Copyright © 2018 Tech Science Press CMC, vol.56, no.2, pp.185-198, 2018

An Empirical Comparison on Multi-Target Regression Learning

Xuefeng Xi[1], Victor S. Sheng[1, 2, *], Binqi Sun[2], Lei Wang[1] and Fuyuan Hu[1]

Abstract: Multi-target regression is concerned with the simultaneous prediction of multiple continuous target variables based on the same set of input variables. It has received relatively small attention from the Machine Learning community. However, multi-target regression exists in many real-world applications. In this paper we conduct extensive experiments to investigate the performance of three representative multi-target regression learning algorithms (i.e. Multi-Target Stacking (MTS), Random Linear Target Combination (RLTC), and Multi-Objective Random Forest (MORF)), comparing the baseline single-target learning. Our experimental results show that all three multi-target regression learning algorithms do improve the performance of the single-target learning. Among them, MTS performs the best, followed by RLTC, followed by MORF. However, the single-target learning sometimes still performs very well, even the best. This analysis sheds the light on multi-target regression learning and indicates that the single-target learning is a competitive baseline for multi-target regression learning on multi-target domains.

Keywords: Multi-target regression, multi-label classification, multi-target stacking.

1 Introduction

Multi-target regression, also known as multivariate or multi-output regression, is another instance of the more general learning task of multi-target prediction. Here the prediction targets are real-valued, as opposed to the closely related task of multi-label classification where the target variables are binary. Multi-output regression has recently emerged and extensively studied for many computer vision tasks, e.g., head pose estimation [Hara and Chellappa (2014)], human body pose estimation [Toshev and Szegedy (2014)] and viewpoint estimation [Torki and Elgammal (2011)]. Moreover, many researchers have found their applications, e.g. camera re-localization [Shotton, Glocker, Zach et al. (2013)] and cardiac volume estimation [Zhen, Wang, Islam et al. (2014)], can be elaborately solved by transferring the corresponding original problem into a multi-output regression task, which not only substantially outperforms conventional approaches but also offers a more compact and exquisite mathematical formulation to circumvent the difficulty in conventional approaches, e.g. the inverse problems [Guzman-Rivera, Kohli, Glocker et al. (2014)].

[1] School of Electronic and Information Engineering, Suzhou University of Science and Technology, Suzhou, China.

[2] Department of Computer Science, University of Central Arkansas, Conway, AR 72035, USA.

[*] Corresponding Author: Victor S. Sheng. Email: ssheng@uca.edu.

Multi-target regression and multi-label classification are supervised machine learning algorithms. They make predictions based on a set of examples. For example, historical stock prices can be used to predict the future prices. Each example used for training can be labeled with the value of interest, for example, the stock price. Both Multi-target regression and multi-label classification algorithms look for patterns in those values, but each algorithm looks for different types of patterns. After an algorithm has found the best pattern, it can use the pattern to make predictions for unlabeled testing data the future price [Microsoft (2017)]. Multi-target regression and multi-label classification are closely related to each other. Despite that multi-target regression is a little more general. Multi-label learning is often treated as a special case of multi-target regression in statistics. However, we could more precisely state that both are instances of learning for predicting multiple targets, which could be real-valued, binary, ordinal, categorical or even of mixed type.

Current existing multi-target regression learning algorithms are developed based on two basic approaches: Algorithm adaptation and problem transformation. Problem transformation is easy to understand. It is to transfer a multi-target regression problem into multiple traditional single-target regression problems. After a multi-target regression problem is transferred into multiple single-target regression ones. All the traditional regression learning algorithms can be applied directly to build a regressor for each single-target dataset and make prediction for its correlated test instances. The prediction for a multi-target instance is made by aggregating outputs from autonomous regressors. Multi-Target Stacking (MTS) [Spyromitros-Xioufis, Tsoumakas, Groves et al. (2016)] is chosen as the representative of this group.

The second method used in multi-target regression is algorithm adaptation. It extends existing traditional regression algorithms to perform multi-target regression directly, for example, Random Linear Target Combination (RLTC) and multi-objective random forest (MORF) [Kocev, Vens, Struyf et al. (2007)]. Algorithm adaption completely differs from problem transformation. Instead, the algorithm learns the structure and correlations that exist among multi-targets directly. Thus, it is very useful to investigate the performance of multi-target regression learning algorithms, which are developed based on the two approaches. The investigating results will guide data mining researchers in their future research on developing better multi-target regression learning algorithms.

