Empirical Asymmetric Selective Transfer in Multi-Objective Decision Trees

Beau Piccart, Jan Struyf, and Hendrik Blockeel

Department of Computer Science, Katholieke Universiteit Leuven,
Celestijnenlaan 200A, 3001 Leuven, Belgium
{Beau.Piccart,Jan.Struyf,Hendrik.Blockeel}@cs.kuleuven.be
http://www.cs.kuleuven.be

Abstract. We consider learning tasks where multiple target variables need to be predicted. Two approaches have been used in this setting: (a) build a separate single-target model for each target variable, and (b) build a multi-target model that predicts all targets simultaneously; the latter may exploit potential dependencies among the targets. For a given target, either (a) or (b) can yield the most accurate model. This shows that exploiting information available in other targets may be beneficial as well as detrimental to accuracy. This raises the question whether it is possible to find, for a given target (we call this the main target), the best subset of the other targets (the support targets) that, when combined with the main target in a multi-target model, results in the most accurate model for the main target. We propose Empirical Asymmetric Selective Transfer (EAST), a generally applicable algorithm that approximates such a subset. Applied to decision trees, EAST outperforms single-target decision trees, multi-target decision trees, and multi-target decision trees with target clustering.

1 Introduction

There has been increasing interest recently in simultaneous prediction of multiple variables, also known as multi-target prediction or multi-objective prediction. In typical classification or regression problems, there is a single target variable that needs to be predicted as accurately as possible. In multi-target prediction, on the other hand, the input is associated with a vector of target variables, and all of them need to be predicted as accurately as possible.

Multi-target prediction is encountered for instance in ecological modelling where the domain expert is interested in (simultaneously) predicting the frequencies of different organisms in river water [1] or agricultural soil [2]. It can also be applied in multi-label classification tasks [3], where a set of labels is to be predicted for each instance instead of a single label; in prediction tasks with a structured output space [4], such as hierarchical multi-label classification [5], where the output is structured as a taxonomy of labels (e.g., newsgroups); and in multi-task or transfer learning, where knowledge gained from learning one task is reused to better learn related tasks [6].

It has been shown that multi-target models can be more accurate than predicting each target individually with a separate single-target model [6]. This is a consequence of the fact that when the targets are related (e.g., if they represent frequently co-occurring organisms in the ecological modelling applications mentioned above), they can carry information about each other; the single-target approach is unable to exploit that information, while multi-target models naturally exploit it. This effect is known as inductive transfer: the information a target carries about the other targets is transferred to those other targets. Note the connection with collective classification [7]: the latter exploits dependencies among targets of different instances, while multi-target models exploit dependencies among the multiple targets of the same instance.

Multi-target models do not, however, always lead to more accurate prediction. As we will show, for a given target variable, the variable's single-target model may be more accurate than the multi-target model. That is, inductive transfer from other variables can be beneficial, but it may also be detrimental to accuracy. Let us focus on one particular target and call this the main target. The subset of targets that, when combined with the main target in a multi-target model, results in the most accurate model for the main target, may be non-trivial, i.e., different from the empty set and from the set of all targets. We call this set the support set for the main target. This paper investigates how we can best approximate this set. Note that the two natural extremes of this approach are the single-target model (the support set is empty) and the full multi-target model (the support set includes all targets).

Based on the above observation, we propose Empirical Asymmetric Selective Transfer (EAST), a greedy algorithm that approximates the support set for a given main target. EAST has the following advantages over other approaches that try to exploit transfer selectively [8–11]: (a) EAST does not assume transfer to be symmetric (in fact, we show that transfer can be asymmetric), (b) EAST estimates transfer empirically and does not rely on heuristic approximations (we show that heuristics may poorly approximate transfer), (c) EAST does not make explicit assumptions about the distribution of the different target variables, and (d) EAST is a general method in the sense that it can be combined with any multi-target learner (other methods are often tied to a particular type of models, such as neural networks).

EAST is the main contribution of this paper. A second contribution is that we show that exploiting transfer selectively is useful in the context of decision trees; previous work focused on other model types such as neural networks [9] and k-nearest neighbor [8]. Decision trees have the well-known advantage over these methods that they are easy to interpret.

