J Comput Aided Mol Des (2012) 26:883–895
DOI 10.1007/s10822-012-9582-x

Multi-task learning for pK_a prediction

Grigorios Skolidis · Katja Hansen · Guido Sanguinetti · Matthias Rupp

Received: 15 November 2011 / Accepted: 11 May 2012 / Published online: 20 June 2012

Abstract Many compound properties depend directly on the dissociation constants of its acidic and basic groups. Significant effort has been invested in computational models to predict these constants. For linear regression models, compounds are often divided into chemically motivated classes, with a separate model for each class. However, sometimes too few measurements are available for a class to build a reasonable model, e.g., when investigating a new compound series. If data for related classes are available, we show that multi-task learning can be used to improve predictions by utilizing data from these other classes. We investigate performance of linear Gaussian process regression models (single task, pooling, and multi-task models) in the low sample size regime, using a published data set ($n = 698$, mostly monoprotic, in aqueous solution) divided beforehand into 15 classes. A multi-task regression model using the intrinsic model of co-regionalization and incomplete Cholesky decomposition performed best in 85 % of all experiments. The presented approach can be applied to estimate other molecular properties where few measurements are available.

Keywords pK_a prediction · Multi-task learning · Quantitative structure–property relationships · Gaussian processes

Electronic supplementary material The online version of this article (doi:10.1007/s10822-012-9582-x) contains supplementary material, which is available to authorized users.

G. Skolidis
Department of Statistical Science, University College London, Gower Street, London WC1E 6BT, UK
e-mail: g.skolidis@ucl.ac.uk

K. Hansen
Theory Department, Fritz Haber Institute of the Max Planck Society, Faradayweg 4-6, 14195 Berlin, Germany
e-mail: hansen@fhi-berlin.mpg.de

K. Hansen · M. Rupp
Machine Learning Group, TU Berlin, Franklinstr. 28/29, 10587 Berlin, Germany

G. Sanguinetti
School of Informatics, University of Edinburgh, 10 Crichton Street, EH8 9AB Edinburgh, Scotland
e-mail: G.Sanguinetti@ed.ac.uk

M. Rupp (✉)
Institute of Pharmaceutical Sciences, ETH Zurich, Wolfgang-Pauli-Str. 10, 8093 Zürich, Switzerland
e-mail: matthias.rupp@pharma.ethz.ch

Introduction

A compound's pharmacokinetic and biochemical properties depend directly on dissociation constants of its acidic and basic groups, commonly expressed as the negative decadic logarithm pK_a of the acid dissociation constant K_a. Its accurate estimation is thus of great interest, and much effort has gone into computational models for pK_a values [1–5]

Empirical (as opposed to ab initio) models assume that a compound's physico-chemical properties are a (mathematical) function of its structure, usually described by computable features. Often, changes in property are assumed to be additive for different substitutions within a class of compounds, e.g., ortho-substituted benzoic acids. Separate models are then built for each class.

This approach does not make use of all information contained in the reference data. Consider, e.g., ortho-substituted phenols. Division into two classes, those that can form internal hydrogen bonds, and those that can not,

improved certain linear pK_a models [6]. However, each class contains information about the other one that separate models don't use.

The ability to use information from related classes might also be beneficial in other settings. An example is the investigation of a new compound series, for which few measurements are likely available, but more measurements might exist for compounds from structurally related series. This is exacerbated for (computationally) designed compounds that have not been synthesized yet.

Multi-task learning [7] is a machine learning approach where multiple related tasks sharing a common representation are learned simultaneously. It has rarely been used in chem- and bioinformatics [8–11].

In this work, we model relationships between pK_a values of related classes using multi-task Gaussian process regression to improve accuracy in situations where only few samples are available.

Acid dissociation constants

In Brønsted–Lowry theory, an acid HA is a proton (hydrogen cation) donor, $HA \rightleftharpoons H^+ + A^-$, and base B is a proton acceptor, $B + H^+ \rightleftharpoons BH^+$. For weak acids in aqueous solution, the dissociation $HA + H_2O \rightleftharpoons A^- + H_3O^+$ is reversible. In the backward reaction, oxonium acts as acid and A^- as base. The equilibrium constant [12], known as the *acid dissociation constant* K_a, is the ratio of activities of products and reagents,

$$K_a = \frac{a(A^-)\, a(H_3O^+)}{a(HA)\, a(H_2O)}, \tag{1}$$

where $a(\cdot)$ is activity, a unit-less measure of "effective concentration". It can be defined in terms of chemical potential, and expressed relative to a standard concentration as $a(x) = \gamma(x)c(x)/c^\ominus$ [1], where $\gamma(\cdot)$ is a dimensionless activity coefficient, $c(\cdot)$ is the molar (or molal) concentration of a species, and, $c^\ominus = 1\,\text{mol/L}$ (or 1 mol/kg) is a standard concentration.

In an ideal solution $\gamma(\cdot) = 1$, and effective concentrations equal analytical ones. Assuming this, $c(H_2O) = c^\ominus = 1$ mol/L, and taking negative decadic logarithm yields the Henderson–Hasselbalch [13] equation

$$pK_a \approx pH + \log_{10}\frac{c(HA)}{c(A^-)}, \tag{2}$$

where $pH = -\log_{10} a(H_3O^+) \approx -\log_{10}(c(H_3O^+)/c^\ominus)$. In an ideal solution, the pK_a of a (monoprotic) weak acid is thus the pH at which 50 % of it is in deprotonated form.

Analogously, protonation of bases leads to pK_b values. Since pK_a and pK_b values use the same scale, pK_a values are used for both acids and bases. Note that these considerations are for *monoprotic* compounds with a single ionizable center (proton to accept/donate).

Linear free energy relationships

Many descriptors and prediction methods have been used to establish quantitative structure–property relationships for pK_a [1–5]. Linear free energy relationships (LFER) [14–17] use the Hammett equation [18]

$$\log_{10}\frac{K_a}{K'_a} = \rho\sum_{i=1}^{m}\sigma_i \iff pK_a = pK'_a - \rho\sum_{i=1}^{m}\sigma_i, \tag{3}$$

where K_a and K'_a are the acid dissociation constants for the substituted and the parent molecule, ρ is a constant specific for the class of the two molecules, m is the number of substituents, and the σ_i are constants expressing the substituent effect on the dissociation constant [15]. The underlying assumptions are that changes in pK_a correspond to changes in free Gibbs energy, and that these changes are additive within a compound class.

This approach has several disadvantages: (1) the σ constants have to be known (experimentally determined) for all involved substituents [19]. (2) non-linear effects within a class are not captured. (3) information from related classes is not used. In previous work by us [20] and others [1, 2], it was shown that (1) influence of substituents on pK_a can be learned from data, i.e., from collections of experimental pK_a values, and that (2) non-linear models improve prediction accuracy. In this work, we demonstrate that (3) using data from other classes can improve prediction accuracy, in particular if few experimental values are known for a class.

Material and methods

Data and descriptors

We use a published data set of structures and pK_a measurements compiled from the literature by Tehan et al. [6, 21] (a curated subset of the PhysProp database [22]). The data set (Table 1) contains 698 compounds (416 acids, 282 bases), partitioned into $6 + 9 = 15$ classes. Figure 1 presents numbering schemes for functional group atoms. Experimental pK_a values were obtained between 15 and 30 °C (mean 23.8 °C, standard deviation 2.5 °C) in aqueous solution.

We focus on investigation of multi-task learning for pK_a prediction. With regard to molecular representation, we therefore limit ourselves to an established pK_a descriptor, electrophilic superdelocalizability (SE). This quantum-mechanical descriptor is based on frontier electron theory [23], and has been shown to be well-suited for pK_a prediction [6, 20]. It is defined as

Table 1 Data set and division into classes (tasks)

T	*n*	S	∅	pK_a range	Description
Aa	57	1	9.17	5.42–10.45 (5)	Phenols, meta/para-substituted
Ab	26	2	6.32	3.03–9.87 (7)	Phenols, ortho-substituted, IHB
Ac	91	3	7.71	0.38–12.23 (12)	Phenols, ortho-substituted, NIHB
Ad	46	4	4.01	2.82–4.85 (2)	Benzoic acids, meta/para-substituted
Ae	53	5	2.90	0.65–5.09 (4)	Benzoic acids, ortho-substituted
Af	143	6	3.70	0.51–6.20 (6)	Aliphatic carboxylic acids
Ba	55	1	1.97	−5.00–5.48 (10)	Anilines
Bb	23	2	9.92	5.70–10.87 (5)	Amines, primary
Bc	23	3	10.42	8.50–11.39 (3)	Amines, secondary
Bd	31	4	9.09	6.57–11.25 (5)	Amines, tertiary
Be	48	5	4.23	0.67–6.47 (6)	Pyridines, meta/para-substituted
Bf	34	6	3.76	−2.86–7.90 (11)	Pyridines, ortho-substituted
Bg	14	7	2.22	−1.63–6.81 (8)	Pyrimidines
Bh	26	8	5.34	−0.53–7.85 (8)	Imidazoles, benzimidazoles
Bi	28	9	4.66	2.69–6.10 (3)	Quinolines

Tasks Aa,…,Af published in Tehan et al. [21], tasks Ba,…,Bi in Tehan et al. [6]. pK_a ranges are given as min–max ranges; the number in brackets indicates the number of spanned orders of magnitude rounded to one digit. T = task; abbreviations were chosen to indicate source (capital letter) and are otherwise consecutively labeled (a,b,c,…). *S* source; number indicates table in original publication. ∅ = average pK_a value. (N)IHB = (not) capable of forming internal hydrogen bonds

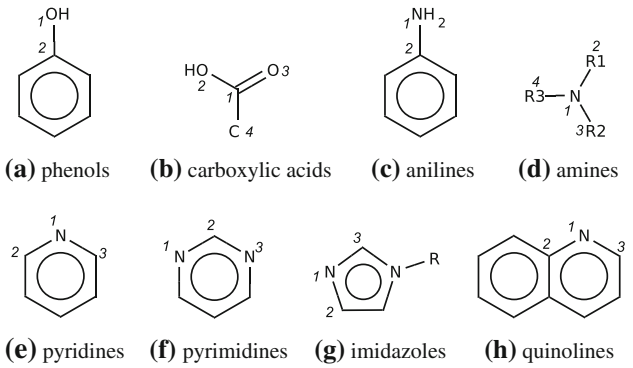


Fig. 1 Chemical classes with atom numbering used for each functional group [6, 21]

$$\mathrm{SE}(p) = 2\sum_{j=1}^{m}\sum_{\alpha=1}^{q}\frac{\mathbf{c}_{\alpha,j}^2}{\lambda_j} \tag{4}$$

where $\mathbf{c}$, λ are eigenvectors and -values of the Hesse energy matrix, p is the atom index, α is the atomic orbital (s, p_x, p_y, p_z, …) index, q is the number of atomic orbitals, j is the molecular orbital index, and m is the number of occupied molecular orbitals.

