Multi-Label Classification Methods for Multi-Target Regression

Eleftherios Spyromitros-Xioufis[1], Grigorios Tsoumakas[1], William Groves[2], and Ioannis Vlahavas[1]

[1] Department of Informatics, Aristotle University of Thessaloniki, Greece
{espyromi,greg,vlahavas}@csd.auth.gr
[2] Department of Computer Science and Engineering, University of Minnesota, USA
groves@cs.umn.edu

Abstract. Real world prediction problems often involve the simultaneous prediction of multiple target variables using the same set of predictive variables. When the target variables are binary, the prediction task is called multi-label classification while when the target variables are real-valued the task is called multi-target regression. Although multi-target regression attracted the attention of the research community prior to multi-label classification, the recent advances in this field motivate a study of whether newer state-of-the-art algorithms developed for multi-label classification are applicable and equally successful in the domain of multi-target regression. In this paper we introduce two new multi-target regression algorithms: multi-target stacking (MTS) and ensemble of regressor chains (ERC), inspired by two popular multi-label classification approaches that are based on a single-target decomposition of the multi-target problem and the idea of treating the other prediction targets as additional input variables that augment the input space. Furthermore, we detect an important shortcoming on both methods related to the methodology used to create the additional input variables and develop modified versions of the algorithms (MTSC and ERCC) to tackle it. All methods are empirically evaluated on 12 real-world multi-target regression datasets, 8 of which are first introduced in this paper and are made publicly available for future benchmarks. The experimental results show that ERCC performs significantly better than both a strong baseline that learns a single model for each target using bagging of regression trees and the state-of-the-art multi-objective random forest approach. Also, the proposed modification results in significant performance gains for both MTS and ERC.

Keywords: multi-target regression, multi-output regression, multivariate regression, multi-label classification, regressor chains, stacking

1 Introduction

Learning from multi-label data has recently received increased attention by researchers working on machine learning and data mining for two main reasons.

The first one is the ubiquitous presence of multi-label data in application domains ranging from multimedia information retrieval to tag recommendation, query categorization, gene function prediction, medical diagnosis, drug discovery and marketing. The other reason is a number of challenging research problems involved in multi-label learning, such as dealing with label rarity, scaling to large number of labels and exploiting label relationships (e.g. hierarchies), with the most prominent one being the explicit modeling of label dependencies [11].

Multi-label learning is closely related to multi-target regression, also known as multivariate or multi-output regression, which aims at predicting multiple real-valued target variables instead of binary ones. Despite that multi-target regression is a less popular task, it still arises in several interesting domains, such as predicting the wind noise of vehicle components [22], stock price prediction and ecological modeling [20]. Multi-label learning is often treated as a special case of multi-target regression in statistics [18]. However, we could more precisely state that both are instances of the more general learning task of predicting multiple targets, which could be real-valued, binary, ordinal, categorical or even of mixed type. The baseline approach of learning a separate model for each target applies to both learning tasks. Furthermore, there exist techniques, such as [5,28,23], that can naturally handle both tasks. Most importantly, they share the same core challenge of modeling dependencies among the different targets.

Given this tight connection between these two learning tasks, it would be interesting to investigate whether recent advances in the more popular multi-label learning task can be successfully transferred to multi-target regression. In particular, this paper adapts two multi-label learning methods that successfully model label dependencies [15,27] to multi-target regression. The results of an empirical evaluation on several real-world datasets, some firstly introduced here, show that the benefits of these two multi-label algorithms apply also to the multi-target regression setting.

Furthermore, by studying the relationship between these two learning tasks, this work aims to increase our understanding in their shared challenge of modeling target dependencies and widen the applicability of existing specialized techniques for either task. This kind of abstraction of key ideas from solutions tailored to related problems offers additional advantages, such as improving the modularity and conceptual simplicity of learning techniques and avoiding reinvention of the same solutions[3].

The rest of the paper is organized as follows: Section 2 discusses related work in the field of multi-label classification and gives some insight into which multi-label classification methods would be more appropriate for multi-target regression. Section 3 gives detailed descriptions of the proposed methods and Section 4 presents the evaluation methodology and introduces the datasets used in the empirical evaluation. The experimental results are shown in Section 5 and finally, Section 6 concludes our study and points to directions for future work.

[3] See the motivation of the NIPS 2011 workshop on relations among machine learning problems at `http://rml.anu.edu.au/`

2 From Multi-Label Classification to Multi-Target Regression

Multi-label learning methods are often categorized into those that *adapt* a specific learning approach (e.g. k nearest neighbors, decision tree, support vector machine) for handling multi-label data and those that *transform* the multi-label task into one or more single-label tasks that can be solved with off-the-shelf learning algorithms [31]. The latter can be further categorized to those that model single labels, pairs of labels and sets of labels [36].

Approaches that model single labels include the typical one-versus-all (also known as binary relevance) baseline, methods based on stacked generalization [15,30,7] and the classifier chains algorithm [27,10]. Such approaches are almost straightforward to adapt to multi-target regression by employing a regression instead of a classification algorithm. It is this kind of approaches that we extend in this work.

Approaches that model pairs of labels [14] follow the paradigm of the one-versus-one decomposition for using binary classifiers on multi-class learning tasks. This concept however is not transferable to multi-target regression. The same holds for approaches that consider sets of labels as different values of a single-label multi-class task [26,32].

The extension of multi-label algorithm adaptation methods for handling multi-target data mainly depends on how easy it is for the underlying learning algorithm to handle classification and regression data interchangeably. For example, decision trees can handle both classification and regression data through different functions for calculating the impurity of internal nodes and the output of leaves. It thus comes as no surprise that there exist decision tree algorithms for both multi-label classification [8] and multi-target regression [3], with the most representative and well-developed ones being based on the predictive clustering trees framework [5,21]. It is interesting to note that the predictive clustering framework is an example of a technique that originally focused on multi-target regression, and only recently showcased its effectiveness on multi-label classification [34,24].

Several multi-label algorithm adaptation methods are based on the definition and optimization of alternative loss functions, compared to the algorithms they extend. For example, the core idea in [37] was the optimization of a ranking loss function that takes into account label pairs, instead of the typical logistic loss of neural networks that looks at individual labels. Similarly, [9] defines a family of online algorithms based on different loss functions considering label relationships. To the best of our knowledge, this type of reasoning for algorithm design has not been transferred to multi-target regression. We believe that it could be an interesting avenue to investigate. For example, consider an application of food sales prediction [38], in particular pastry sales prediction for a patisserie in order to minimize the amount of pastries with short expiration date that are thrown away. In this application, we may like to minimize the sum of prediction errors, but we might also want to minimize the maximum individual prediction error, to avoid for example an early run-out of any of the pastries.

