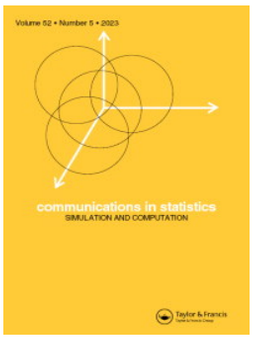

Taylor & Francis
Taylor & Francis Group

Communications in Statistics - Simulation and Computation

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/lssp20

Additive multi-task learning models and task diagnostics

Nikolay Miller & Guoyi Zhang

To cite this article: Nikolay Miller & Guoyi Zhang (2023): Additive multi-task learning models and task diagnostics, Communications in Statistics - Simulation and Computation, DOI: 10.1080/03610918.2023.2212430

To link to this article: https://doi.org/10.1080/03610918.2023.2212430

Published online: 19 May 2023.

Submit your article to this journal

View related articles

View Crossmark data

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=lssp20

COMMUNICATIONS IN STATISTICS - SIMULATION AND COMPUTATION®
https://doi.org/10.1080/03610918.2023.2212430

Taylor & Francis
Taylor & Francis Group

Check for updates

Additive multi-task learning models and task diagnostics

Nikolay Miller and Guoyi Zhang

Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM 87131-0001, USA

ABSTRACT

This paper develops a model for multi-task machine learning that incorporates per-task parametric and nonparametric effects in an additive way. This allows a practitioner the flexibility of modeling the tasks in a customized manner, increasing model performance compared to other modern multi-task methods, while maintaining a high degree of model explainability. We also introduce novel methods for task diagnostics, which are based on the statistical influence of tasks on the model's performance, and propose testing methods and remedial measures for outlier tasks. Additive multi-task learning model with task diagnostics is examined on a well-known real-world multi-task benchmark dataset and shows a significant performance improvement over other modern multi-task methods.

ARTICLE HISTORY
Received 4 June 2022
Accepted 3 May 2023

KEYWORDS
Additive models; Backfitting algorithm; Multi-task learning; Outlier tasks; Support vector machines; Task diagnostics

1. Introduction

Multi-task learning is a subfield of machine learning that learns information from multiple tasks simultaneously, making use of similarities and differences between the tasks. In many research and application contexts, a practitioner is often faced with situations where data can intuitively be separated into tasks, where some information sharing between tasks is evident, while there are also some distinctions between tasks that demand customized attention. The objective of multi-task learning models is to accommodate this information exchange between tasks such that we can obtain a model that has higher predictive power than either training separate models for each task, or one single-task model for all tasks simultaneously.

As we assume that all tasks are somehow related to each other, it is the objective of the multi-task learning model to learn these task relationships. Tasks can be similarly or differently distributed or have clusters of similarly distributed tasks. There may also arise situations where most tasks are similar to each other, while some few tasks differ in their distributions. We denote the latter as outlier tasks. The vast distributional differences may affect the predictive power of the model fit, such that the model may predict poorly on the majority class tasks and on outlier tasks. It is therefore desirable to identify such tasks and to address them specifically to improve the model performance.

Multi-task learning is widely used in fields such as finance, economics, medicine, and education. For example, in finance and economics forecasting, it is often required to predict the value of many possibly related indicators simultaneously (Fiot and Dinuzzo 2015); in stock price prediction, stocks are often related to each other in multi-task fashion (Ghosn and Bengio 1996; Bitvai and Cohn 2015); in bioinformatics, we may want to study tumor prediction from multiple microarray data sets or analyze data from multiple related diseases simultaneously (Y. Zhou et al. (2021)); in small area estimation, we may want to study the small area total estimate from multiple areas simultaneously; in web search, search results can be improved when combining information from multiple geographical markets that exhibit similar or distinct search patterns (Bai, et al. 2009).

CONTACT Nikolay Miller nmill@math.unm.edu Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM 87131-0001, USA.

© 2023 Taylor & Francis Group, LLC

In the context of supervised learning, regularized multi-task learning methods have received significant attention from many researchers. The idea of multi-task mean regularization, where task parameters are penalized individually and with regard to their closeness to the overall mean of all task parameters, was first introduced in Evgeniou and Pontil (2004). This idea was further generalized to be cast into any L2-regularized single-task learning problem using specialized multi-task kernels in Evgeniou, Micchelli, and Pontil (2005). An extensive overview of vector-valued functions and kernels is given in Álvarez, Rosasco, and Lawrence (2012), and learning these functions in RKHS is studied by Micchelli and Pontil (2005). Extension of representer theorem (Kimeldorf and Wahba 1971) to multi-task regularization was studied in Argyriou, Micchelli, and Pontil (2009). Despite a wide variety of models in multi-task learning, there is a need for multi-task model that can account for both parametric and nonparametric effects at the same time.

Identification of outlier tasks has been investigated significantly from an algorithmic perspective. One popular framework is robust multi-task learning, initially proposed in Chen, Zhou, and Ye (2011). Gong, Ye, and Zhang (2012) proposed working with parameters of each task's parametric models, which are defined as $\mathbf{f}_i(\mathbf{x}_j^{(i)}) = (\mathbf{x}_j^{(i)})^T \mathbf{w}_i$, where $\mathbf{w}_i$ are parameter vector of task i. The parameter vectors are then combined as columns in a matrix $W \in \mathbf{R}^{d \times m}$ and decomposed as $W = P + Q$, and separate penalties are imposed on these matrices, such that P encodes the shared features among tasks and Q captures the outlier tasks. The latter step is done in order to simplify the optimization problem. Thus, this method learns both the shared features and outlier tasks. The loss minimization is then performed using a modified version of gradient descent. Gong, Ye, and Zhang (2012) also decomposed the matrix W with group sparsity, while Pu, et al. (2013) proposed to solve a similar problem by using the accelerated proximal method. Chen, Liu, and Ye (2012) introduced a multi-task model which is robust to influence of outlier tasks. Kumar and Daumé (2012) proposed that there exist some basis tasks, and all other tasks can be expressed as linear combinations of these tasks. Zhong et al. (2016) further relax the tasks group structures assumptions to identify them instead. Jeong and Jun (2018) attempt a similar approach by optimizing two coefficient matrices based on a low-rank assumption. It should be noted, however, that a general limitation of these frameworks is that they can only accommodate parametric models, which limits their applicability to nonparametric relationships.

Nonetheless, it is generally the case that the existing works in multi-task learning and task diagnostics for multi-task learning are highly algorithmic in nature, not motivated by the statistical groundwork, and have a major focus on the development of optimization procedures. There is a significant focus on linear and parametric models, limiting their applicability to more complex datasets, where nonparametric and nonlinear effects are often present. Another distinct tendency is the assumption of all tasks are equally related to each other in every cluster. The focus is mostly on clustering the tasks into groups, not on identifying the outlier tasks and performing remedial measures. In this research work, we propose methods that seek to overcome these limitations, have a statistical foundation, and also to generalize the algorithmic clustering methods.

Contributions of this paper are two-fold. First, we introduce an extension of the above framework, the multi-task additive model, that allows combining parametric and nonparametric effects in the same multi-task learning model in an additive way. It allows a practitioner to fully adjust the model to their particular needs, clearly separating linear and nonlinear effects, leading to the maximization of the model's predictive power. Our methods allow for maintaining a high degree of model explainability and require significantly less data for successful training than other modern multi-task methods, such as multi-task neural networks. To achieve this, we use the well-established framework of generalized additive models. As the additive functions of additive models are conventionally found using backfitting algorithm, we introduce its extension to the multi-task additive model. This novel fitting procedure is highly customizable, which allows a practitioner to adjust the fitting process in the most suitable way for the data at hand. Moreover, we introduce a novel model

testing procedure, which allows simultaneous testing of parametric and nonparametric effects in the multi-task additive model, further enhancing the theoretical and statistical qualities of the model.