We represent a compound by three values: The SE of it's ionizable center, and, the binned SE values of all atoms with topological distance to the ionizable center of one and two, respectively [20]. Single 3D conformations were calculated for all molecules using CORINA [24] (version 1.82; Molecular Networks GmbH, http://www.molecular-networks.com). Quantum mechanical calculations were done using MOPAC [25] (version 7.1; Stewart computational chemistry, http://www.openmopac.net) with keywords XYZ, AM1, EF, PRECISE, VECTORS, ALLVEC. Some compounds in the data set may exist, at least partially, in zwitterionic form. Calculations were performed on the neutral species in these cases.

Computed descriptor values are provided via the supplementary material. Structures and pK_a values of the articles by Tehan et al. [6, 12] are available at the "Online Chemical Modeling Environment" [26] (http://www.ochem.eu, accessed 2012-05-08). Matlab (version 7.6.0, The MathWorks, http://www.mathworks.com) source code for STL and MTL Gaussian process regression can be downloaded from the authors web pages at http://homepages.inf.ed.ac.uk/gsanguin/software.html, and, http://www.mrupp.info.

Gaussian process regression

We use Gaussian process (GP) [27] regression, a Bayesian non-parametric[1] technique. In brief, a Gaussian process is a generalization of the (multi-dimensional) normal

[1] As opposed to parametric approaches, where the information from the training data are summarized in the parameters of a distribution, non-parametric approaches require the training data for later predictions. This distinction does not prevent non-parametric approaches from having parameters, here the regression weights $\boldsymbol{\alpha}$ and hyper-parameters $\boldsymbol{\theta}$. Parameters $\boldsymbol{\alpha}$, which directly belong to the model itself, are computed from the data by solving an optimization problem. Hyper-parameters $\boldsymbol{\theta}$ parameterize the kernel, and can be estimated via gradient-based optimization by maximizing the marginal likelihood.

distribution to functions, i.e., a function-valued random variable. For regression, one considers all functions generated by a GP that agree with the training data; the mean of these functions is the predictor.[2] A GP is specified by a covariance function, or *kernel*, that quantifies similarity between two inputs. We use the linear kernel $k(\mathbf{x}_i, \mathbf{x}_j) = e^{\theta}(1 + <\mathbf{x_i}, \mathbf{x_j}>)$, where $\mathbf{x}_i$, $\mathbf{x}_j$ are descriptor vectors, $<\cdot,\cdot>$ is the standard inner product, and θ is a hyper-parameter. Computation of GP regression models essentially amounts to inverting a symmetric positive definite $n \times n$ matrix, where n is the number of training samples, resulting in cubic $O(n^3)$ runtime. For details, see the book by Rasmussen and Williams [27].

Multi-task learning

In supervised learning, one is given a single data set D of n pairs of input and output, $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Here, inputs are SE values, $d = 3$, and outputs are pK_a values. The goal is to learn from the data D a function $\hat{f}$ that maps new inputs $\mathbf{x}$ to their (unknown, i.e., not yet experimentally measured) output y. This is called *single-task learning* (STL).

By contrast, in *multi-task learning* (MTL), one is given M different but *related* data sets $D_1, \ldots, D_M$ (the *tasks*), and the goal is to learn M different functions $\hat{f}_j$. Here, each of the 15 compound classes in Table 1 is a task. We denote the different inputs with $\mathbf{X}_j = [\mathbf{x}_{1j}, \ldots, \mathbf{x}_{n_j j}]$, where the first and second index indicate sample number and task number, respectively. Corresponding outputs are denoted as $\mathbf{y}_j = [y_{1j}, \ldots, y_{n_j j}] \in \mathbb{R}^{n_j}$. For inputs and outputs, $j = 1, \ldots, M$, and $n_1 + \cdots + n_M = N$. Here, the number of samples n_j are given by the second column in Table 1, $M = 15$, and $N = 698$. We define the complete sets of inputs and outputs as $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_M]$, and $\mathbf{y} = [\mathbf{y}_1, \ldots, \mathbf{y}_M]$.

We model measurement errors in experimentally determined pK_a values as noise, i.e., $y_{ij} = f_j(x_{ij}) + \varepsilon$, where x_{ij} is the ith compound in task j, y_{ij} is it's observed pK_a value, f_j is the "true" relationship between inputs and outputs in task j, and ε is the error introduced by measurement. We make the usual assumption of independent, identically distributed Gaussian noise, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, where σ denotes standard deviation of measurement error. Technically, we treat σ as a hyper-parameter.

Multi-task learning can be achieved either by sharing a common set of parameters (*parameter transfer*), or by directly inducing correlations between the task dependent functions (*collocated transfer*). The simplest form of parameter transfer couples the individual functions of the tasks by sharing the same hyper-parameters θ of the covariance function, $\theta_j = \theta$ for all j. Let $\mathbf{f} = (\mathbf{f}_1, \ldots, \mathbf{f}_M)$; the prior distribution of the latent function factorizes as $p(\mathbf{f}|\mathbf{X}, \theta) = \prod_{j=1}^{M} p(\mathbf{f}_j|X_j, \theta)$. The individual $\mathbf{f}_j$'s are independent of the others, and transfer of information occurs only by sharing hyper-parameters θ during the training phase. The prediction stage is the same as for STL GPs. We refer to this method as MTL-SHP (shared hyper-parameters).

The main characteristic of collocated transfer methods is that they require some form of correlation between the functions of the different tasks. A popular way to achieve this is to employ the "*intrinsic model of coregionalization*" [29, 30]. This approach allows the joint prior probability distribution of $\mathbf{f}$ to factorize as the Kronecker product $\otimes$ of two separate matrices, $\mathbf{f}|\mathbf{X} \sim \mathrm{GP}(\mathbf{0}, \mathbf{K^t} \otimes \mathbf{K^x})$, where task matrix $\mathbf{K^t} \in \mathbb{R}^{M \times M}$ captures correlations between tasks, and $\mathbf{K^x} \in \mathbb{R}^{N \times N}$ models correlations between each element of each vector $\mathbf{f}_j$.[3]

Bonilla et al. [30] proposed to use a *free form* task covariance matrix $\mathbf{K^t}$ with hyper-parameters θ^t, both estimated from the data. The entries of $\mathbf{K^t}$ reflect correlations between the tasks. This allows the latent functions of the different tasks to interact during training and during prediction, and is considered a stronger form of transfer learning. We refer to this method as MTL-ICD (incomplete Cholesky decomposition), from the used Cholesky decomposition $\mathbf{K^t} = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L}$ is a lower triangular matrix.

The task covariance matrix $\mathbf{K^t}$ can be restricted to a *correlation* matrix by enforcing a unit diagonal (proper range of off-diagonal elements is ensured by positive definiteness of $\mathbf{K^t}$) [31, 32]. We refer to this method as MTL-COR (correlation matrix). Off-diagonal entries were restricted to positive values for MTL-ICD and MTL-COR. Both models employ the intrinsic model of co-regionalization, and differ only in the way they parameterize the task covariance matrix.

In summary, the three investigated Gaussian process MTL methods can be characterized as follows: (1) MTL-SHP learns shared kernel hyper-parameters, (2) MTL-ICD learns task correlations in the form of a positive definite task matrix, (3) MTL-COR learns task correlations in the form of a correlation matrix (additional restriction of unit diagonal).

[2] Predictions are technically equivalent to those of kernel ridge regression [28], a regularized form of ordinary regression. Here, we do not use additional features of GPs like predictive variance. However, the used GP MTL methods do make use of Bayesian aspects of GPs.

[3] Technically, $\mathbf{K^t} \otimes \mathbf{K^x} \in \mathbb{R}^{MN \times MN}$. In our setting, each sample (compound) occurs in one task only. After removing (marginalizing out) rows and columns corresponding to combinations of compounds and tasks that don't occur, the resulting matrix is $N \times N$. In practice, it is not necessary to construct the $MN \times MN$ matrix explicitly.

Evaluation

We consider three baseline methods: STL, pooling, and pooling with class information. The STL method is simply linear ridge regression on a single task (STL). Pooling is linear ridge regression on the data of all tasks pooled together (Pooling). This is arguably the simplest form of multi-task learning, as it uses information from different tasks, but ignores all task structure. For pooling with class information (PoolingCI), we encode task membership in the descriptor vectors by augmenting them with M components that are either 0 (sample does not belong to a task) or 1 (sample belongs to a task). Since each molecule belongs to exactly one task, this is equivalent to changing the linear kernel to $k(\mathbf{x}_i, \mathbf{x}_j) = e^{\theta}(1 + <\mathbf{x_i}, \mathbf{x_j}> + \delta_{t_i=t_j})$, where $\mathbf{x}_i$, $\mathbf{x}_j$ are the original (not augmented) descriptor vectors, and $\delta_{t_i=t_j} = 1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same task, and 0 otherwise.

For purposes of statistical evaluation, we consider a null model (Null) that uses the average pK_a value of a training set as (constant) predictor for new samples. Any useful model should improve over the null model.

