Data-Driven Structuring of the Output Space Improves the Performance of Multi-Target Regressors

Article *in* IEEE Access · October 2019

DOI: 10.1109/ACCESS.2019.2945084

CITATIONS
3

READS
101

3 authors:

Stevanče Nikoloski
Result d.o.o
9 PUBLICATIONS **23** CITATIONS
SEE PROFILE

Dragi Kocev
Jožef Stefan Institute
135 PUBLICATIONS **2,624** CITATIONS
SEE PROFILE

Sašo Džeroski
Jožef Stefan Institute
566 PUBLICATIONS **15,877** CITATIONS
SEE PROFILE

10.1109/ACCESS.2019.2945084, IEEE Access

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier XX.XXXX/ACCESS.2019.DOI

Data-driven Structuring of the Output Space Improves the Performance of Multi-target Regressors

STEVANCHE NIKOLOSKI[1,3], DRAGI KOCEV[1,2], SAŠO DŽEROSKI[1,2]
[1] Jožef Stefan Institute, Department of Knowledge Technologies, Jamova ulica 39, 1000, Ljubljana, Slovenia
[2] Jožef Stefan International Postgraduate School, Jamova ulica 39, 1000, Ljubljana, Slovenia
[3] Teagasc, Environment Soils and Land-use Department, Johnstown Castle, Co. Wexford, Ireland

Corresponding author: Stevanche Nikoloski (e-mail: stevanche.nikoloski@ijs.si).

This work was supported by the European Commission through the project MAESTRA - Learning from Massive, Incompletely annotated, and Structured Data (Grant number ICT-2013-612944), the project LANDMARK - Land management, assessment, research, knowledge base (H2020 Grant number 635201) and Teagasc Walsh Fellowship Programme

ABSTRACT The task of multi-target regression (MTR) is concerned with learning predictive models capable of predicting multiple target variables simultaneously. MTR has attracted an increasing attention within research community in recent years, yielding a variety of methods. The methods can be divided into two main groups: problem transformation and problem adaptation. The former transform a MTR problem into simpler (typically single target) problems and apply known approaches, while the latter adapt the learning methods to directly handle the multiple target variables and learn better models which simultaneously predict all of the targets. Studies have identified the latter group of methods as having competitive advantage over the former, probably due to the fact that it exploits the interrelations of the multiple targets. In the related task of multi-label classification, it has been recently shown that organizing the multiple labels into a hierarchical structure can improve predictive performance.

In this paper, we investigate whether organizing the targets into a hierarchical structure can improve the performance for MTR problems. More precisely, we propose to structure the multiple target variables into a hierarchy of variables, thus translating the task of MTR into a task of hierarchical multi-target regression (HMTR). We use four data-driven methods for devising the hierarchical structure that cluster the real values of the targets or the feature importance scores with respect to the targets. The evaluation of the proposed methodology on 16 benchmark MTR datasets reveals that structuring the multiple target variables into a hierarchy improves the predictive performance of the corresponding MTR models. The results also show that data-driven methods produce hierarchies that can improve the predictive performance even more than expert constructed hierarchies. Finally, the improvement in predictive performance is more pronounced for the datasets with very large numbers (more than hundred) of targets.

INDEX TERMS clustering, feature ranking, hierarchy, multi-target regression, target space

I. INTRODUCTION

In supervised learning, the main goal is to learn, from a set of examples with known output (target) values, a function predicting the target value of a previously unseen example. The task where the examples refer to one target is called single target prediction and if the examples refer to more than one target is called multi-target prediction. In certain studies, the target components are considered independently and predictive models are built for each component separately. The overall prediction is then generated as a combination of all per-target predictions. In this way, the potential relations between the target components are not taken into account and the gap that is left with this is directly related with the quality of the obtained models.

Considering the t components of the output space, we

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

can distinguish between single ($t = 1$) and multi-target prediction ($t > 1$). If the target space consists of continuous/numeric variables then the task at hand is multi-target regression (MTR). Likewise, if the target space consists of discrete/nominal variables then the task is called multi-target classification. The multi-label classification can be treated as a special case of multi-target classification [1]. Namely, multi-label classification (MLC) is the task of learning from data examples where each example can be associated with multiple labels, which belong to a predefined set of labels. The point of interest in our study is the multi-target regression task.

In many real life problems, for instance, in ecology (predicting the abundance of different species occupying the same habitat [2], estimating different vegetation quality indices for the same site [3] and predicting the composition of a community of organisms [4]), the target space is structured, meaning that there are some internal relations and dependencies (e.g., hierarchical structure) among the targets. Finding those potential dependencies/relations is one of the most challenging problems in machine learning [5].

The methods for multi-target prediction can be categorized into two groups: (1) local methods (problem transformation methods), that create an individual model per target, and then combine the separate models in order to obtain an overall prediction and (2) global methods (known as big-bang methods or algorithm adaptation methods), that predict all targets at once [6], [7]. The main advantage of the global over the local methods is that the latter exploit the potential dependencies among the targets during the learning phase to obtain predictive models with better predictive performance.

A drawback of global models is that they ignore the local modularity in the connections among the target components such as parent-child, siblings relationships etc. In order to address this challenge, we focus on identifying some potential target relations by structuring the output space using a data-driven approach. Here, we approach the problem of structuring the output space by looking into two different spaces coupled with using different clustering approaches (balanced k-means, agglomerative and predictive clustering). First, we cluster the original output space that consists of the target values for each example. We then cluster the space consisting of the feature ranks for each component. At the end, we transform a flat multi-target regression problem into a hierarchical one using the hierarchy obtained by one of the cluster-based approaches. In other words, we translate the MTR task into a hierarchical multi-target regression (HMTR) task. The main research question is to investigate whether a predictive model learned on the transformed problem can achieve better predictive performance compared to a predictive model learned from the flat multi-target regression problem.

The predictive models that we use in the study are predictive clustering trees (PCTs). We selected PCTs since they are global models that can be used for different structured output prediction tasks (including MTR and HMTR) and they are constructed very efficiently. They are able to make a predictions for several types of structured outputs such as tuples of numerical/discrete values, time series, and hierarchies of variables. More details can be found in [8]–[13]. PCTs can be considered as a generalization of standard decision trees towards predicting structured outputs. But the change in just a few of the training examples can sometimes drastically change the structure of the tree. To improve their predictive performance, the predictive models can be combined into an ensemble [14]. An ensemble is a set of single (base) predictive models whose predictions are combined. For basic classification and regression tasks, it is widely accepted that ensemble learners improve the predictive performance of single tree learners [6].

More specifically, we use single PCTs and ensemble of PCTs for both MTR and HMTR setting. We perform an extensive empirical evaluation of the proposed methods on 16 MTR benchmark datasets. Most of the datasets (11 out of 16 datasets) are also used in [15]: The remaining datasets from [15] have small number of targets (2 or 3) and there is not much point in learning hierarchies in such small output spaces. For hierarchy creation, we use agglomerative clustering methods with single and complete linkage, balanced k-means, and predictive clustering trees (PCTs). In order to make our study more comprehensive, we perform experiments on two large datasets (with 111 and 492 targets) thus exploring the effect of including structures in large output spaces.

The results from the evaluation reveal that better predictive performance can be achieved by using data-driven approaches to construct the hierarchies rather than considering either, the flat multi-target regression task, or the pre-defined hierarchy created by a domain expert. Moreover, for large datasets, the results are in line with teh results for MLC [16], [17]: divisive hierarchy creation algorithms (balanced k-means and PCTs for clustering) are the best methods for clustering large output spaces. All in all, constructing a hierarchy of the target variables improves the predictive performance of the predictive models.

The reminder of this paper is organized as follows. In Section 2, we present the related work on the topic of multi-target regression and hierarchical multi-target regression. In Section 3, we show the data-driven approaches for structuring the target space and the space created from feature ranks of the targets for MTR. Furthermore, in this section we present the learning methodology used to create predictive models. Computational

complexity is also discussed at this point. In Section 4, we present the experimental design, where we describe out data, point out the addressed experimental questions and instantiate the parameters used in our study, present the evaluation measures and the used statistical validation as well as the explanation on how the expert hierarchies are created for each data set. Experimental results are given and discussed in Section 5, while Section 6 concludes this paper.

II. BACKGROUND AND RELATED WORK

A. FORMAL DEFINITION OF MULTI-TARGET REGRESSION (MTR)

In our study, we focus on the task of multi-target regression that can be formally defined as follows [6], [18].
Given is:

- A description (input) space X covered by tuples of D independent descriptive instances (examples) i.e., $X = \{X_1, X_2, \ldots, X_D\}$;
- A target (output) space Y covered by tuples of T continuous target variables i.e., $Y = \{Y_1, Y_2, \ldots, Y_T\}$;
- Set of examples E consisting of a pairs of elements, one from input and another from output space, accordingly i.e., $E = \{(x_i, y_i)|x_i \in X, y_i \in Y, 1 \leq i \leq N\}$, where N is a number of examples;
- A quality criterion q, which selects and chooses the models with the lowest predictive error.

Find:

- A function $f : X \rightarrow Y$ which maximizes quality criterion q.

In our study, f is represented with predictive clustering trees (PCTs) or ensembles thereof.

B. METHODS FOR MULTI-TARGET REGRESSION

As mentioned above, we distinguish two groups of MTR methods: local (*problem transformation*) and global (*algorithm adaptation*) methods [6], [7], [19]. Local methods construct t separate models for the t target variables, which are combined to give the overall prediction for all the targets. From the other side, global methods build only one model for predicting all of the t target variables simultaneously. We next present the state-of-the-art MTR algorithms from both groups of methods.

1) Local (problem transformation) methods

Since the local methods transform the problem into t separate single-target models, any known single target regression algorithm can be used to learn the single-target models. Prominent methods addressing the MTR task include: *ridge regression* [20], *support vector regression machines*, *regression trees* [14] and *stochastic gradient boosting* [21]. [20] proposed a separate ridge regression algorithm that deals with MTR problems.

Regressor chain (RC) [22] is another problem transformation method motivated by the multi-label chain classifier [23]. During the training process, RC randomly selects a chain (permutation) of the target space, then builds a separate regression model for each target in consistence with the selected chain. Since RC uses the actual values of all previous targets in a chain, [22], also proposed regressor chain *corrected* (RCC) that uses cross-validation estimates instead of actual values. However, RC and RCC are sensitive to the selected chain ordering. In order to avoid this problem, [15], proposed an approach called ensemble of regressor chains (ERC) and ensemble of regression chains corrected (ERCC), where they randomly select as many models as the number of distinct label chain if the number of labels is less than 10. Otherwise, they randomly selected 10 chains and construct an ensemble of chains.

Multi-target regressor stacking (MTRS) [22] is another problem transformation method inspired by [24] where multi-label classification is performed by using stacked generalization. MTRS training is performed in two stages. First, t different single-target models are learned and then, instead of concatenating the t obtained predictions, MTRS includes additional training stage, where a second collection of t separate single target meta-models are learned. At the end, the predictions are calculated from both stages. The predictions from the second stage use and adjust the predictions from the first stage.

Zhang et al.(2012) [25], presented a new problem transformation method based on multi-output support vector regression approach. Basically, they extend the actual feature space and represent the multi-output problem as equivalent single-output problems, that are solved using the single-output least squares SVRs (LS-SVR) algorithm. The multi-output model takes into account the target correlations by using the vector virtualization method.

Recently, Wang et al. [26] propose a multi-target regression method (MTR-TSF) that embeds the intra-target relationships. First, by using hierarchical clustering on the output space, they reveal the correlation among the targets and create an additional feature vector X_{index} consisting of the indices of the nodes where specific instances belongs to. Next, they use a boosting regression algorithm to learn a similarity matrix for each target. Finally, by querying and clustering of the similarity matrix, a target specific feature vector X_{tsf} is created for all instances and is added to the original feature vector X. At the end, a prediction model per target is learned by considering the 'merged' feature space $X' = X \bigcup X_{index} \bigcup X_{tsf}$.

2) Global (algorithm adaptation) methods

Algorithm adaptation learns a single model for all target variables and thus take into account the dependencies among the targets. There are many advantages over the

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

local methods such as interpretability, better predictive performances, especially, if the targets are related [6]. Below, we briefly discuss various algorithm adaptation methods proposed in the literature.

First attempt to deal with prediction of multiple target variables are the statistical methods such as *reduced-rank regression* [27]. Furthermore, [28] proposed the general version of a multivariate regression problem of the James-Stein estimator, called as *filtered canonical y-variate regression*. Next, lasso regression [29] is a popular regression method for estimation in linear models. It produces interpretable models while at the same time it is stable. Next, gaussian process for MTR are based on the algorithm proposed by [30]. The most prominent statistical approach that deals with multiple targets is the *curds and whey (C & W)* method [31].

Predictive clustering trees (PCTs) are tree-based methods built within the predictive clustering framework [8]. This framework learns decision trees called predictive clustering trees (PCTs) where the top node contains all of the training examples and then it recursively splits into lower partitions (clusters) of the whole train set. PCTs can be used for classical machine learning tasks (clustering, classification and regression), but also, can be applied to multi-target prediction. PCTs can deal with structured outputs prediction, such as vectors, time series, sequences or hierarchies [9]–[13].

In addition, [32] presented an algorithm called multi-target step-wise model tree induction (MTSMOTI) for generation a multi-target model tree on a step-wise manner. The tree model is generated similarly as in PCTs, with TDIDT algorithm. The difference is that each leaf in a tree model is associated with a set of linear models which generate the final target predictions. Conditional Inference Trees (CTrees) are non-parametric regression trees embedding tree-structured regression models into conditional inference procedures and estimate a regression relationship in a multi-target scenario [33].

A different type of MTR algorithm is the rule based algorithm called *FItted Rule Ensemble (FIRE)* method, proposed by [34]. This is a method for learning rule ensembles based on representing an ensemble of regression trees as a large collection of rules. FIRE uses an optimization procedure (minimization) to select the best (much smaller) set of rules and determine their respective weights.

