Metric learning for kernel ridge regression: assessment of molecular similarity

IOP Publishing Mach. Learn.: Sci. Technol. 3 (2022) 035015 https://doi.org/10.1088/2632-2153/ac8e4f

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
20 June 2022

REVISED
18 August 2022

ACCEPTED FOR PUBLICATION
31 August 2022

PUBLISHED
22 September 2022

PAPER

Metric learning for kernel ridge regression: assessment of molecular similarity

Raimon Fabregat[1,4], **Puck van Gerwen**[1,2,4], **Matthieu Haeberle**[1,3,4], **Friedrich Eisenbrand**[3] and **Clémence Corminboeuf**[1,2,*]

[1] Laboratory for Computational Molecular Design, Institute of Chemical Sciences and Engineering, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland
[2] National Center for Competence in Research-Catalysis (NCCR-Catalysis), Zurich, Switzerland
[3] Chair of Discrete Optimisation, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland
[4] These authors contributed equally to this work.
[*] Author to whom any correspondence should be addressed.

E-mail: clemence.corminboeuf@epfl.ch

Keywords: metric learning, kernel-ridge regression, molecular similarity, physics-based machine learning

Supplementary material for this article is available online

Abstract

Supervised and unsupervised kernel-based algorithms widely used in the physical sciences depend upon the notion of *similarity*. Their reliance on pre-defined distance metrics—e.g. the Euclidean or Manhattan distance—are problematic especially when used in combination with high-dimensional feature vectors for which the similarity measure does not well-reflect the differences in the target property. *Metric learning* is an elegant approach to surmount this shortcoming and find a property-informed transformation of the feature space. We propose a new algorithm for metric learning specifically adapted for kernel ridge regression (KRR): *metric learning for kernel ridge regression* (MLKRR). It is based on the Metric Learning for Kernel Regression framework using the Nadaraya-Watson estimator, which we show to be inferior to the KRR estimator for typical physics-based machine learning tasks. The MLKRR algorithm allows for superior predictive performance on the benchmark regression task of atomisation energies of QM9 molecules, as well as generating more meaningful low-dimensional projections of the modified feature space.

1. Introduction

Over the last decade, supervised and unsupervised machine learning methods have established themselves as reliable tools in the physical sciences [1–12]. Supervised models aim to find a mathematical relationship between input data and target properties. Kernel-based regression methods, e.g. kernel ridge regression (KRR) and Gaussian process regression (GPR), are popular in the field [3, 4, 13–15]. While equivariant neural networks have recently also become state-of-the-art models to predict molecular properties [16–19], KRR models remain important, particularly for small datasets. Unsupervised machine learning algorithms instead find underlying structure in unlabelled data. Dimensionality reduction methods like t-distributed stochastic neighbor embedding (t-SNE) [20], PCA, Multidimensional scaling or Isomap [21] enable the visualisation of an otherwise uninterpretable high-dimensional feature space [10–12]. A central concept to many supervised and unsupervised approaches is *similarity*: the underlying assumption being that points close in the feature space should be close in property space. Similarity between two elements in a feature space may be constructed using a kernel, a function that outputs a scalar value between 1 (identical) or 0 (entirely dissimilar). Many kernel functions contain a decreasing exponential of a metric, such as the Euclidean distance (Gaussian kernel) or the Manhattan distance (Laplacian kernel). Alternative measures such as the linear kernel (the dot product between the feature vectors), or the cosine similarity also exist.

Such pre-defined distance metrics pose problems, especially when a feature space is high-dimensional and possibly polluted with irrelevant or redundant features. *Ab-initio* representations widely used in chemistry, materials science and condensed matter physics [3, 4] illustrate these limitations. These

representations are designed to encode the relevant information to describe any molecular structure: typically, using non-linear functions of atom types and positions. Popular examples are SLATM [22], SOAP [23] and FCHL [24, 25], among many others [4, 26, 27]. They are used to develop predictive models for a variety of properties [25, 26, 28–38] or to facilitate the visualisation and interpretation of the chemical space [10–12]. Yet, these representations do not provide a universally meaningful measure of molecular similarity, resulting in sub-optimal performance when more challenging properties are targeted [39]. This deficiency is intrinsically connected to the use of pre-defined metrics that treats all the features on a equal footing regardless of their redundancy or relevance to a particular task. For instance, the Euclidean distance used in the Gaussian kernel ($d_E(\mathbf{a},\mathbf{b}) = \sqrt{\sum_i (a_i - b_i)^2} = ||\mathbf{a}-\mathbf{b}||_2$) is dominated by features with high variance, which we found [39] to not necessarily correlate with predictive capabilities. While redundancy in a feature space is easily eliminated by using unsupervised linear dimensionality reduction algorithms such as PCA and CUR decomposition [40], adapting the relevance of descriptive features in a metric for a particular target requires a supervised approach [41].

Metric learning (or distance metric learning, or similarity learning) [42, 43] provides an elegant solution to adapt the notion of similarity (and distance) according to the target property. Thus, similarity can be transformed from a global concept (two molecules are similar based on their structure) to an application-specific concept (two molecules are similar, in the context of their properties). This conceptual shift enables a more specific definition of similarity, which naturally circumvents many of the pitfalls of pre-defined similarity notions, as well as enabling more accurate machine learning models. Despite its promise, metric learning has remained largely absent in the chemical domain, except for a few examples on graph-structured data [44]. This is likely because most frameworks are designed for clustering or classification [45–47], while chemistry is dominated by regression tasks. We note that alternatives to metric learning exist with the same goal of optimising representations for a specific task: for example in the GPR framework [48], in (supervised) contrastive learning [49–52] or using deep belief networks [53, 54].

One of the few examples of metric learning algorithms focused specifically on regression tasks is metric learning for kernel regression (MLKR) developed by Weinberger and Tesauro [45]. MLKR learns a linear transformation matrix that transforms the distance metric between points in order to optimise the prediction of a particular target in a kernel regression. However, the kernel regression used in MLKR employs the non-parametric Nadaraya-Watson (NW) estimator, which, as demonstrated in this work, is less suited than the KRR estimator for regression tasks. The NW estimator, which is essentially a Nearest-Neighbours estimator, evaluates similarity between points in a relative manner using a notion that is dependent on the distribution of points within a particular dataset. In KRR, the notion of similarity is instead absolute, i.e. independent of the distribution of data. This is an important distinction if the aim is to accurately predict molecular properties. Within this context, identifying the trial molecule that is *close* (in the absolute sense) to the molecular system of interest is more relevant than identifying the *closest* points that potentially correspond to dissimilar molecules.

In order to enable metric learning in the KRR framework, we propose a new algorithm—metric learning for kernel ridge regression (MLKRR)—which modifies the MLKR formalism for the KRR estimator. While this is not the first metric learning algorithm that exists for KRR [42, 55], to our knowledge this is the first effort that is not an adaptation of a algorithm originally designed for clustering or classification. We choose to start from the MLKR algorithm [45] because of its focus placed on the regression task (though not KRR). Finally, we demonstrate the improved performance of MLKRR on the prototypical task of regressing atomisation energies of small molecules in the QM9 dataset [56]. We expect that MLKRR could enable the wide-spread adoption of metric learning for kernel-based machine learning tasks in the physical sciences, thereby re-defining the fundamental notion of similarity upon which they depend.

2. Metric learning

Metric learning algorithms transform a feature space in order to construct a distance function that minimises the prediction error of a specific property. They learn a linear transformation matrix A that generates a *Mahalanobis distance* [57] (d_M) on the original space

$$d_M(\mathbf{a},\mathbf{b}) = ||A\mathbf{a} - A\mathbf{b}||_2 = ||A(\mathbf{a}-\mathbf{b})||_2. \quad (1)$$

Since A^TA is positive semidefinite, there exists an orthogonal matrix Q as well as non-negative diagonal matrix D such that $A^TA = Q^TDQ$, see [45]. Thus, the transformation of the feature space with A corresponds to a rotation and a scaling of the components. In this way, distances of points can be increased or decreased, depending on their relevance for the target property.

