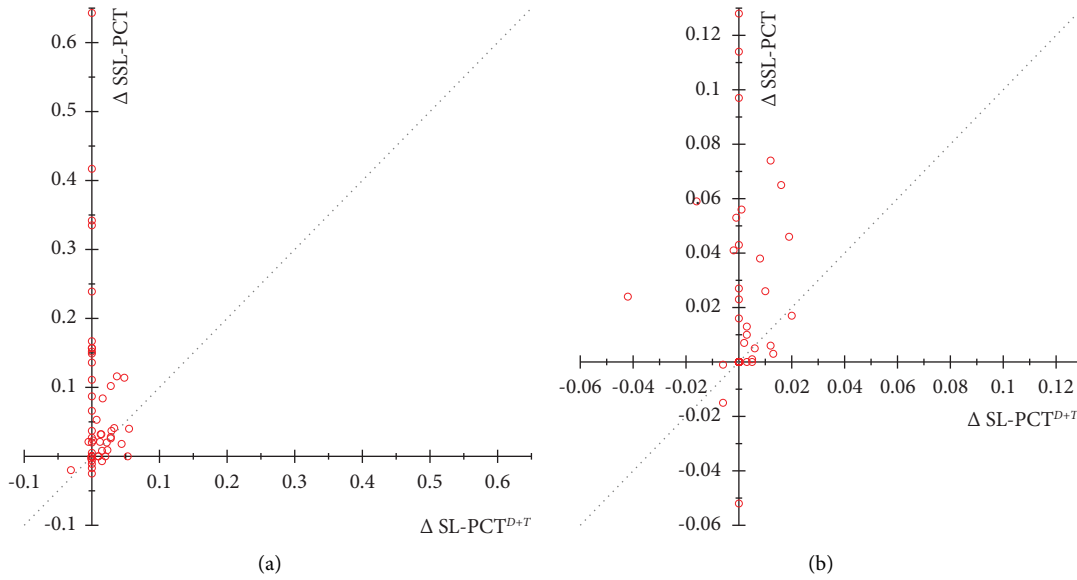


FIGURE 7: The graph depicts the magnitude of improvement in the predictive performance over supervised PCTs enabled by (i) the variance function that considers both the descriptive and target spaces (x -axis) and (ii) unlabeled data and the variance function that considers both the descriptive and target spaces (y -axis). This is measured by the difference in the predictive performance of SL-PCT^{D+T} and SL-PCT ($\Delta \text{SL-PCT}^{D+T}$; x -axis) and of SSL-PCT and SL-PCT ($\Delta \text{SSL-PCT}$; y -axis). The positive values along the x and y axes denote that SL-PCT^{D+T} or SSL-PCT improves over SL-PCT s, respectively. Clearly, the magnitude of improvement over SL-PCT s along the y -axis is much larger than along the x -axis, showing that the unlabeled examples are crucial for the performance of the SSL-PCT s. Each dot represents AUPRC of one experiment (one dataset and one percentage of labeled data; all experiments are considered). (a) Multi-label classification. (b) Hierarchical multi-label classification.

SL-PCT even when this is not the case with SSL-PCT (Figure 7, the values on the positive side of the x -axis, below the dashed line).

7. Conclusions

In this study, we propose an algorithm for multi-label classification and for hierarchical multi-label classification that works in a semi-supervised learning setting. The method is based on predictive clustering trees and uses both the target and the descriptive space for the evaluation of candidate splits. We executed an extensive empirical study using 24 datasets and we summarize the main findings as follows:

- (i) The proposed semi-supervised predictive clustering trees achieve *good predictive performance on both structured output tasks*. On many of the datasets considered, their predictive performance was superior to that of supervised predictive clustering trees.
- (ii) The *control on the amount of supervision* to be used when learning the proposed semi-supervised predictive clustering trees makes them *safe to use*: they do not degrade the performances with respect to their supervised counterparts, i.e., they either outperform them or have the same performance.
- (iii) The degree of superiority of semi-supervised over supervised predictive clustering trees does not translate entirely to the tree ensembles, even though semi-supervised random forests often outperform supervised random forests.
- (iv) Weighting descriptive attributes by their importance may help the predictive performance of semi-supervised predictive clustering in some cases, but the advantages are not great enough to advocate the use of feature weighting by default. Thus, by the principle of Occam's razor, the simpler solution should be preferred, that is, the one without feature weighting.
- (v) The semi-supervised trees produce readily interpretable models and are marginally larger than the supervised trees, though the sizes of the trees are reasonable for manual inspection in most cases. Also, this comes with an increase in the computational cost as evidenced by the theoretical and empirical (runtime) analysis of the computational complexity.

In future work, we intend to extend the proposed semi-supervised (hierarchical) multi-label classification algorithm to the case where examples are not independent and are accommodated in a network data structure. This would

allow us to exploit the semi-supervised learning setting in network data, where the smoothness assumption naturally holds.