Furthermore, Breskvar et al. [35] present an ensemble method with random output selection (ROS). Instead of using all target attributes, they randomly select subsets of target attributes when learning the base predictive models of the ensemble. This additional randomization improves the performance both in terms of time complexity and predictive accuracy.

The most famous non-parametric distance-based method for regression task is the k-nearest neighbour method. It takes the average of the values of the k nearest examples as a prediction. K-nearest neighbour is a flexible algorithm, since it can use any distance function and any number k (nearest neighbours) [36].

Multiple-input multiple-output (MIMO) support vector regression method is a generalization of support vector machines (SVMs) for addressing the MTR task. The generalization is achieved by minimization of a Lagrangian equation which has multi-dimensional parameters that have to be optimized [37], [38].

Partial Least Squares Regression (PLS-PLSR) and Principal Component Regression (PLS-PCR) methods are another methods for multi-target regression which are implemented in the R software package *pls* [39]. These methods are commonly used in many natural sciences and are based on calculation of the scores obtained by decomposition of the product matrix of orthogonal scores and loadings. Then regression coefficients are calculated using the scores.

Multivariate Adaptive Regression Splines (MARS) is a non-parametric regression method implemented in *EARTH* package in R. MARS, as a generalization of step-wise linear regression [40] constructs the dependencies between input and output variables by using a data-driven set of base functions and coefficients.

Another well-known and widely used method for MTR are the artificial neural networks (NN). They are designed based on human brain to recognize patterns in data. They can automatically model the nonlinearity and can deal with multi-input multi-output problems. The most often used algorithm for training artificial neural networks is *backpropagation* algorithm [41]. Backpropagation algorithm is recursive and iterative method which efficiently optimize the network weights by following the gradient descent method that exploits the chain rule. *Deep neural networks* (DNN) are artificial neural networks containing multiple hidden layers. It update the network weights by establishing the correlation between input (past events) and output (future events). There are several variants of DNNs designed based on the specific domains that are used for. Convolutional deep neural networks (CNNs) are used in the domain of computer vision. Recurrent neural networks (RNNs) are used in various cases of language modelling, such as handwriting and speech recognition [42], [43]. Zhen et al. [44] present a deep learning approach for considering the intra-target dependencies. Namely, they propose a multi-layer multi-target regression (MMR) method where intra-target dependencies are explicitly encoded by using matrix elastic nets (MEN) to create the structure of the target space (structure matrix S), which enables learning of the target correlations by minimization of the $rank(S)$. Then, the kernel trick is used in order to solve the problem of non-linearity in the

representation of the target dependencies.

C. FORMAL DEFINITION OF HIERARCHICAL MULTI-TARGET REGRESSION (HMTR)

We follow similar guidelines as for defining the task of MTR to formally define the task of hierarchical multi-target regression [13]:
Given is:

- A description (input) space X covered by tuples of D independent descriptive instances (examples) i.e., $X = \{X_1, X_2, \ldots, X_D\}$;
- A target (output) space $\mathcal{Z}$ covered by tuples of T continuous target variables i.e., $\mathcal{Z} = \{\mathcal{Z}_1, \mathcal{Z}_2, \ldots, \mathcal{Z}_T\}$. We define a hierarchy $\mathcal{H} = (\mathcal{Z}, \leq_p)$ for the variables from the output space $\mathcal{Z}$. The relation " $\leq_p$ " represents a parent-child relationship between tree nodes $(\forall(\mathcal{Z}_1, \mathcal{Z}_2) \in \mathcal{H} : \mathcal{Z}_1 \leq_p \mathcal{Z}_2$ if and only if $\mathcal{Z}_2$ is a parent (meta-label) of $\mathcal{Z}_1)$ and is called *hierarchical constraint*. The meta-labels are result of an aggregation function (for example, sum or average) on their respective children i.e $\mathcal{Z}_k = agg\{\mathcal{Z}_i | \mathcal{Z}_i \leq_p \mathcal{Z}_k\}$;
- Set of examples E consisting of pairs of elements, one from input and another from output space, accordingly i.e., $E = \{(x_i, y_i) | x_i \in X, z_i \in \mathcal{Z}, 1 \leq i \leq N\}$, where N is a number of examples and where the values of the target variables satisfy the hierarchical constraint " $\leq_p$ " i.e $\forall j : \exists i (\mathcal{Z}_i \leq_p \mathcal{Z}_j \Longrightarrow z_j = agg\{z_i | \mathcal{Z}_i \leq_p \mathcal{Z}_j\})$;
- A quality criterion q, which selects and chooses the models with the lowest predictive error and the highest accuracy.

Find:

- a function $f : X \rightarrow Y$ which maximizes the quality criterion q and all predictions $\hat{z} = f(x)$ are satisfying the hierarchical constraint.

The difference to the task of MTR is in the definition of the output space: for HMTR we have a set of numeric variables organized in a hierarchy instead of a flat tuple of numeric variables. The definition of the parent-child relationships (hierarchy constraint) states that the variable belonging to a given hierarchy node automatically contributes to all its parent nodes.

D. METHODS FOR HIERARCHICAL MULTI-TARGET REGRESSION

In this part, we present the existing (state-of-the-art) methods, related to the task of hierarchical multi-target regression. To begin with, *multilevel analysis* refers broadly to the methodology of research and data structures that deal with nested data, i.e., including more than one type of unit. This is directly related with involving several levels of aggregation. Consider an example from educational research, where students from different schools are considered, and their performance (e.g., grades) is being predicted.

Then, a separate regression model can be fitted within each school, and the model parameters from these schools can be modeled as depending on each school properties (such as the socioeconomic status of the school's neighbourhood, whether the school is public or private, and so on). The student-level regression and the school-level regression here are the two levels of a multilevel model. The lowest level is the student-level and each student belonging to this level can be linked with appropriate class, and then each class to appropriate school and so on. With this, a kind of dependency levels (i.e., a hierarchy) is created. Moreover, in the higher levels in the multilevel model, regression parameters (hyper-parameters) can be fitted for the regression model. That is the reason why in most of the research, the term "multilevel analysis" is mostly used interchangeably with "hierarchical linear modeling", although strictly speaking they are distinct.

Another application of the hierarchical linear modeling approach can be found in [45], where a two-level hierarchical linear model with multiple outputs was employed to analyze an information obtained from two different groups of informants (child and parents participants) in order to assess the demographic risk factors on children's exposure to violence (ETV) and how these effects vary by informants.

The main advantage of multilevel modeling is spreading of a residual components through each level of a hierarchy, thus the overall variance is partitioned and moreover, the predictors are included at each level. Hence, with application of multi-target regression at each level, the model can deal with between-level relations in the hierarchy. Latter makes multilevel modelling superior than regression modeling with respect to the model performance [46]. An extensive review for multilevel modeling is given by [47] and [48].

Next, online analytical processing (OLAP) is a method which allows to extract and analyze data from multiple sources at the same time. The data is multidimensional, hence the extracted information can be compared in different ways. For example, a book store might compare their book sales in September with sales in August, then compare those results with the sales from another location, which might be stored in a different database. The OLAP data is stored in multidimensional databases and all attributes are considered as a separate dimension. Considering the multi-dimensionality, the OLAP data is structured in a hierarchical form by using some of the OLAP tools: consolidation (roll-up), drill-down, and slicing and dicing [49]. This structuring and hierarchical representation enables a complex calculations and manipulation of the data (trend analysis, data modeling) [50]. The natural relationships in the data by using OLAP

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

method are also researched by [51] by using a partially ordered set of levels (dimension schema).

Predictive clustering trees (PCTs) for HMTR task is proposed recently by [13]. The original PCTs for MTR are extended to HMTR task with defining prototype function and variance function. All operations for aggregation can be used as a prototype functions, but keeping in mind that with some of them (for example, *minimum* or *maximum*) after averaging, the hierarchical constraint (parent-child relation within the hierarchy) can be violated. For the variance function, the weighted Euclidean distance is used where the weights are defined such that they decrease exponentially with the depth of the node in the hierarchy.

E. METHODS FOR STRUCTURING THE OUTPUT SPACE

The main goal in this article is structuring the output space in MTR. To the best of our knowledge, structuring of the target space for MTR has not been explored yet. Hence, we overview the methods for structuring the output space for the related multi-label classification (MLC) task where learning hierarchies in the output space has been studied to a wider extent [16], [17], [52]–[55].

Joly et al. (2014) [52] propose a method for dimensionaltiy reduction of the output space by random projections of it, mainly focused on MLC task. The projections are made in such a way that preserve distances in projected space. The reduction of the variance function is made on the projected space, while the predictions are made directly in the original output space using a decoding procedure. Similarly, Joly et al. (2017) [56], proposes a gradient boosting method for MTR which automatically adapt the target correlations by random projection of the output space.

Madjarov et al. (2016) [16] present a comprehensive study of different data-derived methods for structuring the label space in the form of hierarchies for MLC. Namely, they use the label co-occurrence matrix to obtain a hierarchy of labels by using several clustering algorithms such as: agglomerative clustering with single and complete linkage, balanced k-means and PCTs. Their results say that divisive clustering methods (balanced k-means and PCTs) perform the best.

Tsoumakas et al. (2007) [55] propose a transformation-based ensemble method for random k-labelsets (RAkEL) for MLC by using existing algorithms for MLC. The RAkEL algorithm creates an ensemble by random sampling a small subset with k labels for each base model. The sampled subsets are structured as a label powerset and multi-class classifier is then used.

Next, Szymanski et al. (2016) [54] present a study which addressed to the question, whether data-driven methods on a graph consisting of label co-occurrences is significantly better than random generated graph of labels for MLC. This method is actually data-driven version of RAkEL method. Their results show that in general data-driven approach is superior to random created graphs of labels.

Nikoloski et al. (2017) [17] propose an algorithm for structuring the output space using feature ranking in MLC. They create a hierarchy from a space constructed by feature rankings for each of the classes. Furthermore, they present a comparative analysis with the approach from [16], where hierarchy is created by clustering the space consisting of label co-occurrences. In both cases, it is shown that some improvements in predictive performance can be achieved if data-driven approach for output space structuring is used, compared to using a flat multi-label classification task, despite the higher complexity added by additional procedure for calculating the feature importance and the clustering procedures.

III. STRUCTURING THE OUTPUT SPACE FOR MTR

The idea for structuring the output space in MLC proposed by [17] and [16] motivates the exploitation of methods for structuring the output space in MTR. In this study, we propose to transform a flat MTR task into a task of hierarchical multi-target regression (HMTR) [13]. Namely, we use the hierarchies created with data-driven clustering approaches to investigate whether the predictive models obtained with the HMTR task yield better predictive performance than predictive models obtained with the flat MTR task.

A. STRUCTURING THE TARGET SPACE

In our paper, we propose a framework that transforms the original multi-target regression (MTR) task into a hierarchical multi-target regression (HMTR) task, by clustering the output space. The flowchart of the framework is given in Figure 1.

The method for structuring the target space is outlined in the procedure *StructuringTargetSpace* from Algorithm 1. First, we take the original training dataset F^{train} and extract the target space W^{train} from the complete dataset. To obtain a hierarchy, we cluster the space W^{train} by using the procedure $Clustering$ (it can use any arbitrary algorithm for clustering). With the function $TransformData$, we transform the original datasets F^{train} and F^{test} to new datasets F_H^{train} and F_H^{test} by including the obtained hierarchy and then, we learn a predictive model and generate the predictions. Next, we calculate the predictions for each node in the hierarchy and extract only the predictions related to the targets, which are in the hierarchy leafs. Finally, using those predictions, we evaluate the predictive performance.

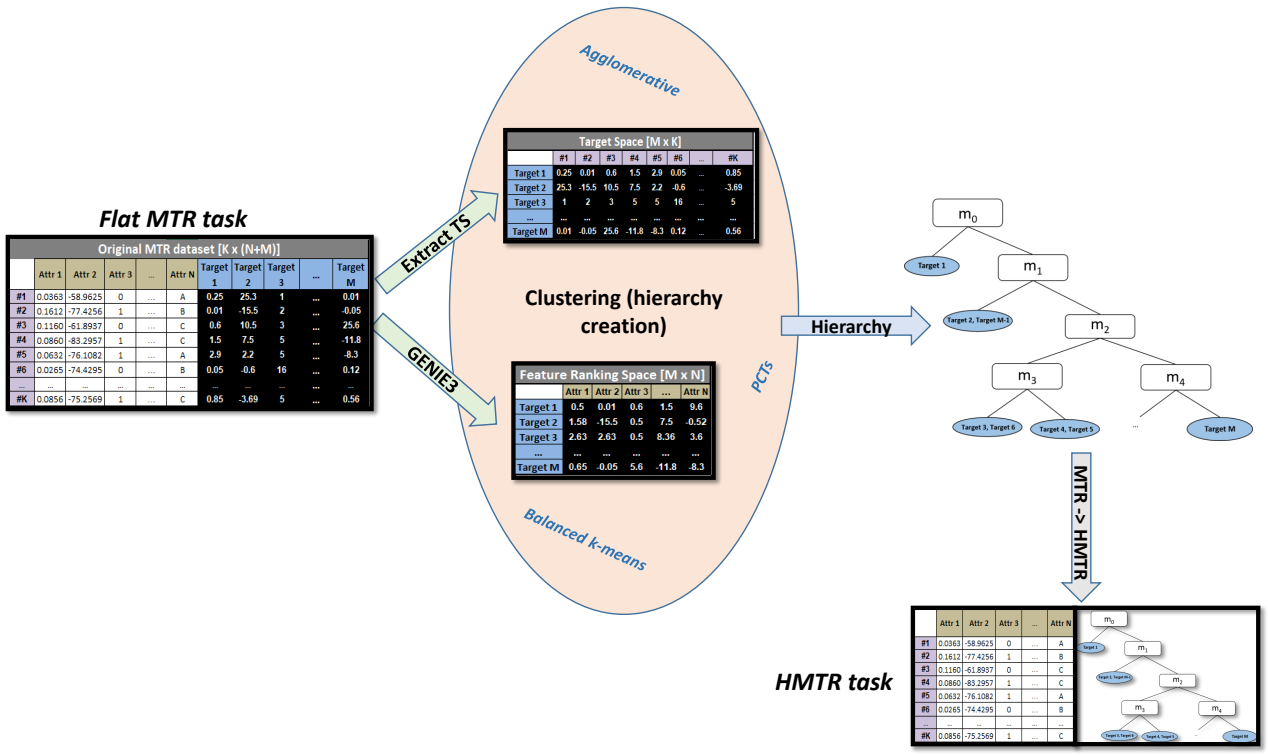

FIGURE 1: An illustration of the proposed framework for structuring the output space. We consider two spaces i.e., representations to cluster the targets: the original target space (TS) i.e., the values of a given target for each example and feature ranking space (FR) i.e., the importance scores for each feature with respect to a given target, for transforming the original MTR task to a HMTR task

B. STRUCTURING THE SPACE OF FEATURE RANKS OF THE TARGETS

The method for structuring the feature importance scores of the targets is outlined in procedure *StructuringFRSpace* from Algorithm 2. First, we take the original training dataset F^{train} and by using an arbitrary feature ranking approach (function *CreateFimp*), we create feature importance scores for each target separately. Then, the F^{ranks} dataset is constructed from the feature importance scores.