Second, we introduce novel task diagnostic procedures, which allow an understanding of task influence on model performance in a fully explainable way. Identifying outlier tasks allows further diagnostics into the model's performance, and can help the practitioner understand their dataset better to further improve their model performance. The methods and ideas we used for task diagnostic are intuitive and are based on conventional statistical techniques, yet we are not aware of applications of similar techniques to multi-task learning.

The rest of the paper is organized as follows. In Sec. 2 we introduce the additive multi-task learning model. The task influence test, which is designed to identify and remedy outlier tasks, is introduced in Sec. 3. We then apply the new methods to a real-world benchmark multi-task dataset in Sec. 4 and compare the results with other popular multi-task learning methods. Finally, in Sec. 5 we discuss the ramifications of our work and consider directions for further research.

2. Additive multi-task learning model

In this chapter we introduce the extension of the additive modeling approach to the multi-task learning framework.

First, for a more rigorous setup of the problem, following the notation in Evgeniou et al. (2005), suppose that we have n datapoints $\{(\mathbf{x}_i, y_i)|i \in [1, ..., n]\}$ with $\mathbf{x}_i \in \mathbf{R}^d$ and $y_i \in \mathbf{R}$. A regularized single task kernel learning criterion is a tradeoff between goodness-of-fit and smoothness/complexity of the function, i.e. $\frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \gamma||f||_k^2$, where $||f||_k^2$ is the norm of f in H_k, and H_k is the reproducing kernel Hilbert space generated by a kernel $K(\cdot,\cdot)$. The goal is to find the unique minimizer in H_k to learn a function of f. Multi-task learning is an extension of the single task learning method to m tasks modeled by $f_1, f_2, ..., f_m$ respectively. Let n_i be the size of task i, and $n = \sum_{i=1}^{m} n_i$ be the total number of observations for all tasks. Regularized multi-task learning estimates $f_1, f_2, ..., f_m$ simultaneously by minimizing

$$\frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i} L(y_{i,j}, f_i(\mathbf{x}_{i,j})) + \gamma J(\cdot) \tag{1}$$

where subscript i, j denotes the jth element of the ith task, and $J(\cdot)$ is the square norm of $f_1, f_2, ..., f_m$ in H_k, the reproducing kernel Hilbert space generated by a kernel $K_{l,q}(\cdot,\cdot)$ with $l, q = 1, 2, ..., m$. Evgeniou et al. (2005) proposed linear regularization penalty (2) with associated kernel function (3),

$$J(\cdot) = \frac{1}{n}\left(\sum_{t=1}^{m}||f_t||^2 + \frac{1-\lambda}{\lambda}\sum_{t=1}^{m}||f_t - \bar{f}||^2\right) \tag{2}$$

$$K_{l,q}(\mathbf{x}, \mathbf{t}) = (1 - \lambda + \lambda n \delta_{l,q})\mathbf{x}'\mathbf{t} \tag{3}$$

where $\delta_{l,q}$ is an indicator function that is equal to 1 if $l = q$ and 0 otherwise, $l, q = 1, 2, ..., m$. The parameter $\lambda \in (0, 1]$ controls tradeoff between closeness of each of these functions to their average and a desirable small size of norm of the functions. If λ is small, the task functions are close to their average. If $\lambda = 1$, the task functions are learned independently for a given γ.

For a brief review, for regression problems, additive model is given by Hastie, Tibshirani, and Friedman (2001) as $Y = \alpha + \sum_{i=1}^{p} f_1(X_i) + \epsilon$ with ϵ being an error term with mean 0, and $\sum_{i=1}^{n} f_j(x_{i,j}) = 0, \forall j$. The functions f are additive in their effect on the response variable Y, and they can be either parametric for linear relationships or nonparametric for nonlinear

relationships. They can be found using backfitting algorithm, which is an iterative procedure that estimates each of the functions sequentially until their convergence.

2.1. Definition

In the following, we define additive multi-task learning model, which allows combining parametric and nonparametric effects in a per-task fashion.

Let $y_{i,j}$ be an observation j in task i, where $i = 1, 2, ..., m$ and $j = 1, 2, ..., n_i$. Let $\mathbf{y} = [y_{1,1}, y_{1,2}, ..., y_{1,n_1}, y_{2,1}, ..., y_{m,n_m}]'$, $\boldsymbol{\epsilon} = [\epsilon_{1,1}, \epsilon_{1,2}, ..., \epsilon_{1,n_1}, \epsilon_{2,1}, ..., \epsilon_{m,n_m}]'$, $\mathbf{f} = [f_1(\mathbf{z}), f_2(\mathbf{z}), ..., f_m(\mathbf{z})]'$; let I_m be an $m \times m$ identity matrix, $\mathbf{1}_{n_i}$ be the vector of n_i 1s, $\mathbf{Z} = I_m \otimes [\mathbf{1}'_{n_1}, \mathbf{1}'_{n_2}, ..., \mathbf{1}'_{n_m}]'$ where $\otimes$ is Kronecker product.

We fit the additive multi-task learning model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{f} + \boldsymbol{\epsilon} \tag{4}$$

where $\mathbf{X}$ is a certain design matrix related to the linear term structure, and error terms $\boldsymbol{\epsilon}$ are uncorrelated with mean zero and separate variance in each task, i.e. Var $(\epsilon_{i,j}) = \sigma_i^2$ for all i. The structure of matrix $\mathbf{Z}$ is inspired by the model matrix in analysis of variance (ANOVA) with a block design. This matrix is fixed, and is designed to separate the multi-task models. Matrix $\mathbf{X}$ is a design matrix for the linear term, and vectors $\mathbf{z}$ denote datapoints. These can be either the same, or different, depending on the model design. Note also that the linear part $\mathbf{X}\boldsymbol{\beta}$ is parametric, while the individual functions $f_i(\mathbf{z})$ are nonparametric in the term $\mathbf{Z}\mathbf{f}$.

Using a backfitting procedure as a motivation (Hastie, Tibshirani, and Friedman 2001), we propose Algorithm 1 to estimate $y_{i,j}$:

Algorithm 1: fitting the additive multi-task model

Initialize $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}\mathbf{y}$ based on a reduced model: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ for i and j;
repeat
  Consider the residual model $\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{Z}\mathbf{f} + \boldsymbol{\epsilon}$, where $\widehat{\mathbf{f}}$ can be solved by regularization criterion such as (1) using support vector regression (SVR);
  Update $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}(\mathbf{y} - \mathbf{Z}\widehat{\mathbf{f}})$;
until $\widehat{\boldsymbol{\beta}}$ *converges*;

where $\boldsymbol{\beta}$ are fitted with weighted least squares in order to adjust for different variances in each tasks, and the reciprocals of group variances are estimated as weights in the weight matrix $\mathbf{W}$, inspired by approach for constant values of response variables described in Montgomery, Peck, and Vining (2012).

The additive linear and nonlinear effects can be designed to fit a particular problem differently for each task, which makes this approach incredibly useful in the multi-task learning framework. Casting the multi-task model in an additive way allows a great flexibility in a range of statistical models that can be applied to a particular problem, and allows the researcher to explicitly specify the parts of the model. From a multi-task learning perspective, it allows clear separation of various task effects, and can lead to a great increase in performance with a suitable design.

To further demonstrate the flexibility of this definition, in the experimental section we define matrix $\mathbf{X}$ to capture multi-task effects through a one-way ANOVA model, while let vectors $\mathbf{z}$ to be the original predictor variables.

The iterative algorithm of fitting the model is inspired by the backfitting algorithm, but it also differs in a major way. The original backfitting algorithm has p components (usually the number of predictors), while our algorithm has only 2 components: the linear and nonlinear parts, as $\mathbf{X}\boldsymbol{\beta}$ and

Zf are additive to each other. The first iteration fits the linear estimator to the design matrix **X**. The initial fitted linear part gets deducted from the response variable in the second step, and the nonparametric model is applied. Then, the parametric coefficients get updated with the response adjusted for the nonlinear part, and the algorithm runs until the linear coefficients converge, which is also the point where the nonlinear part converges. An alternative stopping criterion can be defined, for example the maximization of explained variance or minimization of a loss function.