Finally, we would like to note that there exist approaches, such as [28,23], that have recognized this dual applicability (classification and regression) of their multi-target approach and have given a general formulation of their key ideas.

3 Methods

We first formally describe the multi-target regression task and provide the notation that will be used subsequently for the description of the methods. Let X and Y be two random vectors where X consists of d input variables $X_1, .., X_d$ and Y consists of m target variables $Y_1, .., Y_m$. We assume that samples of the form (x, y) are generated iid by some source according to a joint probability distribution $P(X, Y)$ on $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X} = R^d$ and $\mathcal{Y} = R^m$ are the domains of X and Y and are often referred to as the input and the output space. In a sample (x, y), $x = [x_1, .., x_d]$ is the input vector and $y = [y_1, .., y_m]$ is the output vector which are realizations of X and Y respectively. Given a set $D = \{(x^1, y^1), .., (x^n, y^n)\}$ of n training examples, the goal in multi-target regression is to learn a model $h : \mathcal{X} \rightarrow \mathcal{Y}$ which given an input vector x^q, is able to predict an output vector $\hat{y}^q = h(x^q)$ which best approximates the true output vector y^q.

In the baseline Single-Target (ST) method, a multi-target model h is comprised of m single-target models $h_j : \mathcal{X} \rightarrow R$ where each model h_j is trained on a transformed training set $D_j = \{(x^1, y_j^1), .., (x^n, y_j^n)\}$ to predict the value of a single target variable Y_j. This way, target variables are predicted independently and potential relations between them cannot be exploited.

3.1 Multi-Target Stacking (MTS)

The first method that we consider is inspired by [15] where the idea of stacked generalization was applied in a multi-label classification context. That method works by expanding the original input space of each training example with m additional binary variables, corresponding to the predictions of m binary classifiers, one for each label, for that example. This expanded training set is fed to a second layer of binary classifiers that produce the final decisions. The second-layer classifiers can exploit label dependencies. Variations of this core idea appear in [7,30,2].

Here, we adapt this method for multi-target regression and denote it as Multi-Target Stacking (MTS). The training of MTS consists of two stages. In the first stage, m independent single-target models $h_j : \mathcal{X} \rightarrow R$ are learned as in ST. However, instead of directly using these models for prediction, MTS involves an additional training stage where a second set of m meta models $h_j^* : \mathcal{X} \times R^{m-1} \rightarrow R$ are learned, one for each target Y_j. Each meta model h_j^* is learned on a transformed training set $D_j^* = \{(x^{*1}, y_j^1), .., (x^{*n}, y_j^n)\}$, where $x_j^{*i} = [x_1^i, .., x_n^i, \hat{y}_1^i, .., \hat{y}_{j-1}^i, \hat{y}_{j+1}^i, .., \hat{y}_m^i]$ are expanded input vectors consisting of the original input vectors of the training examples augmented by $m - 1$ predictions (estimates) of the rest of the target variables obtained by the first stage models. We intentionally differentiate slightly our method from the multi-label

method to the point of not including the predictions of the first stage model for target variable Y_j in the input space of the second stage model for this variable as in [2], since this would add redundant information to the second stage models. To obtain predictions for an unknown instance x^q, the first stage models are first applied and an output vector $\hat{y}^q = [\hat{y}^q_1, .., \hat{y}^q_m] = [h_1(x^q), .., h_m(x^q)]$ is obtained. Then the second stage models are applied on a transformed input vector $x^{*q}_j = [x^q_1, .., x^q_n, \hat{y}^q_1, .., \hat{y}^q_{j-1}, \hat{y}^q_{j+1}, .., \hat{y}^q_m]$ to produce the final output vector $\hat{y}^q_j = [h^*_1(x^{*q}), ..., h^*_m(x^{*q})]$.

3.2 Ensemble of Regressor Chains (ERC)

The second method that we consider is inspired by the recently proposed Classifier Chains (CC) method [27] and we henceforth denote it as Regressor Chains (RC). RC is based on the idea of chaining single-target models. The training of RC consists of selecting a random chain (permutation) of the set of target variables and then building a separate regression model for each target. Assuming that the default chain $C = \{Y_1, Y_2, .., Y_m\}$ (where C is represented as an ordered set) is selected, the first model concerns the prediction of Y_1, has the form $h_1 : \mathcal{X} \rightarrow R$ and is the same as the model built by the ST method for this target. The difference in RC is that subsequent models $h_{j,j>1}$ are trained on transformed datasets $D^*_j = \{(x^{*1}_j, y^1_j), .., (x^{*n}_j, y^n_j)\}$, where $x^{*i}_j = [x^i_1, .., x^i_1, y^i_1, .., y^i_{j-1}]$ are transformed input vectors consisting of the original input vectors of the training examples augmented by the actual values of all previous targets in the chain. Thus, the models built for targets $Y_{j,j>1}$ have the form $h_j : \mathcal{X} \times R^{j-1} \rightarrow R$. Given such a chain of models, the output vector $\hat{y}^q$ of an unknown instance x^q is obtained by sequentially applying the models h_j, thus $\hat{y}^q = [h_1(x^q), h_2(x^{*q}_2), .., h_m(x^{*q}_m)]$ where $x^{*q}_{j,j>1} = [x^q_1, .., x^q_d, \hat{y}^q_1, .., \hat{y}^q_{j-1}]$. Note that since the true values $y^q_1, .., y^q_{j-1}$ of the target variables are not available at prediction time, the method relies on estimates of these values obtained by applying the models $h_1, .., h_{j-1}$.

One notable property of RC is that it is sensitive in the selected chain ordering. The main problem arising from the use of a single random chain, is that targets which appear earlier in a chain cannot model potential statistical relationships with targets appearing later in that chain. Additionally, prediction error is likely to by propagated and amplified along a chain when making predictions for a new test instance. To mitigate these effects, [27] proposed an ensemble scheme called Ensemble of Classifier Chains where a set of k (typically k=10) CC models with differently ordered chains are built on bootstrap samples of the training set and the final predictions come from majority voting. This scheme has proven to consistently improve the accuracy of a single CC in the classification domain. We apply the same idea (without sampling) on RC and compute the final predictions by taking the mean of the k estimates for each target. The resulting method is called Ensemble of Regressor Chains (ERC).