The rest of the paper is organized as follows. Section 2 introduces the three representative multi-target regression learning algorithms which we will make comparison empirically. Section 3, we describe the experiments we have conducted. They consist of the setting of the experiments, the experimental results, and the analysis of the experimental results. Section 4 concludes with a summary of our work and a discussion of future work.

2 Multi-target regression algorithms

In this section, we first provide a brief description of multi-target regression learning, and then we briefly review three representative multi-target learning algorithms (i.e. Multi-Target Stacking (MTS), random linear target combination (RLTC), and Multi-Objective Random Forest (MORF)) in this section, which are used in our experiments in Section 3.

2.1 Multi-target regression

Multi-target regression is a statistical process for estimating the relationships among variables. Let us consider a training dataset *D* with *N* instances containing a value assignment for each variable $X_1,...,X_m$, $Y_1,...,Y_d$, i.e. $D=\{(x^1,y^1),...,(x^N,y^N)\}$. Each instance is characterized by an input vector of *m* descriptive or predictive variables $x^l=(X_1,...,X_j,...,X_m)$ and an output vector of d target variables $y^l=(Y_1,...,Y_i,...,Y_d)$ with $i\in\{1,...,d\}$, $j\in\{1,...,m\}$, and $l\in\{1,...,N\}$. The task is to learn a multi-target regression model from *D* consisting of finding a function *h* that assigns to each instance, given by the vector *x*, a vector *y* of *d* target values:

$h: \Omega X_1\times,...,\times\Omega X_m \rightarrow \Omega Y_1\times,...,\times\Omega Y_d$

$x=(x_1,...,x_m)\rightarrow y=(y_1,...,y_d)$,

Where ΩX_j and ΩY_i denote the sample spaces of each predictive variable X_j, for all $j\in\{1,...,m\}$, and each target variable Y_i, for all $i\in\{1,...,d\}$, respectively. Note that, all target variables are considered to be continuous here. The learned multi-target model will be used afterwards to simultaneously predict the values $\{\hat{y}^{(N+1)},...,\hat{y}^{(N')}\}$ of all target variables of the new incoming unlabeled instances $\{x^{(N+1)},...,x^{(N')}\}$ [Hanen, Gherardo, Concha et al. (2015)].

In this way, the dependencies of the target attributes are implicitly modeled as well, producing better predictive performance. The other advantage of described multi-target model is that the size and complexity of the produced model is smaller than the combined size of the single-target models.

2.2 Multi-Target Stacking (MTS)

Multi-Target Stacking (MTS) is a representative multi-target regression learning algorithm developed via problem transformation. Its brief introduction is as follows.

Stacking (also called meta ensembling) is a model ensembling technique used to combine information from multiple predictive models to generate a new model [Gorman (2016)]. The multi-target stacking algorithm is inspired by where stacked generalization was used to deal with multi-label classification [Godbole and Sarawagi (2004)]. Multi-target stacking training is a two-stage process. First, *d* single-target models are learned respectively in a single-target learning mode. However, instead of directly using these models for prediction, multi-target stacking includes an additional training stage where a second set of d meta-models are learned, one for each target Y_i, $i\in\{1,...,d\}$.

Each meta-model is learned on a transformed training set

$D_i^*=\{(x^{*(1)},y_i^{(1)}),...,(x^{*(l)},y_i^{(l)}),...,(x^{*(N)},y_i^{(N)})\}$, where $x^{*(l)}=(x_1^{(l)},...,x_m^{(l)},\hat{y}_1^{(l)},...,\hat{y}_d^{(l)})$ is a transformed input vector consisting of the original input vector of the training set augmented by predictions (or estimates) of their target variables yielded by the first-stage models. In fact, MTS is based on the idea that a second-stage model is able to correct the prediction of the first-stage models by using the predictions of other variables obtained in the first-stage models.

The predictions for a new instance $x^{(N+1)}$ are obtained by generating the first-stage models inducing the estimated output vector $\hat{y}^{(N+1)} = (\hat{y}_1^{(N+1)}, ..., \hat{y}_d^{(N+1)})$, and then MTS applies the second-stage models on the transformed input vector $x^{*(N+1)} = (x_1^{(N+1)}, ..., x_m^{(N+1)}, \hat{y}_1^{(N+1)}, ..., \hat{y}_d^{(N+1)})$ to produce the final estimated multi-output targets $\hat{\hat{y}}^{(N+1)} = (\hat{\hat{y}}_1^{(N+1)}, ..., \hat{\hat{y}}_d^{(N+1)})$.