Algorithm 1 The algorithm for structuring the target space.

procedure StructuringTargetSpace(F^{train}, F^{test})
Input: F^{train} - training dataset
Input: F^{test} - test dataset
Output: $Performance$
1: $W^{train} = \text{ExtractTargetSpace}(F^{train})$;
2: $hierarchy = Clustering(W^{train})$;
3: $F_H^{train} = TransformData(F^{train}, hierarchy)$;
4: $F_H^{test} = TransformData(F^{test}, hierarchy)$;
5: $HMTR\_Model = \text{HMTRMethod}(F_H^{train})$;
6: $predictions = \text{CalculatePredictions}(HMTR\_Model, F_H^{test})$;
7: $P = \text{ExtractLeafsPredictions}(predictions)$;
8: $Performance = \text{Evaluate}(P)$;
9: **return** *Performance*

Next, we obtain a hierarchy with clustering the F^{ranks} space, using an arbitrary clustering algorithm. Same as the previous Algorithm 1, we transform the original datasets F^{train} and F^{test} to new datasets F_H^{train} and F_H^{test} by including the obtained hierarchy and then, we learn a predictive model, generate the predictions and evaluate the predictive performance.

Algorithm 2 The algorithm for structuring the target space using feature importance scores per target.

procedure StructuringFRSpace(F^{train}, F^{test})
Input: F^{train} - training dataset
Input: F^{test} - test dataset
Output: $Performance$
1: $FimpPath = \text{CreateFimp}(F^{train})$;
2: $F^{ranks} = \text{CreateArffFromFimp}(FimpPath)$;
3: $hierarchy = \text{Clustering}(F^{ranks})$;
4: $F_H^{train} = TransformData(F^{train}, hierarchy)$;
5: $F_H^{test} = TransformData(F^{test}, hierarchy)$;
6: $HMTR\_Model = \text{HMTRMethod}(F_H^{train})$;
7: $predictions = \text{CalculatePredictions}(HMTR\_Model, F_H^{test})$;
8: $P = \text{ExtractLeafsPredictions}(predictions)$;
9: $Performance = \text{Evaluate}(P)$;
10: **return** *Performance*

From the abovementioned procedures for structuring the output space, we can notice that in the procedure

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

StructuringFRSpace (Algorithm 2), there is an additional step, compared to the procedure *StructuringTargetSpace* (Algorithm 1). The additional step is the function *CreateFimp* at line 1 (Algorithm 2), which increases the theoretical complexity of the algorithm *StructuringFRSpace*.

Next, we describe the feature ranking approach for calculating the importance of the descriptive variables. Random forests are constructed by using the algorithm for learning PCTs in CLUS, modified according to the original random forest method proposed by [57]. Their use as feature ranking methods has been well studies in the literature (cf. [58]). First, random forests perform bootstrap sampling on the data and then build a decision tree for each bootstrap sample. Next, at each node of the tree, the best test is taken from a randomly selected feature subset.

Huynh-Thu et al. (2010) [59], proposed the GENIE3 algorithm for feature ranking. It uses reduction of the variance (of the target variables) at each node in the tree. The algorithm is checking which of the input variables reduce the variance more, and then, those which reduce more, are more important. Consequently, the ones which reduce the variance less, are less important. For each selected descriptive variable as a splitting variable, the produced reduction of the variance is being measured. The importance will be 0 if the descriptive variable is never been selected as a splitting variable (for any tree in the ensemble), meaning that it was not deemed important enough. The GENIE3 algorithm has been vastly evaluated for single-target regression tasks, for instance, in the domains of gene reconstruction. The random forest algorithm used for feature ranking is adapted with the idea proposed in the GENIE3 algorithm. For building the ensemble, the random forests of PCTs are used. The outcome is a feature ranking algorithm which is adapted to be used for various types of tasks for structure output prediction [60].

C. HIERARCHY CREATION (CLUSTERING) ALGORITHMS

In this part, we overview the clustering methods used to create the hierarchies of the target space. For achieving a good performance of the HMTR methods, it is necessary to construct target hierarchies that are capturing the relations (dependencies) among the target attributes. The main constraint in hierarchy creation is that the original MTR task should be defined by the leafs of the hierarchy. Specifically, each leaf in the hierarchy represents a set of targets from the original MTR problem. At the end, the number of targets in the hierarchy leafs must be the same as the number of targets from the original MTR problem. Furthermore, the internal nodes of the hierarchy (so called meta-labels) represent the potential relations among the original targets.

For creating the hierarchies, we use four different clustering methods (two divisive and two agglomerative):

- balanced k-means clustering (*divisive*);
- predictive clustering trees (*divisive*);
- agglomerative clustering with complete linkage and
- agglomerative clustering with single linkage.

Agglomerative clustering algorithms are bottom-up algorithms for clustering, where in the first iteration, each example is consider as a separate cluster. In the next iterations, the pairs of clusters are merged based on their linkage (distance metric). There are several possibilities for linkage of the examples. Namely, if the maximal distance of two examples from the clusters C_1 and C_2 is used, then this type of linkage is called *complete* linkage, i.e., $\max\{dist(c_1, c_2) : c_1 \in C_1, c_2 \in C_2\}$. Then, if the minimal distance between two examples for two different clusters is used, then we have an agglomerative clustering with *single* linkage i.e., $\min\{dist(c_1, c_2) : c_1 \in C_1, c_2 \in C_2\}$.

Balanced k-means is divisive top-down approach for clustering. First, root node of the hierarchy represents the one common cluster, consisting of all targets from the target space $\mathcal{T}$. Then, consecutively, this cluster is divided into k disjoint sub-clusters (meta-labels) ($k < |\mathcal{T}|$) using the k-means clustering algorithm. The number of cluster divisions k is an input to this algorithm, hence the algorithm output clusters with approximately equal size [61]. The procedure recursively is repeated on each sub-cluster until the number t of targets in each sub-cluster is smaller than $k-1$. In other words, our target space $\mathcal{T}$ is covered by leafs of the hierarchy obtained by the balanced k-means clustering approach.

We also use predictive clustering trees (PCTs), which can be used as another divisive hierarchical clustering method, to build up the target hierarchies. More specifically, we treat the target space as descriptive space. Descriptive and target variables, all together, are used to provide descriptions for the obtained clusters. To calculate the heuristic score, a variance function is used during the learning process until some stopping criterion is met. This means that there is no need for using predefined number of clusters, as required by traditional clustering methods. The focus of using PCTs for clustering is on using predictive clustering framework in unsupervised manner i.e., on the task of clustering instead of predictive modelling [62], [63].

D. LEARNING METHODOLOGIES

1) Predictive clustering trees (PCTs)

The PCT framework views a decision tree as a hierarchy of clusters, where the top-node corresponds to one cluster containing all the data. While moving downwards the tree, this top-cluster is sub-divided into smaller clusters recursively. The PCT framework is implemented in the CLUS

software package (https://sourceforge.net/projects/clus/) [6], [9].

PCTs are obtained with a standard top-down induction of decision trees (TDIDT) algorithm [64]. As an input, TDIDT takes a set of examples to produce a tree as an output. By using a heuristic function, computed on the training instances, the TDIDT procedure selects a test for the root node. The heuristic aims to select a test which maximizes the variance reduction caused by the partitioning of the examples into subsets according to the test outcome. Recursive procedure of partitioning the examples continues until a stopping criterion is satisfied. Further partitioning of examples yields a tree with a lower quality. In this case, we store the prediction (output value of a prototype function) in the corresponding leaf of the tree.

Blockeel (1998) [8], proposed the predictive clustering framework, while predictive clustering trees (PCTs) for multi- target regression (MTR) were proposed by [9]. In PCTs for MTR, the prototype function calculates the mean vector of all target variables Y for the training examples that belong to the leaf. In the prediction phase, for each new example, the algorithm identifies the leaf it belongs to and returns the value predicted by the prototype function associated to that leaf. The PCTs can be instantiated for a specific given learning task by considering specific variance (for split selection) and prototype function (for calculating the predictions in each leaf). Actually, that is the main difference with standard decision tree learning.

The PCTs are developed to work for the task of multi-target regression (MTR) [65], multi-label classification (MLC) [66], prediction of time series [12], hierarchical multi-label classification (HMLC) [11] and recently, for hierarchical multi-target regression (HMTR) [13]. We will now describe how PCTs from hierarchical multi-target regression are build. In order to extend the PCTs for the HMTR task, we need to define variance and prototype functions.

The variance is calculated by applying a distance function on the values of the variables in analogy of the distances for HMLC and the implementation of that task, i.e., the variance is calculated as the average squared distance between each node Π_i of the examples and the mean node vector $\overline{\Pi}$:

$$Var(E) = \frac{1}{|E|} \cdot \sum_{i=1}^{|E|} dist(\Pi_i, \overline{\Pi})^2 \qquad (1)$$

where $\overline{\Pi} = \frac{1}{|E|} \cdot \sum_{i=1}^{|E|} \Pi_i$.

Any distance d can be used as a distance function in Eq (1). [13] proposes for the task of HMTR to use a weighted Euclidean distance:

$$dist(\Pi_1, \Pi_2) = \sqrt{\sum_{s=1}^{|\Pi|} \theta(c_s) \cdot (\Pi_{1,s} - \Pi_{2,s})^2} \qquad (2)$$

where $\Pi_{i,s}$ is the s'th component of the class vector Π_i of the instance E_i, $|\Pi|$ is the size of the class vector, and the class weights $\theta(c) = \theta_0^{depth(c)}$. The class weights $\theta(c)$ decrease exponentially with the depth of the node in the hierarchy thus making the differences in the lower parts of the hierarchy less influential to the overall score.

The prototype function used is averaging the values of the examples belong to a given leaf.

2) Random forests of PCTs

Random forests of PCTs are implemented in the CLUS system [6] following the same method as for the simpler tasks of classification and regreesion [57]. A random forest represents an ensemble of trees where the diversity among the trees is achieved by bootstrap replicates and for each tree node in the learning phase, a randomly selected subset of descriptive attributes is considered for split selection. Bootstrap replicates are generated by random sampling of instances from the training set, with replacements, until the same number of instances as in the original training set is reached.

The difference between the PCT procedure for tree construction in random forest algorithm and the standard PCT procedure is in the selection of descriptive attributes. In the former, selection of the descriptive attributes is randomized. Namely, at each node in the decision tree, a random subset of attributes is taken from the descriptive space and the best attribute is chosen from this subset. There are different ways of retaining the number of attributes from descriptive space. The number of attributes that are chosen from descriptive space is given by function f of the total number of descriptive attributes D (e.g. $f(D) = 1, f(D) = [\sqrt{D} + 1], f(D) = [\log_2 D + 1]$ etc.). This randomness is chosen in order to avoid the correlation between the bootstrap samples. For example, if there are only few relevant descriptive attributes that are important for prediction of the target variables, these will be selected many times in the bootstrap replicates, hence providing more correlated trees.

Prediction of new instances in random forest algorithm for PCTs are made by combining the prediction of all base predictive models. For both MTR and HMTR, the prediction of each target is defined as an average of the predictions obtained from each predictive tree.

E. COMPUTATIONAL COMPLEXITY

1) Single PCTs for MTR/HMTR

In this part, we analyze the computational complexity of PCTs for HMTR and compare it with the computational

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

complexity of PCTs for MTR. We discuss the order of complexity for both single PCTs and ensembles of PCTs for HMTR. Let us assume that the size of the training set, i.e., the number of examples, is e, the number of descriptive attributes is d out of which c are continuous, the number of target attributes is t and the number of meta-labels is m.