The rest of this paper is organized as follows. Sec. 2 introduces single- and multi-target prediction formally and defines our problem setting. Sec. 3 discusses known methods for multi-target prediction and inductive transfer. Sec. 4 describes multi-target decision trees, and shows that their so-called transfer matrix is asymmetric. This motivates EAST, which we introduce in Sec. 5. Sec. 6 presents experiments with EAST, and Sec. 7 states the main conclusions.

2 Single/Multi-Target Prediction and Problem Setting

Assume we have a dataset S containing couples $(\mathbf{x}, \mathbf{y})$ with $\mathbf{x} \in X$ the input vector and $\mathbf{y} \in Y = Y_1 \times \cdots \times Y_n$ the target vector. Denote with $y_i \in Y_i$ the i'th component of $\mathbf{y}$.

A single-target learner learns from a dataset $S = \{(\mathbf{x}, y_i)\}$, with $y_i \in Y_i$ a scalar variable, a function $f_i : X \to Y_i$ such that $\sum_{(\mathbf{x},y_i)\in S} L_i(f_i(\mathbf{x}), y_i)$ is minimized, with L_i some loss function over Y_i.

A multi-target learner learns from a dataset $S = \{(\mathbf{x}, \mathbf{y})\}$, with $\mathbf{y} \in Y$ an n-dimensional vector, a function $F : X \to Y$ such that $\sum_{(\mathbf{x},\mathbf{y})\in S} L(F(\mathbf{x}), \mathbf{y})$ is minimized, with L a loss function over Y. Assume that L is monotonically increasing in each of the L_i (i.e., whenever one L_i increases while the other $L_{(\cdot)}$ remain constant, L increases too). For example, $L(\mathbf{y}, \mathbf{y}') = \sum_i L_i(y_i, y_i')$.

It has been shown that by using multi-target learners, better predictive performance for the targets, on average, can be obtained [6]. That is, for any $(\mathbf{x}, \mathbf{y})$ drawn randomly from the population, on average, $L(F(\mathbf{x}), \mathbf{y}) < L([f_1(\mathbf{x}), \ldots, f_n(\mathbf{x})], \mathbf{y})$.

Under the monotonicity assumption mentioned above, obtaining better predictive performance on average implies that there must be individual targets for which the predictive performance, as measured on this single target, must improve. That is, there must be at least one i for which $L_i(F_i(\mathbf{x}), y_i) < L_i(f_i(\mathbf{x}), y_i)$, with $F_i(\mathbf{x})$ the i'th component of $F(\mathbf{x})$. This observation leads to the question whether single-target models could be improved by following the multi-target approach. That is: even when there is only one single target that we want to predict, we may be able to build a better model for predicting that target if we can exploit the information present in other, related, variables.

Thus, the problem setting becomes as follows. We are given a training set $S = \{(\mathbf{x}, \mathbf{y})\}$, and are interested in predicting y_n from $\mathbf{x}$. The variables $y_i, i \neq n$ need not be predicted, and will not be available at prediction time, but we can use them during the learning phase. We call this setting the single-target setting with support targets: one single target y_n (the main target) needs to be predicted, but a number of additional support targets $y_i, i \neq n$ are available at induction time, and can be used to improve the model for y_n.

Note the following important point: while the support targets are used during learning, they are not assumed to be available during the prediction phase. If they were, then an alternative to the method proposed here would be to learn a model that also uses the support targets as inputs (e.g., in the case of decision tree learning, to learn a tree that is allowed to test these attributes). It is quite likely that that would lead to better prediction. The model that we will try to learn here, is one that will make predictions without having that information; we use the support targets to learn a model that maps $\mathbf{x}$ onto a target attribute more accurately, even though the model itself has no access to the support targets.

3 Related Work

We first discuss known algorithms for multi-target prediction. After that, we turn to methods that selectively exploit transfer by partitioning the target variables. Finally, we discuss methods that explicitly assume transfer to be asymmetric.

