Supplementary information to Big Data meets Quantum Chemistry Approximations: The Δ-Machine Learning Approach

Raghunathan Ramakrishnan[1], Pavlo O. Dral[2,3], Matthias Rupp[1], and O. Anatole von Lilienfeld[1,4*]
[1] *Institute of Physical Chemistry and National Center for Computational Design and Discovery of Novel Materials,*
, Department of Chemistry, University of Basel,
Klingelbergstrasse 80, CH-4056 Basel, Switzerland
[2] *Max-Planck-Institut für Kohlenforschung, Kaiser-Wilhelm-Platz 1, 45470 Mülheim an der Ruhr, Germany*
[3] *Computer-Chemie-Centrum, University of Erlangen-Nuremberg,*
Nägelsbachstr. 25, 91052 Erlangen, Germany and
[4] *Argonne Leadership Computing Facility, Argonne National Laboratory, 9700 S. Cass Avenue, Lemont, IL 60439, USA*
(Dated: January 30, 2015)

I. MACHINE LEARNING

Machine learning (ML) [1–5] is a subfield of artificial intelligence that studies algorithms whose performance improves with data (inductive "learning from experience"). [1] Its main concerns are the systematic identification and exploitation of regularity (non-randomness) in data, e.g., for prediction or analysis. It has been successfully applied in a wide variety of fields, including brain-computer interfaces, recommender systems, robotics, and, chemistry [6, 7].

A. Kernel ridge regression

Here, we use kernel ridge regression (KRR) [5] models for regression. KRR is a nonlinear version of ordinary regression with regularization to prevent overfitting.[1] Let $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathbb{R}^d$ denote a set of N training samples, e.g., vectorial representations of molecules, and let $y_1, \dots, y_N$ denote corresponding labels, e.g., energies, or, differences in energies. Our KRR models take the form

$$f(\mathbf{x}) = \sum_{i=1}^{N} \boldsymbol{\alpha}_i k(\mathbf{x}, \mathbf{x}_i), \tag{1}$$

where $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a positive semi-definite function, called a *kernel*, that measures similarity between samples. The $\boldsymbol{\alpha}_i \in \mathbb{R}$ are regression coefficients, obtained as the solution to the optimization problem

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \sum_{i=1}^{N} \big(f(\mathbf{x}_i) - y_i\big)^2 + \lambda \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha}, \tag{2}$$

where $\mathbf{K}_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ is the kernel matrix of the training data, and $\lambda \in \mathbb{R}$ is a regularization constant that controls the trade-off between minimizing the squared error and a penalty term for large regression coefficients. Setting the derivative of Eq. (2) to zero yields the closed-form solution

$$\boldsymbol{\alpha} = \big(\mathbf{K} + \lambda \mathbf{I}\big)^{-1} \mathbf{y}. \tag{3}$$

Note that the resulting predictions are formally equivalent to those of Gaussian process regression. [8] KRR is a non-parametric form of regression: Each training sample adds another regression coefficient (effectively, a basis function).

In this work, we use the Laplacian kernel

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{||\mathbf{x} - \mathbf{z}||_1}{\sigma}\right), \tag{4}$$

where $||\mathbf{x}||_1 = \sum_i |\mathbf{x}_i|$ denotes the 1-norm. $\sigma \in \mathbb{R}{\geq 0}$ is a free parameter related to the length scale of the problem.

The length scales in Eq. (4), together with the regularization constant λ, are free hyperparameters of the method, not determined by Eq. (2) but rather dialed in by the user. We optimize them using the Nelder-Mead method [9, 10] in combination with cross-validated (see below) mean absolute error as the target function. For performance estimation, this is done in an inner loop of cross-validation.