2.1. Metric learning for kernel regression

Weinberger and Tesauro [45] propose the following method to apply the transformation matrix A in the context of prediction. Suppose that the data points are $x_i \in \mathbb{R}^d$ and the predictions are $y_i \in \mathbb{R}$, $i = 1, \ldots, n$. The transformation is used in the *Gaussian kernel* as follows

$$k(x_i, x_j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{||A(x_i - x_j)||_2^2}{\sigma^2}}. \tag{2}$$

The authors rely on the *NW estimator*

$$\widehat{y}_i = \frac{\sum\limits_{j \neq i} y_j k_{ij}}{\sum\limits_{j \neq i} k_{ij}}. \tag{3}$$

The prediction of a new data point x is then

$$f(x) = \widehat{y} = \frac{\sum\limits_{j} y_j k(x, x_j)}{\sum\limits_{j} k(x, x_j)}. \tag{4}$$

The learning of the matrix A is expressed as an optimisation problem. The goal is to minimise the residual sum of squares $\mathcal{L} := \sum_i (y_i - \widehat{y}_i)^2$. The optimisation then relies on the gradient of the loss with respect to the transformation matrix A

$$\frac{\partial \mathcal{L}}{\partial A} = \frac{4}{\sigma^2} A \sum_i (\widehat{y}_i - y_i) \frac{1}{Z_i} \sum_{j \neq i} (\widehat{y}_i - y_j) k_{ij} x_{ij} x_{ij}^T, \tag{5}$$

where $x_{ij} := x_i - x_j$ and $Z_i := \sum_{j \neq i} k_{ij}$.

2.2. Metric learning for kernel ridge regression

We introduce MLKRR, which uses the standard parametric *KRR estimator* instead:

$$\widehat{y}_i = \sum_j \alpha_j k(x_i, x_j). \tag{6}$$

The prediction of a new data point x is then:

$$f(x) = \widehat{y} = \sum_j \alpha_j k(x, x_j). \tag{7}$$

The coefficients α_j that minimise the sum of squares $\mathcal{L}_\alpha := \sum_i (y_i - \widehat{y}_i)^2$ are obtained by solving the linear equation [58]:

$$\alpha = (K + \lambda I)^{-1} y \tag{8}$$

where $K \in \mathbb{R}^{n \times n}$ is the matrix defined by the Gaussian kernel $k(x_i, x_j)$, λ is a small regularisation parameter and I is the identity matrix. In classical KRR, the α coefficients are optimised for a fixed kernel. In the MLKR approach described above on the other hand, only the matrix A is optimised. The novelty of our method [42] lies in the specific *combination* of both principles. We now explain the two-step procedure that computes first the estimator and then the metric.

We first sample n_α data points $\{x_i^{(\alpha)}\}_{i=1}^{n_\alpha}$ and their corresponding labels $\{y_i^{(\alpha)}\}_{i=1}^{n_\alpha}$ from our dataset in order to compute the estimator from the coefficients α. It follows from the expression (8) above that α may be written in terms of the matrix A, hidden in the kernel matrix $K_{ij} := k(x_i^{(\alpha)}, x_j^{(\alpha)})$. Once the estimator is found, we use its predictions (7) in order to optimise the matrix A. For this, a new loss function is considered on new data points. We sample n_A new data points $\{x_i^{(A)}\}_{i=1}^{n_A}$ with their corresponding labels $\{y_i^{(A)}\}_{i=1}^{n_A}$. The predictions of these points are hence given by

$$\widehat{y}_i^{(A)} := \sum_j \alpha_j k\left(x_i^{(A)}, x_j^{(\alpha)}\right), \tag{9}$$

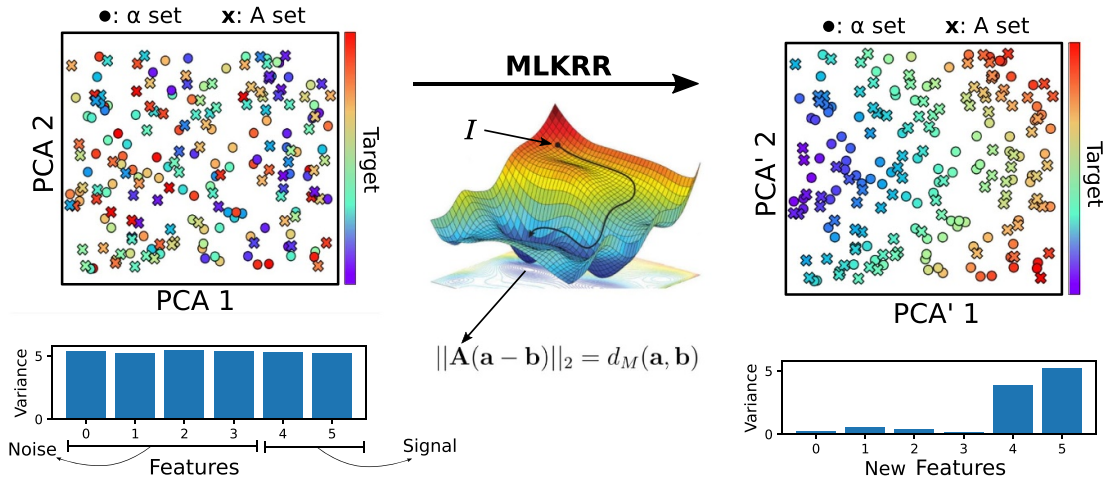

Figure 1. MLKRR transforms and rotates the components of the original feature space. A training set is divided into two sets, labelled α and A. The colours of the points represent the corresponding function values. The projection of the points with standard PCA does not show a correlation of the value with the first principal components, while the PCA projection of the points (PCA') that have been transformed with A illustrate such a correlation. The transformed point-set features are less noisy and the relevant signals are amplified.

which in turn yield the second loss function to minimise $\mathcal{L}_A := \sum_i (y_i^{(A)} - \hat{y}_i^{(A)})^2$. The computation of the matrix A is done by gradient descent using the gradient $\nabla_A \mathcal{L}_A$. To express the gradient in a closed form, we introduce the following notation.

The predictions (9) impose the definition of the additional kernel matrix $Q_{ij} := k(x_i^{(A)}, x_j^{(\alpha)})$. Further, we set $X^{(\alpha)}$ and $X^{(A)}$ the matrices whose rows are the data points of corresponding superscript. Let also $y^{(\alpha)}$ and $y^{(A)}$ be the vectors whose entries are the property labels of corresponding superscript. The gradient $\nabla_A \mathcal{L}_A$ is now given by

$$\begin{aligned}\frac{\partial \mathcal{L}_A}{\partial A} = &-\frac{4}{\sigma^2} A \left(-X^{(A)^T} W X^{(\alpha)} - X^{(\alpha)^T} W^T X^{(A)} + X^{(A)^T} R X^{(A)} + X^{(\alpha)^T} S X^{(\alpha)} \right) \\ &+ \frac{4}{\sigma^2} A X^{(\alpha)^T} \left(-\tilde{W} - \tilde{W}^T + \tilde{R} + \tilde{S} \right) X^{(\alpha)},\end{aligned} \tag{10}$$

where $W_{ij} := (\hat{y}_i^{(A)} - y_i^{(A)}) \alpha_j Q_{ij}$, $\tilde{W}_{ab} := K_{ab} \alpha_b \left[(K + \lambda I)^{-T} Q^T (\hat{y}^{(A)} - y^{(A)}) \right]_a$. The diagonal matrices $R, S, \tilde{R}, \tilde{S}$ are the vertical and horizontal sums of W and $\tilde{W}$, that is $R_{ii} := \sum_j W_{ij}$, and $S_{jj} := \sum_i W_{ij}$. More on the definitions of these matrices, together with the proof of correctness of the expression is given in the appendix.