2.3 Random Linear Target Combinations (RLTC)

Random Linear Target Combinations (RLTC) is a representative multi-target regression learning algorithm [Tsoumakas, Spyromitros-Xioufis, Vrekou et al. (2014)] developed via algorithm adaption. Its brief introduction is as follows.

Consider a set of m input variables $x \in R^m$ and a set of d target variables $y \in R^d$. There are a set of N training examples: $D = (X, Y) = \{(x^{(i)}, y^{(i)})\}_{i=1}^N$, where X and Y are matrices of size $N \times m$ and $N \times d$, respectively. RLTC constructs $r>>d$ new target variables via corresponding random linear combinations of y. To achieve this, this approach defines a coefficient matrix C of size $d \times r$ filled with random values uniformly chosen from [0..1]. Each column of this matrix contains the coefficients of a linear combination of the target variables. Multiplying Y with C leads to a transformed multi-target training set D'=(X, Z), where Z=YC is a matrix of size $N \times r$ with the values of the new target variables. A user-specified multi-target regression learning algorithm is then applied to D' in order to build a corresponding model.

Note that RLTC expects that the original target variables take values from the same domain, as otherwise their linear combinations could be dominated by the values of targets with a much wider domain than the others. To ensure this, it applies 0-1 normalization in order to bring the values of all targets into the range [0…1].

This algorithm considers an additional parameter $k \in \{2,...,d\}$ for specifying the number of original target variables involved in each random linear combination, by setting the coefficients for the rest of the target variables to zero. A higher k means that potential correlations among more targets are being considered. However, at the same time, it means that the new targets are more difficult to predict, especially in the absence of actual correlations among the targets. Therefore, RLTC hypothesizes that low k values will lead to the best results. In practice, when $k<d$, for each linear combination RLTC selects k targets at random, but with priority to targets with the lowest frequency of participation to previously considered linear combinations. This ensures that all targets will participate in C as equivalently (i.e. with similar frequency) as possible.

Given a new test instance, x', the multi-target regression model is first invoked to obtain a vector z' with r predictions. The estimates $\hat{y}'$ for the original target variables are then obtained by solving the following overdetermined (as r>>d) system of linear equations: $C^T \hat{y}' = z'$.

2.4 Multi-Objective Random Forest (MORF)

Multi-Objective Random Forest (MORF) is a direct multi-output learning approach [Kocev, Vens, Struyf et al. (2007)]. Again, it is another representative multi-target regression learning algorithm developed via algorithm adaption. It integrates one of the most popular ensemble meta-learning approach Bagging with Random Forest. For multi-target learning, instead of building single-target random forests, MORF builds multi-target random forests. Specifically, it builds multi-target random forests based on multi-objective decision trees (MODTs) from different random selected feature sets.

Multi-objective decision trees (MODTs) [Blockeel, De Raedt and Ramon (1998)] are decision trees capable of predicting multiple target attributes at once. The main difference between MODTs with single-target standard decision trees is that MODTs treats the variance function and the prototype function that computes a label for each leaf as parameters. For multi-label classification, the variance function is computed as the sum of the entropies of target variables, i.e. $Var(E) = \sum_{i=1}^{d} Entropy(E, y_i)$, and the prototype function returns a vector containing the majority class for each target of the corresponding training examples *E*. For multi-objective regression trees, the sum of the variances of the targets is used, i.e. $Var(E) = \sum_{i=1}^{d} Var(y_i)$, and the prototype of each leaf is the vector mean of the target vectors of the corresponding training examples *E*.

3 Experiments

We conducted an extensive experiments to investigate the performance of three popular and representative multi-target learning algorithms (i.e. MTS, RLTC, and MORF), comparing with the baseline (a single-target learning). Before presenting our experimental results, we first discuss the implementations and parameter settings of these algorithms, and then provide a brief description of each dataset used in our experiments.

3.1 Algorithm implementation and experimental settings

MTS first transforms a multi-target prediction problem into *d* single-target problems, and then learns meta-models from augmented training data (details can be found in Section 2). Since MORF integrates bagging inside and is based on decision trees, to make a fair comparison and to simplify the analysis, we used bagging [Breiman (1996)] to construct 100 regression trees in MTS. To evaluate the performance of MTS, 10-fold cross-validation is applied.