Probably the most influential work on multi-target prediction is that by Caruana [6]. He proposes a neural network based approach where the different target variables are outputs of one single network. Considering the network as containing n different predictive models (one for each output), one could say that these models share the connection weights between the input layer and the hidden layer, but the weights between the hidden and output layers are particular to each individual output. In this way, the network finds a balance between modeling the shared properties of the different targets and modeling their particularities.

Besides neural networks, many other predictive models have been extended to multi-target prediction as well. This includes nearest neighbor methods [6, 8], kernel methods [12], Bayesian approaches [13], logistic regression [14], Gaussian processes [15], and decision trees [16].

Thrun & O'Sullivan [8] explicitly consider the fact that among the target variables, some may be related while others may be unrelated. Hence, better predictive performance may be obtained if multi-target models are built that only include those variables that are indeed related. To this aim, they first cluster the target variables, and then learn a separate multi-target model for each cluster. The variables are clustered based on an empirical measure of relatedness. More recently, also Bakker & Heskes [13], Xue et al. [14], and Evgeniou et al. [12] have proposed methods that are based on clustering targets.

As we will show with an example, transfer may be asymmetric and methods that cluster targets may therefore be suboptimal. The following two approaches assume transfer to be asymmetric and also consider the single-target prediction with support targets setting. Nevertheless, they don't measure transfer empirically. They are also not directly applicable to decision trees.

Silver & Mercer [9] build on the work of Caruana, but use a different learning speed (a parameter of the back-propagation algorithm) for each target in the neural network. They set the learning speed of the support targets based on their relatedness to the main target. In later work, they compare a number of relatedness measures, such as correlation and mutual information, to control the learning speeds [17].

Kaski & Peltonen [11] propose a probabilistic model for each support target that is a mixture of the main target's model and a target specific model. They set the parameters of these mixture models to minimize the conditional log likelihood summed over all targets. The idea is that the support target specific models "explain away" the irrelevant data, and that the relevant data available for the support targets helps improve the model for the main target. The approach is validated with logistic regression models as base models.

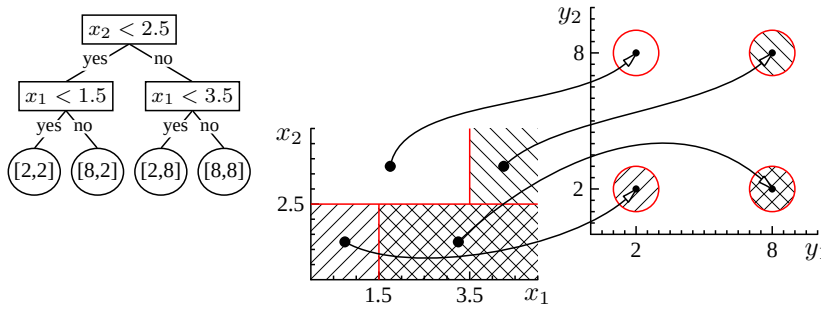

Fig. 1. A multi-target regression tree together with its mapping from the input to the target space. Multi-target regression trees work well if the training data maps hyper-rectangles in the input space to compact clusters in the target space.

4 Single-Target Prediction with Multi-Target Trees

In this section, we briefly describe multi-target decision trees; we will use these as multi-target models in our experiments. Then, we study inductive transfer empirically for decision trees by constructing a so-called transfer matrix.

4.1 Multi-Target Decision Trees

Most descriptions of decision tree learning assume a scalar target, which may be nominal (classification) or numerical (regression). Blockeel et al. [16] argued that decision tree learning can easily be extended towards the case of multi-target prediction, by extending the notion of class entropy or variance towards the multi-dimensional case. They define the variance of a set as the mean squared distance between any element of the set and the centroid of the set. Depending on the definition of distance, which could be Euclidean distance in a multi-dimensional target space, a decision tree will be built that gives accurate predictions for multiple target variables (Fig. 1 shows an example).

Similar to other multi-target models, it has been shown that multi-target trees can be more accurate than single-target trees. This holds for both multi-label classification trees [5] and for multi-objective regression trees [18].