Figure 1 illustrates the MLKRR algorithm at work. Initially, there is no correlation between the two principal components (i.e. high-variance components) of a high dimensional dataset and the target property. The MLKRR algorithm learns a transformation matrix A by minimising the squared distance between the predicted target property and the property labels of training data. The matrix A redefines the distance metric on the original feature space. As illustrated in the right panel of figure 1, the high-variance components of the feature space now correlate with the target property. We note that here the matrix A is square such that the transformation rotates and scales the components, but it could be non-square to reduce the dimensionality of the feature space [42].

We implemented MLKR and MLKRR in a python module found at `github.com/lcmd-epfl/MLKRR` based on the python library `metric-learn` [59].

2.3. Conceptual comparison of MLKR and MLKRR

The different nature of the similarity used by the KRR and NW estimators affects their respective regression performance. Given a datapoint x in two different data distributions, the KRR definition of similarity between x and a neighbour a is independent of any other points in the distribution. On the other hand, the NW definition of similarity adapts to the distribution of data, so that a point x is similar to a if a is the closest point to x (see figure 2(a)). Effectively, functions regressed with NW result in piece-wise surfaces resembling Voronoi diagrams, where each region of the feature space is dominated by the nearest data point. Alternatively, KRR generates surfaces that transition smoothly, which generally results in superior prediction performance. This is clearly observed in figure 2(b), which shows the result of a simple regression on a 2D feature space.

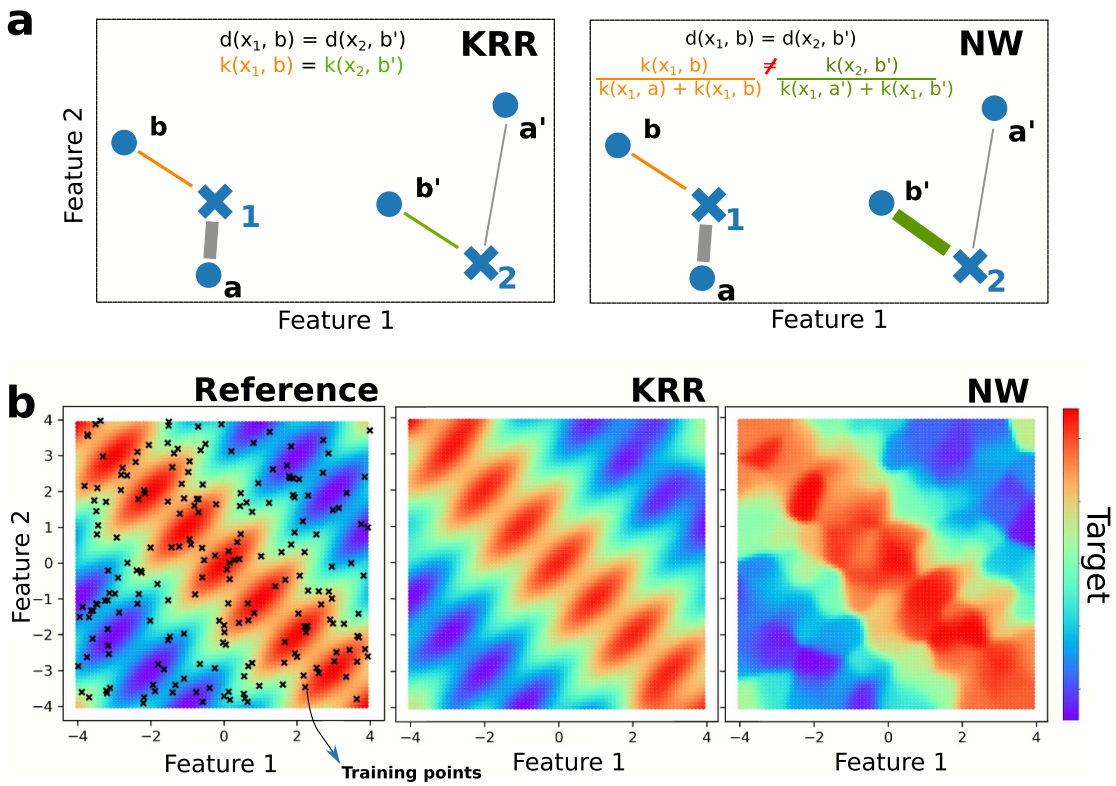

Figure 2. (a) Depiction of the different approaches to define similarity between KRR and NW. (b) Target function depending on two features and the corresponding regressions based on the KRR and the NW estimators. The two models were trained on a limited sample of data points marked as crosses in the first plot.

3. Computational details

3.1. Dataset

All models were built using a random subset of 24 000 molecules from the QM9 dataset [56] of 134 000 three-dimensional structures and corresponding atomisation energies of small drug-like molecules (up to nine heavy atoms). The initial set was then split into 20 000 for training, 2000 for validation (fitting hyperparameters for the KRR, and tracking the accuracy of the models throughout optimisation) and 2000 for testing (used for the out-of-sample error in the learning curves). An equivalent model for the HOMO-LUMO gap (20% gain in performance) is discussed in the supplementary material.

3.2. FCHL representation

Molecules are represented using the Faber-Christensen-Huang-Lilienfeld 19 (FCHL19) representation [25] as implemented in the `qml` python package [60], but other representations could have been used. An equivalent model trained using the BoB [27] representation shows similar improvement and is given in the supplementary material. The default settings were used for all of the FCHL19 parameters. Local representations were converted to global ones by summing all of the atomic contributions, such that the eventual representations were a vector consisting of 720 features.

3.3. Optimisation details

The MLKR and MLKRR algorithms were optimised for enough time to reach convergence, which was around 2000 steps in both cases (see figure 3). The function `minimize` of the SciPy [61] library was used to optimise the matrix A, which itself uses the L-BFGS-S [62] algorithm. A was initialised as the identity matrix. For the MLKRR, we used $n_\alpha = n_A$, although some of our tests suggest that $n_\alpha < n_A$ could be superior. In order to reduce overfitting for the MLKRR, the training data was reshuffled into A and α sets every 30 optimisation steps. The starting kernel width σ for the MLKRR was $\sigma = 55$, and the regularisation parameter was fixed at $\lambda = 1 \times 10^{-9}$. These parameters are the optimal values for the standard KRR, which were optimised using a grid search on the test set.

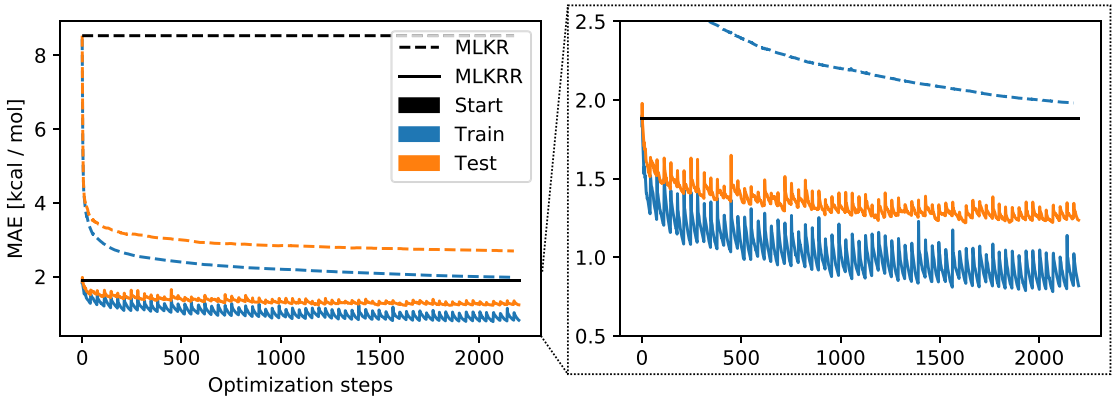

Figure 3. Learning curves showing the evolution of the mean absolute error (MAE) throughout the optimisation process for our implementations of MLKR and MLKRR. Both converge after around 2000 steps.