This algorithm of fitting the additive multi-task model can be seen as a form of continuous bias correstion. At first, some variability is removed by the linear part of the model, then remaining variability is captured by the nonlinear part. This, in turn, allows the linear part to reduce the variability further, and the process continues until the convergence criteria are satisfied. Thus, linear and nonlinear models are continually correcting each other's bias. However, because of the bias-variance tradeoff, correcting the bias can increase variance. This can lead to overfitting, making the model's prediction less generalizable to new, unseen data. Therefore, particular care should be taken when devising the stopping criterion.

2.2. Generalization of the fitting algorithm

In the following, we seek to extend the principle of Algorithm 1, in order to further generalize the fitting procedure and make it more customizable, thus allowing for further increase in the predictive power of the model.

Note that there are two distinct steps in the model fitting procedure. First, the parametric component $\boldsymbol{\beta}$ is fitted, and in the second step, the nonparametric component $\mathbf{f}$ is fitted on a residual model from the first step. This is designed to be one loop of the algorithm. After every loop, the stopping criterion is checked to see whether the performance has improved or worsened. If the performance keeps improving, it is reasonable to run further loops; otherwise, the algorithm should be stopped. In Algorithm 1, no consideration is taken for the intermediate model, inside the loop. However, there may arise circumstances where the intermediate model performs better than the model of one full loop. Thus, in order to improve this procedure, we generalize this algorithm to cover these intermediate cases.

In Algorithm 2, stopping criterion and metric are flexible. Denote this metric as M. Then,

Algorithm 2: algorithmic model selection for additive multi-task model

Initialize $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}\mathbf{y}$ based on a reduced model: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$;
Measure M and denote as M_0;
repeat
  Solve for $\widehat{\mathbf{f}}$ in the residual model $\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{Z}\mathbf{f} + \boldsymbol{\epsilon}$;
  Measure M and denote as M_1;
  if *M_0 is better than M_1* **then**
    Keep the previous model and break the loop
  else
    Set $M_1 \rightarrow M_0$
  Update $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}(\mathbf{y} - \mathbf{Z}\widehat{\mathbf{f}})$;
  Measure M and denote as M_1;
  if *M_0 is better than M_1* **then**
    Keep the previous model and break the loop
  else
    Set $M_1 \rightarrow M_0$
until the loop breaks;

Algorithm 2 is designed to run the model fitting procedure on the additive multi-task model until the performance metric is not improving anymore.

Let $K+1$ be the total number of loops until the stopping criterion is triggered. At the loop $K+1$ it would be decided that the new model is not as good as the previous one, thus the new model is discarded and the previous model from K loops is chosen as the final model.

It is assumed that the performance metric values can be ranked against each other quantitatively, such that it is possible to select one with the highest performance. Thus, M_0 is better than M_1 implies that M_0 and M_1 can be ranked, and depending on the numerical scale, one is higher/-lower than or equal to the other. For example, if explained variance is the metric for the stopping criterion, and the algorithm stops when explained variance starts to decrease. Other examples of performance metrics include mean-squared error *MSE*, Akaike information criterion *AIC*, Bayesian information criterion *BIC*, and deviance information criterion *DIC*.

It can be seen that the Algorithm 1 is a special case of a more general Algorithm 2. The decision of stopping the loop is done at multiples at even multiples of K. Thus the first K value where the loop can stop is at $K=2$, then after the second loop can stop at $K=4$ after the second loop, and so on.

2.3. Testing full and reduced models

In this section we consider a further generalization of the additive multi-task model and propose model testing procedures. These testing procedures further extend the applicability of the additive multi-task model, improving model diagnostics and predictive power.

Recall the Equation 4, which defines the additive multi-task model: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{f} + \boldsymbol{\epsilon}$. The model makes assumptions of uncorrelated error terms, and different variances in each task, and in the Algorithm 1 the model fit of the parametric part $\boldsymbol{\beta}$ is performed using a weighted least-squares algorithm to implement that assumption.

Relaxing these assumptions, consider the model of Equation 4 to have independent and identically distributed error terms $\boldsymbol{\epsilon}$ with mean 0, and assume that the parametric part $\boldsymbol{\beta}$ can be estimated by any regression approach. It can be seen that these conditions make the model more general, as the purpose of this generalization is to open for further customizability. Note that with these assumptions, the only difference between the additive multi-task model and partially linear model is the separation of nonparametric components into tasks and the fitting algorithm which can go in multiple loops, compared to only one loop for the partially linear model (see for example Engle et al. (1986)).

Under these assumptions, the model is very general, and it now includes most of the other previously covered models as special cases. In particular, the additive multi-task model of Sec. 2 is a special case with error terms having different variances within each task, and parametric fit with WLS. However, this is a full model, as both parametric and nonparametric components are present in the model.

Usually, we can't be sure of the correctness of any model until we test all the assumptions. Therefore, the important class of special cases is reduced models. Both the parametric part and the nonparametric part can be evaluated for their suitability to be included in the model.

To test the appropriateness of the parametric part, the decision is to be made on the null hypothesis:

$$H_0 : \boldsymbol{\beta} = \mathbf{0} \tag{5}$$

and to test the nonparametric part, a decision is to be made on

$$H_0 : \mathbf{f} = \mathbf{0} \tag{6}$$

The decisions on these hypotheses can be done with model selection and diagnostics tools. It can be seen that the set of techniques that can be used to test these hypotheses can potentially

include a large part of the statistical inference field. For example, both tests in Equations 5 and 6 can be separately tested using model complexity penalty measures such as Akaike Information Criterion, Bayesian Information Criterion, or minimum description length. Test for the parametric part $\boldsymbol{\beta}$ of Equation 5 can be performed using general linear test (F-test). Some notable nonparametric model selection procedures include those proposed by Yang (1999) and Wegkamp (2003).

In particular, the parametric part of the model in Equation 5 can be tested by applying model testing procedures on $\mathbf{y} - \mathbf{Z}\widehat{\mathbf{f}}$. In the Algorithms 1 and 2, the model fitting happens on the parametric part first, and then the residual model is trained nonparametrically. In order to facilitate testing on $\mathbf{y} - \mathbf{Z}\widehat{\mathbf{f}}$, the algorithms would need to fit the models in a reverse order: first nonparametrically, and then the residual model can be tested with $H_0 : \boldsymbol{\beta} = \mathbf{0}$. For example, a decision on this H_0 can be done with a general linear test. Also, note that since the parametric part has individual components, these can be tested as separately as $H_0 : \beta_j = \mathbf{0}$ for element j in $\boldsymbol{\beta}$. Variable selection procedures can be well-suited, for example, test of model coefficients in OLS regression.

For partially linear models, Eubank (1999) states the conditions for the test of the parametric part of the model in a special case when nonparametric smoothing is used for estimating the nonparametric part $\mathbf{f}$. The estimator of $\boldsymbol{\beta}$ becomes:

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^T(\mathbf{I} - \mathbf{S})^T(\mathbf{I} - \mathbf{S})\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{I} - \mathbf{S})^T(\mathbf{I} - \mathbf{S})\mathbf{Y} \tag{7}$$

where $\mathbf{S}$ is a smoother matrix of a kernel type estimator for the nonparametric fit of $\mathbf{f}$. It can be shown that $\widehat{\boldsymbol{\beta}}$ achieves asymptotic Normal distribution, and the decision on $H_0 : \boldsymbol{\beta} = \mathbf{0}$ can be done on the basis of the t-statistic:

$$t = \frac{\widehat{\boldsymbol{\beta}}}{\hat{s}\sqrt{\mathbf{X}^T(\mathbf{I} - \mathbf{S})^T(\mathbf{I} - \mathbf{S})\mathbf{X}}} \tag{8}$$

where $\hat{s} = \sqrt{\frac{\sum(y_i - \overline{\mathbf{Y}})^2}{n-1}}$ is a sample standard deviation of $\mathbf{Y}$.