RLTC is to solve the overdetermined system of linear equations during prediction. It is to learn a single independent regression model for each target. Each regression model is built using gradient boosting [Friedman (2001)] with a 4-terminal node regression tree as the base learner, a learning rate of 0.1 and 100 boosting iterations. The system of linear equations is solved by the un-regularized least squares approach. In our experiments, we generate r=100 new target variables by combining k=2 the original target variables.

Concerning the parameter settings of MTS and RLTC, in MORF we use an ensemble size of 100 trees and the values suggested by Kocev et al. [Kocev, Vens, Struyf et al. (2007)] for the rest of its parameters.

All of the algorithms are implemented within Mulan. Mulan is an open-source Java library for learning from multi-label datasets, which is built on top of Weka, including implementations of bagging, gradient boosting, and regression tree. MTS and RLTC are already integrated in Mulan [Mulan (2010)].

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors), which is the measure we used to compare the performance of the multi-target learning algorithms. Besides, we chose RMSE because it is commonly used in climatology, forecasting, and regression analysis to verify experimental results.

3.2 Datasets

Although multi-target regression has many interesting applications, quite of a few multi-target regression datasets are publicly available. We conduct experiments on 10 multi-target regression datasets, which can be downloaded from the website of MULAN [Tsoumakas, Katakis and Vlahavas (2010)]. A brief description of each dataset used in our experiments is provided as follows.

Table 1: The characteristics of the 10 datasets used in our experiments

Dataset name	*abbreviation*	*Source*	*#Instances*	*p*	*d*
Electrical Discharge Machining	edm	Karalic and Bratko (1997)	154	16	2
Solar Flare 1	sf1	Lichman (2013)	323	10	3
Solar Flare 2	sf2	Lichman (2013)	1066	10	3
Water Quality	wq	Dzeroski, Demsar and Grbovic et al. (2000)	16	16	14
Energy building	enb	Tsanas and Xifara (2012)	768	8	2
Concrete Slump	slump	Yeh (2007)	103	7	3
Andromeda	andro	Hatzikos, Tsoumakas, Tzanis et al. (2008)	49	30	6
JURA	jura	Goovaerts (1997)	359	15	3
Online Product Sales	osales	Kaggle (2012)	639	413	12
See Click Predict Fix	scpf	Kaggle (2013)	1137	23	3

The characteristics of each dataset are shown in Tab. 1, where the 1st column shows the name of each dataset, the abbreviation of each dataset is shown in the 2nd column, the number of instances of each dataset is shown in the 3rd column, the number of input variables p is shown in the 4th column, and the number of output variables d is shown in the 5th column.

The electrical discharge machining dataset (EDM) [Karalic and Bratko (1997)] was used to study shortening the machining time by reproducing the behavior of a human operator to control the values of two target variables (i.e. gap control and flow control). Each of the target variables takes three distinct numeric values (1, 0, 1). There are 16 continuous input variables, representing mean values and deviations of the observed quantities of considered machining parameters.

The solar flare dataset was used for predicting how often three potential types of solar flare (i.e. common, moderate, and severe) occur in a 24 h period [Lichman (2013)]. That is, each target variable counts the number of solar flares of the corresponding type. There are ten input feature variables, which describe active regions on the sun. There are two versions of this dataset (i.e. sf1 and sf2 in Tab. 1). Sf1 contains the data from year 1969, and sf2 contains the dataset from year 1978.

The water quality dataset (wq) was obtained from the Hydrometeorological Institute of Slovenia. It was used to monitor water quality of Slovenian rivers and maintained a database of water quality samples covering the six year period from 1990 to 1995 [Dzeroski, Demsar and Grbovic (2000)]. There are 16 different measured chemical parameters and 14 target variables representing bioindicator taxa.

The energy building dataset (enb) [Tsanas and Xifara (2012)] was used to study the effect of eight input variables (relative compactness, surface area, wall area, roof area, overall height, orientation, glazing area, glazing area distribution) on two output variables, namely heating load (HL) and cooling load (CL), of residential buildings.

The Concrete Slump dataset (slump) [Yeh (2007)] was used to make predictions on three properties (i.e. slump, flow and compressive strength) of concrete as a function of the content of seven concrete ingredients (i.e. cement, fly ash, blast furnace slag, water, superplasticizer, coarse aggregate, and fine aggregate).