4. Results and discussion

4.1. Learning transformation matrices

Training the MLKR and MLKRR algorithms consists of optimising the relevant transformation matrices A. Figure 3 illustrates the evolution of the optimisation process for the two algorithms. Despite the noisier optimisation of the MLKRR, which is a result of the periodic reshuffling of the datasets to optimise both A and α, both algorithms converge after approximately 2000 steps.

After the optimisation process, we obtain the optimised transformation matrices A, which are visualised in figure 4. Both matrices A_{MLKR} and A_{MLKRR} contain many non-zero values, both along the diagonal and in off-diagonal terms. Modifications to diagonal terms are effectively a re-weighting of the original features, akin to the application-based re-weighting of terms by Ceriotti *et al* [63]. Here however, the re-weighting is entirely learned by the algorithm. Diagonal terms A_{ii} where $|A_{ii}| > 1$ indicate that a higher weight is applied to specific features, whereas $|A_{ii}| < 1$ indicate that a lower weight is applied. $A_{ii} = 0$ indicates that the feature is dropped entirely, reducing the dimensionality of the feature space. Off-diagonal terms $A_{ij}, i \neq j$, are linear combinations of the original features. The highly correlated nature of the FCHL features is illustrated by the fact that the transformation matrices are rather smooth (shown in the panels on the right of figure 4). After convergence, the largest terms in the transformation matrices are the diagonal values. This is partially due to the fact that the initial guess for the optimisation process is the identity matrix. It also indicates that the original features are useful to predict the target property with an appropriate re-weighting (i.e. a selection and amplification of relevant features for the target property). The selection of off-diagonal terms acts to combine features that are originally correlated, i.e. generating new features. Thus, the metric learning process naturally applies a feature selection protocol.

4.2. Improving predictions of atomisation energies of QM9 molecules

With our trained transformation matrices in hand, we now proceed to evaluating the new feature space for the prediction of atomisation energies of QM9 molecules. As we showed in our previous work [39], unless modified by feature selection or metric learning protocols, feature variance is not necessarily correlated to predictive capabilities. Here, since we have modified the feature space and corresponding distance metric, we expect the variance to again correlate with the relevance of features for a target property. In figure 5, the variance of the original FCHL features and modified $\mathrm{FCHL}_{\mathrm{MLKR}}$ and $\mathrm{FCHL}_{\mathrm{MLKRR}}$ features is shown. The original and transformed features do not represent exactly the same information, as the transformed features are constructed as a linear combination of the original ones. Nevertheless, they are still the dominant components, as seen in the diagonal from the A matrices in figure 4. As observed, $\mathrm{FCHL}_{\mathrm{MLKRR}}$, and to a lesser extent, $\mathrm{FCHL}_{\mathrm{MLKR}}$ makes use of almost all of the feature space. This suggests that the metric learning procedure effectively manipulates all of the features to eliminate irrelevance and redundancy.

The goal of the metric learning procedure is to improve predictions on the target property. In figure 6, we compare the relative capabilities of the distance metric learned by MLKR and MLKRR to improve predictions in a KRR model. The learning curves illustrate the evolution of the MAE with increasing training data. In the left panel of figure 6, we observe that the metric obtained with MLKRR offers a significant improvement over the original. The learning curve shows a faster decrease and the MAE of the final model is ∼38% lower. On the other hand, the metric obtained with MLKR in fact performs worse than the original one. The learning curve is shifted upwards and the performance of the final model is ∼24% worse. This suggests that the

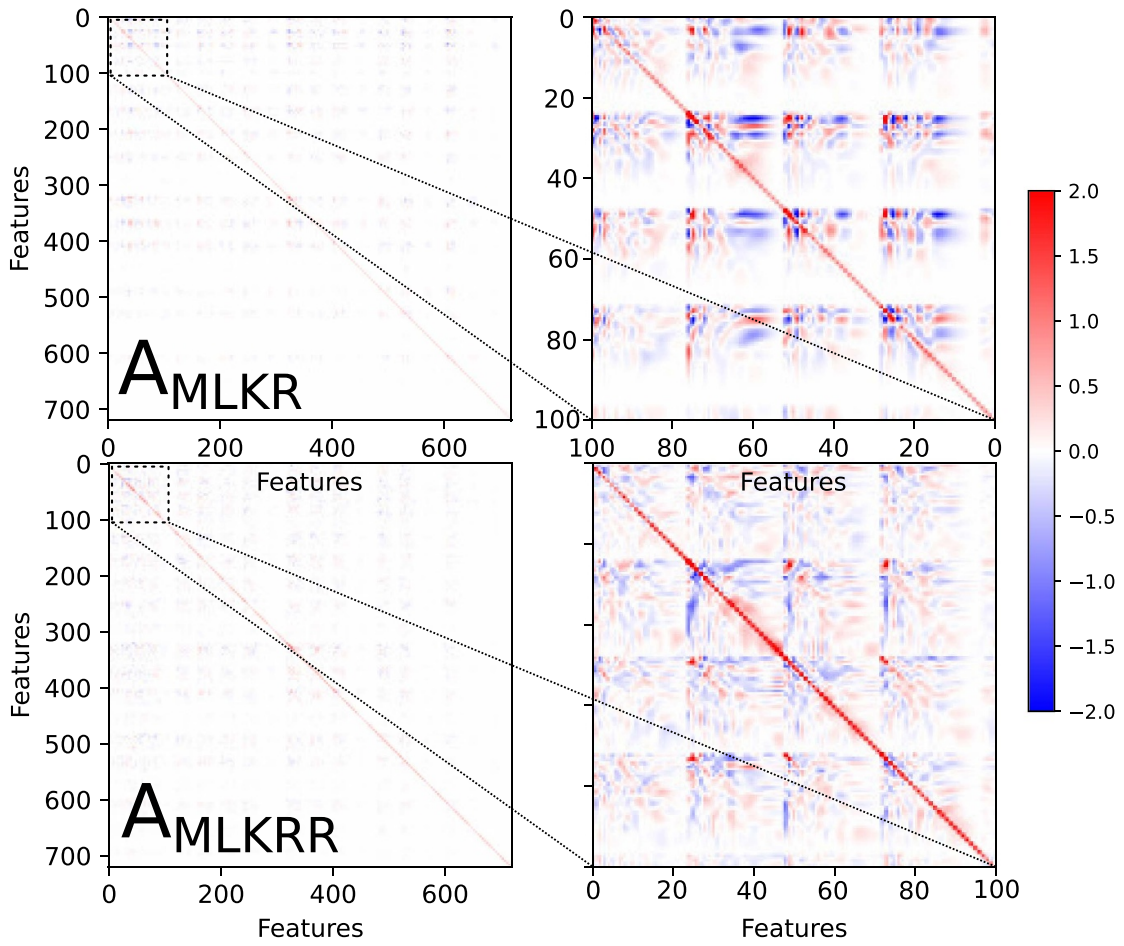

Figure 4. Colourmaps representing the learned transformation matrices A for MLKR (A_{MLKR}) and MLKRR (A_{MLKRR}). The lefthand panels illustrate all matrix elements, whereas the righthand panels illustrate zoomed-in components. Both transformation matrices amplify the diagonal components in order to select relevant features for the target property, as well as choosing off-diagonal components to reduce the number of redundant features. The colour scale is capped in the range $[-2, 2]$ to facilitate the visualisation of the off-diagonal elements (the highest absolute values of the matrices are below 5).