4.2 The Transfer Matrix

To gain insight in the effect of applying multi-target models to single-target prediction, we construct a transfer matrix [8] $C = (c_{i,j})$, where $c_{i,j}$ is the expected gain in predictive performance for target y_i that the two-target model with targets y_i and y_j yields over the single-target model for y_i. In other words, $c_{i,j}$ indicates if inductive transfer from y_j to y_i is useful. We call $c_{i,j}$ the transfer from y_j to y_i.

Table 1 shows the transfer matrix for one of the datasets that we will use in the experimental evaluation. The multi- and single-target models on which the

Table 1. (a) Transfer matrix for the Soil Quality 1, Collembola groups dataset (a subset of S_2, see experimental setup). Cases where transfer is asymmetric are in italic. (b) Correlation matrix for this dataset.

(a)

$j \setminus i$	1	2	3	4	5
1	0	0.01	0.04	0.13	0
2	0	0	0.06	0.13	0
3	0.07	0.01	0	*0.13*	0
4	0.03	0	*-0.02*	0	*-0.01*
5	0.1	0.03	0.31	*0.18*	0

(b)

$j \setminus i$	1	2	3	4	5
1	1	0.03	0.12	0.05	0.07
2	0.03	1	0.08	0.27	0.45
3	0.12	0.08	1	0.31	0.14
4	0.05	0.27	0.31	1	0.19
5	0.07	0.45	0.14	0.19	1

matrix is based are regression trees. The matrix elements are the relative differences in 10-fold cross-validated Pearson correlation (a performance measure that is often used in regression problems) between the multi- and single-target model for y_i, averaged over 5 runs with different random folds; each row corresponds to a different support target y_j in the two-target model.

From the transfer matrix, we see that transfer is an asymmetric quantity: it is possible that y_i can be predicted more accurately if y_j is included as support target ($c_{i,j} > 0$), whereas the prediction for y_j actually deteriorates when y_i is included ($c_{j,i} < 0$).

While empirically measuring transfer as defined above is the most direct way of deciding which support target to use for predicting a given target, two important approximations to this approach have been used in previous work: (a) converting transfer into a symmetric quantity [8], and (b) using other measures, such as correlation, to somehow approximate transfer [17]. The advantage of such approximations is that they reduce the computational cost of the approach. For example, by combining (a) and (b), one could use the pairwise linear correlation between the targets to cluster the targets and then build a multi-target model for each cluster in the partition.

The disadvantage of (a) is that, because transfer is really asymmetric, replacing it by a symmetric approximation will result in suboptimal models for certain targets. Consider again Table 1. Clustering y_3 and y_4 together will result in a suboptimal model for y_3, while putting y_3 and y_4 in different multi-target models may result in a suboptimal model for y_4.

The disadvantage of (b) is that correlation is often not a good approximation for transfer. Table 1 shows the targets' correlation matrix. y_2 and y_5 have the highest correlation. Nevertheless, the transfer from y_2 to y_5 is zero and that from y_5 to y_2 is small. One reason is that transfer does not only depend on the values of the target variables. It also depends on other factors, such as the mapping that the data represents between input and output. Measures that only depend on the targets may therefore poorly approximate transfer. Fig. 1 illustrates this: the correlation between the targets is zero, yet both targets can be predicted accurately with the same tree structure. Therefore, we expect that data for both targets will be beneficial to finding this structure, and that $c_{1,2}$ and $c_{2,1}$ are positive.

Algorithm 1 Empirical Asymmetric Selective Transfer (EAST).

input: dataset S, main target $t = y_n$, candidate support targets $T_s = \{y_i \mid i \neq n\}$.
$T^{(0)} := \{t\}$
$L^{(0)} :=$ cross-validate($T^{(0)}$, L_n, S)
for $i := 1$ **to** $n - 1$
    $L^{(i)} := \infty$
    for each $t_s \in (T_s - T^{(i-1)})$
        $L :=$ cross-validate($T^{(i-1)} \cup \{t_s\}$, L_n, S)
        if $L < L^{(i)}$
            $L^{(i)} := L$
            $T^{(i)} = T^{(i-1)} \cup \{t_s\}$
$i^* := \text{argmin}_{i \in \{0,\ldots,(n-1)\}} L^{(i)}$
return induce($T^{(i^*)}$, S)

These two disadvantages are alleviated by our approach, which we discuss next.