We retrospectively evaluated models as follows: given $M \geq 2$ of the tasks from Table 1, we randomly drew with repetition n samples (compounds) and corresponding labels (pK_a values) from each task. These formed the training set, where Null and STL were trained on each task separately, and Pooling, PoolingCI, MTL-SHP, MTL-COR, MTL-ICD were trained on all M tasks together. Each method was then used to predict the remaining compounds of each task. Based on the predicted values, mean absolute error (MAE) and root mean squared error (RMSE) were computed to quantify predictive performance. This procedure was repeated 100 times for training set sizes n = 5, 10, 15, 20. Shown averages, standard deviations, etc. are over these 100 repetitions. For the sake of simplicity, we consider only the homogeneous case where all tasks contain the same number of data points.

For each experiment, task, and training set size, two tests were done: (1) To test which methods resulted in meaningful models, the 100 repetitions of each method were compared to the 100 repetitions of the null model using a one-sided Wilcoxon signed rank test [33], with null hypothesis that the median performance difference is below 0.1. If the null hypothesis was rejected on a significance level of 0.05, the method was said to perform better than the null model. (2) To test which methods performed best, the best-performing method was compared to the other methods, using the same test, with null hypothesis that the median performance difference is greater than 0.1. If the null hypothesis was rejected, both methods were considered to have performed best.

To analyze task similarity matrices, we converted them to Euclidean distance matrices,[4] took the median of each component over the 100 repetitions, and applied standard hierarchical clustering (cluster agglomeration, single linkage). The results were visualized as a dendrogram.

Results and discussion

We investigated three scenarios: in scenario I, limited data from one or more highly similar tasks is available (e.g., ortho-substituted benzoic acids and meta/para-substituted benzoic acids). There are four experiments in this scenario: (1) phenols (tasks Aa, Ab, Ac), (2) carboxylic acids (tasks Ad, Ae, Af), (3) amines (tasks Bb, Bc, Bd), and (4) pyridines (tasks Be, Bf). In scenario II, limited data from more, but less similar tasks is available (e.g., ortho-substituted benzoic acids and other acids). There are two experiments in this scenario: (1) acids (tasks Aa, …, Af), and (2) bases (tasks Ba, …, Bi). In scenario III, all tasks (Aa, …, Af, Ba, …, Bi) are used. This scenario includes tasks that may be unrelated.

Table 2 presents method performance in terms of MAE. Figures 2 and 3 give a visual overview of performance in scenarios I and II. We limit shown results and discussion to mean and standard deviation of MAE. The supplement contains additional information (mean, standard deviation, median, median absolute deviation of MAE and RMSE for all scenarios, experiments and methods; dendrograms for MTL-COR and MTL-ICD) in tabular and graphical form. Method MTL-SHP consistently did not improve on STL, and Pooling performed consistently worse than PoolingCI. Both methods are therefore not shown or discussed further.

Performance in absolute terms

Prediction errors of one log-unit or less have been deemed acceptable for pK_a values in the literature [34]. Based on Liao and Nicklaus [35], we classify predictions based on MAE as excellent (MAE $\leq$ 0.1), well (0.1 $<$ MAE $\leq$ 0.5), fair (0.5 $<$ MAE $\leq$ 1), poor (1.0 $<$ MAE $\leq$ 2), or awful (2 $<$ MAE). Figure 4 presents MAEs of models in scenario I according to this classification scheme. Figures for scenarios II and III are qualitatively similar (see supplement). We observe that (1) quality of predictions increases with number of training samples; (2) method MTL-ICD delivers the best predictions; (3) for 5 training samples, MTL-COR and MTL-ICD have roughly half as many "awful" predictions as STL

[4] 2Task similarity matrices are positive definite. Their entries thus correspond to evaluations of an inner product in some Hilbert space, which can be converted to Euclidean distance by using $||\mathbf{x}-\mathbf{z}||_2^2 = \sum_{i=1}^{d} |x_i - z_i|^2 = <\mathbf{x}-\mathbf{z}, \mathbf{x}-\mathbf{z}> = <\mathbf{x},\mathbf{x}> - 2<\mathbf{x},\mathbf{z}> + <\mathbf{z},\mathbf{z}>$.

Table 2 Performance in terms of mean absolute error ± standard deviation for 5, 10, 15, and 20 training samples