3.3 MTS & ERC Corrected

Both MTS and ERC are based on the same core idea of treating the other prediction targets as additional input variables that augment the original input space. These meta-variables differ from ordinary input variables in the sense that while their actual values are available at training time, they are missing during prediction. Thus, during prediction, both methods have to rely on estimates of these values which come either from ST (in MTS) or from RC (in ERC) models built on the training set. An important question, which is answered differently by each method, is what type of values should be used at training time for these meta-variables. MTS uses estimates of these variables obtained by applying the first stage models on the training examples, while ERC uses the actual values of these variables to train the RC models. In both cases, *a core assumption of supervised learning is violated*: that the training and testing data should be identically and independently distributed. In the MTS case, the in-sample estimates that are used to form the second stage training examples, will typically be more accurate than the out-of-sample estimates that are used at prediction time. The situation is even more problematic in the case of RC where the actual target values are used at training time. In both cases, the underlying learning algorithm is trained using meta-variables that become noisy (or noisier in the MTS case) at prediction time. As a result, the actual importance and relationship of these meta-variables to the prediction target is falsely estimated. The impact that this discrepancy in the distributions has on CC's accuracy was recently studied in [29], where factors such as the length and order of the chain, the accuracy of the binary classifiers and the degree of dependence between the labels were identified. The latter two factors also apply to MTS.

An alternative to the above approaches is using only a part of the training set for learning the first stage ST models (in MTS) or the RC models (in ERC) and then applying these models to the hold-out part in order to obtain out-of-sample estimates of the meta-variables. While the distribution of the estimates obtained using this approach is expected to be more representative of the distribution of the estimates obtained during prediction, it would lead to reduced second stage training sets for MTS as only the examples of the hold-out set can be used for training the second stage models. The same holds for ERC where the chained RC models would be trained on training sets of decreasing size.

Here, we propose modifications of the original MTS and ERC methods that alleviate the aforementioned problem by employing a procedure that resembles that of f-fold cross-validation in order to obtain unbiased out-of-sample estimates of the meta-variables. The cross-validation approach avoids the problem of the hold-out approach as all training examples are used in the second stage ST models of MTS and the RC models of ERC. We expect that compared to using the actual values or in-sample estimates, the cross-validation estimates will approximate the distribution of the estimates obtained at prediction time more accurately and thus will be more useful for the model. The training procedures the proposed corrected versions denoted as *MTS Corrected* (MTSC) and *RC Corrected* (RCC) are outlined in Algorithms 1 and 3. ERCC consists of simply

repeating the RCC procedure k times with k randomly ordered chains. The prediction procedures are the same for all variants of the methods and are presented in Algorithms 2 and 4. In Section 5 we empirically evaluate the original MTS and ERC methods with the proposed corrected versions (MTSC) and (ERCC).

Algorithm 1: MTSC training

Input: Training set D, number of internal cross-validation folds f
Output: 1st and 2nd stage models h_j and h_j^*, $j = 1..m$

Build 1st stage models
for $j = 1$ **to** m **do**
  $D_j = \{(x^1, y_j^1), .., (x^n, y_j^n)\}$ `// perform the ST transformation of` D
  $h_j : D_j \rightarrow R$ `// build an ST model for` Y_j `using full` D_j
  $D_j^* \leftarrow \emptyset$ `// initialize 2nd stage training set for target` Y_j
  split D_j into f disjoint parts $D_j^i, i = 1..f$
  for $i = 1$ **to** f **do**
    $h_j^i : D_j \setminus D_j^i \rightarrow R$ `// build an ST model for` Y_j `using` $D_j \setminus D_j^i$
Generate 2nd stage training sets
for $i = 1$ **to** f **do**
  $D_j^{*i} \leftarrow \emptyset$ `// initialize 2nd stage training set for target` Y_j `and fold` i
  for $j = 1$ **to** m **do**
    foreach $x_j^i \in D_j^i$ **do**
      $x_j^{*i} = x_j^i$
      for $l = 1$ **to** m , $l \neq j$ **do**
        $\hat{y}_l^i = h_l^i(x_j^i)$
        $x_j^{*i} = [x^{*i}, \hat{y}_l^i]$ `// append` x_j^{*i} `with the estimation` $\hat{y}_l^i$
      $D_j^{*i} = D_j^{*i} \cup x_j^{*i}$
  $D_j^* = D_j^* \cup D_j^{*i}$
Build 2nd stage models
for $j = 1$ **to** m **do**
  $h_j^* : D_j^* \rightarrow R$

4 Experimental Setup

This section describes our experimental setup. We first present the participating methods and their parameters and provide details about their implementation in order to facilitate reproducibility of the experiments. Next, we describe the evaluation measure and justify the process that was followed for the statistical comparison of the methods. Finally, we present the datasets that we used, 8 of which are firstly introduced in this paper.

Algorithm 2: MTS prediction

Input: Unknown instance x^q, 1st and 2nd stage models h_j and h^*_j, $j = 1..m$
Output: Output vector $\hat{y}^q$ of second stage models

$\hat{y}^q = \hat{\hat{y}}^q = \mathbf{0}$ `// initialize and the 1st and 2nd stage output vectors`
Apply 1st stage models
for $j = 1$ **to** m **do**
  $\hat{y}^q_j = h_j(x^q)$
$x^{*q} = [x^q, \hat{y}^q]$ `// concatenate` x^q `and the output vector` $\hat{y}^q$
Apply 2nd stage models
for $j = 1$ **to** m **do**
  $\hat{\hat{y}}^q_j = h^*_j(x^{*q})$

Algorithm 3: RCC training

Input: Training set D, number of internal cross-validation folds f
Output: Chained models $h_j, j = 1..m$

Generate D^*_1
$D^*_1 = \{(x^1, y^1_1), .., (x^n, y^n_1)\}$ `// perform the ST transformation of` D `to create` D^*_1
for $j = 1$ **to** m **do**
  $h_j : D^*_j \rightarrow R$ `// build a chained model for` Y_j `using full` D^*_j
  if $j < m$ **then**
    Generate D^*_{j+1}
    $D^*_{j+1} \leftarrow \emptyset$ `// initialize` D^*_{j+1}
    split D^*_j into f disjoint parts $D^{*i}_j, i = 1..f$
    for $i = 1$ **to** f **do**
      $h^i_j : D^*_j \setminus D^{*i}_j \rightarrow R$ `// build a chained model for` Y_j `using` $D^*_j \setminus D^{*i}_j$
      for $x^{*i}_j \in D^{*i}_j$ **do**
        $x^{*i}_{j+1} = x^{*i}_j$ `// initialize` x^{*i}_{j+1}
        $\hat{y}^i_j = h^i_j(x^{*i}_j)$
        $x^{*i}_{j+1} = [x^{*i}_j, \hat{y}^i_j]$ `// append` x^{*i}_{j+1} `with the estimation` $\hat{y}^i_j$
        $D^*_{j+1} = D^*_{j+1} \cup (x^{*i}_{j+1}, y^i_{j+1})$

Algorithm 4: RC prediction

Input: Unknown instance x^q, chained models h_j, $j = 1..m$
Output: Output vector $\hat{y}^q$

$\hat{y}^q = \mathbf{0}$ `// initialize and the output vector`
$x^{*q}_1 = x^q$
for $j = 1..m$ **do**
  $\hat{y}^q_j = h_j(x^{*q}_j)$
  $x^{*q}_{j+1} = [x^{*q}_j, \hat{y}^q_j]$ `// concatenate` x^{*q}_j `and` $\hat{y}^q_j$

4.1 Methods, Parameters and Implementation

The empirical evaluation compares the performance of the basic MTS and ERC methods against the performance of their corrected versions MTSC and ERCC. All the proposed methods are also compared against the ST baseline as well as the state-of-the-art multi-objective random forest algorithm [21] (MORF).