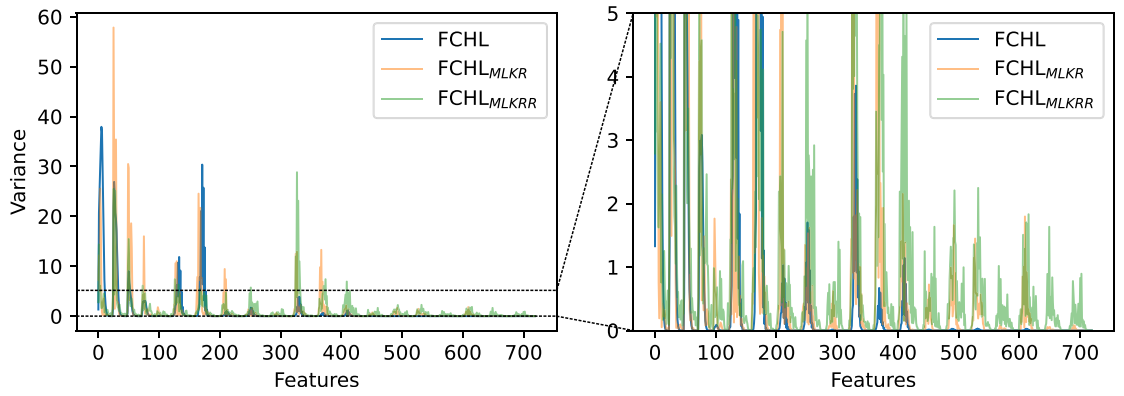

Figure 5. Feature variances of the original FCHL and the metric-learned alternatives ($\mathrm{FCHL}_{\mathrm{MLKR}}$ and $\mathrm{FCHL}_{\mathrm{MLKRR}}$) on the QM9 dataset (left), with a zoom on the $[0, 5]$ range of the y axis (right).

distance metric learned by the MLKR framework does not necessarily improve its capabilities for KRR. Since the MLKR optimisation procedure relies on the NW estimator, we suspect that the nature of this estimator compared to that of the KRR results in a metric that is less suitable for regression tasks.

4.3. Dimensionality reduction

We illustrate the effects of our metric-learned similarity measures by constructing 2D projections of the QM9 dataset using the feature spaces spanned by FCHL, $\mathrm{FCHL}_{\mathrm{MLKR}}$ ($\mathrm{FCHL} \cdot \mathbf{A}_{\mathrm{MLKR}}^{T}$) and $\mathrm{FCHL}_{\mathrm{MLKRR}}$

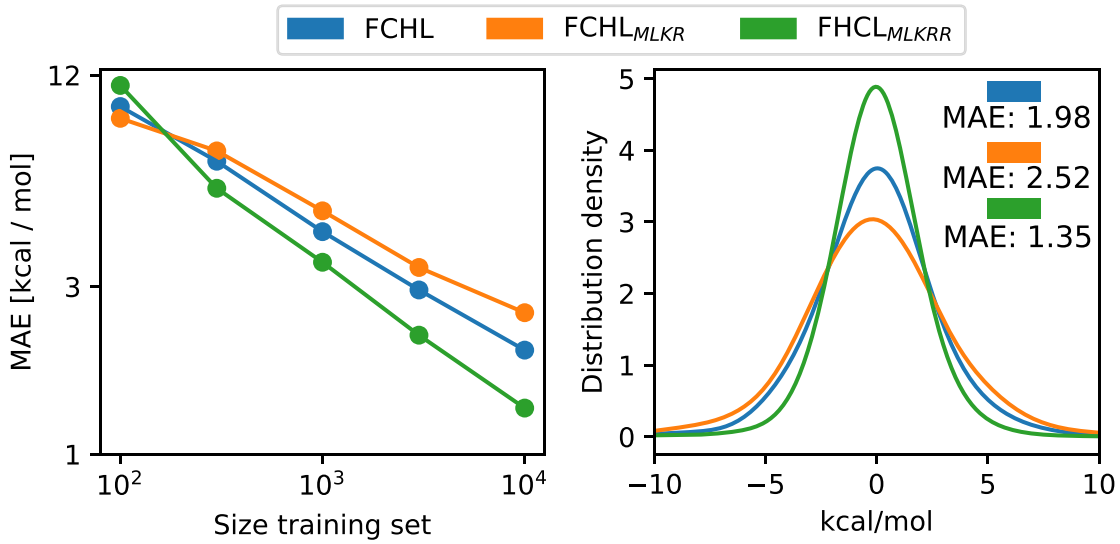

Figure 6. Learning curves with the evolution of the MAE on a validation set of 2k structures obtained using a KRR model (left) and distribution of errors in the same set obtained with a KRR constructed using 10 000 data points from the training set (right), generated using the FCHL representation and the altered variations obtained with MLKR (FCHL$_{\mathrm{MLKR}}$) and MLKRR (FCHL$_{\mathrm{MLKRR}}$).

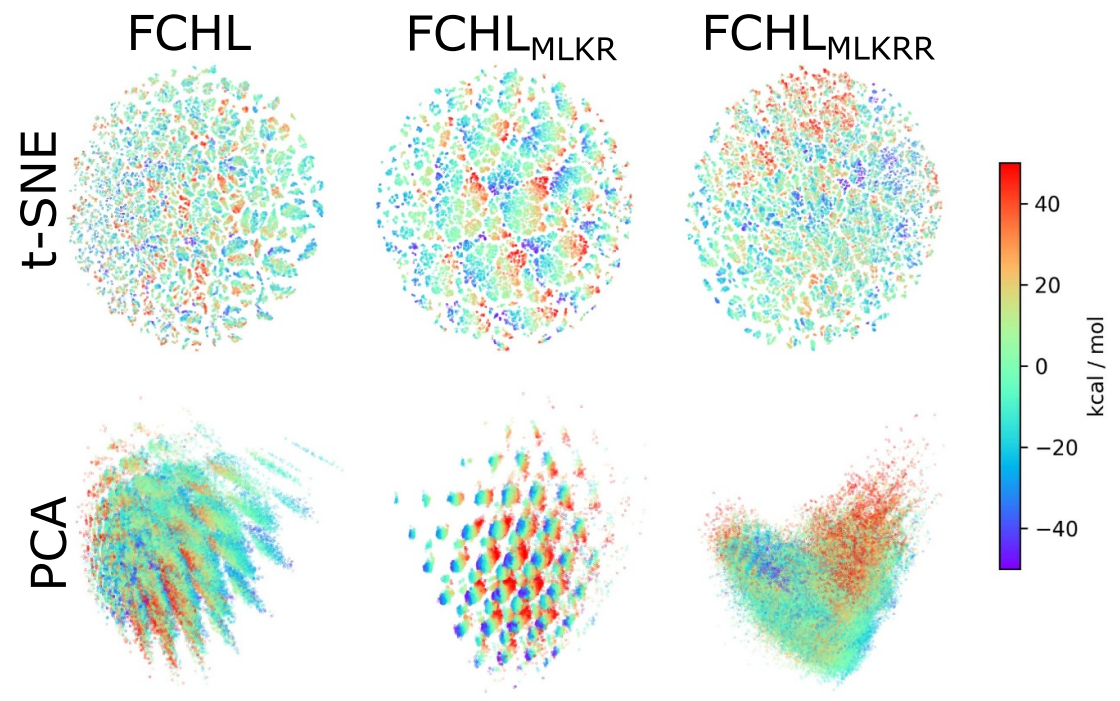

Figure 7. 2D projections of FCHL, FCHL$_{\mathrm{MLKR}}$ and FCHL$_{\mathrm{MLKRR}}$ constructed using t-SNE and PCA algorithms (first and second row, respectively). The colourmap shows the atomisation energy for each compound.