MAE		Number of training samples			
Task/Method		5	10	15	20
Phenols					
Aa	STL	0.77 ± 0.13	0.59 ± 0.10	0.48 ± 0.06	0.42 ± 0.05
	PoolingCI	*0.92 ± 0.14*	**0.43 ± 0.07**	**0.37 ± 0.05**	**0.37 ± 0.07**
	MTL-COR	0.68 ± 0.14	**0.44 ± 0.06**	**0.37 ± 0.06**	**0.35 ± 0.07**
	MTL-ICD	**0.55 ± 0.27**	**0.35 ± 0.06**	**0.33 ± 0.07**	**0.33 ± 0.08**
Ab	STL	*1.47 ± 0.17*	*1.36 ± 0.18*	1.23 ± 0.19	1.13 ± 0.27
	PoolingCI	*1.46 ± 0.16*	0.95 ± 0.15	0.78 ± 0.18	0.69 ± 0.24
	MTL-COR	1.26 ± 0.16	0.96 ± 0.15	0.80 ± 0.20	0.71 ± 0.25
	MTL-ICD	**0.87 ± 0.38**	**0.59 ± 0.15**	**0.56 ± 0.16**	**0.52 ± 0.19**
Ac	STL	2.29 ± 0.25	1.95 ± 0.13	1.69 ± 0.11	1.44 ± 0.21
	PoolingCI	*2.32 ± 0.23*	1.33 ± 0.21	0.96 ± 0.15	0.84 ± 0.11
	MTL-COR	1.86 ± 0.22	1.30 ± 0.19	0.97 ± 0.16	0.84 ± 0.14
	MTL-ICD	**1.18 ± 0.63**	**0.72 ± 0.14**	**0.70 ± 0.16**	**0.71 ± 0.24**
Carboxylic acids					
Ad	STL	**0.28 ± 0.10**	**0.22 ± 0.08**	**0.18 ± 0.03**	**0.18 ± 0.02**
	PoolingCI	*0.39 ± 0.05*	**0.26 ± 0.08**	**0.20 ± 0.04**	**0.19 ± 0.04**
	MTL-COR	**0.21 ± 0.07**	**0.19 ± 0.03**	**0.18 ± 0.04**	**0.18 ± 0.04**
	MTL-ICD	**0.27 ± 0.11**	**0.18 ± 0.04**	**0.17 ± 0.02**	**0.17 ± 0.02**
Ae	STL	*0.87 ± 0.11*	0.68 ± 0.18	**0.50 ± 0.15**	**0.43 ± 0.08**
	PoolingCI	*0.85 ± 0.11*	0.56 ± 0.18	**0.41 ± 0.05**	**0.40 ± 0.05**
	MTL-COR	**0.55 ± 0.17**	**0.44 ± 0.06**	**0.41 ± 0.05**	**0.40 ± 0.05**
	MTL-ICD	0.62 ± 0.22	**0.43 ± 0.06**	**0.40 ± 0.04**	**0.39 ± 0.04**
Af	STL	*0.79 ± 0.09*	0.57 ± 0.19	**0.39 ± 0.07**	**0.36 ± 0.04**
	PoolingCI	*0.79 ± 0.06*	0.52 ± 0.19	**0.38 ± 0.04**	**0.36 ± 0.03**
	MTL-COR	**0.50 ± 0.11**	**0.39 ± 0.05**	**0.38 ± 0.04**	**0.36 ± 0.03**
	MTL-ICD	**0.54 ± 0.19**	**0.36 ± 0.05**	**0.34 ± 0.03**	**0.34 ± 0.03**
Amines					
Bb	STL	*0.98 ± 0.45*	*0.81 ± 0.18*	***0.76 ± 0.23***	*0.73 ± 0.47*
	PoolingCI	***0.83 ± 0.18***	***0.73 ± 0.14***	***0.69 ± 0.22***	**0.61 ± 0.40**
	MTL-COR	***0.83 ± 0.27***	***0.74 ± 0.17***	***0.69 ± 0.22***	***0.68 ± 0.43***
	MTL-ICD	***0.84 ± 0.25***	***0.70 ± 0.20***	***0.67 ± 0.25***	**0.58 ± 0.37**
Bc	STL	*0.93 ± 0.20*	***0.86 ± 0.13***	***0.86 ± 0.12***	*0.84 ± 0.24*
	PoolingCI	***0.88 ± 0.12***	***0.83 ± 0.12***	***0.82 ± 0.14***	**0.69 ± 0.21**
	MTL-COR	*0.92 ± 0.13*	***0.86 ± 0.13***	***0.84 ± 0.13***	0.75 ± 0.23
	MTL-ICD	***0.89 ± 0.13***	***0.80 ± 0.16***	***0.78 ± 0.16***	**0.68 ± 0.26**
Bd	STL	*1.25 ± 0.39*	***1.10 ± 0.14***	***1.04 ± 0.14***	*0.99 ± 0.19*
	PoolingCI	***1.12 ± 0.16***	***1.03 ± 0.11***	***0.99 ± 0.14***	**0.78 ± 0.16**
	MTL-COR	***1.15 ± 0.17***	***1.08 ± 0.14***	***1.02 ± 0.14***	0.90 ± 0.18
	MTL-ICD	***1.15 ± 0.17***	***0.99 ± 0.24***	***0.94 ± 0.23***	**0.80 ± 0.32**
Pyridines					
Be	STL	***1.45 ± 0.19***	***1.35 ± 0.10***	**1.19 ± 0.17**	**1.01 ± 0.13**
	PoolingCI	***1.47 ± 0.20***	***1.39 ± 0.09***	*1.36 ± 0.09*	1.29 ± 0.16
	MTL-COR	***1.44 ± 0.15***	***1.36 ± 0.08***	*1.29 ± 0.12*	1.16 ± 0.14
	MTL-ICD	***1.42 ± 0.26***	***1.30 ± 0.24***	*1.33 ± 0.17*	**0.98 ± 0.37**
Bf	STL	*2.83 ± 0.61*	2.47 ± 0.29	2.38 ± 0.34	2.26 ± 0.30
	PoolingCI	***2.66 ± 0.41***	2.49 ± 0.27	2.40 ± 0.35	2.19 ± 0.34
	MTL-COR	**2.63 ± 0.42**	2.47 ± 0.28	2.36 ± 0.37	2.06 ± 0.35
	MTL-ICD	**2.64 ± 0.69**	**2.27 ± 0.46**	**2.25 ± 0.37**	**1.61 ± 0.64**
Acids					
Aa	STL	0.82 ± 0.17	0.58 ± 0.10	0.48 ± 0.06	0.42 ± 0.06
	PoolingCI	*0.93 ± 0.17*	**0.36 ± 0.04**	**0.34 ± 0.04**	**0.33 ± 0.04**
	MTL-COR	**0.50 ± 0.16**	**0.38 ± 0.05**	**0.35 ± 0.06**	**0.34 ± 0.06**
	MTL-ICD	**0.46 ± 0.15**	**0.35 ± 0.06**	**0.32 ± 0.04**	**0.31 ± 0.04**
Acids (continued)					
Ab	STL	*1.47 ± 0.18*	*1.39 ± 0.22*	1.25 ± 0.17	1.12 ± 0.23
	PoolingCI	*1.43 ± 0.18*	0.85 ± 0.13	0.77 ± 0.17	0.69 ± 0.20
	MTL-COR	1.03 ± 0.17	0.82 ± 0.16	0.72 ± 0.17	0.64 ± 0.20
	MTL-ICD	**0.76 ± 0.27**	**0.59 ± 0.14**	**0.54 ± 0.12**	**0.51 ± 0.16**
Ac	STL	*2.33 ± 0.27*	1.93 ± 0.11	1.72 ± 0.14	1.42 ± 0.22
	PoolingCI	*2.29 ± 0.24*	1.01 ± 0.15	0.86 ± 0.11	0.78 ± 0.09
	MTL-COR	1.44 ± 0.26	0.91 ± 0.15	0.78 ± 0.12	**0.71 ± 0.08**
	MTL-ICD	**0.97 ± 0.46**	**0.70 ± 0.12**	**0.69 ± 0.06**	**0.66 ± 0.06**
Ad	STL	0.31 ± 0.10	**0.20 ± 0.06**	**0.19 ± 0.05**	**0.17 ± 0.02**
	PoolingCI	*0.36 ± 0.04*	*0.32 ± 0.08*	*0.34 ± 0.06*	*0.33 ± 0.06*
	MTL-COR	**0.21 ± 0.07**	**0.24 ± 0.06**	**0.25 ± 0.08**	**0.26 ± 0.09**
	MTL-ICD	**0.19 ± 0.05**	**0.17 ± 0.02**	**0.17 ± 0.02**	**0.16 ± 0.02**
Ae	STL	*0.85 ± 0.11*	0.56 ± 0.16	**0.43 ± 0.05**	**0.42 ± 0.05**
	PoolingCI	*0.80 ± 0.10*	**0.40 ± 0.04**	**0.41 ± 0.05**	**0.39 ± 0.05**
	MTL-COR	**0.47 ± 0.10**	**0.40 ± 0.06**	**0.39 ± 0.07**	**0.38 ± 0.05**
	MTL-ICD	**0.48 ± 0.13**	**0.41 ± 0.06**	**0.39 ± 0.05**	**0.39 ± 0.05**
Af	STL	*0.80 ± 0.09*	0.49 ± 0.15	**0.39 ± 0.07**	**0.37 ± 0.04**
	PoolingCI	*0.77 ± 0.09*	**0.33 ± 0.03**	**0.33 ± 0.03**	**0.32 ± 0.02**
	MTL-COR	**0.45 ± 0.11**	**0.36 ± 0.05**	**0.34 ± 0.03**	**0.34 ± 0.04**
	MTL-ICD	**0.43 ± 0.14**	**0.36 ± 0.05**	**0.34 ± 0.04**	**0.33 ± 0.03**
Bases					
Ba	STL	*1.74 ± 0.40*	*1.57 ± 0.28*	*1.53 ± 0.18*	*1.48 ± 0.15*
	PoolingCI	***1.52 ± 0.25***	*1.40 ± 0.15*	0.92 ± 0.10	0.85 ± 0.10
	MTL-COR	**1.43 ± 0.28**	1.18 ± 0.14	1.09 ± 0.12	1.04 ± 0.15
	MTL-ICD	**1.36 ± 0.31**	**0.80 ± 0.17**	**0.68 ± 0.12**	**0.67 ± 0.10**
Bb	STL	***0.91 ± 0.32***	*0.81 ± 0.19*	*0.73 ± 0.25*	*0.69 ± 0.45*
	PoolingCI	***0.86 ± 0.19***	*0.73 ± 0.12*	**0.47 ± 0.16**	**0.46 ± 0.26**
	MTL-COR	***0.84 ± 0.24***	*0.70 ± 0.15*	0.62 ± 0.22	*0.65 ± 0.37*
	MTL-ICD	***0.81 ± 0.26***	**0.51 ± 0.16**	**0.42 ± 0.16**	**0.43 ± 0.26**
Bc	STL	*0.99 ± 0.26*	*0.88 ± 0.13*	*0.84 ± 0.13*	*0.81 ± 0.21*
	PoolingCI	*0.96 ± 0.17*	*0.86 ± 0.11*	0.62 ± 0.11	**0.58 ± 0.19**
	MTL-COR	***0.94 ± 0.25***	*0.81 ± 0.16*	0.74 ± 0.13	0.71 ± 0.22
	MTL-ICD	***0.95 ± 0.38***	**0.62 ± 0.14**	**0.53 ± 0.13**	**0.51 ± 0.17**
Bd	STL	*1.22 ± 0.26*	*1.09 ± 0.16*	*1.02 ± 0.16*	*0.99 ± 0.19*
	PoolingCI	***1.12 ± 0.17***	*1.02 ± 0.13*	0.68 ± 0.11	0.63 ± 0.14
	MTL-COR	***1.11 ± 0.23***	0.92 ± 0.13	0.83 ± 0.15	0.80 ± 0.17
	MTL-ICD	***1.08 ± 0.27***	**0.67 ± 0.17**	**0.55 ± 0.13**	**0.53 ± 0.13**
Be	STL	*1.43 ± 0.18*	*1.34 ± 0.10*	1.15 ± 0.15	0.99 ± 0.12
	PoolingCI	***1.38 ± 0.12***	*1.31 ± 0.07*	0.82 ± 0.09	0.76 ± 0.08
	MTL-COR	**1.28 ± 0.16**	1.07 ± 0.08	0.96 ± 0.10	0.89 ± 0.11
	MTL-ICD	**1.25 ± 0.26**	**0.68 ± 0.14**	**0.64 ± 0.07**	**0.61 ± 0.08**
Bf	STL	*2.73 ± 0.54*	2.45 ± 0.29	2.37 ± 0.34	2.29 ± 0.34
	PoolingCI	*2.63 ± 0.32*	*2.59 ± 0.22*	1.84 ± 0.24	1.71 ± 0.24
	MTL-COR	2.58 ± 0.61	2.07 ± 0.28	1.86 ± 0.35	1.68 ± 0.31
	MTL-ICD	2.42 ± 0.66	**1.30 ± 0.32**	**1.14 ± 0.23**	**1.10 ± 0.22**
Bg	STL	1.97 ± 0.48	1.81 ± 0.57		
	PoolingCI	*1.98 ± 0.50*	1.88 ± 0.59		
	MTL-COR	**1.89 ± 0.56**	1.67 ± 0.51		
	MTL-ICD	**1.85 ± 0.53**	**1.27 ± 0.49**		
Bh	STL	*2.01 ± 0.37*	*1.88 ± 0.27*	*1.81 ± 0.30*	*1.77 ± 0.49*
	PoolingCI	*1.84 ± 0.28*	*1.73 ± 0.21*	1.31 ± 0.23	1.32 ± 0.32
	MTL-COR	1.81 ± 0.36	1.41 ± 0.24	1.26 ± 0.26	1.16 ± 0.36
	MTL-ICD	**1.67 ± 0.36**	**0.99 ± 0.24**	**0.99 ± 0.22**	**0.93 ± 0.26**
Bi	STL	***0.85 ± 0.17***	***0.80 ± 0.15***	0.76 ± 0.09	0.77 ± 0.21
	PoolingCI	***0.84 ± 0.12***	***0.80 ± 0.09***	**0.67 ± 0.16**	**0.67 ± 0.27**
	MTL-COR	**0.79 ± 0.17**	**0.77 ± 0.29**	**0.71 ± 0.16**	**0.73 ± 0.28**
	MTL-ICD	***0.83 ± 0.14***	**0.84 ± 0.34**	**0.75 ± 0.19**	0.81 ± 0.41