The top-down induction algorithm of PCTs requires sorting of the the c numeric attributes, and it has a cost of $\mathcal{O}(c \cdot e \cdot \log e)$ and $c = \mathcal{O}(d)$. Calculating the best split for multiple variables has the complexity order of $\mathcal{O}(t \cdot d \cdot e)$ and applying the split to the examples has a linear complexity, i.e., $\mathcal{O}(e)$. We assume that the tree is balanced, which means that the depth of the tree is $\log e$. With these calculations, the computational cost of inducing a single MTR tree is:

$$\mathcal{O}(MTRtree) = \mathcal{O}(d \cdot e \log^2 e + t \cdot d \cdot e \cdot \log e + e \cdot \log e) \quad (3)$$

For the HMTR algorithm, we also have the meta-labes (intermediate nodes), which in this case act like targets. This affects the computational cost only when the best split is calculated. More specifically, this costs is given by $\mathcal{O}((t+m) \cdot d \cdot e \log e)$ compare to the $\mathcal{O}(t \cdot d \cdot e \log e)$ for PCTs for MTR. Given this, we can calculate the order of complexity for a HMTR tree, which is very similar to the one for a MTR tree:

$$\mathcal{O}(HMTRtree) = \mathcal{O}(d \cdot e \log^2 e + (t+m) \cdot d \cdot e \cdot \log e + e \cdot \log e) \quad (4)$$

2) Random forest of PCTs

The order of complexity of constructing ensembles depends on the complexity of the base predictive models and their number b. The random forest performs sampling of the instances and sampling of the features. This random sampling reduces the computational complexity of the ensemble and is lower than the intuitive $\mathcal{O}(b \cdot MTRtree)$. Let the number of examples used to train the base predictive model with sampling of the examples be e' and the number of descriptive attributes considered in random forests d' , where $e' < e$ and $d' < d$. The computational complexity of the creation of the bootstrap replicates of the training set for random forests is $\mathcal{O}(e)$ and the complexity of the random sampling of the features at each node for random forests is $\mathcal{O}(d' \cdot \log e')$.

Hence, the computational costs random forest PCT ensembles for MTR is the following:

$$\mathcal{O}(Rforest\_MTR) = \mathcal{O}(b \cdot d' \cdot e' \log^2 e' + b \cdot t \cdot d' \cdot e' \cdot \log e' + b \cdot e' \cdot \log e' + b \cdot e + b \cdot d' \cdot \log e') \quad (5)$$

The computational complexity of the HMTR counterparts of the random forest PCT ensembles for HMTR is the following:

$$\mathcal{O}(Rforest\_HMTR) = \mathcal{O}(b \cdot d' \cdot e' \log^2 e' + b \cdot (t+m) \cdot d' \cdot e' \cdot \log e' + b \cdot e' \cdot \log e' + b \cdot e + b \cdot d' \cdot \log e') \quad (6)$$

In Eq.(6) we can see a linear increasing in complexity with respect to targets with introducing the meta-labels (intermediate nodes). The same translation we already considered for the single PCTs for HMTR (see Eq. (4)).

For all methods (PCTs and ensembles of PCTs for both MTR and HMTR), from their complexity cost, we can see that the dominant elements in the equations are the one containing the second logarithmic power of the number of examples, and the one that is multiplied with the number of targets. For single PCTs, the first element is $\mathcal{O}(d \cdot e \log^2 e)$, and the second is $\mathcal{O}(t \cdot d \cdot e \cdot \log e)$ or $\mathcal{O}((t+m) \cdot d \cdot e \cdot \log e)$ for MTR and HMTR, respectively. If we compare the two terms, we can see that the first term is greater than the second when $\log e > t$ for MTR and $\log e > t + m$ for HMTR. Let us explore the first case where the first term is smaller. This means that when comparing MTR and HMTR, HMTR will have higher computational cost, due to the addition of m. Let us now explore the second case where $\log e$ is higher. In this case, the computational cost is affected only with the first term, hence the linear increase in the second term (i.e., (i.e., the addition of s in $\mathcal{O}((t+m) \cdot d \cdot e \cdot \log e)$) will be insignificant, resulting in comparable performance between MTR and HMTR for all methods on a datasets with a sufficiently large number of examples.

3) Algorithms for structuring the output space

We discuss the computational complexity of the procedures for structuring the output space given in Algorithm 1 and Algorithm 2. In the procedure for structuring the feature ranking space, there is an additional function CreateFimp for calculating the feature importance for each target. Since it is done by random forest method with GENIE3, the order of complexity of this procedure is $\mathcal{O}(CreateFimp) \approx \mathcal{O}(Rforest\_MTR)$.

The most important cost for the clustering procedure is the number of examples e – in the case of datasets with large number of examples, the clustering procedure will take more time to create the hierarchy. When balanced k-means is used as a clustering procedure, the time complexity will be $\mathcal{O}(e \cdot t^3)$. Moreover, if the agglomerative clustering methods are used, the time complexity will be $\mathcal{O}(e \cdot t^3)$ and memory consumption $\mathcal{O}(e \cdot t^2)$, which makes it too slow for even medium data sets. Time complexity of PCTs used as a clustering method is the same as in Eq (3).

Alternatively, the procedure for creating the hierarchy (*Clustering* at line 5 in procedure in Algorithm 2) using

feature rankings has a dimension which depends of the cardinality of the feature space F^{ranks}, denoted as d. The feature space cardinality is typically much smaller than the number of examples (i.e., $|F^{ranks}| \ll |W^{train}|$, i.e., $d \ll e$), meaning that clustering of the rankings will finish faster than clustering of the original target space. Using balanced k-means, it will be $\mathcal{O}(d \cdot t^3)$, where $d \ll e$, then, by using agglomerative it will be $\mathcal{O}(d \cdot t^3)$, and memory consumption $\mathcal{O}(d \cdot t^2)$, where $d \ll e$. Finally, the time complexity of PCTs algorithm used for clustering will be the same as in Eq (3) when we cluster the feature rankings space, considering that $d \ll e$. All in all, the clustering procedure is much more efficient when feature ranking space is considered, since the number of features and number of targets, in most of our datasets are significantly smaller than number of instances.

IV. EXPERIMENTAL DESIGN

A. EXPERIMENTAL QUESTIONS

We set the experimental design focusing on the following research questions:

(1.) Does structuring the output space (using a hierarchies) improves the predictive performance compared to the original flat MTR task?
(2.) Which clustering method yields better hierarchy?
  (2.1.) Can we achieve better predictive models by using the hierarchies obtained by structuring the feature ranking or target space?
(3.) Are the data-driven hierarchies better than the hierarchies created by a domain expert?
(4.) How the structuring of the output space scales from small to large output spaces?
(5.) How the performance difference translates from single model to ensemble of models?

In order to answer the above questions, we perform an extensive evaluation on a diverse datasets from the environmental and socio-economic domain. In the following part, we will describe the data we use.

B. DATA DESCRIPTION

We use 16 datasets for multi-target regression benchmark problems from 2 different domains (8 from the domain of socio-economic sciences and 8 from the domain of environmental sciences, from which 14 with small and 2 with large number of targets). The number of targets in the datasets range from 6 to 492 and the number of descriptive attributes from 16 to 576. The datasets with large number of targets (> 100) are inspected separately. The number of instances is also diverse ranging from 42 to 16976. The basic information and statistics about these datasets are given in Table 1.

The Andromeda (andro) dataset is for prediction of 6 water quality variables in Thermaikos Gulf of Thessaloniki, Greece [67]. The Airline Ticket Price datasets are used to infer the minimal price of an airline ticket for the next day (*atp1d*) i.e., next 7 days (*atp7d*) [68]. Metal data (*mdv2*) is the data for meta-learning of an automated assistant system for choosing appropriate machine learning algorithms for a specific data mining process [69]. The Occupational Employment Survey datasets are from the US Bureau of Labor Statistics for the years 1997 (*oes97*) and 2010 (*oes10*) [15]. The Online sales (*osales*) dataset deals with the prediction of online sales of products described with various product features. The dataset is

TABLE 1: Properties of the used benchmark datasets in terms of number of instances (**#inst**), number of descriptive attributes (**D**), number of targets (**T**), percentage of missing values (**MissVal**) and sorted by number of instances. The datasets with * as superscript will be considered separately, since they have large number of targets.

Dataset name	Abbr.	#inst	D	T	MissVal
Water quality	**wq**	1060	16	14	/
Andromeda	**andro**	49	30	6	/
Online Sales	**osales**	639	413	12	3.79%
Occupational Employment Survey for 1997	**oes97**	334	263	26	/
Occupational Employment Survey for 2010	**oes10**	403	298	16	/
Metal data	**mdv2**	42	53	10	24%
Prespa Lake Diatoms*	**pd***	349	16	111	0.11%
Prespa Lake Diatoms Top 10	**pdt**	248	16	10	0.54%
Airline Ticket Price (1 day)	**atp1d**	337	411	6	/
Airline Ticket Price (7 days)	**atp7d**	296	411	6	/
Vegetation conditions	**vgc**	16967	40	7	/
River Flows 1	**rf1**	9125	64	8	0.5%
River Flows 2	**rf2**	9125	576	8	6.68%
Slovenian Rivers*	**SloRiv***	1060	16	492	/
Supply Chain Management tournament (1 day)	**scm1d**	9803	280	16	/
Supply Chain Management tournament (20 days)	**scm20d**	8966	61	16	/

ublication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

from the Kaggle's Online Product Sales competition in 2012 [70]. Prespa Diatoms Lake (*pd*) and Prespa Diatoms Lake top 10 (*pdt*) datasets investigate the effect of the environmental conditions of Lake Prespa in the Republic of Macedonia on diatom communities [71]. The former (*pd*) is the complete data set with all 111 targets and examples, while the latter (*pdt*) consists of only top 10 the most abundant diatoms. River Flows (*rf1* and *rf2*) are datasets for prediction of the river network flows in the Mississippi river in the United States obtained from the US National Weather Service consists of 8 sites, with 8 attributes from each site [15]. The difference between *rf1* and *rf2* is that the latter includes the forecast information about the precipitation. The SCM datasets are from the 2010 Trading Agent Competition in the Supply Chain Management tournament (TAC SCM). It consists of 4-time delayed observations for traded prices of various computing equipment for the specific day (i.e., prices from 1, 2, 4 and 8 days ago vs. the price today) and trying to predict the forward trend of the next tournament day price (*scm1d*), i.e., the mean price of the next 20 tournament days (*scm20d*) [72]. The Vegetation condition(*vgc*) dataset concerns the prediction of the vegetation condition for the Victoria State in Australia and provided by the Arthur Rylah Institute for Environmental Research, Department of Sustainability and Environment (DSE) [3]. Water quality (*wq*) and Slovenian Rivers (*SloRiv*) are two datasets for predicting species abundance in water in Slovenian rivers using 16 chemical parameters as a descriptors. The *wq* data set consists of only 14 the most abundant species, while the *SloRiv* dataset consists of 492 different species which occur more than 5 times in the samples [73], [65].

C. EVALUATION MEASURES

We follow the literature recommendations regarding the evaluation measures [19]. We present the values of the average relative root mean squared error ($aRRMSE$) (Eq 7) for performance of the tested methods. To perform a fair comparison, we calculate these errors only for the target variables at the leafs of the hierarchy.

Let us assume that t is the number of target variables and N_{test} is the size of the test set. The actual value of a target variable of an example is Y, and $\hat{Y}$ is the predicted value using the model for that example. Similarly, $\bar{Y}$ is the average of the actual values for that target variable. The $aRRMSE$ can be define as follows:

$$aRRMSE = \frac{1}{t}\sum_{i=1}^{t} RRMSE_i = \\ = \frac{1}{t}\sum_{i=1}^{t}\sqrt{\frac{\sum_{k=1}^{N_{test}}(Y_i^{(k)} - \hat{Y}_i^{(k)})^2}{\sum_{k=1}^{N_{test}}(Y_i^{(k)} - \bar{Y}_i)^2}} \tag{7}$$

If $aRRMSE \approx 0$, then we have much better performance, but if $aRRMSE \approx 1$, we have a closer value to the default prediction that predicts the average value for each target.

D. PARAMETER INSTANTIATION

The majority of our experiments are performed using the CLUS software package (https://sourceforge.net/projects/clus/), where the predictive clustering framework for MTR and HMTR tasks, including PCTs for MTR/HMTR, random forests of PCTs for MTR/HMTR and feature ranking [6], [9] are implemented. The algorithms are developed to natively handle missing values.

A hierarchical tree defined by the used clustering methods in HMTR are defined as tree shaped hierarchies. For obtaining a hierarchy using the agglomerative clustering method, we use the non-commercial version of OCTAVE software package (functions *pdist()*, *linkage()* and *dendrogram()*). Furthermore, in OCTAVE, we used balanced k-means clustering for numerical type values, which is based on Hungarian (Munkres') assignment algorithm to assign the examples to the clusters [74]. Since most of the datasets have a relatively small number of targets (except the two with more than 100), we selected the value $k = 2$ for balanced k-means in order to obtain more branched hierarchies.

We use Euclidean distance metric in all our algorithms that require distance. In HMTR, as defined in previous sections, we use weighted Euclidean distance. Moreover, for random forest for feature ranking, we use GENIE3 as a feature importance method based on variable selection with ensembles of PCTs [59], [60]. We use 100 base ppredictive models for the random forests in all tasks (MTR, HMTR and feature ranking). For PCTs for HMTR task, we use sum as an aggregation function with the weight set to 0.75 [13].

E. HIERARCHIES CREATED BY A DOMAIN EXPERT

In our analysis, we also use hierarchies created by the domain experts, defined as a class ontology or domain-specific class structure. In the following part, we explain the creation of the hierarchies for each dataset.

The hierarchy in *mdv2* (Metal data) dataset is created based on a type of machine learning algorithm in three hierarchy levels. For *andro* (Andromeda) dataset is created based on correlation matrix given in [67]. For *pdt* (Prespa Lake Top 10) data set, the top 10 most abundant diatoms are grouped into a hierarchy based on their taxonomic rank. For *atp1d* and *atp7d* (Airline ticket prices) datasets, the target classes are grouped based on the type of the flight, either non-stop flight or with any number of stops. For *oes97* and *oes10* (Occupational Employment Survey 1997 and 2010), the target classes are organized into a hierarchy based on the type of the occupation and specific job position. For *osales* (Online Sales) data set,

the hierarchies are created based on sales products in first and the second half of the year. For *wq* (Water Quality) and *SloRiv* (Slovenian Rivers) datasets the hierarchies are created based on the taxonomic tanks of the species. The expert hierarchy for *rf1* and *rf2* (River Flows) datasets is constructed based on three different river network flows (Illionis, Iowa and Missouri). The hierarchy for *scm1d* and *scm20d* (Supply Chain Management) datasets is created based on the grouping the 16 PC configurations (targets) on 3 main market segments (low, medium and high) consisting of a combination of 10 different components, as it is given in Table 5 in the report [75]. Finally, the hierarchy for the *vgc* (Vegetation conditions) data set in created based on grouping of the target classes, either to tree related scores or other type of scores [3].