5 Empirical Asymmetric Selective Transfer

Since addition of extra support targets may increase the predictive accuracy for the main target, but is not guaranteed to do so, and moreover some target variables may help while others are detrimental, we can consider the following procedure: add extra target variables one by one, always selecting that target variable that results in the best model.

How can we select the "best" target to add to our current support set? As explained before, any measure that is symmetric or takes only relations among the target values into account will be suboptimal. Therefore, we directly measure the increase in predictive performance that a candidate support target yields using (internal) cross-validation. This takes into account all possible effects of including a certain support target.

Our "Empirical Asymmetric Selective Transfer (EAST)" procedure is described in Algorithm 1. It essentially implements the approach outlined above. The internal loop finds the next best candidate support target that can be added to the multi-target model. To do so, it calls the procedure cross-validate(T, L_n, S), which computes the 10-fold cross-validated loss L_n with regard to the main target y_n of a multi-target model with targets T constructed from S. The outer loop repeats this process until the support set is equal to the set of all targets. In the end, the algorithm returns the best support set found in this way.

The computational cost of EAST compares as follows to the execution time required for building a single-target decision tree (T_{ST}). Iteration i of EAST's outer loop costs $9 \cdot T_{\text{ST}} \cdot (i+1) \cdot (n-i)$, because it tries $(n-i)$ candidate support targets and cross-validates for each candidate a $(i+1)$-target tree. Building one single $(i+1)$-target tree costs $\approx T_{\text{ST}} \cdot (i+1)$; cross-validating it costs 9 times more (10 folds, each with a training set of $0.9 \cdot |S|$). $(n-1)$ iterations of EAST

therefore cost $9 \cdot T_{\mathrm{ST}} \sum_{i=1}^{(n-1)}(i+1)(n-i)$, i.e., EAST is a factor $O(n^3)$ slower than building a single-target decision tree.

In our experiments, EAST's runtime proved to be acceptable because T_{ST} was relatively small. For example, for a dataset with 9 targets, EAST took on average 25 minutes; a factor 290 slower than T_{ST}. For large datasets, an alternative is to replace the cross-validation in EAST's internal loop by a single train/test split. This modification would make the algorithm about a factor 10 faster.

6 Experimental Evaluation

The aim of our experiments is to test to which extent EAST, for a given main target, succeeds in finding a good set of support targets. We do this by comparing EAST to two common baseline models: a single-target model for the main target (ST) and a multi-target model that includes all targets (MT). We also compare to the TC algorithm by Thrun & O'Sullivan [8], which we briefly describe next.

6.1 The TC Algorithm

We compare EAST to the task (target) clustering algorithm (TC) by Thrun & O'Sullivan [8], because clustering of targets is a straightforward and frequently used approach to exploit transfer selectively; TC is also quite general and can easily be used with multi-target decision trees as base models.

TC exhaustively searches for the clustering $C_1, \ldots, C_K$ of targets that maximizes $\frac{1}{n}\sum_{k=1}^{K}\sum_{y_i \in C_k}\frac{1}{|C_k|}\sum_{y_j \in C_k} c_{i,j}$, with $c_{i,j}$ the transfer from y_j to y_i. Next, it builds a multi-target model for each cluster C_k. As in EAST, transfer is estimated empirically using cross-validation (but TC only estimates the pairwise transfer between two targets, while EAST compares candidate support sets of arbitrary size).

The number of possible partitions grows exponentially with the number of targets. As a result, TC quickly becomes computationally infeasible, even for moderate numbers of targets. For example, computing all partitions for 9 targets took 4.5 minutes; for the 39 targets in dataset S_3 (Table 2) this would take about 2 years.

An alternative and faster approach would be to use an approximate method to compute the clustering, such as hierarchical agglomerative clustering. We chose not to pursue this because this was also not done by Thrun & O'Sullivan (they consider a relatively small number of tasks) [8]. An exact method is also a stronger baseline to compare our approach to.