Since all the proposed methods (including ST) transform the multi-target prediction problem into several single-target problems, they have the advantage of being combinable with any single-target regression algorithm. To facilitate a fair comparison and to simplify the analysis we use bagging [6] of 100 regression trees as the base regression algorithm in all methods. MTSC and ERCC are run with $f = 10$ internal cross-validation folds and the ensemble size of ERC and ERCC is set to 10 models, each one trained using a different random chain (in cases where the number of distinct chains is smaller than 10 we create exactly as many models as the number of distinct label chains). In order to ensure a fair comparison of ERC and ERCC with the non-ensemble methods, we do not perform bootstrap sampling (i.e. each ensemble model is build using all training examples). Finally, for MORF we use an ensemble size of 100 trees and the values suggested in [21] for the rest of its parameters.

All the proposed methods and the evaluation framework were implemented within the open-source multi-label learning Java library Mulan[4] [33] by expanding the library to handle multi-target regression tasks. Mulan is built on top of Weka[5] [35], which includes implementations of bagging and regression tree (via the REPTree class). A wrapper of the CLUS[6] software, which includes support for MORF, was also implemented and included in Mulan, enabling the evaluation of all methods under a single software framework. In support of open science and to ease replication of the experimental results of this paper, we have included a class called ExperimentMTR in Mulan's experiments package.

4.2 Evaluation

We use Relative Root Mean Squared Error (RRMSE) to measure the accuracy of a multi-target model on each target variable. The RRMSE of a multi-target model h that has been induced from a train set D_{train} is estimated based on a test set D_{test} according to the following equation:

$$RRMSE(h, D_{test}) = \sqrt{\frac{\sum_{(x,y_j)\in D_{test}} (\hat{y}_j - y_j)^2}{\sum_{(x,y_j)\in D_{test}} (\bar{Y}_j - y_j)^2}} \quad (1)$$

where $\bar{Y}_j$ is the mean value of target variable Y_j over D_{train} and $\hat{y}_j$ is the estimation of $h(x)$ for Y_j. More intuitively, RRMSE for a target is equal to the Root Mean Squared Error (RMSE) for that target divided by the RMSE

[4] `http://mulan.sourceforge.net`
[5] `http://www.cs.waikato.ac.nz/ml/weka`
[6] `http://dtai.cs.kuleuven.be/clus/`

of predicting the average value of that target in the training set. The RRMSE measure is estimated using the hold-out approach for large datasets, while 10-fold cross-validation is employed for small datasets.

To test the statistical significance of the observed differences between the methods, we follow the methodology suggested in [12]. When comparing two methods on multiple datasets we use the Wilcoxon signed-ranks test. When the comparison involves multiple methods we first apply the non-parametric Friedman test that operates on the average ranks of the methods and checks the validity of the hypothesis (null-hypothesis) that all methods are equivalent. If the null-hypothesis is rejected, we proceed to the Nemenyi post-hoc test that computes the critical distance (between average ranks) required in order for two methods to be considered significantly different. Finally, we graphically present the results with appropriate diagrams which plot the average ranks of the methods and show groups of methods whose average rank differences are less than the critical distance for a p-value of 0.05.

As the above methodology requires a single performance measurement for each method on each dataset, it is not directly applicable to multi-target evaluation where we have multiple performance measurements (one for each target) for each method on each dataset. One option is to take the average RRMSE (aRRMSE) across all target variables within a dataset as a single performance measurement. However, this may not always be a meaningful choice since: a) different targets may represent different things and b) we may not be always interested into the best average performance (see patisserie example in Section 2). Another option is to treat the RRMSE performance of each method on each different target as a different measurement. In this case, however, Friedman's test assumption of independence between performance measurements might be violated. In the absence of a better solution, we perform the two dimensional analysis of [1], where statistical tests are conducted using both aRRMSE but also considering the RRMSE value per target as an independent measurement.

4.3 Datasets

Despite the numerous interesting applications of multi-target regression, there are few publicly available datasets of this kind, perhaps because most applications are industrial. In fact, among the datasets used in other multi-target regression studies, e.g. [1], only Solar Flare [4], Water Quality [13] and EDM [19] are publicly available while the rest are proprietary and could not be acquired. This lack of available data, motivated the collection of real-world multi-target regression data and the composition of 8 new benchmark datasets which have been made publicly available[7]. A description of each dataset follows, while Table 1 summarizes their statistics.

EDM The Electrical Discharge Machining dataset [19] represents a two-target regression problem. The task is to shorten the machining time by reproducing

[7] http://users.auth.gr/espyromi/datasets.html

Table 1: Statistics of the datasets used in the evaluation. Those on the right are firstly introduced in this paper. A two-value entry for the Examples column indicates the sample counts for the train and test set respectively, d denotes the number of input variables and m denotes the number of target variables.

Dataset	Examples	d	m	Dataset	Examples	d	m
EDM	154	16	2	OES97	334	263	16
SF1	323	10	3	OES10	403	298	16
SF2	1066	10	3	ATP1d	337	411	6
WQ	1060	16	14	ATP7d	296	411	6
				RF1	4108/5017	64	8
				RF2	4108/5017	576	8
				SCM1d	8145/1658	280	16
				SCM20d	7463/1503	61	16

the behavior of a human operator which controls the values of two variables. Each of the target variables takes 3 distinct numeric values (−1,0,1) and there are 16 continuous input variables.

SF1 & SF2 The Solar Flare dataset [4] has 3 target variables that correspond to the number of times 3 types of solar flare (common, moderate, severe) are observed within 24 hours. There are two versions of this dataset. SF1 contains data from year 1969 and SF2 from year 1978.

WQ The Water Quality dataset [13] has 14 target attributes that refer to the relative representation of plant and animal species in Slovenian rivers and 16 input attributes that refer to physical and chemical water quality parameters.