Table 2 continued

MAE		Number of training samples			
Task/Method		5	10	15	20
All tasks					
Aa	STL	0.83 ± 0.17	0.57 ± 0.10	0.46 ± 0.06	**0.42 ± 0.06**
	PoolingCI	*0.95 ± 0.20*	0.52 ± 0.21	**0.35 ± 0.04**	**0.35 ± 0.04**
	MTL-COR	0.67 ± 0.15	0.54 ± 0.22	**0.43 ± 0.12**	**0.41 ± 0.10**
	MTL-ICD	**0.62 ± 0.26**	**0.38 ± 0.11**	**0.37 ± 0.09**	**0.35 ± 0.07**
Ab	STL	*1.46 ± 0.18*	*1.39 ± 0.17*	1.25 ± 0.16	1.16 ± 0.26
	PoolingCI	*1.44 ± 0.16*	1.11 ± 0.22	0.91 ± 0.15	0.83 ± 0.21
	MTL-COR	1.24 ± 0.18	1.12 ± 0.19	0.98 ± 0.18	0.89 ± 0.27
	MTL-ICD	**1.01 ± 0.38**	**0.58 ± 0.14**	**0.54 ± 0.12**	**0.51 ± 0.19**
Ac	STL	*2.29 ± 0.22*	1.99 ± 0.17	1.69 ± 0.13	1.44 ± 0.20
	PoolingCI	*2.26 ± 0.19*	1.48 ± 0.50	0.96 ± 0.12	0.87 ± 0.09
	MTL-COR	1.76 ± 0.23	1.47 ± 0.23	1.17 ± 0.17	1.02 ± 0.17
	MTL-ICD	**1.31 ± 0.72**	**0.69 ± 0.09**	**0.66 ± 0.06**	**0.64 ± 0.04**
Ad	STL	**0.31 ± 0.10**	**0.20 ± 0.07**	**0.18 ± 0.03**	**0.18 ± 0.02**
	PoolingCI	*0.38 ± 0.04*	**0.23 ± 0.08**	**0.21 ± 0.03**	**0.23 ± 0.05**
	MTL-COR	**0.26 ± 0.09**	**0.22 ± 0.10**	**0.25 ± 0.15**	**0.26 ± 0.12**
	MTL-ICD	**0.27 ± 0.09**	**0.19 ± 0.09**	**0.18 ± 0.03**	**0.18 ± 0.03**
Ae	STL	*0.86 ± 0.11*	0.64 ± 0.19	**0.49 ± 0.13**	**0.40 ± 0.05**
	PoolingCI	*0.83 ± 0.11*	0.58 ± 0.13	**0.47 ± 0.04**	**0.45 ± 0.04**
	MTL-COR	**0.63 ± 0.13**	0.57 ± 0.18	**0.51 ± 0.09**	**0.49 ± 0.07**
	MTL-ICD	**0.62 ± 0.18**	**0.44 ± 0.06**	**0.42 ± 0.06**	**0.40 ± 0.04**
Af	STL	*0.82 ± 0.13*	0.52 ± 0.18	**0.40 ± 0.06**	**0.36 ± 0.03**
	PoolingCI	*0.80 ± 0.10*	0.60 ± 0.10	0.51 ± 0.03	0.48 ± 0.03
	MTL-COR	0.67 ± 0.13	0.59 ± 0.09	0.53 ± 0.07	0.50 ± 0.11
	MTL-ICD	**0.59 ± 0.20**	**0.39 ± 0.06**	**0.36 ± 0.03**	**0.35 ± 0.03**
Ba	STL	*1.76 ± 0.43*	*1.53 ± 0.23*	*1.48 ± 0.15*	*1.46 ± 0.15*
	PoolingCI	*1.57 ± 0.39*	1.03 ± 0.23	0.77 ± 0.07	0.74 ± 0.07
	MTL-COR	1.35 ± 0.37	1.12 ± 0.15	1.01 ± 0.15	0.99 ± 0.14
	MTL-ICD	**1.08 ± 0.37**	**0.75 ± 0.11**	**0.67 ± 0.10**	**0.65 ± 0.09**
All tasks (continued)					
Bb	STL	*0.85 ± 0.22*	*0.80 ± 0.14*	*0.76 ± 0.25*	*0.66 ± 0.42*
	PoolingCI	*0.82 ± 0.14*	0.56 ± 0.15	**0.42 ± 0.12**	**0.38 ± 0.19**
	MTL-COR	*0.76 ± 0.18*	0.68 ± 0.16	0.61 ± 0.24	0.52 ± 0.36
	MTL-ICD	**0.65 ± 0.21**	**0.46 ± 0.11**	**0.41 ± 0.14**	**0.38 ± 0.21**
Bc	STL	*1.04 ± 0.52*	*0.85 ± 0.13*	*0.81 ± 0.12*	*0.81 ± 0.21*
	PoolingCI	*0.96 ± 0.20*	0.71 ± 0.14	**0.57 ± 0.12**	**0.53 ± 0.19**
	MTL-COR	***0.93 ± 0.33***	*0.75 ± 0.18*	0.67 ± 0.16	0.62 ± 0.19
	MTL-ICD	***0.85 ± 0.40***	**0.55 ± 0.11**	**0.50 ± 0.11**	**0.47 ± 0.17**
Bd	STL	*1.27 ± 0.56*	*1.09 ± 0.18*	*1.06 ± 0.18*	*0.97 ± 0.21*
	PoolingCI	*1.10 ± 0.18*	0.78 ± 0.16	**0.62 ± 0.10**	**0.55 ± 0.12**
	MTL-COR	*1.07 ± 0.27*	0.86 ± 0.14	0.78 ± 0.13	0.68 ± 0.14
	MTL-ICD	**1.06 ± 1.09**	**0.61 ± 0.14**	**0.57 ± 0.13**	**0.50 ± 0.13**
Be	STL	*1.43 ± 0.15*	*1.33 ± 0.10*	1.16 ± 0.16	0.98 ± 0.14
	PoolingCI	*1.37 ± 0.11*	0.94 ± 0.25	**0.69 ± 0.06**	**0.65 ± 0.06**
	MTL-COR	1.15 ± 0.14	0.99 ± 0.11	0.88 ± 0.11	0.80 ± 0.10
	MTL-ICD	**0.98 ± 0.34**	**0.63 ± 0.09**	**0.61 ± 0.07**	**0.60 ± 0.06**
Bf	STL	*2.85 ± 0.56*	2.47 ± 0.30	2.36 ± 0.30	2.30 ± 0.30
	PoolingCI	*2.65 ± 0.30*	2.07 ± 0.41	1.59 ± 0.20	1.52 ± 0.21
	MTL-COR	2.28 ± 0.45	1.90 ± 0.33	1.65 ± 0.28	1.60 ± 0.27
	MTL-ICD	2.01 ± 0.73	**1.18 ± 0.20**	**1.12 ± 0.16**	**1.12 ± 0.23**
Bg	STL	1.98 ± 0.49	1.74 ± 0.63		
	PoolingCI	1.97 ± 0.43	1.67 ± 0.56		
	MTL-COR	1.80 ± 0.48	1.39 ± 0.44		
	MTL-ICD	**1.69 ± 0.56**	**1.14 ± 0.54**		
Bh	STL	*2.00 ± 0.39*	*1.83 ± 0.29*	*1.81 ± 0.32*	*1.77 ± 0.45*
	PoolingCI	*1.78 ± 0.21*	1.32 ± 0.33	1.19 ± 0.26	1.25 ± 0.32
	MTL-COR	1.61 ± 0.42	1.24 ± 0.27	1.05 ± 0.23	**1.02 ± 0.29**
	MTL-ICD	**1.43 ± 0.54**	**0.95 ± 0.18**	**0.95 ± 0.19**	**0.97 ± 0.26**
Bi	STL	***0.85 ± 0.15***	***0.79 ± 0.13***	0.77 ± 0.14	0.75 ± 0.13
	PoolingCI	***0.83 ± 0.10***	**0.71 ± 0.11**	**0.66 ± 0.18**	**0.67 ± 0.31**
	MTL-COR	**0.81 ± 0.19**	***0.76 ± 0.13***	**0.72 ± 0.20**	**0.73 ± 0.30**
	MTL-ICD	***0.85 ± 0.29***	***0.85 ± 0.36***	0.81 ± 0.39	**0.74 ± 0.28**

Each experiment was repeated 100 times. Figures not significantly different from the null model (best model) are set in italics (bold) typeface; to increase readability, entries not in italics have a gray backbround

and PoolingCI; (4) the fractions of "fair" or better predictions are 64.4–82.9, 52.7–75.7, 51.1–77.1 % and 49.4–63.7 % for MTL-ICD, MTL-COR, PoolingCI, and STL, respectively. This indicates that for very few samples ($n = 5$), MTL can reduce the number of "awful" predictions, and that it can markedly increase the share of "fair" or better predictions for up to $n = 20$ samples.

Absolute performance values compare[5] favorably with previously published [20] results for larger models: For n=20 training samples and over tasks Aa, …, Af, Bb, …, Bf (scenario I), methods STL, PoolingCI, MTL-COR, MTL-ICD achieve, respectively, 0.63, 0.71, 0.71, 0.83 % of the median MAE of linear ridge regression models trained there using the same descriptors, but 90 % of all available training samples (see Table 1 for task sizes).

[5] Comparison is based on Table S2 of the supplement of Ref. [20], using column R' and third lines from each row of the common tasks.

Variance in performance tends to decrease with training set size: median ± median absolute deviation over all tasks of the correlation between standard deviation of MAE and training set size is −0.51 ± 0.39, −0.43 ± 0.46, −0.69 ± 0.26, −0.79 ± 0.1 for STL, PoolingCI, MTL-COR, MTL-ICD. There are occasional deviations from this general trend; e.g., in scenario I, the performance of MTL-ICD varies more strongly for 20 than for fewer training samples for tasks Bd, Be, Bf. These might be attributed to the fact that all experiments take place in a low sample regime. In accordance with this argument, the variance in the given examples is much reduced in scenario II.