(FCHL $\cdot \mathbf{A}^{T}_{\mathrm{MLKRR}}$). t-SNE projects data points in a N-dimensional space ($N = 2$ in general) by minimising the difference of the pairwise affinity between points in the original and new spaces. The definition of affinity between two points in t-SNE is:

$$p_{ij} = e^{-\gamma d(x_i,x_j)^2} \Big/ \sum_{k \neq i} e^{-\gamma d(x_i,x_k)^2} \tag{11}$$

akin to the weights of the kernel average for the predictions in the NW estimator of MLKR (see equation (3)). PCA instead performs the projection such that the new dimensions are those with the highest variance in the original data. It does not rely on an explicit distance as in the t-SNE, but still selects the principal components by evaluating $\mathbf{X}^T\mathbf{X}$ (where $\mathbf{X}$ is the feature vector), which is analogous to a distance.

The t-SNE and PCA maps obtained with FCHL, $\text{FCHL}_{\text{MLKR}}$ and $\text{FCHL}_{\text{MLKRR}}$ are shown in figure 7, where each of the points representing a molecule are coloured by their atomisation energy. Distinct differences between the maps are observed. The t-SNE and PCA maps obtained with the $\text{FCHL}_{\text{MLKR}}$ features tend to organise local clusters. Yet, there is no global coherence between the organisation of the projected points and the coloured target property. Instead, the $\text{FCHL}_{\text{MLKRR}}$ maps resemble more closely those of the original FCHL, albeit organised into larger regions offering a better coherence with respect to the target property. Overall, MLKRR improves the organisation of data to uncover patterns (figure 7) relevant to optimise the prediction of the targeted properties (figure 6).

The proposed MLKRR algorithm offers additional functionalities beyond maximising prediction accuracy for specific applications. Careful analysis of the learned transformation matrices provide valuable insights as to why some representations behave better than others [15]. In addition, the algorithm offers a mathematical route to construct a hierarchy of molecular representations adaptable and tailored to specific targets as an alternative to building representations encoding the relevant information in absolute terms [4]. Comparisons between the similarities and metrics learned for a wide range of applications might also uncover hidden trends useful to develop improved and more general molecular representations. Finally, MLKRR could also be exploited in transfer-learning or meta-learning approaches: a concept which has so far been limited to neural network applications [64].

5. Conclusions

Similarity-based machine learning methods are widely used in the physical sciences. Their dependence on a pre-defined distance metric is problematic in combination with high-dimensional feature vectors often containing irrelevant or redundant features. To address this shortcoming, we introduce an algorithm, MLKRR, which re-defines the notion of similarity between points to optimise the prediction of specific target properties in KRR tasks. MLKRR was shown to offer improved performance (38%) on the prototypical regression task of atomisation energies of the QM9 dataset [56], as well as generating more meaningful low-dimensional projections of transformed feature vectors. In addition, we illustrate why the MLKRR algorithm is more suited to the prediction of continuous target properties, as is typical in the physical sciences, than the related MLKR algorithm based on the NW estimator.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://github.com/lcmd-epfl/MLKRR.

Acknowledgments

R F and C C acknowledge funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant 817977). P v G and C C acknowledge the National Centre of Competence in Research (NCCR) 'Sustainable chemical process through catalysis (Catalysis)' of the Swiss National Science Foundation (SNSF, Grant No. 180544) for financial support. F E acknowledges support from the Swiss National Science Foundation (SNSF) (Grant 185030). M H, F E and C C acknowledge the FSB Intrafaculty Seed Funding. We thank Mojmir Mutny and Alberto Fabrizio for helpful discussions.

Appendix. Derivation of MLKR and MLKRR gradients

Derivation of the MLKR gradient in equation (5)
Starting from the top, we find

$$\frac{\partial \mathcal{L}}{\partial A} = 2\sum_i (\widehat{y}_i - y_i)\frac{\partial \widehat{y}_i}{\partial A},$$

where $\widehat{y}_i = \frac{1}{Z_i}\sum_{j\neq i} k_{ij}y_j$ and $Z_i = \sum_{j\neq i} k_{ij}$.

This naturally leads us to compute $\frac{\partial \widehat{y}_i}{\partial A}$, and hence $\frac{\partial k_{ij}}{\partial A}$. Firstly, the kernel derivative is

$$\begin{aligned}\frac{\partial k_{ij}}{\partial A} &= -\frac{1}{\sigma^2}k_{ij}\frac{\partial}{\partial A}\left(d(x_i,x_j)^2\right),\\ &= -\frac{2k_{ij}}{\sigma^2}Ax_{ij}x_{ij}^T,\end{aligned} \tag{12}$$

where $x_{ij} := x_i - x_j$.

Consequently, one has

$$\begin{aligned}\frac{\partial \widehat{y}_i}{\partial A} &= \frac{1}{Z_i}\sum_{j\neq i}\frac{\partial k_{ij}}{\partial A}y_j - \frac{1}{Z_i}\widehat{y}_i\sum_{j\neq i}\frac{\partial k_{ij}}{\partial A},\\ &= -\frac{2}{\sigma^2 Z_i}A\sum_{j\neq i}k_{ij}y_j x_{ij}x_{ij}^T + \frac{2}{\sigma^2 Z_i}A\widehat{y}_i\sum_{j\neq i}k_{ij}x_{ij}x_{ij}^T,\\ &= \frac{2}{\sigma^2 Z_i}A\sum_{j\neq i}k_{ij}(\widehat{y}_i - y_j)x_{ij}x_{ij}^T.\end{aligned}$$

From which the conclusion stems naturally.

Derivation of the MLKRR gradient of equation (10)

In the following, the derivation of the gradient (10) and its precise definition are given. The notations defined in section 2.1 will be reused along with the subsequent new ones.

We consider two different kernel matrices: one between $X^{(\alpha)}$ and itself and one between $X^{(A)}$ and $X^{(\alpha)}$ (seen as K and Q respectively in equation (10)). We also denote by H the regularised kernel defining the coefficients α for KRR.

To simplify notations, we let the index i range over $\{1,\dots,n_A\}$, while the indices j, a, and b range over $\{1,\dots,n_\alpha\}$. We further lose the superscripts on x and y when the indices suffice to determine them, and use the same letter K for both kernels.

$$k_{i,j} := \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{d\left(x_i,x_j\right)^2}{\sigma^2}}, \qquad k_{j,j'} := \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{d\left(x_j,x_{j'}\right)^2}{\sigma^2}},$$
$$H := (K+\lambda I), \qquad \alpha := H^{-1}y^{(\alpha)}.$$

As above, we start from the following expression

$$\frac{\partial \mathcal{L}}{\partial A} = 2\sum_i(\widehat{y}_i - y_i)\frac{\partial \widehat{y}_i}{\partial A},$$

where $\frac{\partial \widehat{y}_i}{\partial A} = \sum_j \frac{\partial}{\partial A}(\alpha_j k_{ij})$. Remembering that both α and K depend on A, we therefore aim to compute $\frac{\partial k_{ij}}{\partial A}$ and $\frac{\partial \alpha_j}{\partial A}$.

Remember that the kernel derivative is already computed in (12).

$$\frac{\partial k_{ij}}{\partial A} = -\frac{2k_{ij}}{\sigma^2}Ax_{ij}x_{ij}^T,$$

where $x_{ij} := x_i - x_j$.