The Andromeda dataset (andro) [Hatzikos, Tsoumakas, Tzanis et al. (2008)] was used to predict the future values of six water quality variables (temperature, pH, conductivity, salinity, oxygen, turbidity) in Thermaikos Gulf of Thessaloniki, Greece. Measurements of the target variables are taken from under-water sensors with a sampling interval of nine seconds and then averaged to get a single measurement for each variable over each day.

The Jura dataset [Goovaerts (1997)] contains the measurements of concentrations of seven heavy metals (cadmium, cobalt, chromium, copper, nickel, lead, and zinc), recorded at 359 locations in the topsoil of a region of the Swiss Jura. The type of land use (Forest, Pasture, Meadow, and Tillage) and rock type (Argovian, Kimmeridgian, Sequanian, Portlandian, and Quaternary) were also recorded for each location. Specifically, the concentration of three metals (i.e. cadmium, copper and lead) is more expensive to measure than other metals. Therefore, the concentration of three metals (i.e. cadmium, copper and lead) are treated as target variables while the remaining metals along with land use type, rock type and the coordinates of each location (15 features in total) are used as input features.

The online product sale dataset (osales) [Kaggle (2012)] was used to predict monthly online sales of consumer products. Each row in this dataset represents a different consumer product that is described by various product features as well as features of an advertising campaign (413 input features in total). There are 12 target variables corresponding to the monthly sales for the first 12 months after the product launches.

The "See Click Predict Fix" dataset (scpf) [Kaggle (2013)] is to quantify and predict the number of views, votes, and comments that a specific issue has received to date, in terms of 23 input features, such as the number of days that an issues stayed online, the source from which the issue was created (e.g. android, iphone, remote api, etc.), the type of the issue (e.g. graffiti, pothole, trash, etc.), and the geographical co-ordinates of the issue. The issues have been collected from four cities (Oakland, Richmond, New Haven, Chicago) in the U.S. and span a period of 12 months (01/2012-12/2012).

3.3 Experimental results

We have conducted the experiments on ten datasets, comparing the performance of three multi-target learning methods (i.e. MORF, MTS, and RLTC) with a single-target learning, in terms of the relative root mean squared error (RMSE). Our experimental results of four learning algorithms on ten datasets are shown in Tab. 2.

We further conducted analysis on our experimental results in Tab. 2. We first ranked the performance of the four algorithms on each dataset and have their number of rank #1, rank #2, rank #3, and rank #4 shown in Tab. 3.

Table 2: Comparisons among the four algorithms in terms of RMSE on the ten datasets. The lowest RMSE is in bold, and the second lowest is in *italic*

Rank	Single-Target	MORF	MTS	RLTC
edm	0.7421	**0.7338**	0.7430	*0.7352*
sf1	*1.1353*	1.2825	**0.1127**	1.1631
sf2	*1.1494*	1.4248	**0.9448**	1.2282
wq	0.9083	**0.8994**	0.9110	*0.9014*
enb	**0.1167**	0.1210	*0.1185*	0.1203
slump	*0.6878*	0.6939	**0.6758**	0.6904
andro	0.6016	**0.5098**	*0.5109*	0.5701
jura	*0.5891*	0.5968	**0.5861**	0.5964
osales	0.7479	0.7533	**0.7376**	*0.7406*
scpf	0.8371	*0.8335*	**0.8260**	0.8348
average	*0.7515*	0.7849	**0.6166**	0.7581

From the above Tab. 2 and the following Tab. 3, we can see that MTS performs the best on six out the ten datasets, MORF performs the best on three out of the ten datasets, and Single-Target performs the best on one of the ten datasets. According to the average performance in terms of RMSE (shown in the last row of Tab. 2) and the average ranking (shown in the last row of Tab. 3), which supports each other, we can see that MTS performs the best in general, followed by Single-Target. Although Single-Target only performs the best on one of the ten datasets, it is in the second on four out of the ten dataset, which makes the second best among the four learning algorithms. MORF performs the worst, followed by RLTC.

The results shown in Tabs. 2 and 3 are not consistent with our thoughts on multi-target learning. Especially, the Single-Target learning performs the second, which is better than MORF and RLTC. We usually think that multi-target learning can improve the performance of the single-target learning in multi-target domains. Therefore, we further investigate the detailed performance of each learning algorithm on each target of each dataset, instead of investigating the performance of the four algorithms on each dataset. Our detailed experimental results are shown in Tab. 4.