For the two-sided alternative $H_A : \boldsymbol{\beta} \neq \mathbf{0}$, we reject H_0 at α confidence level if $|t| > t_{\alpha/2}$, where $t_{\alpha/2,(n-1)}$ is $\alpha/2$-th percentile of the t-distribution with $n-1$ degrees of freedom.

Reduced versions of the additive multi-task model include conventional parametric and nonparametric models as special cases. As the model fit is done with Algorithm 1, or its general version in Algorithm 2, the model performance can be further strengthened even in cases of reduced models. Note that the initial definition of the additive multi-task model in Equation 4 is also a special case of this generalization, although it is not reduced in the parameter space. In the subsequent real dataset example, an analysis of the full version and two restricted versions of the model is performed.

3. Task influence test

In multi-task learning, we assume that all tasks are somehow related to each other, and it is the objective of the multi-task learning model to learn these task relationships. Tasks can be similarly or differently distributed, or have clusters of similarly distributed tasks. There may also arise situations where most tasks are similar to each other, while some few tasks differ in their distribution. We denote the latter as outlier tasks. The vast distributional differences may affect the predictive power of the model fit, such that the model may predict poorly on the majority class tasks and on outlier tasks. It is therefore desirable to identify such tasks and to address them specifically to improve the model performance.

We propose an approach to identify outlier tasks inspired by Cook's distance procedure in regression analysis in Cook (1977) and Cook (1979). For each observation in a regression model, Cook's distance measures the influence of deleting that observation on a predictive performance

of that model. The observations with large Cook's distance values may be outliers and/or highly influential, and merit closer examination.

For multi-task learning, we seek to find outlier tasks instead, as it may not be suitable to consider each observation individually as in Cook's distance procedure. Thus, we can consider the influence of a task by deleting this task from the training data, refitting the model, and checking the its influence on the predictive power on the training set. We also seek to keep the deleted task observations in the training set for the purposes of measuring the performance of a model with that task deleted. That way, all the models with the deleted tasks can have their performance measured on the same dataset, which allows the easier identification of the outlier tasks.

Using this procedure we obtain *deleted explained variance* (*DEV*) for each task. For task i, DEV_i is defined as

$$DEV_i(\mathbf{Y}_{train}, \hat{\mathbf{Y}}^{(i)}_{train}) = \frac{\mathrm{Var}(\mathbf{Y}_{train}) - MSE(\mathbf{Y}_{train}, \hat{\mathbf{Y}}^{(i)}_{train})}{\mathrm{Var}(\mathbf{Y}_{train})} \tag{9}$$

where $\hat{\mathbf{Y}}^{(i)}_{train}$ are fitted values of all tasks' training set, based on a model which is trained with task i deleted from the training set.

If the DEV is high, then deleting this task has improved the overall model fit, and this task can be considered to be deleted. If the DEV is low, then deleting this task has significantly worsened the overall model fit, therefore such task should be kept in the model. Thus we seek to remove tasks with high DEV. We can do so by computing the difference of the overall explained variance, when all tasks are present in the model, with the DEV. This defines the deleted explained variance test statistic for task i as D_i:

$$D_i = EV(\mathbf{Y}_{train}, \hat{\mathbf{Y}}_{train}) - DEV_i(\mathbf{Y}_{train}, \hat{\mathbf{Y}}^{(i)}_{train}) \tag{10}$$

where EV is the overall explained variance with all tasks' training sets used both for training and testing.

When D_i is high, then the DEV is low, and vice versa. Using the overall explained variance in this way gives us a natural threshold level to identify outlier tasks. Even though only tasks with positive D_i are candidates for being outlier tasks, it may not be the best strategy to delete all tasks with the positive D_i. A good strategy can be to start with investigation and deleting the task with the lowest D_i, and move forward in a sequential procedure, removing tasks one-by-one until we find a task combination which results in the highest explained variance in the final model (or the lowest error rate).

Algorithm 3: task influence testing procedure

```
fit the model to training data of all tasks together;
use the model to predict the training set data of all tasks together;
measure the overall explained variance EV;
for each task do
    remove its observations from the training data;
    fit the model on the data with this task deleted;
    use the model to find fitted values for the training set of all tasks together;
    measure and record the deleted explained variance (DEV_i) for this task;
    compute D_i, the difference of overall explained variance and DEV;
rank the tasks by D_i from lowest to largest;
remove the tasks with D_i lower than some threshold value τ;
use the remaining tasks to fit the final model;
```

The task influence testing procedure is summarized in Algorithm 3. For illustration, this example uses threshold value τ, such that tasks with lower D_i than τ are regarded as outlier tasks. The value of τ can be chosen using cross-validation for the best performance. In the following, we introduce two ways that the parameter τ can be chosen.

3.1. t-distribution cutoff values for task outliers

In this section, we establish a distributional approach to choosing the parameter τ for identification of outlier tasks. Figure 1 displays each task's D_i and ranks them from lowest to highest. Note that this plot has characteristics of a QQ plot and that D_i values look similar to a sample from a normal distribution, although with rather fat tails. This means that the cutoff values can be devised in a manner of a hypothesis test. If the D_i have approximate normal distribution, we assume that the standardized values of D_i are drawn from a t-distribution with $m-1$ degrees of freedom.

In this setup, it is natural to consider a cutoff value that separates the t-distribution to acceptance and rejection regions. The acceptance region is defined as an area in the t-distribution with $m-1$ degrees of freedom where tasks are not considered outliers. The rejection region is an area where tasks are considered to be outliers and are deleted from the training set. Since tasks with very low D_i are considered outlier tasks, their standardized values of D_i are far in the left tail of the t-distribution with $m-1$ degrees of freedom. It means that the rejection region should be defined to be an area of certain probability in the left tail of t-distribution.

In other words, let α be a chosen confidence level. Let $t(\alpha, m-1)$ be a α-quantile of t-distribution with $m-1$ degrees of freedom. Define the acceptance region as

$$\Theta_0 = \left\{t(\alpha, m-1), +\infty\right\} \tag{11}$$

and the rejection region as

$$\Theta_1 = \left\{-\infty, t(\alpha, m-1)\right\} \tag{12}$$

Let $\bar{D} = \frac{1}{L}\sum_{i=1}^{m} D_i$ be a sample average and $SD(D)$ be a standard deviation of D_i for all m tasks. The implication of acceptance region is that $P(\frac{D_i-\bar{D}}{SD(D)} \in \Theta_0) = 1-\alpha$, such that we can conclude with $1-\alpha$ confidence that if D_i for task i is in the acceptance region, then the task is drawn from the same distribution as other tasks. Otherwise, if the D_i of task i is in the rejection region, such that $\frac{D_i-\bar{D}}{SD(D)} \in \Theta_1$, then it is an outlier task, and it is deleted from the training set.

This method of choosing τ is inspired by traditional statistical hypothesis testing methods. Usually, such inferential testing is done to make a conclusion about a particular statistic that has a sampling distribution and its properties often depend on the sample and its size. However, the task outlier procedure is quite different. Every task has its individual D_i value, and every D_i can

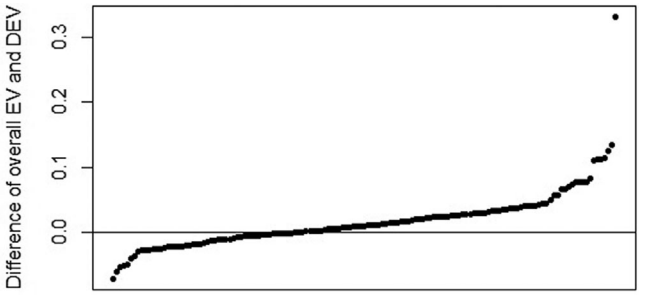

Figure 1. Tasks are ranked according to their D_i values. Most differences are positive, meaning that deleting these tasks worsens the model performance. The task furthest to the right has a particularly positive DEV difference, indicating its high importance.

be considered a statistic as it is based on a sample for task i. And these can have either the same or different distributions across the tasks in the data. The initial assumption in the cutoff values approach is that all D_i have the same distribution, thus they come from the same population. Then, for each D_i a test is performed, to measure whether this task is consistent with the overall distribution of all tasks' D_i values. Then if the task deviates substantially from the overall task distribution, it means that it is potentially an outlier task. However, it should only be considered for deletion if it influences the overall model performance in a negative way, i.e. if D_i is lower than for other tasks. If a task has unusually high D_i compared to other tasks, then it influences the model performance in a positive way, so it should not be deleted, despite it being highly influential.