The second term requires treating the derivative of H^{-1}, the details of which are spared. In fact, routine computation yields that

$$\frac{\partial}{\partial A}(H^{-1})_{jj'} = \frac{2}{\sigma^2}A\sum_{a,b}(H^{-1})_{ja}(H^{-1})_{bj'}k_{ab}x_{ab}x_{ab}^T,$$

and hence that

$$\frac{\partial \alpha_j}{\partial A} = \frac{2}{\sigma^2}A\sum_{a,b}(H^{-1})_{ja}k_{ab}\alpha_b x_{ab}x_{ab}^T.$$

Combining both derivatives gives the following expression for the gradient.

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial A} &= 2\sum_{i,j}(\widehat{y}_i - y_i)\left[\frac{\partial k_{ij}}{\partial A}\alpha_j + \frac{\partial \alpha_j}{\partial A}k_{ij}\right],\\ &= -\frac{4}{\sigma^2}A\sum_{i,j}(\widehat{y}_i - y_i)\alpha_j k_{ij}x_{ij}x_{ij}^T\\ &\quad + \frac{4}{\sigma^2}A\sum_{a,b}k_{ab}\alpha_b\left[H^{-T}K^T(\widehat{y} - y^{(A)})\right]_a x_{ab}x_{ab}^T.\end{aligned} \tag{13}$$

We recall that K has two different definitions depending on its indexing. To clarify, we set $K \in \mathbb{R}^{n_\alpha \times n_\alpha}$ the kernel between $X^{(\alpha)}$ and itself; and we set $Q \in \mathbb{R}^{n_A \times n_\alpha}$ the kernel between $X^{(A)}$ and $X^{(\alpha)}$.

The lemma below allows us to turn this expression into matrix form. Indeed, note that both sums are of the form $\sum_{i,j} W_{ij}(x_i - x_j)(x_i - x_j)^T$, where x_i and x_j are lines of $X^{(A)}$ or $X^{(\alpha)}$ depending on their index.

Lemma 1. *Consider two matrices A and B with lines $\{a_i\}$, $\{b_j\}$ of matching dimensions, and let $x_{ij} = a_i - b_j$. Set further the matrix*

$$\Sigma = \sum_{i,j} W_{ij} x_{ij} x_{ij}^T, \tag{14}$$

with some coefficients W_{ij} making up a matrix W. Then

$$\Sigma = -A^T W B - B^T W^T A + A^T R A + B^T S B,$$

where R and S are both diagonal matrices with $R_{ii} = \sum_j W_{ij}$, and $S_{jj} = \sum_i W_{ij}$.

Applying the lemma to both expressions in (13) leads us to construct the matrices $W \in \mathbb{R}^{n_A \times n_\alpha}$ and $\tilde{W} \in \mathbb{R}^{n_\alpha \times n_\alpha}$ in the following way.

$$\begin{cases} W_{ij} := (\hat{y}_i - y_i)\alpha_j Q_{ij}, \\ \tilde{W}_{ab} := K_{ab}\alpha_b \left[H^{-T} Q^T (\hat{y} - y^{(A)})\right]_a. \end{cases}$$

Finally, the diagonal matrices $R, S, \tilde{R}, \tilde{S}$ given by the lemma conclude.

Proof of lemma 1. The right-hand side of (14) splits into four sums which make up each term of the result.

$$\begin{aligned} \Sigma &= \sum_{i,j} W_{ij}(a_i - b_j)(a_i - b_j)^T, \\ &= \sum_i \left(\sum_j W_{ij} \right) a_i a_i^T + \sum_j \left(\sum_i W_{ij} \right) b_j b_j^T, \\ &\quad - \sum_{i,j} W_{ij} a_i b_j^T - \sum_{i,j} W_{ij} b_j a_i^T. \end{aligned}$$

Note that the last two terms are transpose of each others so that only one has to be treated. Additionally, for a general matrix M of appropriate dimensions, one can easily verify that

$$A^T M B = \sum_{i,j} M_{ij} a_i b_j^T.$$

Taking $M = W, R$, and S yields the desired result. □

ORCID iDs

Raimon Fabregat https://orcid.org/0000-0002-7946-817X
Puck van Gerwen https://orcid.org/0000-0002-7992-5529
Clémence Corminboeuf https://orcid.org/0000-0001-7993-2879

References

[1] von Lilienfeld O A 2020 Introducing machine learning: science and technology *Mach. Learn.: Sci. Technol.* **1** 010201
[2] Pyzer-Knapp E O, Cuff J, Patterson J, Isayev O and Maskell S 2020 Welcome to the first issue of applied AI letters *Appl. AI lett.* **1** e8
[3] Huang B and von Lilienfeld O A 2021 *Ab initio* machine learning in chemical compound space *Chem. Rev.* **121** 10001–36
[4] Musil F, Grisafi A, Bartók A P, Ortner C, Csányi G and Ceriotti M 2021 Physics-inspired structural representations for molecules and materials *Chem. Rev.* **121** 9759–815
[5] Butler K T, Davies D W, Cartwright H, Isayev O and Walsh A 2018 Machine learning for molecular and materials science *Nature* **559** 547–55
[6] Unke O T, Chmiela S, Sauceda H E, Gastegger M, Poltavsky I, Schütt K T, Tkatchenko A and Müller K-R 2021 Machine learning force fields *Chem. Rev.* **121** 10142–86
[7] Aspuru-Guzik A, Lindh R and Reiher M 2018 The matter simulation (R)evolution *ACS Cent. Sci.* **4** 144–52
[8] Kitchin J R 2018 Machine learning in catalysis *Nat. Catal.* **1** 230–2
[9] Carleo G, Cirac I, Cranmer K, Daudet L, Schuld M, Tishby N, Vogt-Maranto L and Zdeborová L 2019 Machine learning and the physical sciences *Rev. Mod. Phys.* **91** 045002
[10] Ceriotti M 2019 Unsupervised machine learning in atomistic simulations, between predictions and understanding *J. Chem. Phys.* **150** 150901