6.2 Experimental Procedure

The datasets that we use are listed, together with their properties, in Table 2. Many datasets are of ecological nature. We omit the description of each dataset; the interested reader can find details in Ženko [19]. Each dataset represents a

Table 2. Dataset properties. Datasets S_1 to S_6 are regression tasks, the remaining ones are classification tasks. N is the number of examples, $|\mathbf{x}|$ the number of input variables, and $|\mathbf{y}|$ is the number of target variables.

(a)

	Dataset	N	$\|\mathbf{x}\|$	$\|\mathbf{y}\|$
S_1	Sigmea Real	817	4	2
	Soil Quality 1	1944	139	
S_2	Acari/Coll. groups	"	"	9
S_3	Coll. species	"	"	39
S_4	Soil Quality 2	393	48	3
	Water quality	1060		
S_5	Plants/Animals	"	16	14
S_6	Chemical	"	836	16

(b)

	Dataset	N	$\|\mathbf{x}\|$	$\|\mathbf{y}\|$
S_7	Mediana	7953	78	5
S_8	Bridges	104	7	5
S_9	Monks	432	6	3
S_{10}	Thyroid	9172	29	7

multi-target regression or classification task, and the number of targets varies from 2 to 39.

EAST has been implemented in the decision tree induction system Clus, which is available as open source software from `http://www.cs.kuleuven.be/~dtai/clus`. Clus also implements single- and multi-target decision trees, so all results that we present next are obtained with the same system and parameter settings. All parameters are set to their default values. To avoid overfitting we prune the trees using CART validation set based pruning, i.e., we use 70% training data for tree building and 30% for pruning (as suggested by Torgo [20]). We normalize each numerical target to zero mean and unit variance.

We compare for each dataset and target variable, the predictive performance of a traditional single-target tree (ST), a tree constructed by EAST with all other targets as candidate support targets, a multi-target tree including all targets (MT), and a multi-target tree from the clustering created by TC for datasets where this is computationally feasible. We estimate predictive performance as the 10-fold cross-validated misclassification error (for classification tasks) or Pearson correlation (for regression tasks). Correlation is often used as evaluation measure in ecological modelling. To compare EAST to ST, MT, and TC we count the number of targets on which it performs better (wins) and the number on which it performs worse (losses) and apply the sign test. Besides strict wins and losses, we also report significant wins and losses; to this end we use the corrected paired t-test [21] with significance level 0.05.

6.3 Results and Discussion

Table 3 gives an overview of the results. EAST outperforms the three other algorithms. The sign test applied to the wins/losses counts shows that EAST is significantly better than ST ($p = 0.0003$) and TC ($p = 0.029$). At the 5% level, it is just not significantly better than MT ($p = 0.057$). This result is mainly due to the losses on dataset S_3. If we would disregard this dataset, then EAST is clearly better than MT ($p = 0.0003$).

Table 3. Pairwise comparison of methods. For each pair A/B, the number of targets are given that represent (significant) wins and losses for A when compared to B. Entries marked n/a were computationally infeasible (see Section 6.1).

	Dataset	EAST/ST				EAST/MT				EAST/TC			
		#win		#loss		#win		#loss		#win		#loss	
S_1	Sigmea Real	1	(0)	0	(0)	0	(0)	0	(0)	0	(0)	1	(0)
S_2	SQ1 - Acari/Coll. groups	7	(0)	2	(0)	3	(1)	6	(0)	8	(0)	1	(0)
S_3	SQ2 - Coll. species	25	(4)	8	(0)	15	(0)	24	(1)	n/a			
S_4	Soil Quality 2	2	(0)	1	(0)	2	(0)	1	(0)	1	(0)	2	(0)
S_5	WQ - Plants/Animals	8	(1)	6	(0)	9	(1)	5	(0)	n/a			
S_6	WQ - Chemical	11	(3)	5	(0)	14	(3)	2	(0)	n/a			
S_7	Mediana	3	(2)	2	(0)	4	(0)	1	(0)	4	(0)	1	(0)
S_8	Bridges	2	(2)	3	(0)	5	(0)	0	(0)	4	(0)	1	(0)
S_9	Monks	1	(0)	0	(0)	2	(1)	0	(0)	0	(0)	1	(0)
S_{10}	Thyroid	5	(2)	2	(0)	6	(1)	1	(1)	5	(0)	2	(0)
	Total	65	(14)	29	(0)	60	(7)	40	(2)	22	(0)	9	(0)