F. STATISTICAL EVALUATION

To validate our predictive models, we use 10-fold cross validation in all settings. More specifically, the whole dataset is first randomly split into 10 folds. Next, 9 folds are used for training, and the remaining one for testing. The procedure is repeated 10 times so that each fold is used exactly once as test set. The reported results represent an average of all 10 runs.

For statistical evaluation of the results, we adhered to the recommendations by [76]. For assessing the statistical significance of the differences, we used the non-parametric Friedman test [77] with the correction recommended by [78]. In order to compare the methods and to check the statistical significance among them, we used the Nemenyi post-hoc test [79]. The result from Nemenyi post-hoc test is presented with an average ranks diagram [76]. For statistical comparison between two algorithms, we used the Wilcoxon signed-rank non-parametric statistical hypothesis test [80].

V. RESULTS

In this section, we present the obtained results from the performed experiments using the procedures for structuring the output space. In our study, as output spaces, we consider the space consisting of the target values or the space consisting of feature ranks for each target. We compare the following methods for hierarchy construction:

- flat MTR problem (no hierarchy) (*MTR*);
- agglomerative clustering with single linkage (*AggS*);
- agglomerative clustering with complete linkage (*AggC*)
- balanced k-means clustering (*BkM*)
- clustering using predictive clustering trees (*PCT*).
- hierarchy created by an expert (*Expert*)

Since we have two different models (single PCTs model and random forest of PCTs) and two different structured output spaces, we show separately the results for single PCTs (Fig 2) and random forest of PCTs (Fig 7). To clarify the notation, we need to distinguish between using either single tree or random forest of PCTs and different methods of structuring the output space (target space and feature ranking space). To achieve this, we use prefixes (*PCT-* and *RF-*) and suffixes (*-TS* and *-FR*) before and after the hierarchy construction method name, accordingly. For example, *RF-BkM-TS* refers to the balanced k-means method used on the original target space using random forest of PCTs for model creation. Then, *PCT-PCT-FR* refers to the clustering method with PCTs of the output space consisting of feature rankings using single PCTs for building the model, etc.

Fig 2 visually presents the results of the predictive performance of single PCTs for each dataset. Examining the figure, it is clear that data-driven hierarchies, generally, improve the predictive performance over the flat MTR task, except on five datasets (*andro*, *pdt*, *atp1d*, *scm1d* and *scm20d*). It is interesting to notice that, for most of the datasets with more than 12 targets (*oes97, oes10, osales, wq*), using hierarchies noticeably improve the performance over flat MTR (with no hierarchies). Those results give an insight that, for the datasets with large number of targets, there is an improvement of the performance if the hierarchies obtained by structuring the target space, are used.

In order to figure out which data-driven clustering method for hierarchy creation performed the best, we created an average rank diagrams for aRRMSE values per output space for $p - value = 0.05$. More specifically, Fig 3 (left) illustrates the average diagram for clustering methods over the target space and Fig 3 (right) gives the average rank diagram for clustering methods over the feature ranking space. We can see that the best method for hierarchy creation over target space is *PCT-BkM-TS*, and it is only significantly better than *PCT-AggS-TS*. From the other side, in the average rank diagrams for the clustering methods over the feature ranking space, we can see that *PCT-BkM-FR* is the best performing method and it is significantly better than all others. Therefore, for task of MTR with single PCTs, we can easily recommend using balanced k-means clustering method for creation of hierarchies from the output space (either target or feature rankings space).

In order to check the significance of the performance between the two best approaches for hierarchy creation (considering the two target spaces), we perform non-parametric Wilcoxon hypothesis test for p-value = 0.05 for the *PCT-BkM-FR* and *PCT-BkM-TS* algorithms. The results show that *PCT-BkM-FR* $>$ *PCT-BkM-TS*; p-value = $0.0325 < 0.05$, which means that *PCT-BkM-FR* is statistically significantly better method than *PCT-BkM-TS*.

Considering this, we have that the hierarchies constructed over the space consisting of feature importances

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

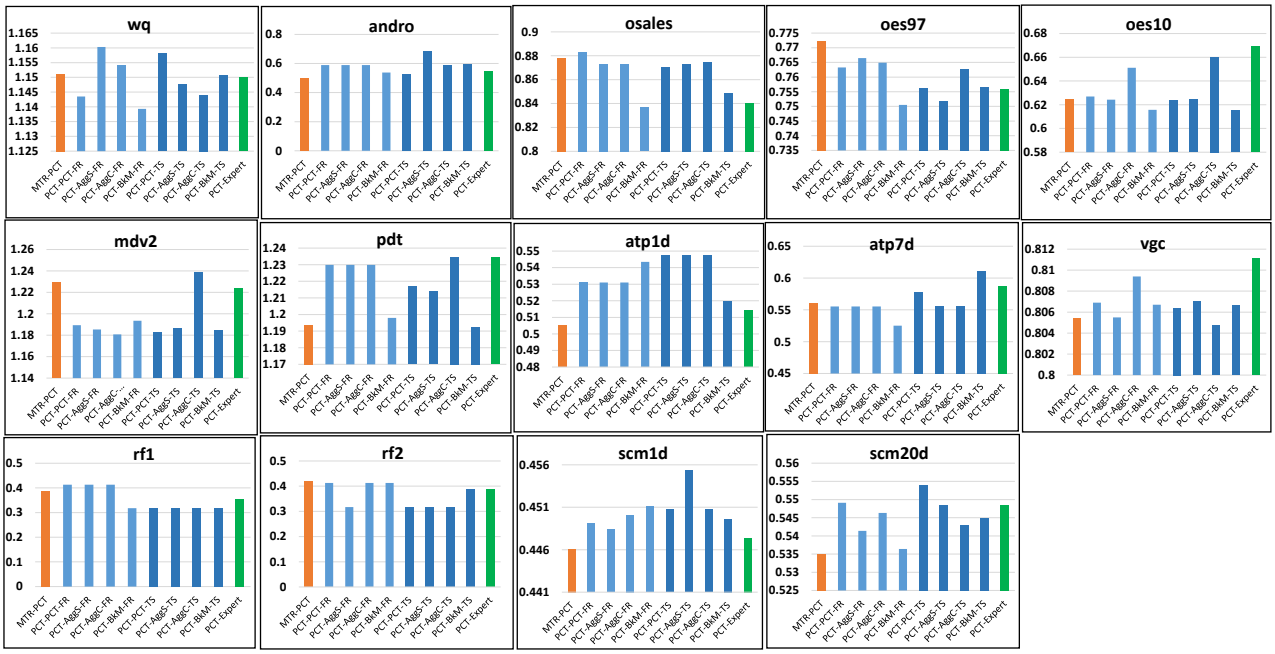

FIGURE 2: Results for the predictive performance of *single PCTs* from experiments per dataset represented by aRRMSE. Green bars represent hierarchies created by an expert and orange bars represent the flat MTR results.

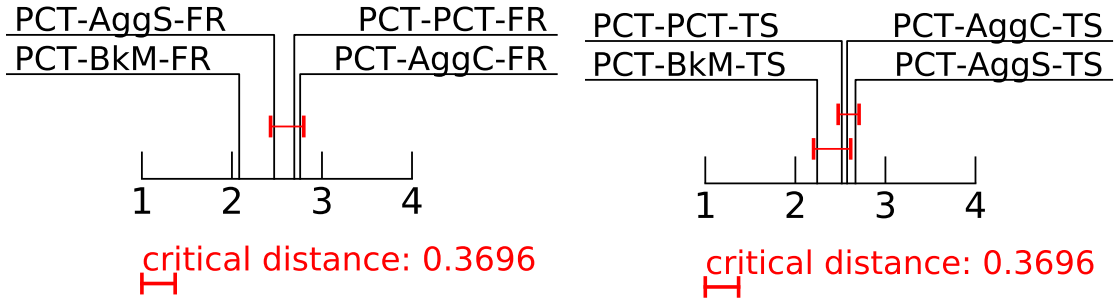

FIGURE 3: Average rank diagrams for algorithms that cluster the target space (left) and feature ranking space (right) using single PCTs.

are superior to the hierarchies constructed over the target space, both using balanced k-means method for clustering.

For a clearer picture over the best clustering method performance and the performance of the flat MTR method and using the hierarchy created by an expert in HMTR task, we took the best performing methods for structuring the output space (*PCT-BkM-TS* and *PCT-BkM-FR*) and compare together with flat MTR task performance (*MTR-PCT*) and the performance of the hierarchy created by an expert (*PCT-Expert*). The average rank diagram from statistical evaluation is given in Fig 4. We can see that *PCT-BkM-FR* is the superior algorithm, and significantly better than *MTR-PCT* and *PCT-Expert*. All in all, data-driven hierarchies improve the predictive performance in multi-target regression problems.

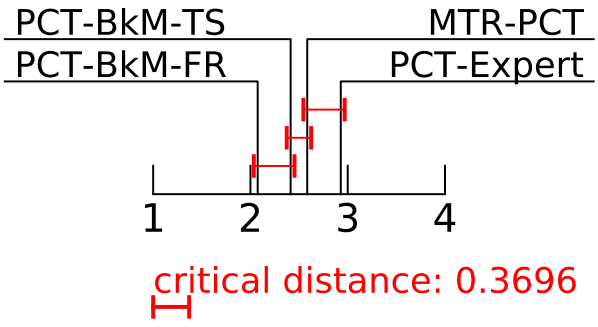

FIGURE 4: Average rank diagrams for the best algorithms from Fig. 3 compared to the flat MTR task and the use of an expert created hierarchy on the target space.

If we consider the performances for aRRMSE using random forest of PCT algorithm, we can see that in all data sets, the aRRMSE is reduced, which is in accordance with the general rule-of-thumb for the random forest. In the Appendix part of the paper, the results for $aRRMSE$ using random forest of PCTs per dataset are given in Fig 7.

To investigate the translation of predictive performance from single PCTs to ensemble of PCTs, we performed the same experimental analysis and statistical evaluation. Similar conclusions can be made as for the single PCTs. Generally, hierarchies improve the predictive performance over the flat MTR or expert created hierarchies (in eleven out of sixteen datasets). But, there is no statistically significant difference between the performances of used clustering algorithms and the flat MTR algorithm. The average rank diagrams for aRRMSE using random forest are given in Fig 8 and Fig 9 in the Appendix.

The detailed results of the predictive performance ($aRRMSE$) for each dataset that were used to draw the graphs in Figure 2 for single PCTs, i.e., in Figure 7 for random forests of PCTs, are given in Figure 12 in the Appendix.

We must note here that we exclude both large datasets (PD and SLORIV) from the statistical analysis, because the high number of targets will influence the overall per-target evaluation and will guide us towards the statistically incorrect conclusions. For that reason, we consider those two datasets separately in the next subsection.

A. STRUCTURING LARGE OUTPUT SPACES

In this subsection, we present the results from the experiments performed on the two datasets with large number of targets: Prespa Diatoms Lake (*pd*) with 111 targets and Slovenian rivers (*SloRiv*) with 492 targets. The main goal here is to make a more comprehensive and sustainable study which will take into consideration the size of the output space, i.e., the target space cardinality.

The balance k-means clustering algorithm for hierarchy creation, especially on the space consisting of feature rankings, is the best performing method based on above results. Furthermore, in the study of [66], they recommend to use the divisive methods for hierarchy creation and to some extend this relates with our results from the statistical evaluation. For that reason, we use the divisive methods (balanced k-means and predictive clustering trees) for clustering the output spaces for the two big datasets. More precisely, we show the results for clustering the target space using predictive clustering trees (PCT-PCT-TS and RF-PCT-TS) and for clustering the feature rankings space using balanced k-means (PCT-BkM-FR and RF-BkM-FR). The results are analysed as per target performance of the data-driven hierarchy creation methods and expert constructed hierarchy compared to the performance of the flat MTR task.

To better illustrate the results, we calculate the difference $\Delta RRMSE$, which is the difference between $RRMSE$ value of flat MTR and the $RRMSE$ from the appropriate method for hierarchy creation. The results for the *pd* dataset using single PCTs are shown in Fig 5. The green bars present the per target $RRMSE$ values that denote that HMTR models are better than flat MTR models (positive value for $\Delta RRMSE$), while the red bars present the per target $RRMSE$ values where MTR models are better than HMTR models (negative values for $\Delta RRMSE$). Examining the results, we can see that using the *PCT-BkM-FR* method, we obtain the best per-target performance. Specifically, by using *PCT-Expert* compared to *PCT-MTR* in the *pd* dataset, we have 60 out of 111 targets where *PCT-Expert*>*PCT-MTR*, then using *PCT-BkM-FR* compared to *PCT-MTR*, we have 76 out of 111 (68.5%) targets, where *PCT-BkM-FR* > *PCT-MTR* and finally, using *PCT-PCT-TS* v.s *PCT-MTR*, we have 72 out of 111 target, where *PCT-PCT-TS* > *PCT-MTR*.

The results from the *SloRiv* dataset are shown in Fig 10 (in Appendix). Here, by visual inspection of the results, we can see that using the hierarchy created by *PCT-PCT-TS* algorithm we obtain a better performance on the most of the targets compared to the *PCT-MTR* algorithm, i.e., *PCT-PCT-TS* > *PCT-MTR* in 325 out of 492 (66%) targets.