3.2. Kernel density estimation of D_i

For cases where the distribution of D_i deviates from Normal, we propose to apply kernel density estimation for D_i and to use low percentile values from this density as cutoff criteria for the task rejection regions. The theory of kernel density estimation is covered in detail by Scott (1992) and Eubank (1999). To implement it, we use R built-in package "stats" (R Core Team (2022)) and its function "*density*".

For the purposes of this procedure, assume that deleted explained variance statistics $(D_1, D_2, ..., D_m)$ are independent and identically distributed from a univariate distribution given by probability density function s at any given point D. We are interested in estimating this probability density function with the kernel density estimation method, given by:

$$\hat{s}(D) = \frac{1}{mh}\sum_{i=1}^{m} K\left(\frac{D - D_i}{h}\right), \tag{13}$$

where $h > 0$ is smoothing bandwidth, K is a non-negative kernel function and m is the number of tasks in the data.

The function "*density*" has an option for 7 different kernels, which will all be attempted for the experiment on the ILEA schools data: Gaussian, Epanechnikov, rectangular, triangular, biweight, cosine, and optcosine. For the smoothing bandwidth rule, which decides how the parameter h is estimated, we choose to use Silverman's "rule of thumb" approach (Silverman (1986)), which is built-in to function "*density*" as parameter "*nrd0*" for smoothing bandwidth.

Negative values of D_i indicate that deleting the task i from the training data has increased the explained variance. Thus, the cutoff value is defined in terms of low percentiles of estimated density. Let $\hat{s}_\alpha$ denote its $\alpha \times 100\%$-th percentile. Thus the rejection region is $\Theta_1 = \{-\infty, \hat{s}_\alpha\}$ and the acceptance region is $\Theta_0 = \{\hat{s}_\alpha, +\infty\}$. The tasks in the rejection region are to be deleted from the final model's training data, while tasks in the acceptance region are kept for the model to be trained on.

4. Application to a real-world dataset

In this chapter, we demonstrate the effectiveness of our methods by applying it to the real-world dataset. The school effectiveness data by Inner London Education Authority has become popular benchmark in the multi-task learning literature. Some of the analyses of this dataset were performed in Evgeniou and Pontil (2004), Evgeniou et al. (2005), Liao and Carin (2005), Argyriou, Evgeniou, and Pontil (2008), Zacharia (2009), Agarwal et al. (2010), Romera-Paredes et al. (2013), Fang and Tao (2015), Kim and Mowakeaa (2019). The metric that is often considered in the literature when analyzing this dataset is explained variance, which is defined in Bakker and Heskes (2003) as percentage of test set variance minus sum of squared errors of the model on the test

set, taken as a percentage over the test set variance. The best performance of Evgeniou, Micchelli, and Pontil (2005) is ca. 34.5% using SVM with linear mean-regularized multi-task kernel.

The dataset contains records of 15362 students information in 139 schools. The primary response variable is exam score, and predictors are years, % of students eligible for free school meals in the school, gender, VR band of the student, percent of students in VR band in the school, student ethnicity, school gender and school denomination. The school number plays a role of a predictor variable, but is not considered a separate column in the data matrix; rather it is a task indicator which informs the multi-task learning model. We find 15 observations with VR band equal to 0, which is a level that is not mentioned in the dataset description, so we make an assumption that these students belong to VR band 1. Therefore, after one-hot encoding, there are a total of 26 predictor variables. Moreover, we find one female student in an all-male school 44, which we assume is a typo and edit the gender variable to male. We follow the procedure of Evgeniou, Micchelli, and Pontil (2005) and split the data with 75%-25% train-test split ratio within each task.

The literature on this dataset often considers only 10 train-test splits. However, in our experiments we've compared results of different 10 train-test splits and found that the results can be unstable between different 10 splits. Thus, in order to give our results more robustness and stability, we are using 100 train-test splits, and average out our results across these splits.

4.1. Additive multi-task model application

The motivation of the model construction in this section comes from our observation of differing distributions of response variable in each task.

We observe that the means and standard deviations for each task have quite strong deviations from their estimates for all tasks together. These can dramatically affect the model performance, and we seek to address this specifically through our framework.

4.1.1. Overview

I apply the additive multi-task model (4) in the following way. The matrix $\mathbf{X}$ is defined to be a design matrix of one-way ANOVA to estimate each task's training set mean. This captures each task's group effect, and after decentering, the residual model with the response variable $\mathbf{y} - \mathbf{X}\boldsymbol{\beta}$ allows the distributions of tasks to be closer to each other. This can be seen as a variation of a block design where each task is a block. By applying a simple one-way ANOVA, the block effect is estimated and is removed from the analysis by calculating the residual for the further estimation with the residual model in the next step. Moreover, we adjust for different variances in each task by fitting $\boldsymbol{\beta}$ by weighted least squares with the reciprocals of task sample variances as weights in the weight matrix $\mathbf{W}$. In combination with centering, it has the effect of standardization of all observations in each task by using estimates of each task's mean and standard deviation.

As mentioned above, the purpose of the one-way ANOVA is to decenter the tasks separately. We can regard each task as a separate level of factor so that the ANOVA design matrix can be created to find the group (task) means after fitting the parametric part of the model. Defined in Kutner, et al. (2005) and Christensen (2016), the design matrix $\mathbf{X}$ for one-way ANOVA is $n \times m$ matrix, where n is the total number of observations and m is the number of tasks (factor levels). The column j indices observations for task j with integers 1 in rows that belong to task j, and 0 in all other rows. The parameters $\boldsymbol{\beta}$ are encoded in a $m \times 1$ column vector of unknown task (group) means:

$$\boldsymbol{\beta} = [\mu_1, \mu_2, ..., \mu_m]^T \tag{14}$$

where μ_j is the unknown mean of task j.

Decentering the tasks in this way addresses the implicit assumption that every task has its own unknown mean of the response variable. Moreover, fitting the parameters with weighted least squares addresses the assumption in 2 that every task j has its own variance, different from other tasks.

In the next step, we solve for $\widehat{\mathbf{f}}$ in $\mathbf{Zf}$ by applying SVR with Gaussian kernel on a dataset which is transformed with a mean-regularized multi-task kernel by Evgeniou, Micchelli, and Pontil (2005) as described in Sec. 1.

After the above two steps, we may consider that the model training is finished, completing one full loop of Algorithm 1. Note that this is equivalent to running $K=2$ repetitions in the Algorithm 2.

As the dataset has the highest performance with low lambdas, for demonstration purposes we choose two low values of lambdas and unite their fitted values using the multi-task combined estimate, such that the best value of θ_l is chosen for each task using cross-validation on the test set. The combinations of λ values to be considered are 0 together with 0.01 and 0.1.

4.1.2. Definitions

Let's first consider how this configuration can be stated in a form of statistical model. Recall the definition of multi-task additive model in Equation 4: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Zf} + \boldsymbol{\epsilon}$. As already mentioned, we define the matrix $\mathbf{X}$ to be a $n \times m$ one-way ANOVA matrix, with weighted least squares fit on $\boldsymbol{\beta} = [\mu_1, \mu_2, ..., \mu_m]^T$ that together facilitates standardization of response variable in each task. Recall that $\mathbf{f} = [f_1(\mathbf{z}), f_2(\mathbf{z}), ..., f_m(\mathbf{z})]'$, where $\mathbf{z}$ are inputs to the functions, and that matrix $\mathbf{Z}$ facilitates functions $f_j(\mathbf{z})$ to link with their respective tasks.