We conjecture that the losses of EAST compared to MT on S_3 are the result of a form of overfitting due to the many targets in this dataset and high variance of the performance estimates. EAST loses to MT when its internal cross-validation incorrectly estimates the performance of a subset that does not include all other targets higher than that of the subset corresponding to MT. The chance that this happens depends on the number of targets (more targets implies more candidate subsets, which in turn implies a higher chance that at least one of these is incorrectly estimated as better than MT) and on the variance of the performance estimates obtained in EAST's internal cross-validation. While this kind of overfitting is unavoidable (it can happen in general when performing model selection), we expect it to remain small and expect few significant losses. This is confirmed by the results in Table 3.

Table 4 shows detailed results for dataset S_6. First consider the comparison ST versus MT. ST works best on 8 targets and MT works best on 7, so neither is a clear winner. This illustrates again the advantage of selective transfer. For some targets, using all other targets as support targets may be best, while for others using no support targets may be best. EAST successfully discovers this and may find a non-trivial subset that performs even better. This is confirmed by the results: EAST outperforms the best of ST and MT on 10 targets.

7 Conclusions and Future Work

This paper addresses the single-target with support targets prediction task, where the goal is to build a model for the main target (or one model for each target in case of multiple targets), and where a number of candidate support targets are available (only) at model induction time, which may carry information about the main target. The paper's chief contribution is Empirical Asymmetric

Table 4. Cross-validated Pearson correlation for EAST, ST, and MT for each target t of dataset S_6 *WQ - Chemical*. •, ○ denote a statistically significant improvement or degradation of EAST when compared to ST or MT. ■,□ denote a statistically significant improvement or degradation of MT when compared to ST.

t	EAST	ST	MT	t	EAST	ST	MT
1	0.50±0.07	0.51±0.07	0.11±0.17•□	9	0.26±0.15	0.22±0.19	0.06±0.16□
2	0.34±0.09	0.36±0.09	0.22±0.12•□	10	0.44±0.22	0.11±0.21•	0.34±0.20■
3	0.42±0.10	0.38±0.14	0.31±0.16•	11	0.35±0.18	0.11±0.20•	0.32±0.12■
4	0.41±0.10	0.39±0.06	0.38±0.16	12	0.49±0.12	0.51±0.07	0.42±0.18
5	0.32±0.13	0.32±0.15	0.37±0.16	13	0.26±0.14	0.20±0.16	0.20±0.11
6	0.37±0.16	0.16±0.15•	0.24±0.15	14	0.39±0.23	0.46±0.22	0.43±0.16
7	0.38±0.14	0.42±0.09	0.26±0.17 □	15	0.48±0.21	0.38±0.24	0.44±0.16
8	0.27±0.13	0.19±0.20	0.26±0.16	16	0.56±0.22	0.51±0.16	0.52±0.16

Selective Transfer (EAST), an algorithm that approximates the subset of support targets that, when predicted together with the main target in a multi-target model, maximally improves predictive performance of the main target.

Experiments show that EAST, on top of a multi-target decision tree learner, outperforms single-target decision trees, multi-target decision trees, and multi-target decision trees with target clustering.

We would like to address a few questions in future work. First, we will analyze in more depth to which degree the subset selected by EAST approximates the optimal support set. This analysis will include experiments to gain more insight in the overfitting behavior of EAST. Second, it is not clear if different multi-target learners can exploit the additional information available in other targets equally well. Therefore, we will test EAST in combination with different base-learners. This will give us more insight in the behavior of selective inductive transfer with those base-learners.