Furthermore, using random forest of PCTs yields quite similar situation. The difference here with single PCTs is that *RF-PCT-TS* clustering method gives the best results on the *pd* dataset. Specifically, we have *RF-PCT-TS* > *RF-MTR* in 89 out of 111 (80%) targets. This is a very good improvement compared to the other clustering methods for hierarchy creation. The results for the *pd* dataset are shown in Fig 6. Examining the results for the *SloRiv* dataset, again, same as single PCTs, we can see that by using *RF-PCT-TS* method we can obtain the best per-target performances i.e., *RF-PCT-TS* > *RF-MTR* in 287 out of 492 (59.5%) targets. The results for the *SloRiv* dataset are shown in Fig 11 from the Appendix.

Generally, on the larger datasets, there is an improvement of the performance, when the hierarchies are used. More precisely, divisive methods for clustering (hierarchy creation) are the best methods for structuring the output space, which is in accordance with the conclusions from the recent literature [17], [66]. Furthermore, data-driven hierarchies are generally better than the hierarchies created by an domain expert. It is confirmed by our results as well.

Examining the arrows in Fig 10 and Fig 11 (in Appendix) shown for Slovenian Rivers (*SloRiv*) dataset, we can see that for example, considering the target number 170 (which is taxa *Euglena viridis* from taxonomic group *EUGLENOPHYTA*), there is a significant improvement in

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

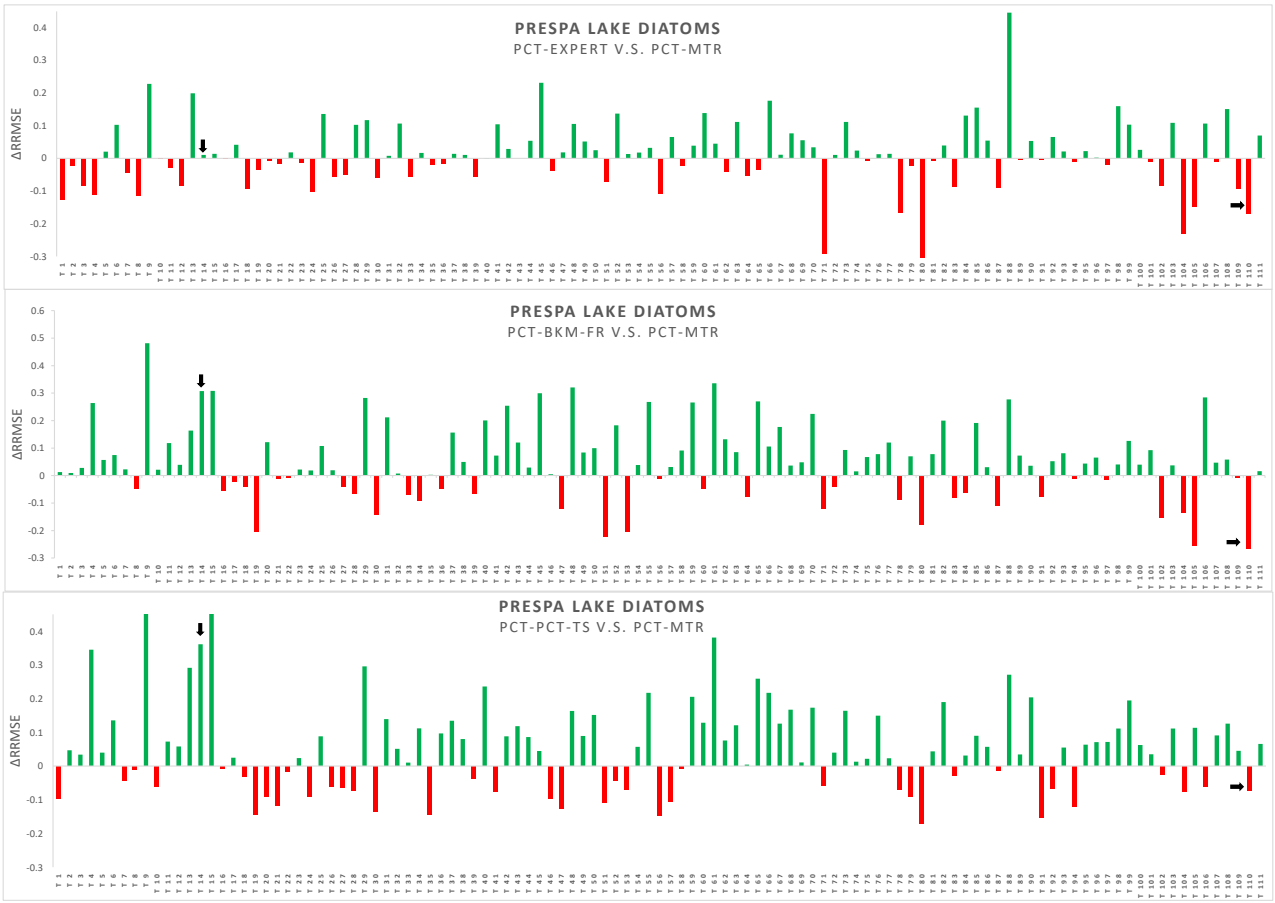

FIGURE 5: $\Delta RRMSE$ values for Prespa Diatom Lake dataset *pd* using single PCTs and using expert created hierarchy (*PCT-Expert*), balanced k-means on a feature ranking space (*PCT-BkM-FR*) and predictive clustering trees for clustering the target space (*PCT-PCT-TS*). The arrows represent the chosen examples with good/bad performance.

the performance, if the hierarchies are used rather than they are not used. The average abundance of all species in the examples is 71.8. The target 170 occurs 13 times in the examples, which is quite below the average. This confirms the fact that with small occurrence of the target in the examples, the model performance will be lower than considering a whole hierarchy (target dependence), where the target will be included. This is in accordance with the fact that, if we build a model with structuring of the output space (HMTR task), we can improve the predictive performance compared to the models built on a flat MTR task. Alternatively, if we want to check why the hierarchies do not help on some of the targets, as an example, we can select the target 353, which represents the taxa *Heptagenia sulphurea* from the *EPHEMEROPTERA* taxonomic group.

Similarly, examining the arrows in Fig 5 and Fig 6 for Prespa Diatoms dataset *pd* with 111 targets, we can make similar conclusions as for the previous dataset. For example, if we select the target number 14, on which we have the best performance by using hierarchies compared to the flat MTR task, the occurrence of this target in the examples is 5 times, but the average occurrence of the targets is 33.5. Therefore, as less the target occur in the examples, as much better performance can be achieved by using the structure of the output space (hierarchy) rather than using a flat MTR task, where no hierarchy is considered.

VI. CONCLUSIONS

In this paper, we present two data-driven approaches for structuring the output space. Namely, we present an algorithm for clustering the targets and the algorithm for clustering the targets according to the importance scores of each feature per target. Our research is focused on the question whether the two data-driven methods for structuring the output space can improve the predictive performance on the original flat multi-target regression task, and, moreover, whether data-driven hierarchies are better than expert created hierarchies.

For constructing the hierarchies, we investigate the use of agglomerative clustering method with single and com-

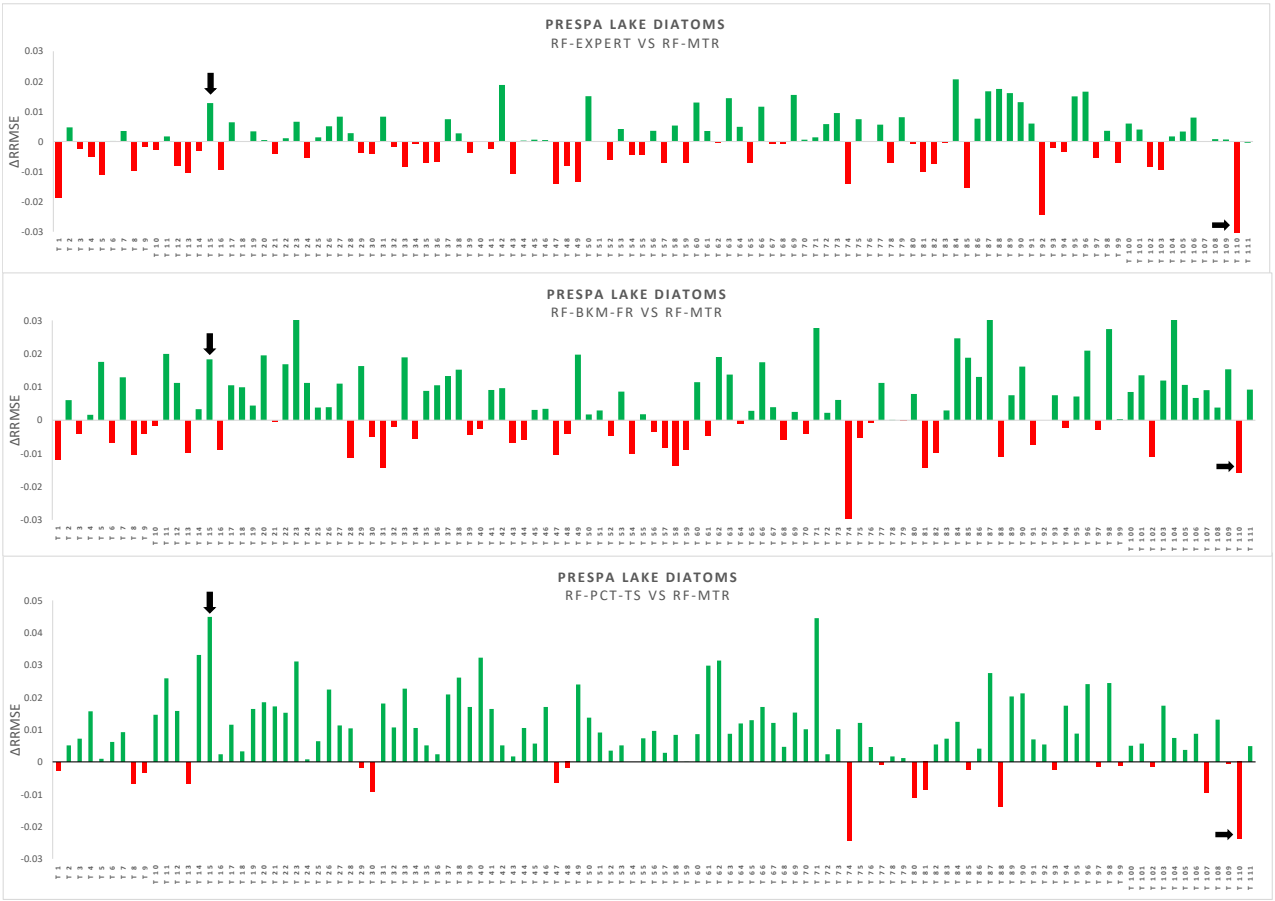

FIGURE 6: $\Delta RRMSE$ values for Prespa Diatom Lake dataset *pd* using random forest of PCTs and using expert created hierarchy (*RF-Expert*), balanced k-means on a feature ranking space (*RF-BkM-FR*) and predictive clustering trees for clustering the target space (*RF-PCT-TS*). The arrows represent the chosen examples with good/bad performance.

plete linkage, balanced k-means clustering and clustering using PCTs. The resulting problem is then transformed into a HMTR problem, and finally addressed by using PCTs and random forests of PCTs for HMTR. We use 16 benchmark datasets to evaluate the performance of all methods. Two datasets have a large number of targets ($>$ 100 targets). After obtaining the results for the average RMMSE ($aRRMSE$), we perform a statistical evaluation by using Friedman non-parametric test with Nemenyi post-hoc testing and Wilcoxon statistical test for testing the two best methods for structuring the output space.

The results show that for single PCTs, the data-driven approach for structuring (clustering) the output space significantly increases the predictive performance over the original MTR task and over the performance obtained by using an expert created hierarchy. A recommendation that comes out from the statistical evaluation is that balanced k-means algorithm can be used for clustering the output space. Moreover, by using hierarchies created over the feature ranking space there is an improvement in the performance. The same, but to a lesser extent, conclusions can be made by using ensembles of PCTs, since they are not improving the predictive performance significantly.

For large output spaces, datasets with a large number of targets (greater than 100), the results show that hierarchies improve the performance compared to using the flat MTR task, where no hierarchy is considered. For structuring the large output spaces, the divisive methods for hierarchy creation are the best choice, since they are constructing good hierarchies that improve the predictive performance. Moreover, data-driven hierarchies are a better choice than expert created hierarchies, which implies that we could obtain good structure of the target space if we discover the knowledge from the data directly rather than using the structure based on some pre-defined relations defined by a domain expert.

For further work, we plan to make more extensive evaluation on more datasets with a larger number of targets and to investigate different feature ranking methods (for example, RReliefF and attention mechanism based feature ranking with NNs). There are some insights that there might be potential improvements of the performance

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

that can be achieved with cutting the obtained hierarchies based on data density, distance between the nodes etc. and addressing the task of MTR as multiple smaller MTR tasks.

ACKNOWLEDGEMENTS

We would like to acknowledge the support of the European Commission through the project MAESTRA - Learning from Massive, Incompletely annotated, and Structured Data (Grant number ICT-2013-612944), the project LANDMARK - Land management, assessment, research, knowledge base (H2020 Grant number 635201) and Teagasc Walsh Fellowship Programme.

REFERENCES

[1] G. Tsoumakas and I. Katakis, "Multi Label Classification: An Overview," International Journal of Data Warehouse and Mining, vol. 3, no. 3, pp. 1–13, 2007.

[2] D. Demšar, S. Džeroski, T. Larsen, J. Struyf, J. Axelsen, M. Pedersen, and P. Krogh, "Using multi-objective classification to model communities of soil.," Ecological Modelling, vol. 191, pp. 131–143, 2006.

[3] D. Kocev, S. Džeroski, M. White, G. Newell, and P. Griffioen, "Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition," Ecological Modelling, vol. 220, no. 8, pp. 1159–1168, 2009.

[4] J. Levatić, D. Kocev, M. Debeljak, and S. Džeroski, "Community structure models are improved by exploiting taxonomic rank with predictive clustering trees," Ecological Modelling, vol. 306, pp. 294–304, 2015.