Comparison of methods

In scenario I (Fig. 2), for phenols MTL-ICD performs best, followed by PoolingIC and MTL-COR. For 5 samples, MTL-ICD's winning margin is greatest, albeit at high

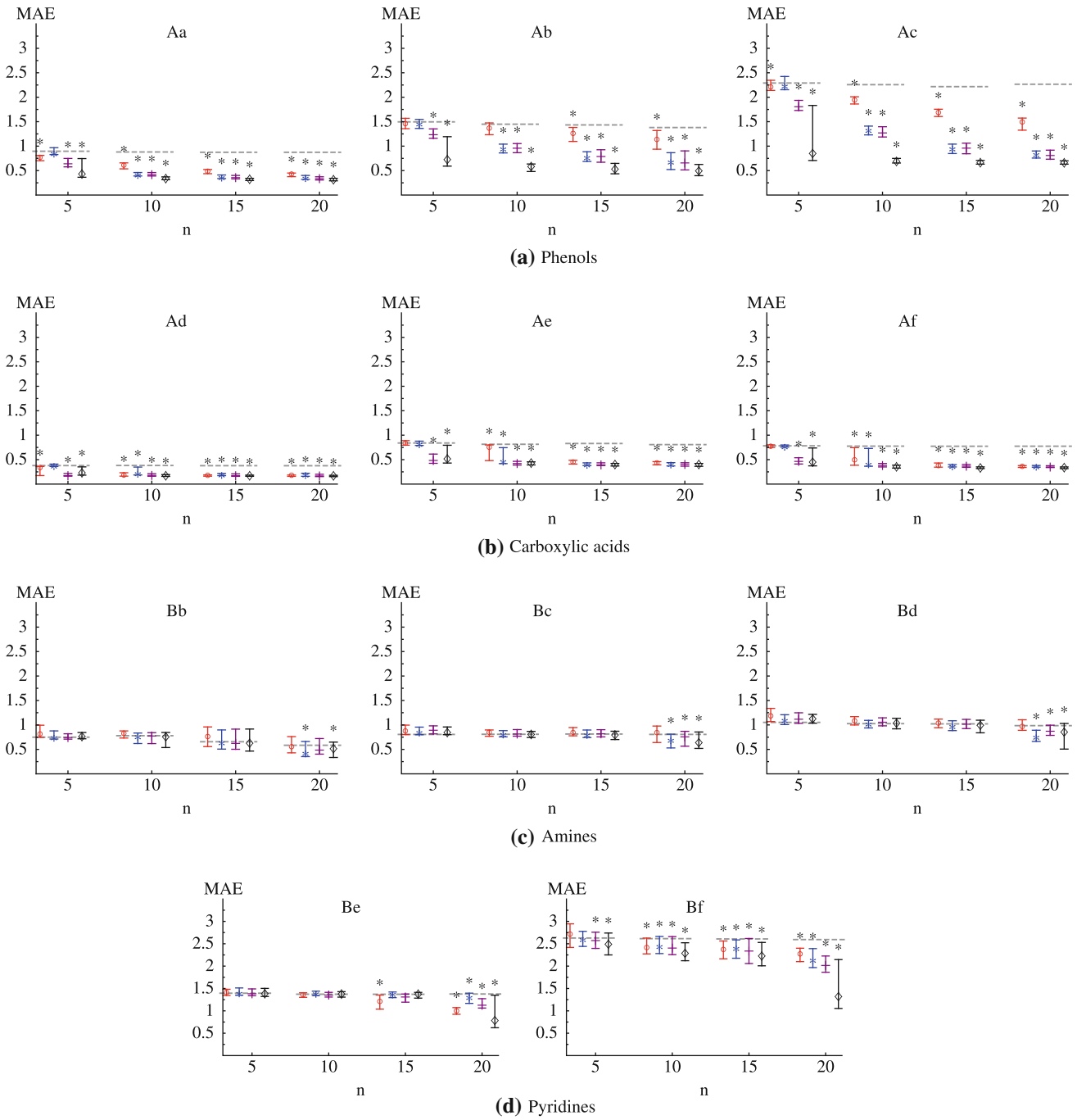
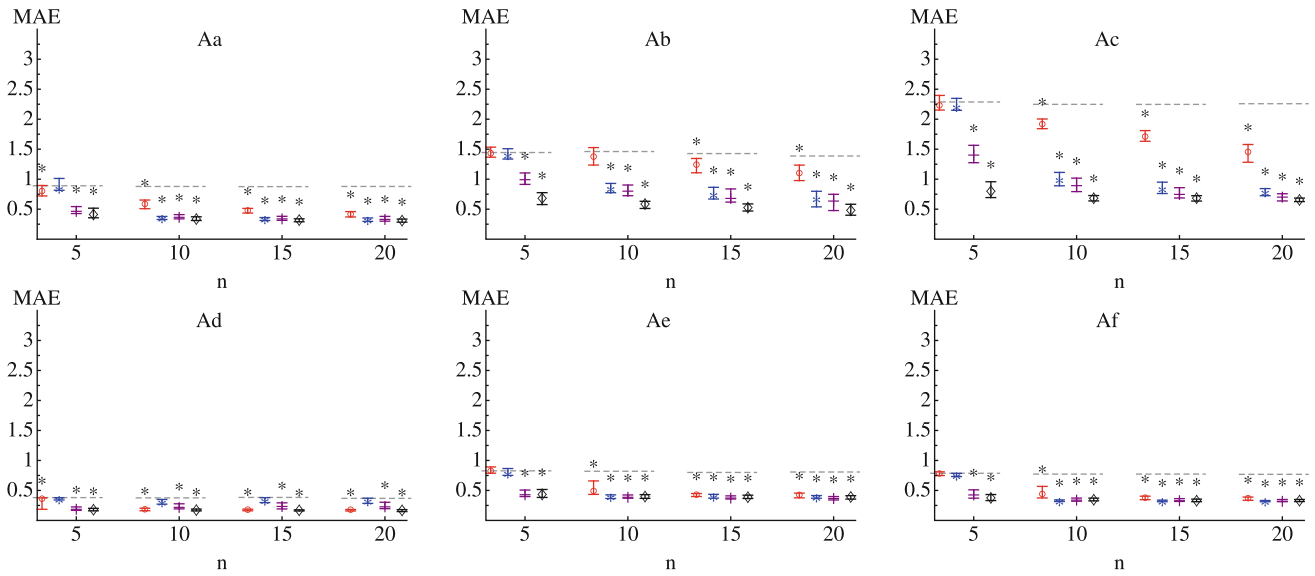


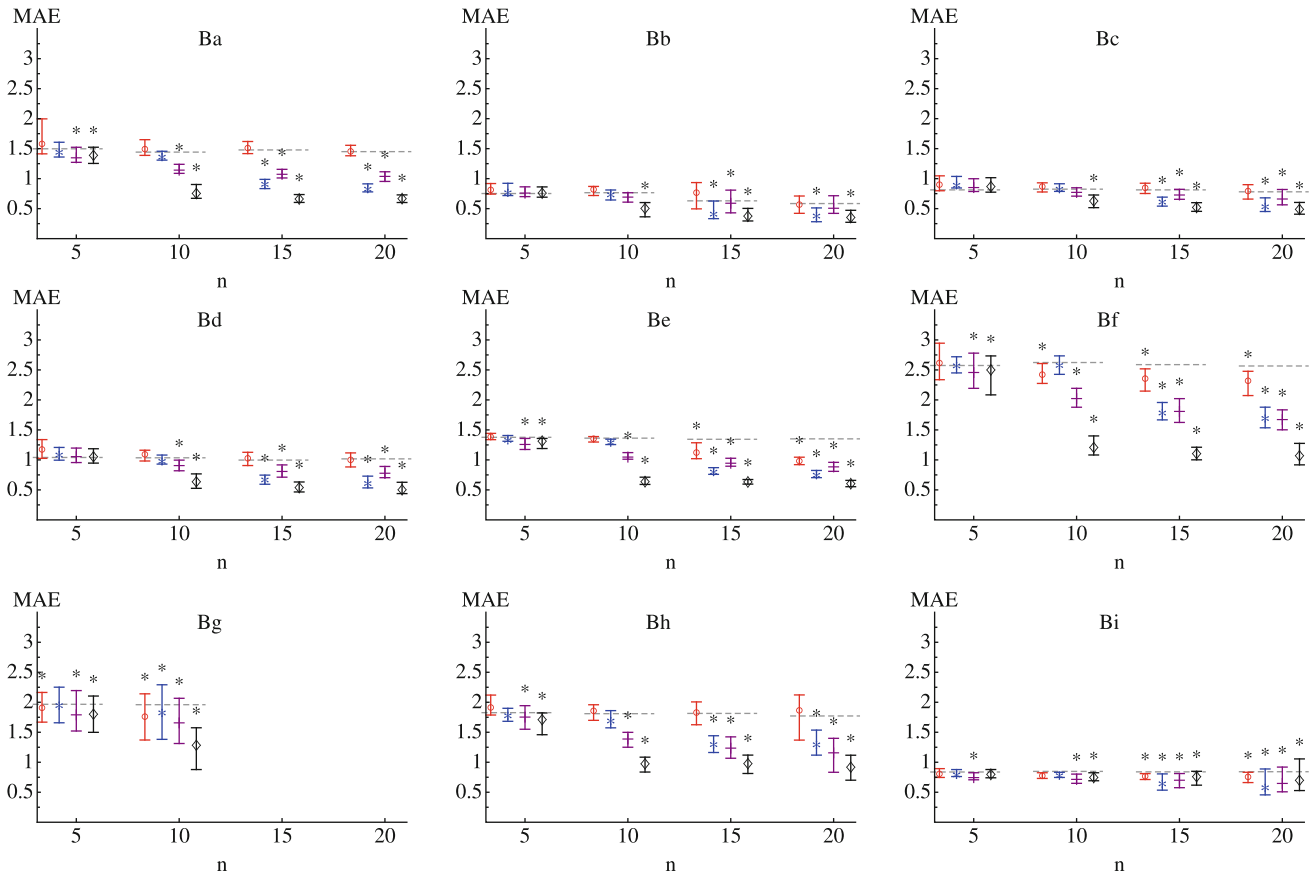
Fig. 2 Scenario I (few, highly similar tasks) performance. Shown are mean absolute error (MAE) 25 % quantile (*lower bars*), median (*symbols*), and 75 % quantiles (*upper bars*), for increasing number of training samples n = 5, 10, 15, 20 and methods STL (*red circle*), PoolingCI (*blue star*), MTL-COR (*purple bar*), and MTL-ICD (*black diamond*). *Dashed gray horizontal lines* indicate performance of the null model; *stars* indicate that a model performs significantly better than the null model. Each experiment was repeated 100 times. Plots are intended to provide an overview of larger overall differences in performance across tasks; the y-axis scaling intentionally does not resolve minor differences, as these are likely not significant due to noise, e.g., from experimental measurements and cross-validation

variance. For carboxylic acids, all methods perform on par, except for STL and PoolingIC for 5 and 10 samples. For amines, models outperform the null model only for 20 samples, except for STL, which never does. Behaviour differs for meta/para- and ortho-substituted pyridines, with large gains by MTL-ICD for 20 samples, again at high variance.