OES97 & OES10 The Occupational Employment Survey datasets were obtained from years 1997 (OES97) and 2010 (OES10) of the annual Occupational Employment Survey compiled by the US Bureau of Labor Statistics[8]. Each row provides the estimated number of full-time equivalent employees across many employment types for a specific metropolitan area. There are 334 and 403 cities in the 1997 and May 2010 datasets, respectively. The input variables in these datasets are a randomly sequenced subset of employment types (i.e. doctor, dentist, car repair technician) observed in at least 50% of the cities (some categories had no values for particular cities). The targets for both years are randomly selected from the entire set of categories above the 50% threshold. Missing values in both the input and the target variables were replace by sample means for these results. To our knowledge, this is the first use of the OES dataset for benchmarking of multi-target prediction algorithms.

[8] `http://www.bls.gov/`.

ATP1d & ATP7d The Airline Ticket Price dataset concerns the prediction of airline ticket prices. The rows are a sequence of time-ordered observations over several days. Each sample in this dataset represents a set of observations from a specific observation date and departure date pair. The input variables for each sample are values that may be useful for prediction of the airline ticket prices for a specific departure date. The target variables in these datasets are the next day (ATP1d) price or minimum price observed over the next 7 days (ATP7d) for 6 target flight preferences (any airline with any number of stops, any airline non-stop only, Delta Airlines, Continental Airlines, Airtrain Airlines, and United Airlines). The input variables include the following types: the number of days between the observation date and the departure date (1 feature), the boolean variables for day-of-the-week of the observation date (7 features), the complete enumeration of the following 4 values: 1) the minimum price, mean price, and number of quotes from 2) all airlines and from each airline quoting more than 50% of the observation days 3) for non-stop, one-stop, and two-stop flights, 4) for the current day, previous day, and two days previous. The result is a feature set of 411 variables. For specific details on how these datasets are constructed please consult [17]. The nature of these datasets is heterogeneous with a mixture of several types of variables including boolean variables, prices, and counts.

SCM1d & SCM20d The Supply Chain Management datasets are derived from the Trading Agent Competition in Supply Chain Management (TAC SCM) tournament from 2010. The precise methods for data preprocessing and normalization are described in detail in [16]. Some benchmark values for prediction accuracy in this domain are available from the TAC SCM Prediction Challenge [25], these datasets correspond only to the "Product Future" prediction type. Each row corresponds to an observation day in the tournament (there are 220 days in each game and 18 tournament games in a tournament). The input variables in this domain are observed prices for a specific tournament day. In addition, 4 time-delayed observations are included for each observed product and component (1,2,4 and 8 days delayed) to facilitate some anticipation of trends going forward. The datasets contain 16 regression targets, each target corresponds to the next day mean price (SCM1d) or mean price for 20-days in the future (SCM20d) for each product in the simulation. Days with no target values are excluded from the datasets (i.e. days with labels that are beyond the end of the game are excluded).

RF1 & RF2 The river flow datasets concern the prediction of river network flows for 48 hours in the future at specific locations. The dataset contains data from hourly flow observations for 8 sites in the Mississippi River network in the United States and were obtained from the US National Weather Service. Each row includes the most recent observation for each of the 8 sites as well as time-lagged observations from 6, 12, 18, 24, 36, 48 and 60 hours in the past. In RF1, each site contributes 8 attribute variables to facilitate prediction. There are a total of 64 variables plus 8 target variables.The RF2 dataset extends the

RF1 data by adding precipitation forecast information for each of the 8 sites (expected rainfall reported as discrete values: 0.0, 0.01, 0.25, 1.0 inches). For each observation and gauge site, the precipitation forecast for 6 hour windows up to 48 hours in the future is added (6, 12, 18, 24, 30, 36, 42, and 48 hours). The two datasets both contain over 1 year of hourly observations (>9,000 hours) collected from September 2011 to September 2012. The training period is the first 40% of observations, and the test period is the remainder. The domain is a natural candidate for multi-target regression because there are clear physical relationships between readings in the contiguous river network.

5 Results

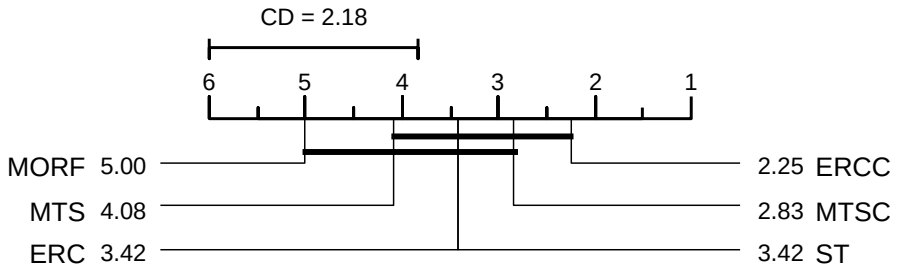
Table 2 shows the aRRMSE obtained by each multi-target method on each dataset while per target RRMSE results are shown in Table 3 (Appendix). The two last rows of Table 2 show the average rank of each method calculated over dataset RRMSE averages (aRRMSE) and over per target RRMSE results.

Table 2: aRRMSE obtained by each multi-target method on each dataset. In each row, the lowest error is typeset in bold. The two last rows show the average rank of each method calculated over dataset RRMSE averages (d) and over per target RRMSEs (t).

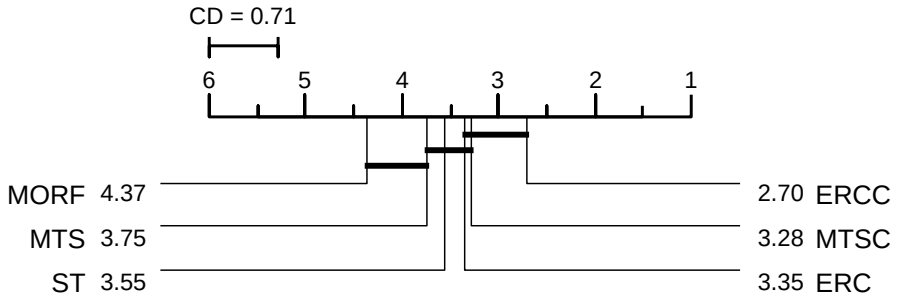
Dataset	MORF	ST	MTS	MTSC	ERC	ERCC
EDM	**73.38**	74.21	74.30	73.96	74.35	74.07
SF1	128.25	113.54	112.70	106.80	**105.01**	108.87
SF2	142.48	114.94	**94.48**	105.53	105.32	108.79
WQ	**89.94**	90.83	91.10	90.95	90.97	90.59
OES97	54.90	52.48	52.59	52.43	52.54	**52.39**
OES10	45.18	42.00	42.01	42.05	42.02	**41.99**
ATP1d	42.22	37.35	37.16	37.17	**37.10**	37.24
ATP7d	55.08	52.48	51.43	**50.74**	53.43	51.24
SCM1d	56.63	47.75	47.41	47.01	47.09	**46.63**
SCM20d	77.75	77.68	78.62	78.54	77.55	**75.97**
RF1	85.13	**69.63**	82.37	69.82	79.47	69.89
RF2	91.89	**69.64**	81.75	69.86	79.61	69.82
Avg. rank d	5.00	3.42	4.08	2.83	3.42	**2.25**
Avg. rank t	4.37	3.55	3.75	3.28	3.35	**2.70**