Define $\mathbf{f}^{[q]} = [f_1(\mathbf{z})^{[q]}, f_2(\mathbf{z})^{[q]}, ..., f_m(\mathbf{z})^{[q]}]'$, $\mathbf{X}^{[q]}$ and $\boldsymbol{\beta}^{[q]}$ to be nonparametric functions, parametric design matrix, and parametric component vector for model q. Then, the output of model q is:

$$\mathbf{y}^{[q]} = \mathbf{X}^{[q]}\boldsymbol{\beta}^{[q]} + \mathbf{Zf}^{[q]} + \boldsymbol{\epsilon} \tag{15}$$

where the matrix $\mathbf{Z}$ is constant between the models since its function is unaffected by a model choice.

In the following, we introduce combined estimation. We assume that we operate with Q model outputs, and combine them together in a general form of combined estimate. For task l, we have that:

$$\hat{y}_l^{comb} = \sum_{q=1}^{Q} \theta_{l,q} \hat{y}_l^{[q]} \tag{16}$$

where $\theta_{l,q}$ is a weight parameter for task l trained from model q.

For all the tradeoff parameters, we have conditions that $\sum_{q=1}^{Q} \theta_{l,q} = 1$, and $0 \leq \theta_{l,q} \leq 1 \ \forall q \in \{1, ..., Q\}$. Therefore, as $\theta_{l,Q} = 1 - \sum_{q=1}^{Q-1} \theta_{l,q}$, it is only necessary to search among the other $Q-1$ tradeoff parameters for every task l. It follows that for task l:

$$\hat{y}_l^{comb} = \sum_{q=1}^{Q-1} \theta_{l,q} \hat{y}_l^{[q]} + \left(1 - \sum_{q=1}^{Q-1} \theta_{l,q}\right) \hat{y}_l^{[Q]} \tag{17}$$

Let $\boldsymbol{\Theta}$ be a $m \times (Q-1)$ matrix of all parameters, where (l, q)-th element is $\theta_{l,q}$. This matrix omits a column for the Q-th model since its weight is constrained by all other $Q-1$ weights.

Let $\odot$ be a Hadamard product, i.e. element-wise product of two vectors or matrices of the same dimensions. Let n_l be the number of observations in task l, such that $\sum_{l=1}^{m} n_l = n$ is the total number of observations in the whole data (without loss of generality). Let $\boldsymbol{\Theta}_q$ be a $n \times 1$ column vector where elements 1 to n_1 are $\theta_{1,q}$ for $l=1$, and $(\sum_{i=1}^{l} n_i)$ to $(\sum_{i=1}^{l+1} n_i - 1)$ are $\theta_{l,q}$ for $l \in \{2, ..., m\}$. Then the general version of combined estimate is:

Table 1. Results of the mean and standard deviation of explained variance on the test set for the ILEA schools data.

Method	mean EV ± s.d.
Regularized MTL (Evgeniou and Pontil 2004)	34.8 ± 0.5
MTL-FEAT (Gaussian kernel) (Argyriou, Evgeniou, and Pontil 2008)	37.6 ± 1.0
Multi-boost–unweighted (Chapelle et al.(2011))	37.7 ± 1.2
Additive MTL - combined estimate of $\lambda \in \{0, 0.01\}$	38.86 ± 1.07
Additive MTL - combined estimate of $\lambda \in \{0, 0.1\}$	39.07 ± 1.06

The results for regularized MTL, MTL-FEAT, and multi-boost are as in Chapelle et al. (2011) and averaged over 10 train-test splits. Additive multi-task learning model results are averaged over 100 train-test splits.

$$\mathbf{y}^{[comb]} = \sum_{q=1}^{Q-1} \Theta_q \odot \mathbf{y}^{[q]} + \left(1 - \sum_{q=1}^{Q-1} \Theta_q\right) \odot \mathbf{y}^{[Q]} \tag{18}$$

In this demonstration on ILEA schools data, we apply a case of $Q = 2$, which is the conventional version of the combined estimate with two model outputs.

In the following, we describe the algorithm for this example to combine the model outputs, combining two models that use a mean-regularized multi-task kernel with λ_1 and λ_2. Notation wise, $\widehat{\mathbf{Y}}_{test}^{[j]}$ denotes prediction of model j on the test set.

Algorithm 4: combined estimation model training for the mean-regularized kernel with λ_1 and λ_2

Select $\mathbf{Y}_{train}$ and $\mathbf{X}_{train}$;
Train the models 1 and 2 with mean-regularization parameters λ_1 and λ_2, respectively;
Predict on $\mathbf{X}_{test}$ with models 1 and 2, obtaining $\widehat{\mathbf{Y}}_{test}^{[1]}$ and $\widehat{\mathbf{Y}}_{test}^{[2]}$;
foreach *task* $l \in \{1, 2, ..., m\}$ **do**
  Select observations of task l in $\mathbf{X}_{test}$, $\mathbf{Y}_{test}$, $\widehat{\mathbf{Y}}_{test}^{[1]}$ and $\widehat{\mathbf{Y}}_{test}^{[2]}$;
  foreach $\theta \in \{0, 0.01, 0.02, ..., 1\}$ **do**
    For all task observations, compute combined estimate
    $\hat{y}_l^{comb} = \theta \hat{y}_l^{[1]} + (1 - \theta)\hat{y}_l^{[2]}$, obtaining $\widehat{\mathbf{Y}}_{l,test}^{[comb]}$;
    Compute and record $EV(\mathbf{Y}_{l,test}, \widehat{\mathbf{Y}}_{l,test}^{[comb]})$ that is associated with θ;
  Choose $\theta_l = \arg\max_\theta EV(\mathbf{Y}_{l,test}, \widehat{\mathbf{Y}}_{l,test}^{[comb]})$;
  The final prediction for the task l are to be computed as
  $\hat{y}_l^{comb} = \theta_l \hat{y}_l^{[1]} + (1 - \theta_l)\hat{y}_l^{[2]}$;

To summarize, the additive MTL models in this experiment implement the setup of ANOVA and SVR, together with the combined estimation of models trained on datasets transformed with mean-regularized MTL kernel with $\lambda = 0$ and another small value of λ. For each task i, the optimal θ_i is found in the test set of that task. The optimal C and γ parameters of SVR with Gaussian kernel are found using cross-validation for each λ.

4.1.3. Experimental results

The results of the experiments are stated in Table 1, along with the results of state-of-the-art methods in the literature. The additive multi-task model outperforms the best performance by Chapelle et al. (2011) by about 1.3%, indicating a significant improvement over the existing methods. The model with $\lambda = 0.1$ allows a wider tradeoff between single-task learning and mean-regularization and achieves better performance than the model with $\lambda = 0.01$, where this tradeoff is not so wide. A deeper examination of other values of λ, running further loops in the Algorithm 1, and

customizing the linear and nonlinear components for this particular dataset are the directions that are some possible ways to further improve the performance, and are left for further research.

4.2. Task outlier identification

In this section, we demonstrate the use of task outlier identification procedures in Sec. 3. We consider two possible methods of choosing τ from t-distribution and through kernel density estimation. It is an open research question to investigate other methods of finding τ cutoff values.

To assist with the procedures, we introduce a new performance metric, inspired by explained variance of Bakker and Heskes (2003).

Definition 1. *The fraction of total variation in the dataset that is explained by a model trained on this dataset is measured by the total explained variance as:*

$$TEV(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{\mathrm{Var}(\mathbf{Y}) - MSE(\mathbf{Y}, \hat{\mathbf{Y}})}{\mathrm{Var}(\mathbf{Y})} \times 100\% \tag{19}$$

The purpose of the total explained variance *TEV* metric is to cover those cases where no train-test splitting is done.