[5] C. N. Silla and A. Freitas, "A survey of hierarchical classification across different application domains," Data Mining and Knowledge Discovery, vol. 22, pp. 31 – 72, 2011.

[6] D. Kocev, C. Vens, J. Struyf, and S. Džeroski, "Tree ensembles for predicting structured outputs," Pattern Recognition, vol. 46, no. 3, pp. 817–833, 2013.

[7] G. H. Bakır, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan, Predicting Structured Data. Neural Information Processing. The MIT Press., 2007.

[8] H. Blockeel, Top-down induction of first order logical decision trees. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1998.

[9] J. Struyf and S. Džeroski, "Constraint Based Induction of Multi-Objective Regression Trees," in Proc. of the 4th International Workshop on Knowledge Discovery in Inductive Databases KDID - LNCS 3933, pp. 222–233, Springer, 2006.

[10] D. Kocev, C. Vens, J. Struyf, and S. Džeroski, "Ensembles of Multi-Objective Decision Trees," in Proc. of the 18th European conference on Machine Learning, pp. 624–631, 2007.

[11] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," Machine Learning, vol. 73, no. 2, pp. 185–214, 2008.

[12] I. Slavkov, V. Gjorgjioski, J. Struyf, and S. Džeroski, "Finding explained groups of time-course gene expression profiles with predictive clustering trees," Molecular BioSystems, vol. 6, no. 4, pp. 729–740, 2010.

[13] V. Mileski, S. Džeroski, and D. Kocev, "Predictive clustering trees for hierarchical multi-target regression," in Advances in Intelligent Data Analysis XVI (N. Adams, A. Tucker, and D. Weston, eds.), pp. 223–234, Springer International Publishing, 2017.

[14] L. Breiman, "Bagging Predictors," Machine Learning, vol. 24, no. 1, pp. 123–140, 1996.

[15] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas, "Multi-target regression via input space expansion: treating targets as inputs," Machine Learning, vol. 104, no. 1, pp. 55–98, 2016.

[16] G. Madjarov, D. Gjorgjevikj, I. Dimitrovski, and S. Džeroski, "The use of data-derived label hierarchies in multi-label classification," Journal of Intelligent Information Systems, vol. 47, no. 1, pp. 57–90, 2016.

[17] S. Nikoloski, D. Kocev, and S. Džeroski, "Structuring the output space in multi-label classification using feature ranking," International Workshop NFMCP in conjunction with ECML-PKDD 2017 (Skopje, Macedonia), pp. 151–166, 2018.

[18] S. Džeroski, V. Gjorgjioski, I. Slavkov, and J. Struyf, "Analysis of time series data with predictive clustering trees," in Knowledge Discovery in Inductive Databases, 5th International Workshop, KDID 2006, Revised Selected and Invited Papers - LNCS 4747, pp. 63–80, Springer, 2007.

[19] H. Borchani, G. Varando, C. Bielza, and P. Larrañaga, "A survey on multi-output regression," Wiley Int. Rev. Data Min. and Knowl. Disc., vol. 5, no. 5, pp. 216–233, 2015.

[20] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonothogonal problems," Technometrics, vol. 12, pp. 55–67, 1970.

[21] J. Friedman, "Stochastic gradient boosting," Computational Statistical Data Analysis, vol. 38, no. 4, pp. 367–378, 2002.

[22] E. Spyromitros-Xioufis, W. Groves, G. Tsoumakas, and I. Vlahavas, "Multi-label classification methods for multi-target regression," arXiv preprint arXiv:1211.6581 Cornall University Library, pp. 1159–1168, 2012.

[23] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier Chains for Multi-label Classification," in Proc. of the 20th European Conference on Machine Learning, pp. 254–269, 2009.

[24] S. Godbole and S. Sarawagi, "Discriminative Methods for Multi-labeled Classification," in Advances in Knowledge Discovery and Data Mining, pp. 22–30, Springer Berlin / Heidelberg, 2004.

[25] W. Zhang, X. Liu, Y. Ding, and D. Shi, "Multi-output ls-svr machine in extended feature space.," in Proc. of the 2012 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, pp. 130–134, 2012.

[26] J. Wang, Z. Chen, K. Sun, H. Li, and X. Deng, "Multi-target regression via target specific features," Knowledge-Based Systems, vol. 170, pp. 70 – 78, 2019.

[27] A. Izenman, "Reduced-rank regression for the multivariate linear model," Journal of Multivariate Analysis, vol. 5, pp. 248–264, 1975.

[28] A. van der Merwe and J. V. Zidek, "Multivariate regression analysis and canonical variates," Can J Stat, vol. 8, pp. 27–39, 1980.

[29] R. Tibshirani, "Regression shrinkage and selection via the lasso.," Journal of the Royal Statistical Society. Series B (Methodological), p. 267–288, 1996.

[30] C. E. Rasmussen and C. K. Williams, "Gaussian processes for machine learning.," The MIT Press, Cambridge, MA, USA, vol. 38, p. 715–719, 2006.

[31] L. Breiman and J. H. Friedman, "Predicting multivariate responses in multiple linear regression," J. R. Stat. Soc. Series B, vol. 59, no. 1, pp. 3–54, 1997.

[32] A. Appice and S. Džeroski, "Stepwise induction of multi-target model trees," in In Proc: 18th ECML 2007, Warsaw, Poland, pp. 502–509, 2007.

[33] T. Hothorn, K. Hornik, and A. Zeileis, "Unbiased recursive partitioning: A conditional inference framework," Journal of Computational and Graphical statistics, vol. 15, pp. 651–674, 2006.

[34] T. Aho, B. Ženko, and S. Džeroski, "Rule ensembles for multi-target regression," in In Proc. of Ninth IEEE International Conference on Data Mining, pp. 21–30, IEEE Press, 2009.

[35] M. Breskvar, D. Kocev, and S. Džeroski, "Ensembles for multi-target regression with random output selections," Machine Learning, vol. 107, pp. 1673–1709, Nov 2018.

[36] M. Pugelj and S. Džeroski, "Predicting structured outputs k-nearest neighbours method," in Discovery Science, LNCS vol. 6926, pp. 262–276, 2011.

[37] W. J. Brouwer, J. D. Kubicki, J. O. Sofo, and C. L. Gilesd, "An investigation of machine learning methods applied to structure prediction in condensed matter.," arXiv preprint arXiv:1405.3564, 2014.

[38] M. Sánchez-Fernández, M. de Prado-Cumplido, J. Arenas-García, and F. Pérez-Cruz, "SVM multiregression for nonlinear channel estimation in multiple-input multiple output systems.," IEEE Transactions on Signal Processing, vol. 52, no. 8, pp. 2298–2307, 2004.

[39] B.-H. Mevik and R. e. a. Wehrens, "The pls package: Principal component and partial least squares regression in R.," Journal of Statistical Software, vol. 18, no. 2, p. 1–24, 2007.

[40] T. Hastie, J. Friedman, and R. Tibshirani, "Additive models, trees, and related methods.," In The Elements of Statistical Learning (Springer), p. 321–329, 2001.
[41] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm.," In IEEE International Conference on Neural Networks (IEEE), p. 586–591, 1993.
[42] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for improved unconstrained handwriting recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, p. 855–868, 2009.
[43] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," CoRR, vol. abs/1402.1128, 2014.
[44] X. Zhen, M. Yu, X. He, and S. Li, "Multi-target regression via robust low-rank learning," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 2, pp. 497–504, 2018.
[45] M. Kuo, B. Mohler, S. L. Raudenbush, and F. J. Earls, "Assessing exposure to violence using multiple informants: Application of hierarchical linear model," Journal of Child Psychology and Psychiatry, vol. 41, no. 8, pp. 1049–1056, 2000.
[46] A. Gelman, "Multilevel (hierarchical) modeling: What it can and cannot do.," Technometrics, vol. 48, p. 432–435, 2006.
[47] J. de Leeuw and E. Meijer, Handbook of Multilevel Analysis. Springer, New York, NY, 2008.
[48] T. A. B. Snijders, Multilevel Analysis, pp. 879–882. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.
[49] A. O'Brien, James and M. Marakas, George, Management Information Systems. McGraw-Hill/Irwin, 2010.
[50] R. Agrawal, A. Gupta, and S. Sarawagi, "Modeling multidimensional databases.," In Proceedings of the 13th International Conference on Data Engineering (IEEE Computer Society), p. 232–243, 1997.
[51] T. Nguyen, A. M. Tjoa, and R. Wagner, "An object oriented multidimensional data model for olap.," In Proceedings of the 1st International Conference on Web-Age Information Management (WAIM) in LNCS (Springer-Verlag), vol. 1846, pp. 69–69, 2000.
[52] A. Joly, P. Geurts, and L. Wehenkel, "Random forests with random projections of the output space for high dimensional multi-label classification," In Joint European conference on machine learning and knowledge discovery in databases, pp. 607–622, 2014.
[53] J. Levatić, D. Kocev, and S. Džeroski, "The importance of the label hierarchy in hierarchical multi-label classification.," Journal of Intelligent Information Systems, vol. 45, p. 247–271, 2015.
[54] P. Szymanski, T. Kajdanowicz, and K. Kersting, "How is a data-driven approach better than random choice in label space division for multi-label classification?," Entropy, vol. 18, p. 282, 2016.
[55] G. Tsoumakas and I. Vlahavas, "Random k-Labelsets: An Ensemble Method for Multilabel Classification," in Proc. of the 18th European conference on Machine Learning, pp. 406–417, 2007.
[56] A. Joly, "Exploiting random projections and sparsity with random forests and gradient boosting methods—application to multi-label and multi-output learning, random forest model compression and leveraging input sparsity," arXiv preprint arXiv:1704.08067, 2017.
[57] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
[58] A. Verikas, A. Gelzinis, and M. acauskiene, "Mining data with random forests: A survey and results of new tests," Pattern Recognition, vol. 44, no. 2, pp. 330–349, 2011.
[59] V. A. Huynh-Thu, L. Irrthum, Wehenkel, and P. Geurts, "Inferring regulatory networks from expression data using tree-based methods," PLos One, vol. 5, no. 9, 2010.
[60] M. Petković, D. Kocev, and S. Džeroski, "Feature ranking for multi-target regression," Machine Learning Journal, vol. to appear, 2019.
[61] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective and Efficient Multilabel Classification in Domains with Large Number of Labels," in Proc. of the ECML/PKDD Workshop on Mining Multidimensional Data, pp. 30–44, 2008.
[62] D. Kocev, Ensembles for predicting structured outputs. PhD thesis, IPS Jožef Stefan, Ljubljana, Slovenia, 2011.
[63] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Džeroski, "Fast and scalable image retrieval using predictive clustering trees," International Conference on Discovery Science, pp. 33–48, 2013.
[64] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone, Classification and Regression Trees. Chapman & Hall/CRC, 1984.
[65] H. Blockeeel, S. Džeroski, and J. Grbović, "Simultaneous prediction of multiple chemical parameters of river water quality with Tilde," in Proceedings of the 3rd European Conference on PKDD - LNAI 1704, pp. 32–40, Springer, 1999.
[66] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski, "An extensive experimental comparison of methods for multi-label learning," Pattern Recognition, vol. 45, no. 9, pp. 3084–3104, 2012.
[67] E. V. Hatzikos, G. Tsoumakas, G. Tzanis, B. Nick, and I. P. Vlahavas, "An empirical study on sea water quality prediction," Knowl.-Based Syst., vol. 21, no. 6, pp. 471–478, 2008.
[68] W. Groves and M. Gini, "On optimizing airline ticket purchase timing.," ACM Transactions on Intelligent Systems and Technology (TIST), vol. 7, no. 3, p. 1–28, 2015.
[69] L. Todorovski, H. Blockeel, and S. Dzeroski, "Ranking with predictive clustering trees," in Machine Learning: ECML 2002, 13th European Conference on Machine Learning, Helsinki, Finland, August 19-23, 2002, Proceedings, pp. 444–455, 2002.
[70] Kaggle, "Kaggle: Online product sales." https://www.kaggle.com/c/online-sales, 2012. Accessed: 2017-05-05.
[71] D. Kocev, A. Naumoski, K. Mitreski, S. Krtić, and S. Džeroski, "Learning habitat models for the diatom community in Lake Prespa," Ecological Modelling, vol. 221, no. 2, pp. 330–337, 2010.
[72] W. Groves and M. Gini, "Improving prediction in tac scm by integrating multivariate and temporal aspects via pls regression.," In Agent-Mediated Electronic Commerce. Designing Trading Strategies and Mechanisms for Electronic Markets, pp. 28–43, 2013.
[73] S. Džeroski, D. Demšar, and J. Grbović, "Predicting chemical parameters of river water quality from bioindicator data," Applied Intelligence, vol. 13, no. 1, pp. 7–17, 2000.
[74] M. Malinen and P. Fränti, "Balanced k-means for clustering," Joint Int. Workshop on Structural, Syntactic, and Statistical Pattern Recognition (S+SSPR 2014), LNCS 8621, pp. 32–41, 2014.
[75] J. Collins, R. Arunachalam, N. Sadeh, J. Eriksson, N. Finne, and S. Janson, "The supply chain management game for the 2007 trading agent competition," 2006.
[76] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," Journal of Machine Learning Research, vol. 7, pp. 1–30, 2006.
[77] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," Journal of Machine Learning Research, vol. 11, no. 1, pp. 86–92, 1940.
[78] R. L. Iman and J. M. Davenport, "Approximations of the critical region of the Friedman statistic," Communications in Statistics - Theory and Methods, vol. 9, no. 6, pp. 571–595, 1980.
[79] P. B. Nemenyi, Distribution-free multiple comparisons. PhD thesis, Princeton University, Princeton, NY, USA, 1963.
[80] F. Wilcoxon, "Individual Comparisons by Ranking Methods," Biometrics Bulletin, vol. 1, no. 6, pp. 80–83, 1945.