We observe that in both types of analyses ERCC obtains the lowest average rank, followed by MTSC. Surprisingly, the ST baseline obtains a lower average rank than MORF which has the worst average rank in both cases. We also see that both MTSC and ERCC obtain lower average ranks than the non-modified

versions. When the Friedman test is applied to compare the six algorithms, it finds significant differences between them at $p = 0.01$ in both the per dataset and the per target analysis. Thus, in both cases we proceed to the Nemenyi post-hoc test, whose results at $p = 0.05$ are presented in the average rank diagrams of Figures 1a and 1b. In the per dataset analysis case, the performance of ERCC is significantly better than that of MORF while the experimental data is not sufficient to reach statistically significant conclusions regarding the performance of the other methods. In the per target analysis case, the following significant performance differences are found: $ERCC > \{MORF, MTS, ST\}$ and $MORF < \{ST, ERC, MTSC\}$ where $>$ ($<$) denotes a statistically significant performance improvement (degradation).



(a) Per dataset analysis.



(b) Per target analysis.

Fig. 1: Comparison of all methods with the Nemenyi test calculated on average RRMSE per dataset (Fig 1a) and on RRMSE per target (Fig 1b). Groups of methods that are not significantly different (at $p = 0.05$) are connected.

To further study the impact of the proposed modification on MTS and ERC we apply the Wilcoxon signed-ranks test between MTS and MTSC and between ERC and ERCC. In the MTS/MTSC comparison, the p-value is 0.0425 and 0.0445 in the per dataset and the per target analysis respectively suggesting

that the differences are statistically significant for a = 0.05. In the ERC/ERCC comparison, the p-value is 0.1763 and 0.0098 in the per dataset and the per target analysis respectively suggesting that the differences are not found statistically significant when the analysis is performed per dataset while in the per target analysis ERCC is found significantly better for a = 0.01.

Summarizing the comparative results, we see that the novel multi-label-inspired approaches obtain a better performance than the state-of-the-art MORF method in 10 out of 12 datasets and 82 out of 114 targets, indicating that the knowledge transfer between the two domains was successful. Nevertheless, the fact that there is a large variation in relative performance between the proposed methods across different prediction targets, suggests that their performance requires a further analysis especially compared to the performance of the ST baseline that without taking any target dependencies into account obtains the best performance in 2 out of 12 datasets and 12 out of 114 targets.

6 Conclusions and Future Work

This paper attempts to highlight the connection between multi-label classification and multi-target regression and investigates the applicability of methods proposed to solve the former task into the later task. The analysis of Section 2 reveals that transformation multi-label methods that model each label independently are directly applicable to multi-target regression, simply by employing regression models.

In Section 3 we introduce two new techniques for multi-target regression, MTS and ERC, through straightforward adaptation of two corresponding multi-label learning methods that model label dependencies by using targets as inputs to meta-models. Furthermore, we detect an important shortcoming of these techniques, namely the discrepancy between the distribution of the target variables during training and during prediction, and propose modified versions of MTS (MTSC) and ERC (ERCC), that use cross-validation estimates of the targets during training.

As revealed by the empirical evaluation, this modification has a significant positive impact on the performance of the two methods. Equipped with this modification, MTSC and ERCC exhibit better performance than both the ST baseline and the state-of-the-art MORF approach. In particular, ERCC has the best overall performance and is found statistically significantly more accurate than both ST and MORF (when the analysis is performed per target).

Another important contribution of this paper is the introduction of 8 new publicly available multi-target datasets. This is important as most multi-target datasets mentioned in the literature are kept private due to their industrial nature, a fact that hinders benchmarking techniques and advancing state-of-the-art.

As future work, we would like to perform an analysis into why different multi-target methods perform better on different targets/datasets and to study the connection of these performance differences with dataset characteristics such

as the degree and type of dependence between target variables. It would also be interesting to explore whether we could complete the circle back to multi-label classification, by investigating to what extend the discrepancy problem in MTS and ERC affects the corresponding multi-label classification algorithms and whether our modification can increase the generalization performance in this case as well.

Appendix

References

1. Aho, T., Ženko, B., Džeroski, S.: Rule ensembles for multi-target regression. In: Proc. of the 9th IEEE International Conference on Data Mining (2009)
2. Alvares-Cherman, E., Metz, J., Monard, M.C.: Incorporating label dependency into the binary relevance framework for multi-label classification. Expert Systems with Applications 39(2), 1647 – 1655 (2012)
3. Appice, A., Džeroski, S.: Stepwise induction of multi-target model trees. In: Proc. of the 18th European conference on Machine Learning. pp. 502–509. Springer-Verlag (2007)
4. Asuncion, A., Newman, D.: UCI machine learning repository (2007), `http://www.ics.uci.edu/$\sim$mlearn/MLRepository.html`
5. Blockeel, H., Raedt, L.D., Ramong, J.: Top-down induction of clustering trees. In: In Proc. of the 15th International Conference on Machine Learning. pp. 55–63. Morgan Kaufmann (1998)
6. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
7. Cheng, W., Hüllermeier, E.: Combining instance-based learning and logistic regression for multilabel classification. Machine Learning 76(2-3), 211–225 (September 2009)
8. Clare, A., King, R.: Knowledge discovery in multi-label phenotype data. In: Proc. 5th European Conference Principles of Data Mining and Knowledge Discovery. pp. 42–53. Germany (2001)
9. Crammer, K., Singer, Y.: A family of additive online algorithms for category ranking. Journal of Machine Learning Research 3, 1025–1058 (2003)
10. Dembczyński, K., Cheng, W., Hüllermeier, E.: Bayes optimal multilabel classification via probabilistic classifier chains. In: Proc. of the 27th International Conference on Machine Learning (2010)
11. Dembczyński, K., Waegeman, W., Cheng, W., Hüllermeier, E.: On label dependence and loss minimization in multi-label classification. Machine Learning (2012)
12. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
13. Džeroski, S., Demšar, J., Grbović, J.: Predicting chemical parameters of river water quality from bioindicator data. Applied Intelligence 1(13), 7–17 (2000)
14. Fürnkranz, J., Hüllermeier, E., Mencia, E.L., Brinker, K.: Multilabel classification via calibrated label ranking. Machine Learning 73(2), 133–153 (2008)
15. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Proc. of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 22–30 (2004)

Table 3: RRMSE obtained by each multi-target method on each target of each dataset. In each row, the lowest error is typeset in bold.