Acknowledgments. B. Piccart is supported by project G.0255.08 "Efficient microprocessor design using machine learning" funded by the Research Foundation - Flanders (FWO-Vlaanderen). J. Struyf and H. Blockeel are post-doctoral fellows of the Research Foundation - Flanders (FWO-Vlaanderen). The authors are grateful to S. Džeroski and B. Ženko for providing the datasets. This research utilizes the high performance computational resources provided by the K.U. Leuven (`http://ludit.kuleuven.be/hpc`).

References

1. Blockeel, H., Džeroski, S., Grbović, J.: Simultaneous prediction of multiple chemical parameters of river water quality with TILDE. In: 3rd European Conf. on Principles of Data Mining and Knowledge Discovery. (1999) 32–40
2. Demšar, D., Džeroski, S., Larsen, T., Struyf, J., Axelsen, J., Bruus Pedersen, M., Henning Krogh, P.: Using multi-objective classification to model communities of soil microarthropods. Ecol. Model. **191**(1) (2006) 131–143

3. Clare, A., King, R.: Knowledge discovery in multi-label phenotype data. In: 5th European Conf. on Principles of Data Mining and Knowledge Discovery. (2001) 42–53
4. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. J. Mach. Learn. Res. **6** (2005) 1453–1484
5. Blockeel, H., Schietgat, L., Struyf, J., Džeroski, S., Clare, A.: Decision trees for hierarchical multilabel classification: A case study in functional genomics. In: 10th European Conf. on Principles and Practice of Knowledge Discovery in Databases. (2006) 18–29
6. Caruana, R.: Multitask learning. Mach. Learn. **28** (1997) 41–75
7. Jensen, D., Neville, J., Gallagher, B.: Why collective inference improves relational classification. In: 10th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. (2004) 593–598
8. Thrun, S., O'Sullivan, J.: Discovering structure in multiple learning tasks: The TC algorithm. In: 13th Int'l Conf. on Machine Learning. (1996) 489–497
9. Silver, D., Mercer, R.: The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness. Connect. Sci. **8**(2) (1996) 277–294
10. Rosenstein, M.T., Marx, Z., Kaelbling, L.P., Dietterich, T.G., Whistler, B.: To transfer or not to transfer. In: NIPS'05 Workshop on Transfer Learning. (2005) 4 pages.
11. Kaski, S., Peltonen, J.: Learning from relevant tasks only. In: 18th European Conf. on Machine Learning. (2007) 608–615
12. Evgeniou, T., Micchelli, C.A., Pontil, M.: Learning multiple tasks with kernel methods. J. Mach. Learn. Res. **6** (2005) 615–637
13. Bakker, B., Heskes, T.: Task clustering and gating for Bayesian multitask learning. J. Mach. Learn. Res. **4** (2003) 83–99
14. Xue, Y., Liao, X., Carin, L., Krishnapuram, B.: Multi-task learning for classification with Dirichlet process priors. J. Mach. Learn. Res. **8** (2007) 35–63
15. Yu, K., Tresp, V., Schwaighofer, A.: Learning Gaussian processes from multiple tasks. In: 22nd Int'l Conf. on Machine Learning. (2005) 1012–1019
16. Blockeel, H., De Raedt, L., Ramon, J.: Top-down induction of clustering trees. In: 15th Int'l Conf. on Machine Learning. (1998) 55–63
17. Silver, D.L., Mercer, R.E.: Selective functional transfer: Inductive bias from related tasks. In: IASTED Int'l Conf. on Artificial Intelligence and Soft Computing. (2001) 182–189
18. Struyf, J., Džeroski, S.: Constraint based induction of multi-objective regression trees. In: Knowledge Discovery in Inductive Databases, 4th Int'l Workshop, Revised, Selected and Invited Papers. (2006) 222–233
19. Ženko, B.: Learning predictive clustering rules. PhD thesis, University of Ljubljana, Slovenia (2007)
20. Torgo, L.: Error estimators for pruning regression trees. In: 10th European Conf. on Machine Learning. (1998) 125–130
21. Nadeau, C., Bengio, Y.: Inference for the generalization error. Mach. Learn. **52** (2003) 239–281