APPENDIX

ublication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

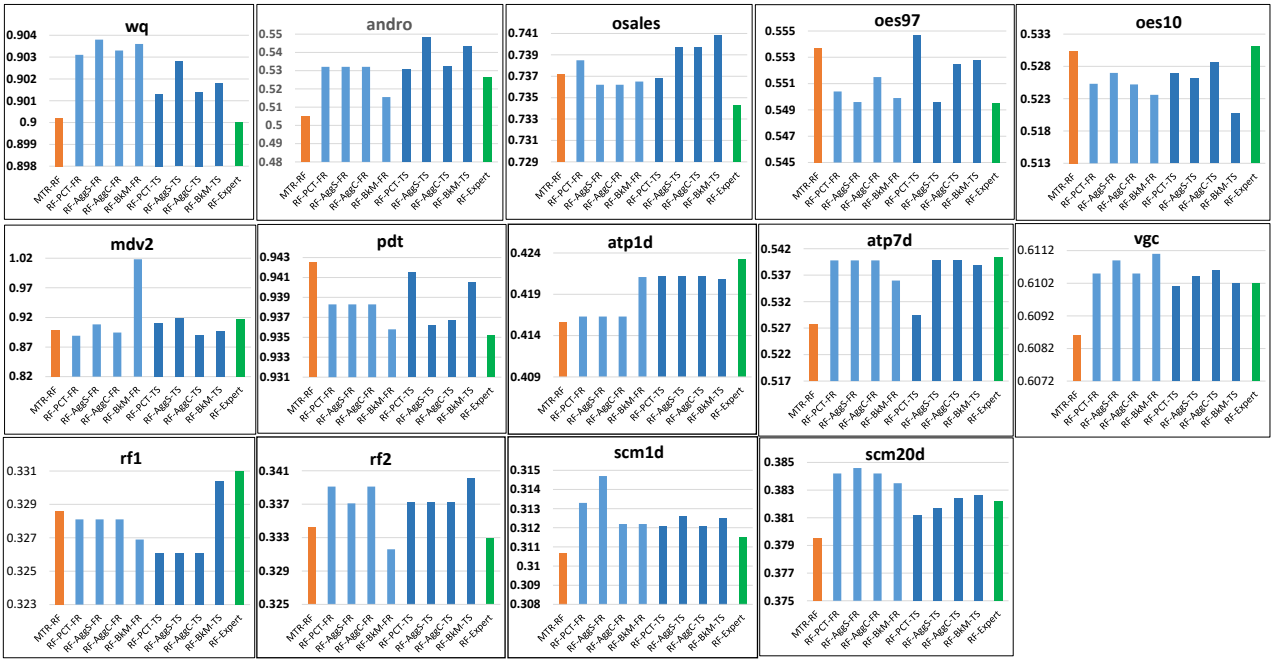

FIGURE 7: Results for the predictive performance of *Random forest of PCTs* from experiments per dataset represented by aRRMSE. Green bars represent hierarchies created by an expert and orange bars represent the flat MTR results.

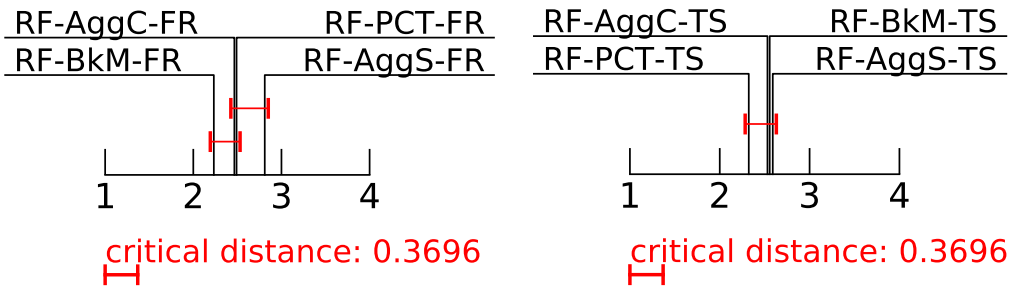

FIGURE 8: Average rank diagrams for algorithms cluster the target space (left) and feature ranking space (right) using random forest of PCTs.

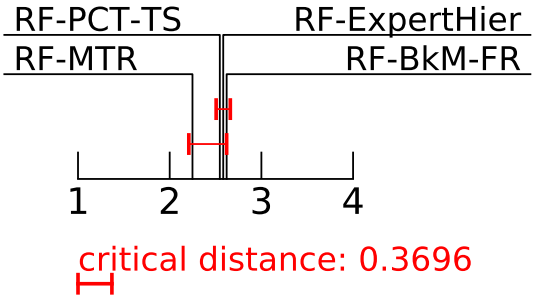

FIGURE 9: Average rank diagrams for the best algorithms from Fig.8 compared to the flat MTR task and the use of an expert created hierarchy on the target space.

FIGURE 10: $\Delta RRMSE$ values for Slovenian rivers *SloRiv* dataset using single PCTs and using expert created hierarchy (*PCT-Expert*), balanced k-means on a feature ranking space (*PCT-BkM-FR*) and predictive clustering trees for clustering the target space (*PCT-PCT-TS*). The arrows represent the chosen examples where we have good/bad performance.

publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2945084, IEEE Access

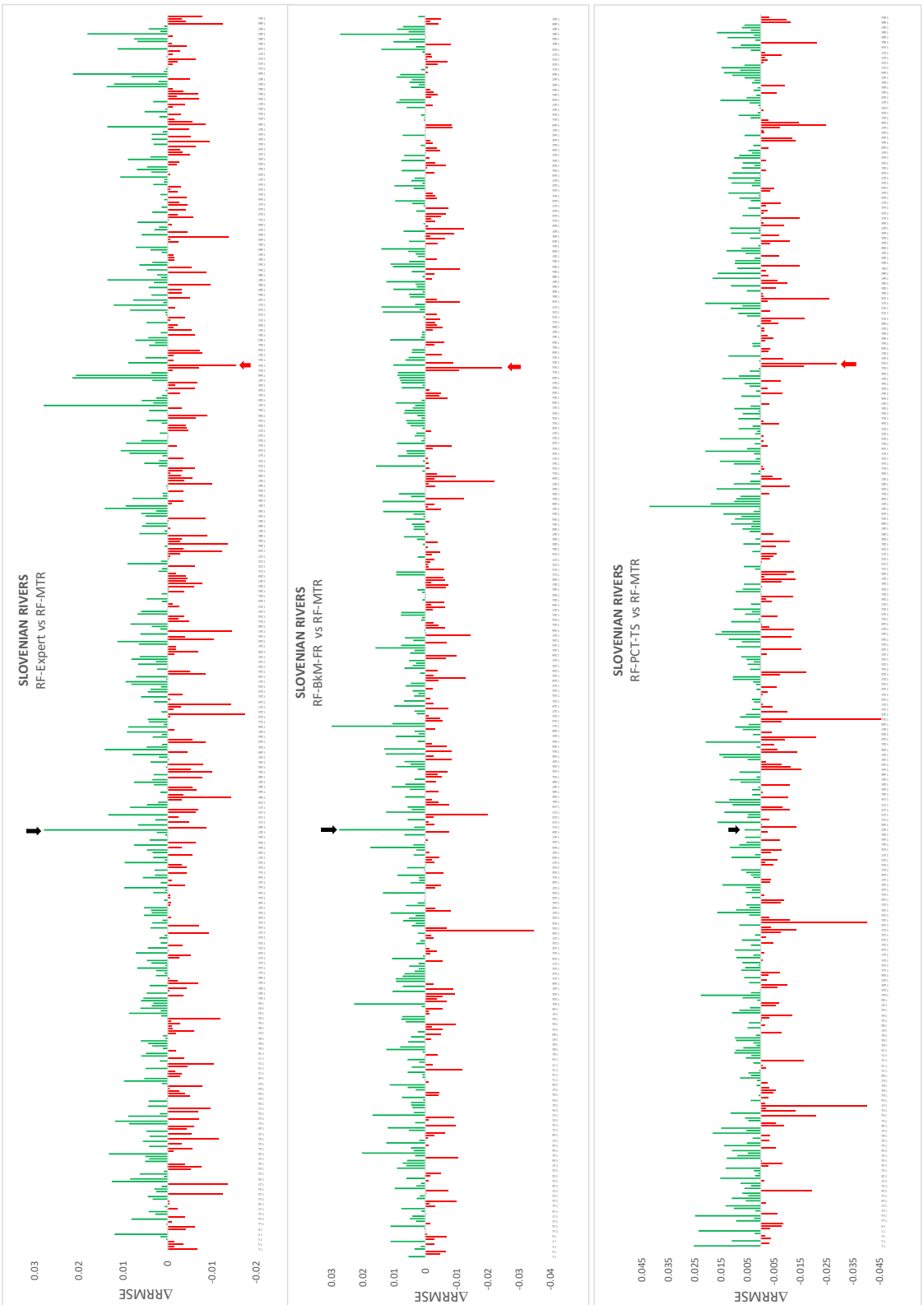

FIGURE 11: $\Delta RRMSE$ values for Slovenian rivers dataset *SloRiv* using random forest of PCTs and using expert created hierarchy (*PCT-Expert*), balanced k-means on a feature ranking space (*PCT-BkM-FR*) and predictive clustering trees for clustering the target space (*PCT-PCT-TS*). The arrows represent the chosen examples where we have good/bad performance.

Single PCTs

aRRMSE	wq	andro	osales	oes97	oes10	pdt	atp1d	atp7d	vgc	rf1	rf2	scm1d	scm20d	mdv2
MTR-PCT	1.1511	**0.5004**	0.878	0.7724	0.6244	1.1937	**0.5052**	0.5603	0.8054	0.3871	0.417	**0.4461**	**0.535**	1.2295
PCT-PCT-FR	1.1435	0.5888	0.883	0.7633	0.6269	1.2299	0.531	0.5554	0.8069	0.4133	0.4129	0.4491	0.5491	1.1893
PCT-AggS-FR	1.1603	0.5888	0.8733	0.7665	0.6243	1.2299	0.531	0.5554	0.8055	0.4133	0.3169	0.4484	0.5414	1.1853
PCT-AggC-FR	1.154	0.5888	0.8733	0.7649	0.6512	1.2299	0.531	0.5554	0.8094	0.4133	0.4129	0.4501	0.5463	**1.1808**
PCT-BkM-FR	**1.1393**	0.5375	**0.8366**	**0.7505**	0.6157	1.198	0.5435	**0.5251**	0.8067	0.3175	0.4126	0.4511	0.5364	1.1935
PCT-PCT-TS	1.158	0.5247	0.8707	0.7562	0.6236	1.2169	0.5474	0.5774	0.8064	0.3181	0.3174	0.4507	0.554	1.1825
PCT-AggS-TS	1.1475	0.6809	0.8733	0.7517	0.6244	1.2139	0.5474	0.5554	0.807	0.3181	0.3174	0.4553	0.5485	1.1863
PCT-AggC-TS	1.144	0.5888	0.8745	0.7628	0.6604	1.2347	0.5474	0.5554	**0.8047**	0.3181	0.3174	0.4507	0.543	1.2383
PCT-BkM-TS	1.1507	0.5931	0.8486	0.7566	**0.6151**	**1.1923**	0.5193	0.6105	0.8066	**0.317**	0.3858	0.4496	0.5448	1.1843
PCT-Expert	1.15	0.545	0.8404	0.756	0.6689	1.2344	0.514	0.5879	0.8111	0.3532	0.3859	0.4474	0.5486	1.224

Random Forests

aRRMSE	wq	andro	osales	oes97	oes10	pdt	atp1d	atp7d	vgc	rf1	rf2	scm1d	scm20d	mdv2
MTR-RF	0.9002	**0.5051**	0.7372	0.5537	0.5304	0.9425	**0.4156**	**0.5277**	**0.6086**	0.3286	0.3342	**0.3107**	**0.3795**	0.8982
RF-PCT-FR	0.9031	0.5321	0.7385	0.5504	0.5253	0.9383	0.4163	0.5398	0.6105	0.3281	0.3391	0.3133	0.3842	0.8891
RF-AggS-FR	0.9038	0.5321	0.7362	0.5496	0.527	0.9383	0.4163	0.5398	0.6109	0.3281	0.3371	0.3147	0.3846	0.9083
RF-AggC-FR	0.9033	0.5321	0.7362	0.5515	0.5252	0.9383	0.4163	0.5398	0.6105	0.3281	0.3391	0.3122	0.3842	0.8946
RF-BkM-FR	0.9036	0.5155	0.7365	0.5499	0.5236	0.9358	0.4211	0.536	0.6111	0.3269	**0.3316**	0.3122	0.3835	1.0182
RF-PCT-TS	0.9013	0.5304	0.7368	0.5547	0.5269	0.9415	0.4212	0.5294	0.6101	**0.3261**	0.3373	0.3121	0.3812	0.9104
RF-AggS-TS	0.9028	0.5482	0.7397	0.5496	0.5261	0.9362	0.4212	0.5398	0.6104	**0.3261**	0.3373	0.3126	0.3817	0.9195
RF-AggC-TS	0.9014	0.5321	0.7397	0.5525	0.5286	0.9367	0.4212	0.5398	0.6106	**0.3261**	0.3373	0.3121	0.3824	**0.8903**
RF-BkM-TS	0.9018	0.5433	0.7408	0.5528	**0.5207**	0.9405	0.4208	0.5389	0.6102	0.3304	0.3401	0.3125	0.3826	0.8962
RF-Expert	**0.9**	0.5263	**0.7343**	**0.5495**	0.5311	**0.9352**	0.4232	0.5404	0.6102	0.331	0.3329	0.3115	0.3822	0.9165

FIGURE 12: Detailed results for the predictive performance ($aRRMSE$) per dataset corresponding to the graphical results in Figure 2 for single PCTs i.e., in Figure 7 for random forests of PCTs