4.2.1. τ chosen in t-distribution

In order to evaluate the procedure, we choose to use the whole dataset, without splitting it into training and test sets, just as is often done in statistical practice. We will use a conventional value of $\alpha = 0.05$. $m = 139$ in ILEA schools data. Therefore, the cutoff value in t-distribution is $t(0.05, 138) = -1.656$. It implies the acceptance region of $\Theta_0 = \{-1.656, +\infty\}$ and the rejection region of $\Theta_1 = \{-\infty, -1.656\}$. All tasks with a standardized D_i in the rejection region are to be considered outlier tasks and deleted from training the final model, and all tasks in the acceptance region are to be kept for training the final model.

I model the data with mean-regularized multi-task kernel support vector regression with Gaussian kernel. The optimal parameters γ and C were found using cross-validation on the whole dataset. The task coupling parameter $\lambda = 0.1$. Under these parameters for ILEA schools data, $TEV(\mathbf{Y}, \hat{\mathbf{Y}}) = 52.605\%$, $\bar{D} = 0.38$ and $SD(D) = 0.24$. Plot of estimated density for the standardized D_i is illustrated in Figure 2.

Note that this estimated density curve shows that the distribution of D_i has two peaks, the right tail is rather far out, while there is no left tail, so there is a violation of the assumption that D_i have a normal distribution. There are some tasks with unusually high D_i which perform exceptionally well compared to other tasks, thus making the distribution right-skewed. Deleting these tasks would diminish the overall model performance.

The lowest standardized D_i for this dataset is -1.45, which does not fall into the rejection region of $\Theta_1 = \{-\infty, -1.656\}$. Therefore, with 95% confidence, we conclude that there are no outlier tasks, and all tasks need to be included for training the final model.

ILEA data seems to be strongly affected by a few tasks with extraordinarily great performance. For example, the highest standardized D_i is 3.77. It is a known fact that right-skewed distributions have their mean skewed to higher values.

Nonetheless, the possible benefits of using the t-distribution cutoff values are potential of great use. An open research question is whether the positive task outliers can also be considered to be outside of the overall task distribution, and deleted from the standardization procedure for the purposes of this test only. In this way, the standardized D_i values of other tasks can become more stable, which can potentially ease the identification of the detrimental task outliers with cutoff values from t-distribution.

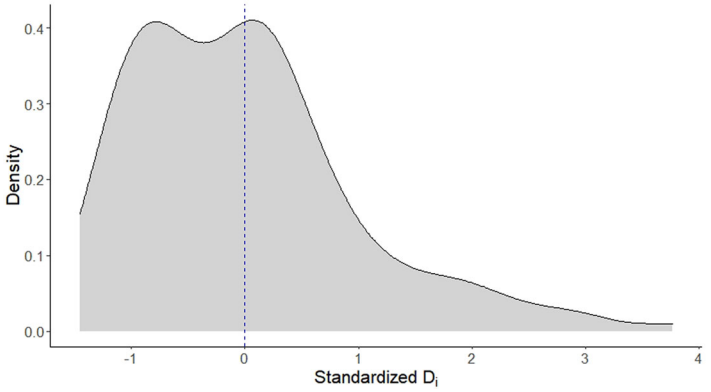

Figure 2. Estimated density of standardized D_i.

Table 2. The number of tasks identified as outliers per different combinations of kernel and significance level α.

Kernel / Significance Level	5%	10%
Gaussian	1	7
Epanechnikov	1	7
Rectangular	1	7
Triangular	1	7
Biweight	1	7
Cosine	1	7
Optcosine	1	7

The identified tasks are the same for all kernels at each significance level.

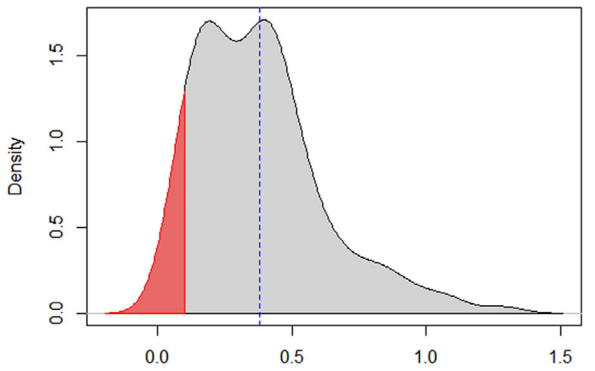

Figure 3. Kernel density estimation for $\hat{s}$ with Gaussian kernel. The blue line is mean, and the shaded red area is the rejection region $\Theta_1 = \{-\infty, \hat{s}_{0.10}\}$.

4.2.2. τ Chosen by kernel density estimation

The results of task identification are stated in Table 2. Increasing the significance level includes more tasks that are identified as outlier groups. Notably, the choice of the kernel doesn't change the tasks that are identified as outliers at every significance level.

Figure 3 displays estimated kernel density $\hat{s}$. Note the difference from Figure 2 where D_i were standardized, while in the current procedure they aren't. The analysis shows that in ILEA schools data, no task has $D_i < 0$, and the reason the estimated density in Figure 4 extends to negative values is the estimation algorithm. As no tasks have negative D_i, all tasks that are identified to be deleted in Table 2 have $D_i > 0$.

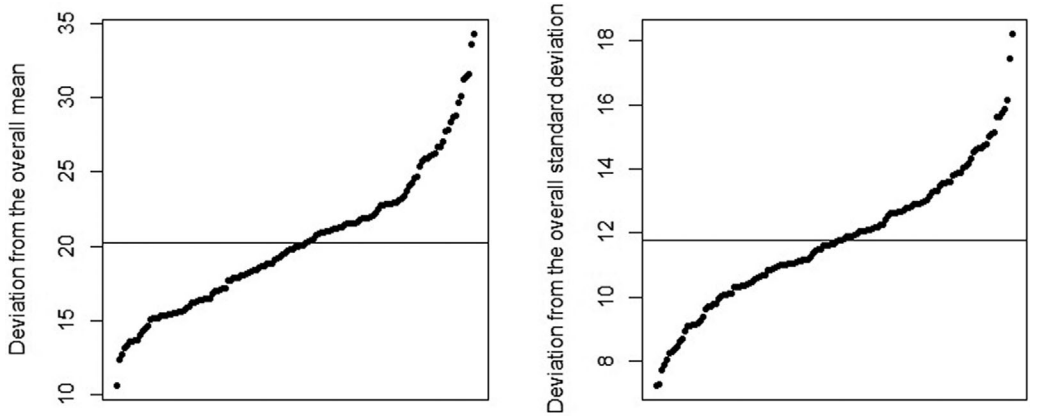

Figure 4. Tasks ranked separately according to their deviation from the overall mean score and standard deviation, which are marked by horizontal lines.

When all tasks are included in the model, $TEV(\mathbf{Y}, \hat{\mathbf{Y}}) = 52.605\%$. At $\alpha = 5\%$, task 99 is identified as outlier. Deleting this task from training data, training the model, and predicting on all tasks' training data is equivalent to its deleted explained variance, with $DEV_{99} = 52.024\%$. At $\alpha = 10\%$, tasks 14, 78, 83, 99, 109, 119 and 138 that are identified as outliers. Deleting these tasks from the training set ultimately leads to $TEV(\mathbf{Y}, \hat{\mathbf{Y}}^{(i)}) = 50.428\%$. We find that $TEV(\mathbf{Y}, \hat{\mathbf{Y}})$ is larger than performance both at $\alpha = 5\%$ and $\alpha = 10\%$. Therefore the procedure to use kernel density estimation for task outlier detection hasn't yielded improvement of performance on the ILEA schools data. Nonetheless, the usefulness of the procedure can potentially be high in other multi-task learning applications.

5. Conclusion

In this paper, we propose a new model, which we call the additive multi-task learning model, that allows the combining of parametric and nonparametric effects in a per-task fashion. The model is highly customizable in its structure and fitting procedure while providing high explanatory power, which is often required in many modern applications of machine learning. As the model is based on statistical theory, we also propose testing procedures that enable proper model diagnostics and can facilitate further increases in predictive power.

Further, we propose task diagnostics methods to identify task outliers by their influence on model output in a leave-task-out fashion. We consider cases of task influence on model performance and satisfaction of model assumptions. As this framework is inspired by statistical testing procedures, we also propose empirical methods for working with the proposed test statistics, enabling decisions on testing hypotheses and the creation of confidence intervals.