In scenario II (Fig. 3), the overall picture stays the same for the tasks from scenario I, except that the variance of MTL-ICD is reduced. On the other tasks (Ba, Bg, Bh, Bi),



(a) Acids



(b) Bases

Fig. 3 Scenario II (more, but less similar tasks) performance. Shown are mean absolute error (MAE) 25 % quantile (*lower bars*), median (*symbols*), and 75 % quantiles (*upper bars*), for increasing number of training samples n = 5, 10, 15, 20 and methods STL (*red circle*), PoolingCI (*blue star*), MTL-COR (*purple bar*), and MTL-ICD (*black diamond*). *Dashed gray horizontal lines* indicate performance of the null model; *stars* indicate that a model performs significantly better than the null model. Each experiment was repeated 100 times. Plots are intended to provide an overview of larger overall differences in performance across tasks; the y-axis scaling intentionally does not resolve minor differences, as these are likely not significant due to noise, e.g., from experimental measurements and cross-validation

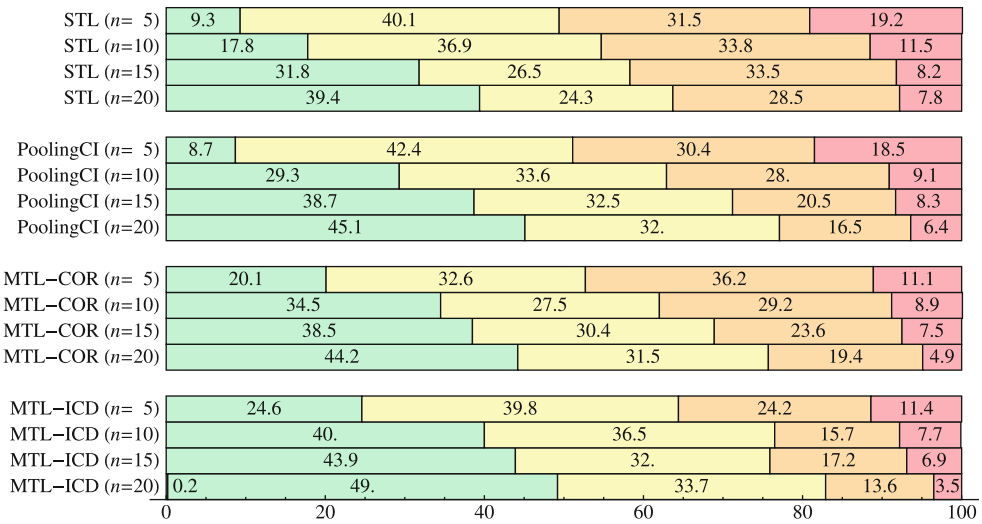


Fig. 4 Classification of predictive performance. Model's MAE (over 100 repetitions of all experiments in scenario I) are classified (from *left* to *right*) as excellent (*blue*, MAE $\leq$ 0.1), well (*green*, 0.1 $<$ MAE $\leq$ 0.5), fair (*yellow*, 0.5 $<$ MAE $\leq$ 1), poor (*orange*, 1 $<$ MAE $\leq$ 2), and awful (*red*, 2 $<$ MAE), for different methods and numbers of training samples. Numbers are percentages. Category excellent is occupied only once (MTL-ICD, $n = 20$, 0.2 %)

MTL-ICD mostly leads, followed by PoolingIC and MTL-COR, with STL often not able to outperform the null model. Scenario III (see supplement) resembles scenario II, except for occasionally increased variance in MAE.

Table 3 presents the fraction of tasks across all experiments in which a given method performed both best and better than the null model. Due to ties (if performance of two methods could not be statistically distinguished, both were considered to have performed best), columns do not add up to 100 %. We observe that (1) all methods improve with increasing number of samples (the only exception is MTL-COR from 5 to 10 samples). (2) STL performs worst; MTL methods improve over STL over all training set sizes. (3) PoolingCI and MTL-COR are on par in total, but MTL-COR performs better for small (5 and 10) training set sizes, whereas PoolingCI is better for 15 and 20 samples. (4) MTL-ICD outperforms all other methods by a wide margin.

In scenario I, improvements of MTL over STL are mostly seen on acid tasks (phenols, carboxylic acids). For bases (amines, pyridines), all methods perform equally bad, often not improving over the null model, the exception being ortho-substituted pyridines (task Bf). For 20 samples,

Table 3 Best performance by number of training samples

Method	Number of training samples				
	5	10	15	20	All
STL	4.9	7.3	25.6	28.2	16.2
PoolingCI	0.0	17.1	43.6	53.8	28.1
MTL-COR	36.6	24.4	33.3	38.5	33.1
MTL-ICD	68.3	87.8	87.2	97.4	85.0

Shown is the percentage of tasks over all scenarios for which a method performed best and better than the null model. Due to ties (when the performance of several methods could not be statistically distinguished), columns add up to more than 100 %

this improves somewhat. The introduction of additional related tasks in scenario II improves MTL performance, in particular for tertiary amines and pyridines. Scenario III does not introduce marked changes.

Average MAEs over all scenarios, tasks, repetitions, and training set sizes for STL, PoolingCI, MTL-COR, MTL-ICD are 1.12, 0.94, 0.90, 0.74 (median MAEs were 0.97, 0.80, 0.80, 0.63), in accordance with values reported in the literature [1, 6, 21], taking reduced training set size into account (see also subsection on performance in absolute terms).

In summary, methods rank by increasing overall performance in the order STL, PoolingCI, MTL-COR, MTL-ICD.

Performance and number of tasks

In scenario I, for phenols, carboxylic acids, and ortho-substituted pyridines, models improve on the null model from $n = 5$ training samples on. For amines and meta/para-substituted pyridines, this happens not until $n = 20$. Increasing the number of related tasks (scenario II) shifts this to smaller n, i.e., models improve on the null model from $n = 10$ (amines) and $n = 5$ (pyridines) on. Tasks not in scenario I (Ba, Bg, Bh, Bi) improve from $n = 5$ onwards. In scenario III, models improve starting from $n = 5$ for all tasks except for secondary amines (Bc, $n = 10$). This indicates that increasing the number of tasks allows for meaningful models earlier on, i.e., with fewer training samples per task, even if tasks are increasingly less related to the task of interest. However, adding more, but less related tasks (scenario III) also leads to marked increase in variance of MTL-ICD for very few ($n = 5$) samples.

Figure 5 presents improvements in MAE when increasing the number of tasks. In the transitions from

scenario I to scenario II to scenario III, more, but increasingly less related tasks are added; e.g., for a task in the phenols experiment, the other tasks are two phenols (scenario I), five acids (scenario II), and, 14 acids and bases (scenario III). STL does not improve as it does not make use of the additional tasks. MTL methods improve in MAE when related tasks are added (Fig. 5a), with MTL-ICD profiting the most. The average improvement of PoolingCI and MTL-COR is similar, with MTL-COR showing more stable performance. When adding more, but less related tasks (Fig. 5b), performance degrades for acids (−1.2, −8, −23.3, −9.9 % MAE for STL, PoolingCI, MTL-COR, MTL-ICD over tasks Aa,...,Af and n=5,10,15,20), but improves for bases (0.5, 12.3, 8.1, 7 %), leaving overall performance unchanged (except for a tendency of MTL-COR to degrade). A potential explanation is that acids have better initial performance and profit more from adding related tasks than bases do. Adding unrelated tasks does not help them further, but introduces noise, whereas bases can still profit from more information, even if it is only marginally related.

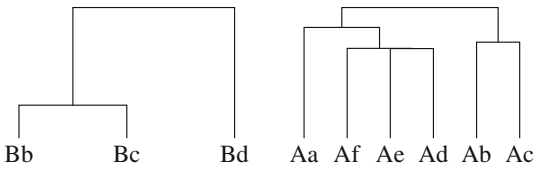
This indicates that adding related tasks tends to improve performance, whereas for less related tasks, this depends.

Task similarity

We show exemplarily how to interpret task similarity kernel matrices of methods MTL-COR and MTL-ICD.

Figure 6 shows dendrograms of learned dependencies for amines (based on MTL-COR with 5 samples) and acids (MTL-ICD with 5 samples). See the supplement for dendrograms for MTL-COR and MTL-ICD over all experiments and numbers of training samples. In Fig. 6a, primary and secondary amines are more correlated with each other than with tertiary amines. A possible reason could be that the solvent accessible area is smaller for tertiary amines with their three substituents, i.e., the quaternary aminium ion is less well solvated for tertiary amines than for secondary ones. In Fig. 6b, carboxylic acids are grouped together, separate from the phenols.

While these examples appear reasonable, other correlations are harder to explain; e.g., in scenarios II and III, ortho-substituted pyridines are often separate from all other tasks. Altogether, dendrograms tend to be shallow, with few pronounced subgroups.