Dataset Target	MORF	ST	MTS	MTSC	ERC	ERCC
EDM						
dflow	**77.54**	81.53	81.68	81.23	81.79	81.37
dgap	69.22	66.89	66.92	**66.70**	66.90	66.77
SF1						
c-class	103.46	101.68	103.72	100.08	**100.03**	100.67
m-class	121.16	109.63	113.68	100.50	**99.18**	103.35
x-class	160.12	129.29	120.70	119.81	**115.84**	122.61
SF2						
c-class	99.59	98.01	97.45	**96.36**	97.33	96.43
m-class	115.95	107.54	99.35	**99.20**	102.18	102.98
x-class	211.89	139.28	**86.64**	121.04	116.46	126.97
WQ						
17300	**89.53**	90.21	92.10	91.39	90.63	90.79
19400	**82.79**	83.42	82.93	82.94	83.55	82.85
25400	**92.38**	92.45	93.05	92.81	92.95	92.72
29600	**97.63**	98.65	98.62	98.38	98.54	98.26
30400	**94.20**	94.52	94.49	95.11	94.37	94.61
33400	**89.32**	91.20	90.37	90.23	91.21	90.18
34500	95.95	96.95	96.14	96.34	96.50	**95.71**
37880	85.08	85.57	84.82	**84.69**	85.68	84.87
38100	**90.70**	91.20	90.77	90.80	91.24	91.05
49700	**79.31**	79.48	81.49	80.82	79.90	79.57
50390	**89.15**	89.16	90.11	89.91	89.21	89.19
55800	**90.32**	92.36	93.11	92.88	92.59	92.40
57500	**89.63**	91.76	91.73	91.86	91.75	91.42
59300	**93.10**	94.67	95.66	95.09	95.42	94.69
OES97						
13008	**27.17**	33.94	33.97	33.74	33.92	33.79
15014	46.50	35.83	36.54	**35.16**	35.88	35.24
15017	40.25	36.49	36.96	36.79	**36.47**	36.61
21114	**29.07**	30.69	30.72	30.82	30.70	30.73
27108	63.64	58.02	**57.63**	57.66	58.07	57.70
27311	**56.59**	60.08	59.47	59.51	59.90	59.80
32314	70.43	61.41	61.54	61.72	**61.29**	61.52
32511	75.16	71.67	71.50	**71.37**	71.58	71.42
53905	60.21	**56.89**	57.41	57.17	57.00	57.00
58028	**26.46**	33.62	33.48	33.69	33.67	33.64
65032	**53.46**	55.20	55.43	55.01	55.18	55.14
85110	63.60	56.69	57.45	56.83	56.68	**56.63**
92965	69.84	64.76	64.84	64.73	64.92	**64.72**
92998	70.27	64.36	**64.16**	64.42	64.69	64.25
98502	71.49	**66.49**	66.61	66.54	67.05	66.50
98902	54.28	53.59	53.72	53.63	53.63	**53.52**

Continued on the next page...

Table 3: Continued from previous page.

Dataset Target	MORF	ST	MTS	MTSC	ERC	ERCC
OES10						
119032	40.17	36.97	**36.89**	37.09	36.97	37.01
151131	45.70	43.05	43.14	43.07	43.10	**43.03**
151141	49.61	**43.64**	43.88	44.05	43.80	43.80
172141	59.30	**49.44**	49.63	49.56	49.49	49.45
291051	26.65	26.46	26.72	26.58	26.51	**26.40**
291069	65.01	61.64	61.70	**61.63**	61.72	61.64
291127	**44.16**	46.15	45.28	45.60	46.08	45.78
292037	38.24	**33.49**	33.70	33.69	33.53	33.55
292071	39.32	38.50	38.27	**38.16**	38.70	38.28
392021	**39.72**	41.22	41.25	41.36	41.26	41.24
412021	46.15	37.66	37.85	37.88	**37.63**	37.79
419022	65.44	64.41	64.44	64.44	64.43	**64.40**
431011	**22.72**	26.86	26.72	27.09	26.84	26.90
432011	39.69	**39.15**	39.22	39.22	39.19	39.17
513021	45.34	43.81	43.79	43.72	**43.64**	43.79
519061	55.66	**39.48**	39.67	39.66	39.52	39.55
ATP1d						
acominpa	35.63	24.19	**23.32**	24.16	24.17	23.94
adlminpa	42.44	41.57	40.46	40.96	**39.76**	41.20
aflminpa	48.69	47.12	47.97	47.30	**46.97**	47.31
allminp0	43.65	42.97	**42.00**	42.20	42.64	42.79
allminpa	**47.36**	48.23	48.51	48.19	48.54	48.18
auaminpa	35.57	**20.03**	20.71	20.19	20.52	20.05
ATP7d						
acominpa	43.71	31.62	**26.90**	27.81	29.36	29.05
adlminpa	**52.39**	54.64	53.32	53.70	55.29	54.21
aflminpa	**67.44**	68.92	72.84	70.04	70.74	69.38
allminp0	**60.16**	66.99	67.22	65.30	67.50	65.99
allminpa	63.56	64.07	**61.89**	62.93	69.57	62.60
auaminpa	43.20	28.61	26.43	**24.65**	28.10	26.23
SCM1d						
lbl	51.18	37.19	38.01	**36.58**	37.28	36.87
mtlp10	51.57	45.13	**40.33**	44.24	41.83	44.56
mtlp11	52.21	47.31	**43.74**	46.34	44.40	46.56
mtlp12	56.19	43.06	43.80	43.21	**42.93**	42.96
mtlp13	49.34	39.18	39.30	39.09	**38.67**	38.86
mtlp14	45.45	37.56	38.27	37.37	37.23	**37.17**
mtlp15	45.39	39.62	39.05	39.95	**38.99**	39.54
mtlp16	44.17	40.97	**39.28**	40.08	39.72	40.30
mtlp2	50.47	38.18	37.06	37.32	**36.61**	37.19
mtlp3	53.12	46.85	44.02	**43.59**	44.01	44.10
mtlp4	56.22	46.98	50.71	48.48	47.34	**46.89**
mtlp5	72.79	59.56	63.64	58.87	64.17	**58.20**
mtlp6	76.07	66.05	65.82	64.95	65.63	**64.77**
mtlp7	73.23	64.51	64.42	63.59	63.82	**61.05**
mtlp8	77.48	70.56	69.13	67.74	70.40	**66.17**
mtlp9	51.11	41.31	42.07	40.82	**40.35**	40.83

Continued on the next page...