Table 3: Comparisons among the four algorithms in terms of the ranks of RMSE on the ten datasets

Rank	**Single-Target**	**MORF**	**MTS**	**RLTC**
1st	1	3	6	0
2nd	4	1	2	3
3rd	3	0	0	7
4th	2	6	2	0
average	2.6	2.9	**1.8**	2.7

We further conducted analysis on our detailed experimental results in Tab. 4. We first ranked the performance of the four algorithms on each target of each dataset and have their number of rank #1, rank #2, rank #3, and rank #4 shown in Tab. 5.

From Tabs. 4 and 5, we can see that Single-Target takes the first rank seven times out of 51 targets of the ten datasets, MORF takes the first rank 13 times out of 51 targets of the ten datasets, MTS takes the first rank 17 times out of 51 targets of the ten datasets, and RLTC takes the first rank 15 times out of 51 targets of the ten datasets. Note that there exist 51 targets from the ten dataset, which is the summation of the last column in Tab. 1. This analysis can make us understand that the Single-Target learning can perform better than all three multi-target learning algorithms (i.e. MORF, MTS, and RLTC) for some targets of some domains. However, in general, MTS performs the best, followed by RLTC, followed by MORF. Single-Target performs the worst. This analysis sheds the light on multi-target learning and also indicates that the single-target learning is a competitive baseline for multi-target learning on multi-target domains.

Table 4: Our detailed experimental results of the four comparison algorithms in terms of RMSE on each target of each dataset. Again, the lowest RMSE is in bold, and the second is in *italic*

Dataset	Target	Single-Target	MORF	MTS	RLTC
edm	DFlow	0.8153	**0.7754**	0.8168	*0.8009*
edm	DGap	**0.6689**	*0.6922*	*0.6692*	0.6695
sf1	c-class	*1.0168*	1.0346	**0.1037**	1.0186
sf1	m-class	*1.0963*	1.2116	**0.1137**	1.1296
sf1	x-class	*1.2929*	1.6012	**0.1207**	1.3412
sf2	c-class	*0.9801*	0.9959	**0.9745**	0.9849
sf2	m-class	*1.0754*	1.1595	**0.9935**	1.0798
sf2	x-class	*1.3928*	2.1189	**0.8664**	1.6198
wq	25400	0.9245	*0.9238*	0.9305	**0.9224**
wq	29600	0.9865	**0.9763**	0.9862	*0.9794*
wq	30400	0.9452	*0.9420*	0.9449	**0.9367**
wq	33400	0.9120	**0.8932**	*0.9037*	0.9040
wq	17300	*0.9021*	**0.8953**	0.9210	0.9027
wq	19400	0.8342	**0.8279**	*0.8293*	0.8321
wq	34500	0.9695	*0.9595*	0.9614	**0.9570**
wq	38100	0.9120	*0.9070*	0.9077	**0.9042**
wq	49700	0.7948	**0.7931**	0.8149	*0.7933*
wq	50390	0.8916	*0.8915*	0.9011	**0.8840**
wq	55800	0.9236	**0.9032**	0.9311	*0.9155*
wq	57500	0.9176	**0.8963**	0.9173	*0.9070*
wq	59300	0.9467	**0.9310**	0.9566	**0.9310**
wq	37880	0.8557	*0.8508*	**0.8482**	0.8508
enb	Y1	0.0534	0.0600	*0.0532*	**0.0530**
enb	Y2	**0.1799**	*0.1820*	0.1837	0.1875
slump	SLUMP_cm	0.7950	*0.7754*	**0.7636**	0.7922
slump	FLOW_cm	0.7424	*0.7331*	**0.7311**	0.7397
slump	Compressive_Strength_Mpa	**0.5260**	0.5732	*0.5327*	0.5393
andro	Target_1	0.5150	**0.4359**	0.4835	*0.4519*
andro	Target_2	0.3404	0.4027	*0.3181*	**0.2958**
andro	Target_3	0.5884	*0.4988*	**0.3910**	0.5935
andro	Target_4	0.5302	*0.4669*	**0.4588**	0.5874