Real-world data experiments have shown that a relatively simple configuration of an additive multi-task learning model achieves a significant performance boost compared to the existing published results of other multi-task learning models. Further, we examine cases of task influence on performance metrics through task influence tests; however, they do not yield performance improvement on this particular dataset but have strong potential to be useful in other real-world data applications. Theoretically tracking the sampling distributions of the proposed task statistics is an open research question, which can potentially shed light on more optimal values for identifying and remedying task outliers.

ORCID

Nikolay Miller http://orcid.org/0000-0002-3643-3559

References

Agarwal, A., H. Daumé III, S. Gerber, et al. 2010. Learning multiple tasks using manifold regularization. In *Advances in Neural Information Processing Systems 23, NIPS*, ed. John D. Lafferty, 46–54. Curran Associates, Inc.

Álvarez, M. A., L. Rosasco, and N. D. Lawrence. 2012. Kernels for vector-valued functions: a review. *Foundations and Trends® in Machine Learning* 4 (3):195–266. doi:10.1561/2200000036.

Argyriou, A., T. Evgeniou, and M. Pontil. 2008. Convex multi-task feature learning. *Machine Learning* 73 (3):243–72. doi:10.1007/s10994-007-5040-8.

Argyriou, A., C. A. Micchelli, and M. Pontil. 2009. When is there a representer theorem? Vector versus matrix regularizers. *Journal of Machine Learning Research* 10:2507–29.

Bai, J., K. Zhou, G. Xue, H. Zha, G. Sun, B. Tseng, Z. Zheng and Y. Chang. 2009. Multi-task learning for learning to rank in web search. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management. CIKM '09*. Hong Kong, China: Association for Computing Machinery, 1549–52. doi:10.1145/1645953.1646169.

Bakker, B., and T. Heskes. 2003. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research* 4:83–99.

Bitvai, Z., and T. Cohn. 2015. Day trading profit maximization with multi-task learning and technical analysis. *Machine Learning* 101 (1–3):187–209. doi:10.1007/s10994-014-5480-x.

Chapelle, O., P. Shivaswamy, S. Vadrevu, K. Weinberger, Y. Zhang, and B. Tseng. 2011. Boosted multi-task learning. *Machine Learning* 85 (1–2):149–73. doi:10.1007/s10994-010-5231-6.

Chen, J., J. Liu, and J. Ye. 2012. Learning incoherent sparse and low-rank patterns from multiple tasks. *ACM Transactions on Knowledge Discovery from Data* 5 (4):22. doi:10.1145/2086737.2086742.

Chen, J., J. Zhou, and J. Ye. 2011. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 42–50. doi:10.1145/2020408.2020423.

Christensen, R. 2016. *Analysis of variance, design, and regression - Linear modeling for unbalanced data*. 2nd ed. Chapman & Hall/CRC Texts in Statistical Science.

Cook, R. D. 1977. Detection of influential observation in linear regression. *Technometrics* 19 (1):15–8. doi:10.2307/1268249.

Cook, R. D. 1979. Influential observations in linear regression". *Journal of the American Statistical Association* 74 (365):169–74.

Engle, R. F., C. W. J. Granger, J. Rice, and A. Weiss. 1986. Semiparametric estimates of the relation between weather and electricity sales. *Journal of the American Statistical Association* 81 (394):310–20. doi:10.1080/01621459.1986.10478274.

Eubank, R. L. 1999. Nonparametric regression and spline smoothing. In *Statistics: Textbooks and monographs*. 2nd ed. 157. New York, NY: Dekker.

Evgeniou, T., C. A. Micchelli, and M. Pontil. 2005. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research* 6.21:615–37.

Evgeniou, T., and M. Pontil. 2004. Regularized multi–task learning. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '04. Seattle, WA, USA: Association for Computing Machinery, 109–17.

Fang, M., and D. Tao. 2015. Active multi-task learning via bandits. *Proceedings of the 2015 SIAM International Conference on Data Mining*, 505–13. SIAM. doi:10.1137/1.9781611974010.57.

Fiot, J.-B., and F. Dinuzzo. 2015. Electricity demand forecasting by multi-task learning. CoRR abs/1512.08178

Ghosn, J., and Y. Bengio. 1996. Multi-task learning for stock selection. In *NIPS'96: Proceedings of the 9th International Conference on Neural Information Processing Systems*, ed. Michael Mozer, Michael I. Jordan, and Thomas Petsche, 946–52. MIT Press.

Gong, P., J. Ye, and C. Zhang. 2012. Robust multi-task feature learning. In: *KDD'12: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ed. Qiang Yang, Deepak Agarwal, and Jian Pei, 895–903. ACM. doi:10.1145/2339530.2339672.

Hastie, T., R. Tibshirani, and J. Friedman. 2001. *The elements of statistical learning. Springer series in statistics*. New York, NY, USA: Springer New York Inc.

Jeong, J.-Y., and C.-H. Jun. 2018. Variable selection and task grouping for multi-task learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ed. Yike Guo and Faisal Farooq, 1589–98. ACM. doi:10.1145/3219819.3219992.

Kim, S.-J., and R. Mowakeaa. 2019. Kernel-based efficient lifelong learning algorithm. *2019 IEEE Data Science Workshop (DSW)*, 175–9. IEEE.

Kimeldorf, G., and G. Wahba. 1971. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications* 33 (1):82–95. doi:10.1016/0022-247X(71)90184-3.

Kumar, A., and H. Daumé. III. 2012. Learning task grouping and overlap in multi-task learning. *ICML* icml.cc/Omnipress.

Kutner, M. H., C. J. Nachtsheim, J. Neter and W. Li. 2005. *Applied linear statistical models*. 5th ed. New York: McGraw-Hill, Irwin.

Liao, X., and L. Carin. 2005. Radial basis function network for multi-task learning. *Advances in Neural Information Processing Systems 18 (NIPS 2005)*, 792-802.

Micchelli, C. A., and M. Pontil. 2005. On learning vector-valued functions. *Neural Computation* 17 (1):177–204. doi:10.1162/0899766052530802.

Montgomery, D. C., E. A. Peck, and G. G. Vining. 2012. *Introduction to linear regression analysis*. 5th ed. Hoboken: John Wiley & Sons.

Pu, J, Y.-G. Jiang, J. Wang, and X. Xue. 2013. Multiple task learning using iteratively reweighted least square. In *JCAI'13: Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, ed. Francesca Rossi, 1607–13. IJCAI/AAAI.

R Core Team 2022. *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing.

Romera-Paredes, B., and M. Pontil 2013. A new convex relaxation for tensor completion. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, ed. Christopher J. C. Burges, 2967–75.

Scott, D. W. 1992. Multivariate density estimation: Theory, practice, and visualization. In *Wiley series in probability and statistics*, 1–317. Hoboken: Wiley.

Silverman, B. W. 1986. *Density estimation for statistics and data analysis*. London: Chapman & Hall.

Wegkamp, M. 2003. Model selection in nonparametric regression. *The Annals of Statistics* 31 (1):252–73. doi:10.1214/aos/1046294464.

Yang, Y. 1999. Model selection for nonparametric regression. *Statistica Sinica* 9:475–99.

Zacharia, G. 2009. *Regularized algorithms for ranking, and manifold learning for related tasks*. PhD thesis. Massachusetts Institute of Technology.

Zhong, S., J. Pu, Y.-G. Jiang, R. Feng, and X. Xue. 2016. Flexible multi-task learning with latent task grouping. *Neurocomputing* 189:179–88. doi:10.1016/j.neucom.2015.12.092.

Zhou, Y., H. Chen, Y. Li, Q. Liu, X. Xu, S. Wang, P.-T. Yap, and D. Shen. 2021. Multi-task learning for segmentation and classification of tumors in 3D automated breast ultrasound images. *Medical Image Analysis* 70:101918. doi:10.1016/j.media.2020.101918.