Table 3: Continued from previous page.

Dataset Target	MORF	ST	MTS	MTSC	ERC	ERCC
SCM20d						
lbl	65.64	62.86	64.90	62.28	64.47	**61.76**
mtlp10a	76.43	74.16	74.04	77.84	**73.94**	74.76
mtlp11a	77.15	73.82	72.48	76.30	74.24	**72.43**
mtlp12a	81.80	74.52	77.69	78.00	77.25	**73.03**
mtlp13a	77.53	72.92	**71.98**	72.00	72.76	72.22
mtlp14a	75.35	72.49	**71.41**	73.78	71.48	71.93
mtlp15a	74.51	70.22	72.80	72.19	71.28	**69.19**
mtlp16a	75.02	73.92	73.98	73.81	73.35	**71.91**
mtlp2a	65.68	64.91	63.84	64.58	64.96	**62.20**
mtlp3a	64.21	71.19	65.77	67.85	**62.58**	65.17
mtlp4a	69.04	71.74	74.78	72.93	**67.33**	69.11
mtlp5a	**91.85**	92.67	103.83	93.72	98.98	92.68
mtlp6a	**94.61**	97.17	99.64	99.38	99.02	95.80
mtlp7a	**89.68**	100.13	97.33	96.79	96.72	94.43
mtlp8a	**94.01**	99.00	103.54	101.70	100.99	99.27
mtlp9a	71.51	71.12	69.90	73.44	71.43	**69.62**
RF1						
chsi2	54.98	49.32	53.25	**47.64**	50.98	48.65
clkm7	87.23	73.81	74.72	**72.89**	73.71	73.53
dldi4	94.55	**92.55**	96.02	100.65	93.67	96.60
eadm7	55.52	52.36	**49.79**	50.50	50.04	51.85
napm7	119.36	58.01	62.89	**57.48**	59.52	57.83
nasi2	101.19	103.84	**100.82**	102.33	102.83	103.15
sclm7	75.09	66.82	**62.56**	64.95	62.93	65.86
vali2	93.17	**60.34**	158.93	62.15	142.05	61.62
RF2						
chsi2	59.04	49.30	53.06	**47.56**	51.06	48.63
clkm7	83.62	74.09	75.01	**73.57**	74.00	74.02
dldi4	**92.33**	92.51	96.00	100.47	93.72	96.43
eadm7	56.91	52.42	50.32	50.35	**50.08**	51.88
napm7	148.90	58.01	62.41	**57.09**	59.44	57.81
nasi2	**100.55**	103.90	100.76	102.49	103.01	103.25
sclm7	95.89	67.41	**62.50**	66.12	62.81	66.38
vali2	97.87	**59.50**	153.91	61.21	142.76	60.15

16. Groves, W., Gini, M.: Improving prediction in TAC SCM by integrating multivariate and temporal aspects via pls regression. In: IJCAI11: Workshop on Trading Agent Design and Analysis (2011)
17. Groves, W., Gini, M.: A regression model for predicting optimal purchase timing for airline tickets. Tech. rep., University of Minnesota, Minneapolis, MN (Oct 2011)
18. Izenman, A.J.: Modern Multivariate Statistical Techniques : Regression, Classification, and Manifold Learning. Springer Texts in Statistics, Springer New York (2008)
19. Karalič, A., Bratko, I.: First order regression. Machine Learning (1997)
20. Kocev, D., Džeroski, S., White, M.D., Newell, G.R., Griffioen, P.: Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition. Ecological Modelling 220(8) (2009)
21. Kocev, D., Vens, C., Struyf, J., Džeroski, S.: Tree ensembles for predicting structured outputs. Pattern Recognition (2013)
22. Kužnar, D., Možina, M., Bratko, I.: Curve prediction with kernel regression. In: Proc. ECML/PKDD 2009 Workshop on Learning from Multi-Label Data. pp. 61–68 (2009)
23. Lebanon, G., Balasubramanian, K.: The landmark selection method for multiple output prediction. In: Proc. of the 29th International Conference on Machine Learning. Edinburgh, Scotland, UK (2012)
24. Madjarov, G., Kocev, D., Gjorgjevikj, D., Džeroski, S.: An extensive experimental comparison of methods for multi-label learning. Pattern Recognition 45(9), 3084 – 3104 (2012)
25. Pardoe, D., Stone, P.: The 2007 TAC SCM prediction challenge. In: AAAI 2008 Workshop on Trading Agent Design and Analysis (2008)
26. Read, J., Pfahringer, B., Holmes, G.: Multi-label classification using ensembles of pruned sets. In: Proc. 8th IEEE International Conference on Data Mining (2008)
27. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Machine Learning 85(3), 333–359 (2011)
28. Seeger, M., Teh, Y.W., Jordan, M.I.: Semiparametric latent factor models. In: Proc. of the 10th International Workshop on Artificial Intelligence and Statistics. pp. 333–340 (2005)
29. Senge, R., del Coz, J.J., Hüllermeier, E.: On the problem of error propagation in classifier chains for multi-label classification. In: Proc. GFKL-2012, 36th Annual Conference of the German Classification Society. Springer-Verlag (Forthcoming)
30. Tsoumakas, G., Dimou, A., Spyromitros, E., Mezaris, V., Kompatsiaris, I., Vlahavas, I.: Correlation-based pruning of stacked binary relevance models for multi-label learning. In: Proc. ECML/PKDD 2009 Workshop on Learning from Multi-Label Data (2009)
31. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Data Mining and Knowledge Discovery Handbook (2010)
32. Tsoumakas, G., Katakis, I., Vlahavas, I.: Random k-labelsets for multi-label classification. IEEE Transactions on Knowledge and Data Engineering (2011)
33. Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I.: Mulan: a java library for multi-label learning. Journal of Machine Learning Research (2011)
34. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. Machine Learning 73(2), 185–214 (2008)
35. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques, 3rd Edition. Morgan Kaufmann (2011)
36. Zhang, M.L.: Lift: Multi-label learning with label-specific features. In: IJCAI (2011)

37. Zhang, M.L., Zhou, Z.H.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Transactions on Knowledge and Data Engineering 18(10), 1338–1351 (2006)
38. Zliobaite, I., Bakker, J., Pechenizkiy, M.: Towards context aware food sales prediction. In: Proc. IEEE ICDM Workshops on Domain Driven Data Mining. pp. 94–99. IEEE Computer Society (2009)