andro	Target_5	0.8088	*0.6320*	**0.6271**	0.7454
andro	Target_6	0.8268	**0.6222**	0.7867	*0.7466*
jura	Cd	0.7108	*0.6943*	**0.6862**	0.7017
jura	Co	**0.5428**	0.5661	*0.5503*	0.5579
jura	Cu	**0.5137**	0.5301	*0.5217*	0.5296
osales	Outcome_M1	**0.6528**	0.6759	*0.6547*	0.6567
osales	Outcome_M2	0.7539	**0.7195**	0.7517	*0.7496*
osales	Outcome_M3	0.7856	*0.7782*	0.7787	**0.7719**
osales	Outcome_M4	0.6889	0.7361	*0.6825*	**0.6761**
osales	Outcome_M5	0.7363	0.7380	*0.7199*	**0.7035**
osales	Outcome_M6	**0.6964**	0.7528	*0.7032*	0.7100
osales	Outcome_M7	0.7427	0.7682	*0.7403*	**0.7378**
osales	Outcome_M8	0.7641	0.7887	*0.7614*	**0.7587**
osales	Outcome_M9	0.8119	*0.7461*	**0.7325**	0.7931
osales	Outcome_M10	0.7725	*0.7697*	**0.7643**	0.7760
osales	Outcome_M11	0.7490	0.7604	*0.7423*	**0.7356**
osales	Outcome_M12	0.8205	**0.8063**	0.8193	*0.8181*
scpf	num_views	0.8153	*0.8085*	**0.8048**	0.8144
scpf	num_votes	0.7200	*0.7036*	**0.7021**	0.7242
scpf	num_comments	0.9760	0.9883	*0.9710*	**0.9659**
average		0.7885	0.8018	**0.7025**	0.7878

Table 5: Comparisons among the four algorithms in terms of the ranks of RMSE

Rank	**Single-Target**	**MORF**	**MTS**	**RLTC**
1st	7	13	17	15
2nd	7	18	16	10
3rd	18	0	11	21
4th	19	20	7	5
average	2.9608	2.5294	**2.1569**	2.3137

We further analyzed why we have different comparison conclusions shown in Tabs. 3 and 5. This is because the granularity of comparisons is different. Tabs. 2 and 3 are based on datasets. Each dataset is the basic unit in comparisons. However, Tabs. 4 and 5 are based on targets. Each target is the basic unit in comparisons. Since different datasets have a different number of targets. Tab. 5 is a weighted comparison summarization, where the number of targets in each dataset works as the corresponding weight. When the number of targets is great, multi-target learning is preferred. This is reasonable because when there exist a great number of targets in a multi-target domain, there exist some targets that could improve the learning performance of others.

4 Conclusions and future work

In this paper, we conducted extensive experiments to investigate the performance of four multi-target regression learning algorithms (i.e. Single-Target, MTS, RLTC, and MORF). Our experimental results in terms of RMSE showed that in general MTS performs the best, followed by RLTC, followed by MORF. Single-Target performs the worst. However, Single-Target performs the best on one of the ten datasets, and the second best on four out of the ten datasets. This analysis sheds the light on multi-target learning and also indicates that the single-target learning is a competitive baseline for multi-target learning on multi-target domains.

All of the algorithms that used above, including Single-Target, MTS, RLTC, and MORF, are categorized as problem transformation methods in multi-target learning. All of them first transform a multi-output regression problem into multiple single-target regression problems, then build a model for each target, and finally concatenate all predictions. The main drawback of the Single-Target learning is that the relationships among the targets are ignored, and the targets are predicted independently, which may affect the overall quality of the predictions [Hanen, Gherardo, Concha et al. (2015)]. However, the Single-Target learning is the simplest approach to learn from multi-output regression domains. Both MTS and RLTC employ the correlations between targets to improve performance of multi-target regression learning.

Considering potential real-world applications of multi-target regression, we will continue to evaluate the performance of existing multi-target regression learning algorithms. In the same time, we are going to design novel algorithms for multi-target regression with the insights found in the experiments.

Acknowledgments: This research has been supported by the US National Science Foundation under grant IIS-1115417, the National Natural Science Foundation of China under grant 61728205, 61472267, and Foundation of Key Laboratory in Science and Technology Development Project of Suzhou under grant SZS201609.

References

Blockeel, H.; De Raedt, L.; Ramon, J. (1998): Top-down induction of clustering trees. *Machine Learning, Proceedings of the 15th International C*onference, pp. 55-63.

Borchani, H.; Varando, G.; Bielza, C.; Larrañaga, P. (2015): A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 5, pp. 216-233.