Data Availability

The datasets supporting this study are from previously reported studies, which have been cited, and are available at repositories referenced therein. The datasets are also available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the Slovenian Research and Innovation Agency (formerly ARRS; recently renamed to ARIS), via a young researcher grant to the first author, the Research Program Knowledge Technologies (grant P2-0103), and the projects J1-3033, J2-2505, J2-4452, J2-4460, J3-3070, J4-3095, J5-4575, J7-4636, J7-4637, and N2-0236. It was also supported by the European Commission (EC) via the project MAESTRA (grant 612944), as well as the projects ASSAS (grant number 101059682), ELIAS (grant 101120237), INQUIRE (grant 101057499), PARC (grant 101057014), and TAILOR (grant 952215).

References

- [1] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised Learning*, MIT Press, Cambridge, MA, USA, 2006.
- [2] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.
- [3] J. Jeong, S. Lee, J. Kim, and N. Kwak, "Consistency-based semi-supervised learning for object detection," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., pp. 10758–10767, Springer, Vancouver, Canada, 2019.
- [4] D. Yu, B. Varadarajan, L. Deng, and A. Acero, "Active learning and semi-supervised learning for speech recognition: a unified framework using the global entropy reduction maximization criterion," *Computer Speech and Language*, vol. 24, no. 3, pp. 433–444, 2010.
- [5] J. Levatic, M. Ceci, T. Stepisnik, S. Dzeroski, and D. Kocev, "Semi-supervised regression trees with application to QSAR modelling," *Expert Systems with Applications*, vol. 158, Article ID 113569, 2020.
- [6] Y. Liu, X. Zhou, H. Kou et al., "Privacy-preserving point-of-interest recommendation based on simplified graph convolutional network for geological traveling," *ACM Transactions on Intelligent Systems and Technology*, 2023.
- [7] M. Mojdeh and G. V. Cormack, "Semi-supervised spam filtering using aggressive consistency learning," in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 751–752, New York, NY, USA, January 2010.
- [8] D. Kocev, C. Vens, J. Struyf, and S. Dzeroski, "Tree ensembles for predicting structured outputs," *Pattern Recognition*, vol. 46, no. 3, pp. 817–833, 2013.
- [9] Y. Guo and D. Schuurmans, "Semi-supervised multi-label classification-A simultaneous large-margin, subspace learning approach," in *Machine Learning and Knowledge Discovery in Databases-European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II*, P. A. Flach, T. D. Bie, and N. Cristianini, Eds., pp. 355–370, Springer, Berlin, Germany, 2012.
- [10] X. Kong, M. K. Ng, and Z. Zhou, "Transductive multi-label learning via label set propagation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 3, pp. 704–719, 2013.
- [11] W. Shi, V. S. Sheng, X. Li, and B. Gu, "Semi-supervised multi-label learning from crowds via deep sequential generative model," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, Eds., pp. 1141–1149, ACM, Berlin, Germany, 2020.
- [12] A. M. Santos and A. M. P. Canuto, "Applying semi-supervised learning in hierarchical multi-label classification," *Expert Systems with Applications*, vol. 41, no. 14, pp. 6075–6085, 2014.
- [13] H. Blockeel, L. De Raedt, and J. Ramon, "Top-down induction of clustering trees," in *Proceeding of the 15th International Conference on Machine Learning*, pp. 55–63, Morgan Kaufmann, San Francisco, CA, USA, 1998.
- [14] L. Wang, Y. Liu, C. Qin, G. Sun, and Y. Fu, "Dual relation semi-supervised multi-label learning," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pp. 6227–6234, AAAI Press, New York, NY, USA, 2020.
- [15] F. Zhao and Y. Guo, "Semi-supervised multi-label learning with incomplete labels," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina*, Q. Yang and M. J. Wooldridge, Eds., pp. 4062–4068, AAAI Press, New York, NY, USA, 2015.
- [16] Z. Song, Z. Meng, Y. Zhang, and I. King, "Semi-supervised multi-label learning for graph-structured data," in *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, G. Demartini, G. Zuccon, J. S. Culpepper, Z. Huang, and H. Tong, Eds., pp. 1723–1733, ACM, New York, NY, USA, 2021.
- [17] D. Li and S. Dick, "Semi-supervised multi-label classification using an extended graph-based manifold regularization," *Complex and Intelligent Systems*, vol. 8, no. 2, pp. 1561–1577, 2022.
- [18] J. Levatic, M. Ceci, D. Kocev, and S. Dzeroski, "Semi-supervised predictive clustering trees for (hierarchical) multi-label classification," 2022, <https://arxiv.org/abs/2207.09237>.
- [19] J. Levatic, M. Ceci, D. Kocev, and S. Dzeroski, "Self-training for multi-target regression with tree ensembles," *Knowledge-Based Systems*, vol. 123, pp. 41–60, 2017.
- [20] J. Levatic, D. Kocev, M. Ceci, and S. Dzeroski, "Semi-supervised trees for multi-target regression," *Information Sciences*, vol. 450, pp. 109–127, 2018.
- [21] M. Petković, J. Levatic, D. Kocev, M. Breskvar, and S. Dzeroski, "Clusplus: a decision tree-based framework for predicting structured outputs," *SoftwareX*, vol. 24, Article ID 101526, 2023.
- [22] R. J. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, New York, NY, USA, 1st edition, 1993.