B. Cross-validation

The simplest approach to model validation is to set aside part of the data as a *hold-out set*, train the model on the remaining data, then evaluate its performance on the hold-out set. The disadvantage of this approach is that it requires many data points. Cross-validation [11] is a statistical validation method for efficient utilization of limited available reference data. In k-fold cross-validation,

*anatole.vonlilienfeld@unibas.ch

[1] Fitting data points too closely leads to unwanted behavior when interpolating between them ("over-fitting"). This is often reflected in large regression coefficients of opposite sign that cancel each other only on the training data. Regularization prevents this by penalizing large coefficients. Note that this particularly affects noisy data such as outcomes of experimental measurements as the model's flexibility is misused to adapt to noise. For noise-free data such as results of computational procedures this is much less so, although other factors like wrong choice of model class can cause noise-like effects.

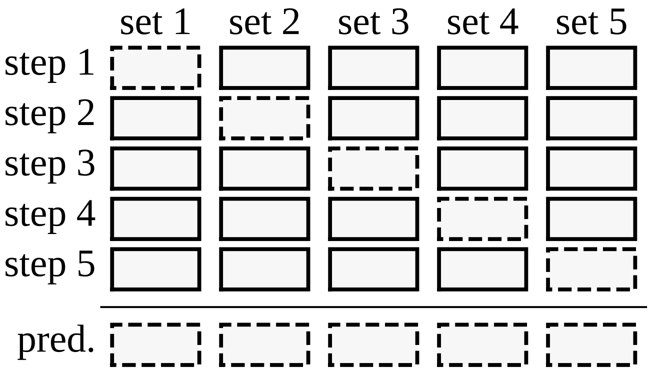

FIG. 1. Schematic for 5-fold cross-validation. Solid boxes represent training data, dashed boxes represent predicted test data.

the data is split into k parts, each of which serves as test (hold-out) set in turn, while the others jointly serve as training set. This results in one prediction for each datum, made by a model *that was not trained using this datum* (Fig. 1). The following pseudocode summarizes the procedure:

```
1. Randomly partition the data set into $k$ subsets of (almost) equal size.
2. For $i = 1, \ldots, k$,
    (a) Train model on data from sets $\{1, \ldots, k\} \backslash \{i\}$
    (b) Use model to predict set $i$
```

Cross-validation can be used to estimate the performance of a model on a given data set. Frequently used measures for this are the mean absolute error (MAE) $\frac{1}{N}\sum_{i=1}^{N} |f(\mathbf{x}_i) - y_i|$, the root mean squared error (RMSE) $\left(\frac{1}{N}\sum_{i=1}^{N} |f(\mathbf{x}_i) - y_i|^2\right)^{1/2}$, and the squared Pearson product-moment correlation coefficient R^2 for

$$R = \frac{\sum_{i=1}^{N}\big(f(\mathbf{x}_i) - \overline{f(\mathbf{x})}\big)\big(y_i - \bar{y}\big)}{\sqrt{\sum_{i=1}^{N}\big(f(\mathbf{x}_i) - \overline{f(\mathbf{x})}\big)^2}\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}}, \qquad (5)$$

where $\overline{f(\mathbf{x})}$ and $\bar{y}$ denote the respective means. Cross-validated performance can be used to select hyperparameters. If both is done at the same time, *nested* cross-validation must be used to avoid erroneously over-optimistic performance estimates. This is done by using an outer loop of cross-validation for the performance estimation. For each predicted fold, the training set (consisting of all other folds) is subjected to a new (inner) loop of cross-validation to select hyperparameters. A model is then built on the (outer loop's) training data using these hyperparameters, and the (outer loop's) test data fold is predicted. See ref. [12] for further details and a related application.