Breiman, L. (1996): Bagging predictors. *Machine Learning*, vol. 24, no. 2, pp. 123-140.

Dembczynski, K.; Waegeman, W.; Cheng, W.; Hullermeier, E. (2012): On label dependence and loss minimization in multi-label classification. *Machine Learning*, vol. 88, no. 1-2, pp. 5-45.

Dzeroski, S.; Demsar, D.; Grbovic, J. (2000): Predicting chemical parameters of river water quality from bioindicator data. *Applied Intelligence*, vol. 13, no. 1, pp. 7-17.

Friedman, J. H. (2001): Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, vol. 29, no. 5, pp. 1189-1232.

Godbole, S.; Sarawagi, S. (2004): Discriminative methods for multi-labeled classification. *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 22-30.

Goovaerts, P. (1997): *Geostatistics for Natural Resources Evaluation*. Oxford University Press, New York.

Gorman, B. (2016): A kaggler's guide to model stacking in practice. http://blog.kaggle.com/2016/12/27/a-kagglers-guide-to-model-stacking-in-practice/.

Guzman-Rivera, A.; Kohli, P.; Glocker, B.; Shotton, J.; Sharp, T. et al. (2014): Multi-output learning for camera relocalization. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1114-1121.

Hara, K.; Chellappa, R. (2014): Growing regression forests by classification: Applications to object pose estimation. *European Conference on Computer Vision*, pp. 552-567.

Hatzikos, E. V.; Tsoumakas, G.; Tzanis, G.; Bassiliades, N.; Vlahavas, I. (2008): An empirical study on sea water quality prediction. *Knowledge-Based Systems*, vol. 21, no. 6, pp. 471-478.

Kaggle (2012): Kaggle competition: Online product sales. https://www.kaggle.com/c/online-sales.

Kaggle (2013): Kaggle competition: See click predict fix. https://www.kaggle.com/c/see-click-predict-fix.

Karalic A.; Bratko, I. (1997): First order regression. *Machine Learning*, vol. 26, no. 2-3, pp. 147-176.

Kocev, D.; Vens, C.; Struyf, J.; Dzeroski, S. (2007): Ensembles of multi-objective decision trees. *Proceedings of the 18th European Conference on Machine Learning*, pp. 624-631.

Lichman, M. (2013): UCI machine learning repository. http://archive.ics.uci.edu/ml.

Microsoft (2017): Microsoft azure. https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-algorithm-choice.

Mulan (2010): Mulan: A java library for multi-label learning. http://mulan.sourceforge.net/index.html.

Shotton, J.; Glocker, B.; Zach, C.; Izadi, S.; Criminisi, A. et al. (2013): Scene coordinate regression forests for camera relocalization in RGB-D images. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2930-2937.

Spyromitros-Xioufis, E.; Groves, W.; Tsoumakas, G.; Vlahavas, I. (2014): Drawing parallels between multi-label classification and multi-target regression.

Spyromitros-Xioufis, E.; Tsoumakas, G.; Groves, W.; Vlahavas, I. (2016): Multi-target regression via input space expansion: Treating targets as inputs. *Machine Learning*, vol. 104, no. 1, pp. 55-98.

Torki, M.; Elgammal, A. (2011): Regression from local features for viewpoint and pose estimation. *IEEE International Conference on Computer Vision*, pp. 2603-2610.

Toshev, A.; Szegedy, C. (2014): Deeppose: Human pose estimation via deep neural networks. *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1653-1660.

Tsanas, A.; Xifara, A. (2012): Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy and Buildings*, vol. 49, pp. 560-567.

Tsoumakas, G.; Katakis, I.; Vlahavas, I. (2010): Mining multi-label data. In O. Maimon, L. Rokach (Ed.), *Data Mining and Knowledge Discovery Handbook. 2nd Edition*, Springer.

Tsoumakas, G.; Spyromitros-Xioufis, E.; Vrekou, A.; Vlahavas, I. (2014, September): Multi-target regression via random linear target combinations. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 225-240.

Yeh, I. C. (2007): Modeling slump flow of concrete using second-order regressions and artificial neural networks. *Cement and Concrete Composites*, vol. 29, no. 6, pp. 474-480.

Zhen, X.; Wang, Z.; Islam, A.; Bhaduri, M.; Chan, I. et al. (2014): Direct estimation of cardiac bi-ventricular volumes with regression forests. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 586-593.