- [23] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: bagging, boosting, and variants," *Machine Learning*, vol. 36, no. 1/2, pp. 105–139, 1999.
- [24] A. Clare, *Machine learning and data mining for Yeast functional genomics*, Ph.D. thesis, University of Wales, Aberystwyth, UK, 2003.
- [25] C. Vens, J. Struyf, L. Schietgat, S. Dzeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," *Machine Learning*, vol. 73, no. 2, pp. 185–214, 2008.
- [26] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, CA, USA, 1984.
- [27] P. Cunningham and S. J. Delany, "k-nearest neighbour classifiers," *Multiple Classifier Systems*, vol. 34, pp. 1–17, 2007.
- [28] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [29] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, New York, NY, USA, 2005.
- [30] I. Katakis, G. Tsoumakas, and I. Vlahavas, "Multi-label text classification for automated tag suggestion," in *Proceedings of the ECML/PKDD 2008 Discovery Challenge* Vol. 75, Springer, Berlin, Germany, 2008.
- [31] F. Briggs, Y. Huang, R. Raich et al., "The 9th annual MLSP competition: new methods for acoustic classification of multiple simultaneous bird species in a noisy environment," in *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing*, pp. 1–8, IEEE, Southampton, UK, September 2013.
- [32] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. P. Vlahavas, "Multi-label classification of music into emotions," in *Proceedings of the 9th International Conference on Music Information Retrieval*, vol. 8, pp. 325–330, Drexel University, Philadelphia, PA, USA, January 2008.
- [33] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth, *Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary*, Springer, Berlin, Germany, 2002.
- [34] B. Klimt and Y. Yang, "The Enron corpus: a new dataset for email classification research," *Lecture Notes in Computer Science*, pp. 217–226, Springer, Berlin, Germany, 2004.
- [35] S. Diplaris, G. Tsoumakas, P. A. Mitkas, and I. P. Vlahavas, "Protein classification with multiple algorithms," in *Advances in Informatics, 10th Panhellenic Conference on Informatics, PCI 2005, Volos, Greece, November 11-13, 2005, Proceedings*, P. Bozaris and E. N. Houstis, Eds., pp. 448–456, Springer, Berlin, Germany, 2005.
- [36] M. Skrjanc, M. Grobelnik, and D. Zupanic, "Insights offered by data-mining when analyzing media space data," *Informatica*, vol. 25, no. 3, pp. 357–363, 2001.
- [37] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine Learning*, vol. 85, no. 3, pp. 333–359, 2011.
- [38] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.
- [39] D. Demšar, M. Debeljak, C. Lavigne, and S. Džeroski, "Modelling pollen dispersal of genetically modified oilseed rape within the field," in *Proceedings of the Annual Meeting of the Ecological Society of America*, p. 152, New York, NY, USA, August 2005.
- [40] S. Dzeroski, D. Demšar, and J. Grbovic, "Predicting chemical parameters of river water quality from bioindicator data," *Applied Intelligence*, vol. 13, no. 1, pp. 7–17, 2000.
- [41] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds., pp. 681–687, MIT Press, Berlin, Germany, 2001.
- [42] D. Demšar, S. Džeroski, T. Larsen et al., "Using multi-objective classification to model communities of soil microarthropods," *Ecological Modelling*, vol. 191, no. 1, pp. 131–143, 2006.
- [43] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Džeroski, "Hierarchical annotation of medical images," *Pattern Recognition*, vol. 44, no. 10-11, pp. 2436–2449, 2011.
- [44] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Džeroski, "Hierarchical classification of diatom images using ensembles of predictive clustering trees," *Ecological Informatics*, vol. 7, no. 1, pp. 19–29, 2012.
- [45] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.
- [46] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometric Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.
- [47] M. Petković, S. Džeroski, and D. Kocev, "Feature ranking for semi-supervised learning," *Machine Learning*, vol. 112, no. 11, pp. 4379–4408, 2022.
- [48] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Machine Learning*, vol. 39, no. 2/3, pp. 103–134, 2000.
- [49] F. Cozman, I. Cohen, and M. Cirelo, "Unlabeled data can degrade classification performance of generative classifiers," in *Proceedings of the 15th International Florida Artificial Intelligence Research Society Conference*, pp. 327–331, Palo Alto, CA, USA, May 2002.
- [50] Z. Zhou and M. Li, "Semi-supervised regression with cotraining-style algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 11, pp. 1479–1493, 2007.
- [51] Y. Guo, X. Niu, and H. Zhang, "An extensive empirical study on semi-supervised learning," in *ICDM 2010, the 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, G. I. Webb, B. Liu, C. Zhang, D. Gunopulos, and X. Wu, Eds., pp. 186–195, IEEE Computer Society, New York, NY, USA, 2010.
- [52] N. Chawla and G. Karakoulas, "Learning from labeled and unlabeled data: an empirical study across techniques and domains," *Journal of Artificial Intelligence Research*, vol. 23, no. 1, pp. 331–366, 2005.
- [53] B. Ženko, *Learning predictive clustering rules*, Ph.D. thesis, University of Ljubljana, Ljubljana, Slovenia, 2007.