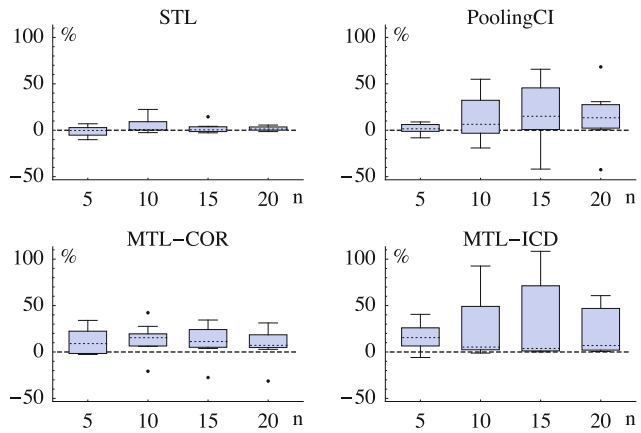


(a) Amines, MTL-COR **(b)** Acids, MTL-ICD

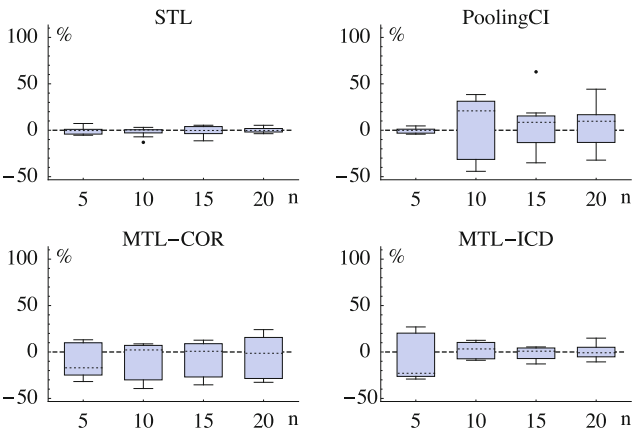
Fig. 6 Dendrograms for **a** amines by MTL-COR, and **b** acids by MTL-ICD, both for 5 training samples

Conclusions

We show that multi-task learning (MTL) methods can improve prediction performance in quantitative structure–property relationship modeling when few experimental measurements are available for the target task, but data is available for related (similar) tasks. Using prediction of acid dissociation constants (pK_a values) of small molecules as a model application, we conducted a retrospective



(a) From scenario I to scenario II.



(b) From scenario II to scenario III.

Fig. 5 Improvement in MAE when increasing the number of tasks. Shown are *box-whisker plots* of the improvement in MAE, expressed in percent, when going **a** from scenario I to scenario II, and **b** from scenario II to scenario III, for methods STL, PoolingCI, MTL-COR, MTL-ICD, and training set sizes n = 5, 10, 15, 20. Data for each *box-whisker plot* are the improvement in average MAE, expressed in percent, of the 11 tasks Aa, …, Af, Bb, …, Bf that are part of all three scenarios

validation study on a published data set ($n = 698$) divided into 15 chemically motivated compound classes that constitute the related tasks.

We compared performance of three MTL methods, a model trained only on the target task, and two models trained on multiple tasks data pooled together. All models are linear Gaussian process regression models. MTL methods outperform the model trained on only one task; adding related tasks increases their performance. One MTL method, based on an intrinsic model of co-regionalization and incomplete Cholesky decomposition, outperforms the other models, performing best in 85 % of all experiments. For as few as 5 training samples in each task, its mean absolute error is below one log-unit in 64 % of all cases. For 20 training samples per task, this increases to 83 %. This model also makes the most efficient use of data from additional tasks. The investigated MTL methods provide a measure of correlation between the tasks, and thus a limited form of insight into relationships between tasks.

MTL methods might prove useful in situations where computational estimates of physico-chemical or other molecular properties are required and data is scarce, but related data is available. An example is the investigation of new compound series, where few measurements exist for compounds of the new series, but more measurements might exist for structurally related compound series. This might be particularly useful if experimental determination is expensive, e.g., computationally designed compounds that would need to be synthesized prior to experimental measurement.

Acknowledgments The authors thank Klaus-Robert Müller, Gisbert Schneider, Tiago Rodrigues, and an anonymous referee for helpful suggestions, and David Manallack for the provision of data. M. Rupp and K. Hansen acknowledge partial support by FP7-ICT programme of the European Community (PASCAL2) and DFG (grant MU 987/4-2). M. Rupp acknowledges partial support by FP7 programme of the European Community (Marie Curie IEF 273039). G. Sanguinetti and G. Skolidis acknowledge support from the Engineering and Physical Sciences Research Council (EPSRC, grant EP/F009461/2). G. Sanguinetti is funded by the Scottish government through the SICSA initiative.

References

1. Rupp M, Körner R, Tetko IV (2010) Predicting the pK_a of small molecules. Comb Chem High Throughput Screen 14(5):307–327
2. Lee A, Crippen G (2009) Predicting pK_a. J Chem Inf Model 49(9):2013–2033
3. Fraczkiewicz R (2006) In silico prediction of ionization. In: Testa B, Waterbeemd H (eds) Comprehensive medicinal chemistry II, vol 5, Elsevier, Oxford, pp 603–626
4. Wan H, Ulander J (2006) High-throughput pK_a screening and prediction amenable for ADME profiling. Expert Opin Drug Metab Toxicol 2(1):139–155
5. Ho J, Coote M (2010) A universal approach for continuum solvent pK_a calculations: are we there yet? Theor Chim Acta 125(1–2):3–21
6. Tehan B, Lloyd E, Wong M, Pitt W, Gancia E, Manallack D (2002) Estimation of pK_a using semiempirical molecular orbital methods. Part 2: application to amines, anilines and various nitrogen containing heterocyclic compounds. Quant Struct Act Rel 21(5):473–485
7. Caruana R (1997) Multi-task learning. Mach Learn 28:41–75
8. Jacob L, Vert JP (2008) Protein-ligand interaction prediction: an improved chemogenomics approach. Bioinformatics 24(19): 2149–2156
9. Varnek A, Gaudin C, Marcou G, Baskin I, Pandey A, Tetko I (2009) Inductive transfer of knowledge: application of multi-task learning and feature net approaches to model tissue-air partition coefficients. J Chem Inf Model 49(1):133–144
10. Ning X, Rangwala H, Karypis G (2009) Multi-assay-based structure-activity relationship models: improving structure-activity relationship models by incorporating activity information from related targets. J Chem Inf Model 49(11):2444–2456
11. Mordelet F, Vert JP (2011) ProDiGe: PRioritization of disease genes with multitask machine learning from positive and unlabeled examples. BMC Bioinf 12:389
12. Rossotti F, Rossotti H (1961) The determination of stability constants and other equilibrium constants in solution. McGraw-Hill, New York
13. Hasselbalch KA (1916) Die Berechnung der Wasserstoffzahl des Blutes aus der freien und gebundenen Kohlensäure desselben, und die Sauerstoffbindung des Blutes als Funktion der Wasserstoffzahl. Biochem Z 78:112–144
14. Clark J, Perrin D (1964) Prediction of the strength of organic bases. Q Rev Chem Soc 18:295–320
15. Perrin DD, Dempsey B, Serjeant EP (1981) pK_a Prediction for organic acids and bases. Chapman and Hall/CRC Press, Boca Raton
16. Lyman W, Reehl W, Rosenblatt D (eds) (1982) Handbook of chemical property estimation methods: environmental behavior of organic compounds. McGraw-Hill, New York
17. Livingstone D (2003) Theoretical property predictions. Curr Top Med Chem 3(10):1171–1192
18. Hammett L (1937) The effect of structure upon the reactions of organic compounds. Benzene derivatives. J Am Chem Soc 59(1):96–103
19. Ertl P (1997) Simple quantum chemical parameters as an alternative to the Hammett sigma constants in QSAR studies. Quant Struct Act Rel 16(5):377–382
20. Rupp M, Körner R, Tetko IV (2010) Estimation of acid dissociation constants using graph kernels. Mol Inf 29(10):731–740
21. Tehan B, Lloyd E, Wong M, Pitt W, Montana J, Manallack D, Gancia E (2002) Estimation of pK_a using semiempirical molecular orbital methods. Part 1: application to phenols and carboxylic acids. Quant Struct Act Rel 21(5):457–472
22. Howard P, Meylan W (1999) Physical/chemical property database (PHYSPROP). Syracuse Research Corporation, Environmental Science Center, 6225 Running Ridge Road, North Syracuse, New York
23. Fukui K, Yonezawa T, Nagata C (1954) Theory of substitution in conjugated molecules. Bull Chem Soc Jpn 27(7):423–427
24. Sadowski J, Gasteiger J (1993) From atoms and bonds to three-dimensional atomic coordinates: automatic model builders. Chem Rev 93(7):2567–2581
25. Stewart J (1997) MOPAC: a general molecular orbital package. Quant Chem Prog Exch 10:86
26. Sushko I, Novotarskyi S, Körner R, Pandey AK, Rupp M, Teetz W, Brandmaier S, Abdelaziz A, Prokopenko VV, Tanchuk VY, Todeschini R, Varnek A, Marcou G, Ertl P, Potemkin V, Grishina M, Gasteiger J, Schwab C, Baskin II, Palyulin VA, Radchenko EV, Welsh WJ, Kholodovych V, Chekmarev D, Cherkasov A, de Sousa JA, Zhang QY, Bender A, Nigsch F, Patiny L, Williams A,

Tkachenko V, Tetko IV (2011) Online chemical modeling environment (OCHEM): web platform for data storage, model development and publishing of chemical information. J Comput Aided Mol Des 25(6):533–554
27. Rasmussen CE, Williams CK (2005) Gaussian processes for machine learning. MIT Press, Cambridge
28. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning. Data mining, inference, and prediction, 2nd edn. Springer, New York
29. Cressie NA (1993) Statistics for spatial data. Wiley, New York
30. Bonilla E, Chai KM, Williams C (2008) Multi-task Gaussian process prediction. In: Platt J, Koller D, Singer Y, Roweis S (eds) Advances in neural information processing systems 20. MIT Press, Cambridge, pp 153–160
31. Rebonato R, Jäckel P (1999) The most general methodology for creating a valid correlation matrix for risk management and option pricing purposes. J Risk 2(2):17–27
32. Skolidis G, Sanguinetti G (2011) Bayesian multitask classification with Gaussian process priors. IEEE Trans Neural Netw 22(12):2011–2021
33. Wilcoxon F (1945) Individual comparisons by ranking methods. Biometr Bull 1(6):80–83
34. Manallack D (2007) The pK_a distribution of drugs: application to drug discovery. Perspect Med Chem 1:25–38
35. Liao C, Nicklaus M (2009) Comparison of nine programs predicting pK_a values of pharmaceutical substances. J Chem Inf Model 49(12):2801–2812