C. Representation

We numerically represent molecules, given by atomic numbers and coordinates $\{Z_i, \mathbf{R}_i\}$, using the *Coulomb matrix* [13]

$$\mathbf{M}_{i,j} = \begin{cases} 0.5 Z_i^{2.4} & i = j \\ \frac{Z_i Z_j}{||\mathbf{R}_i - \mathbf{R}_j||_2} & i \neq j \end{cases}. \qquad (6)$$

This symmetric matrix representation, based on internal distances, is invariant with respect to all translational and rotational degrees of freedom. Invariance with respect to indexing of atoms is enforced by sorting the atom index according to 2-norm (Euclidean) of an atom's row and column (simultaneously swapping them). [2] Two matrices are compared via their 1-norm (Manhattan),

$$||\mathbf{M} - \mathbf{M}'||_1 = \sum_{i=1}^{m}\sum_{j=1}^{m} |\mathbf{M}_{i,j} - \mathbf{M}'_{i,j}|. \qquad (7)$$

For matrices of different size (i.e., molecules with different numbers of atoms), the smaller matrix is extended with zeros.

II. ELIMINATING SYSTEMATIC ERRORS

Evaluating the performance of a model using mean absolute error MAE, $\overline{|P_{\text{pred.}} - P_{\text{ref.}}|}$, is often not conveneint if the model introduces a systematic shift in the property w.r.t. the reference property values. For example the method B3LYP has been shown to introduce errors that grow with increasing system size [14]. In such cases, one can eliminate the systematic error using the generalized MAE, $\overline{|P_{\text{pred.}} - P_{\text{ref.}} + \eta|}$. In the main paper, we used this formula to calculate the MAE of different baseline/targetline combinations.

[2] Invariance with respect to translation, rotation, and atom indexing is important because otherwise the model would have to explicitly learn these, leading to a combinatorial blow-up of required training set size.

TABLE I. Shifts [kcal/mol] used in the calculation of MAE for estimating the internal energy of atomization at $T = 0$ K (U_0) (Fig. 1 in the paper) and enthalpy of atomization at 298.15 K (Table 1 in the paper) at G4MP2 level of theory using various baseline theories.

base	target	
	U_0 - G4MP2	H - G4MP2
H_{f} - PM7	22.0	0.1
E_e- PBE	177.8	155.5
E_e- B3LYP	95.3	73.0

TABLE II. Shifts [kcal/mol] used in the calculation of MAE for estimating atomization energies for various combinations of increasingly correlated post Hartree-Fock methods as target and baseline methods (Fig. 3 in the paper).

base	target		
	MP2	CCSD	CCSD(T)
HF	-439.9	-375.8	-399.5
MP2		64.0	40.4
CCSD			-23.7

[1] T. M. Mitchell, *Machine Learning* (McGraw Hill, 1997).

[2] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. (Wiley, New York, 2001).

[3] D. MacKay, *Information theory, Inference, and Learning Algorithms* (Cambridge University Press, Cambridge, 2005).

[4] C. Bishop, *Pattern Recognition and Machine Learning* (Springer, Berlin, 2006).

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*, 2nd ed. (Springer, New York, 2009).

[6] O. Ivanciuc, in *Reviews in Computational Chemistry*, Vol. 23, edited by K. Lipkowitz and T. Cundari (Wiley, Hoboken, 2007) Chap. 6, pp. 291–400.

[7] A. Varnek and I. Baskin, J. Chem. Inf. Model. **52**, 1413 (2012).

[8] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning* (MIT Press, Cambridge, 2006).

[9] J. A. Nelder and R. Mead, Comput. J. **7**, 308 (1965).

[10] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes. The Art of Scientific Computing*, 3rd ed. (Cambridge University Press, Cambridge, 2007).

[11] G. Cawley and N. Talbot, J. Mach. Learn. Res. **11**, 2079 (2010).

[12] K. Hansen, G. Montavon, F. Biegler, S. Fazli, M. Rupp, M. Scheffler, O. A. von Lilienfeld, A. Tkatchenko, and K.-R. Müller, J. Chem. Theor. Comput. **9**, 3543 (2013).

[13] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, Phys. Rev. Lett. **108**, 058301 (2012).

[14] M. D. Wodrich, C. Corminboeuf, P. R. Schreiner, A. A. Fokin, and P. v. R. Schleyer, Org. Lett. **9**, 1851 (2007).