[11] Glielmo A, Husic B E, Rodriguez A, Clementi C, Noé F and Laio A 2021 Unsupervised learning methods for molecular simulation data *Chem. Rev.* **121** 9722–58
[12] Cheng B, Griffiths R-R, Wengert S, Kunkel C, Stenczel T, Zhu B, Deringer V L, Bernstein N, Margraf J T, Reuter K *et al* 2020 Mapping materials and molecules *Acc. Chem. Res.* **53** 1981–91
[13] Deringer V L, Bartók A P, Bernstein N, Wilkins D M, Ceriotti M and Csányi G 2021 Gaussian process regression for materials and molecules *Chem. Rev.* **121** 10073–141
[14] Kamath A, Vargas-Hernández R A, Krems R V, Carrington T Jr and Manzhos S 2018 Neural networks vs Gaussian process regression for representing potential energy surfaces: a comparative study of fit quality and vibrational spectrum accuracy *J. Chem. Phys.* **148** 241702
[15] Faber F A, Hutchison L, Huang B, Gilmer J, Schoenholz S S, Dahl G E, Vinyals O, Kearnes S, Riley P F and von Lilienfeld O A 2017 Prediction errors of molecular machine learning models lower than hybrid DFT error *J. Chem. Theory Comput.* **13** 5255–64
[16] Klicpera J, Groß J and Günnemann S 2020 Directional message passing for molecular graphs (arXiv:2003.03123)
[17] Anderson B, Hy T S and Kondor R 2019 Cormorant: covariant molecular neural networks *Advances in Neural Information Processing Systems* vol 32 (*Vancouver, Canada*)
[18] Batzner S, Musaelian A, Sun L, Geiger M, Mailoa J P, Kornbluth M, Molinari N, Smidt T E and Kozinsky B 2022 E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials *Nat. Commun.* **13** 1–11
[19] Satorras V G, Hoogeboom E and Welling M 2021 E(n) equivariant graph neural networks *Proceedings of the 38th International Conference on Machine Learning PMLR* vol 139 ed M Meila and T Zhang pp 9323–32
[20] Van der Maaten L and Hinton G 2008 Visualizing data using t-SNE *J. Mach. Learn. Res.* **9** 2579–605
[21] Tenenbaum J B, Silva V D and Langford J C 2000 A global geometric framework for nonlinear dimensionality reduction *Science* **290** 2319–23
[22] Huang B and von Lilienfeld O A 2020 Quantum machine learning using atom-in-molecule-based fragments selected on the fly *Nat. Chem.* **12** 945–51
[23] Bartók A P, Kondor R and Csányi G 2013 On representing chemical environments *Phys. Rev.* B **87** 184115
[24] Faber F A, Christensen A S, Huang B and von Lilienfeld O A 2018 Alchemical and structural distribution based representation for universal quantum machine learning *J. Chem. Phys.* **148** 241717
[25] Christensen A S, Bratholm L A, Faber F A and von Lilienfeld O A 2020 FCHL revisited: faster and more accurate quantum machine learning *J. Chem. Phys.* **152** 044107
[26] Rupp M, Tkatchenko A, Müller K-R and von Lilienfeld O A 2012 Fast and accurate modeling of molecular atomization energies with machine learning *Phys. Rev. Lett.* **108** 058301
[27] Huang B and von Lilienfeld O A 2016 Communication: understanding molecular representations in machine learning: the role of uniqueness and target similarity *J. Chem. Phys.* **145** 161102
[28] Von Lilienfeld O A 2018 Quantum machine learning in chemical compound space *Angew. Chem., Int. Ed.* **57** 4164–9
[29] Li Z, Kermode J R and De Vita A 2015 Molecular dynamics with on-the-fly machine learning of quantum-mechanical forces *Phys. Rev. Lett.* **114** 096405
[30] Chmiela S, Tkatchenko A, Sauceda H E, Poltavsky I, Schütt K T and Müller K-R 2017 Machine learning of accurate energy-conserving molecular force fields *Sci. Adv.* **3** e1603015
[31] Chmiela S, Sauceda H E, Müller K-R and Tkatchenko A 2018 Towards exact molecular dynamics simulations with machine-learned force fields *Nat. Commun.* **9** 3887
[32] Bereau T, Andrienko D and von Lilienfeld O A 2015 Transferable atomic multipole machine learning models for small organic molecules *J. Chem. Theory Comput.* **11** 3225–33
[33] Grisafi A, Wilkins D M, Csányi G and Ceriotti M 2018 Symmetry-adapted machine learning for tensorial properties of atomistic systems *Phys. Rev. Lett.* **120** 036002
[34] Wilkins D M, Grisafi A, Yang Y, Lao K U, DiStasio R A and Ceriotti M 2019 Accurate molecular polarizabilities with coupled cluster theory and machine learning *Proc. Natl Acad. Sci. USA* **116** 3401–6
[35] Grisafi A, Fabrizio A, Meyer B, Wilkins D M, Corminboeuf C and Ceriotti M 2018 Transferable machine-learning model of the electron density *ACS Cent. Sci.* **5** 57–64
[36] Bartók A P, Payne M C, Kondor R and Csányi G 2010 Gaussian approximation potentials: the accuracy of quantum mechanics, without the electrons *Phys. Rev. Lett.* **104** 136403
[37] Fabrizio A, Grisafi A, Meyer B, Ceriotti M and Corminboeuf C 2019 Electron density learning of non-covalent systems *Chem. Sci.* **10** 9424–32
[38] Westermayr J, Gastegger M, Menger M F S J, Mai S, González L and Marquetand P 2019 Machine learning enables long time scale molecular photodynamics simulations *Chem. Sci.* **10** 8100–7
[39] Gallarati S, Fabregat R, Laplaza R, Bhattacharjee S, Wodrich M D and Corminboeuf C 2021 Reaction-based machine learning representations for predicting the enantioselectivity of organocatalysts *Chem. Sci.* **12** 6879–89
[40] Mahoney M W and Drineas P 2009 CUR matrix decompositions for improved data analysis *Proc. Natl Acad. Sci. USA* **106** 697–702
[41] Kuhn M and Johnson K 2013 *Applied Predictive Modelling* vol 26 (New York: Springer) (https://doi.org/10.1007/978-1-4614-6849-3)
[42] Kulis B *et al* 2013 Metric learning: a survey *Found. Trends Mach. Learn.* **5** 287–364
[43] Yang L and Jin R 2006 *Distance Metric Learning: A Comprehensive Survey* (available at: www.cs.cmu.edu/~./liuy/frame_survey_v2.pdf) (Accessed 12 September 2022)
[44] Coupry D E and Pogány P 2022 Application of deep metric learning to molecular graph similarity *J. Cheminformatics* **14** 1–12
[45] Weinberger K Q and Tesauro G 2007 Metric learning for kernel regression *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics Proc. Mach. Learn. Res.* (*San Juan, Puerto Rico*) vol 2 ed Meila and Shen Xiatong pp 612–9
[46] Koch G, Zemel R, Salakhutdinov R *et al* 2015 Siamese neural networks for one-shot image recognition *Proceedings of the 32nd International Conference on Machine Learning JMLR: W&CP* vol 37 (*Lille, France*)
[47] Hoffer E and Ailon N 2015 Deep metric learning using triplet network *International Workshop on Similarity-Based Pattern Recognition SIMBAD* vol 9370 ed Feragen A, M Pelillo and M Loog (Springer) pp 84–92
[48] Rasmussen C E and Williams C K I 2006 *Gaussian Processes for Machine Learning* (Adaptive computation and machine learning) 2 edn (Cambridge, MA: The MIT Press)
[49] Chopra S, Hadsell R and LeCun Y 2005 Learning a similarity metric discriminatively, with application to face verification *IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR* vol 1 (IEEE) pp 539–46

[50] Oh Song H, Xiang Y, Jegelka S and Savarese S 2016 Deep metric learning via lifted structured feature embedding *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition CVPR* (*Las Vegas, Nevada*) pp 4004–12
[51] Khosla P, Teterwak P, Wang C, Sarna A, Tian Y, Isola P, Maschinot A, Liu C and Krishnan D 2020 Supervised contrastive learning *Advances in Neural Information Processing Systems NeurIPS* vol 33 ed H Larochelle, M Ranzato, R Hadsell, M F Balcan and H Lin pp 18661–73
[52] Stärk H, Beaini D, Corso G, Tossou P, Dallago C, Günnemann S and Liò P 2022 3D infomax improves GNNs for molecular property prediction *Proceedings of the 39th International Conference on Machine Learning* vol 162 (*Baltimore, Maryland*) pp 20479–502
[53] Larochelle H, Erhan D, Courville A, Bergstra J and Bengio Y 2007 An empirical evaluation of deep architectures on problems with many factors of variation *Proc. 24th Int. Conf. on Machine Learning* pp 473–80
[54] Nasser M, Salim N, Hamza H, Saeed F and Rabiu I 2020 Improved deep learning based method for molecular similarity searching using stack of deep belief networks *Molecules* **26** 128
[55] Zhu P, Qi R, Hu Q, Wang Q, Zhang C and Yang L 2018 Beyond similar and dissimilar relations: a kernel regression formulation for metric learning *Proceedings of the 27th International Joint Conference on Artificial Intelligence IJCAI* (*Stockholm, Sweden*) pp 3242–8
[56] Ramakrishnan R, Dral P O, Rupp M and von Lilienfeld O A 2014 Quantum chemistry structures and properties of 134 kilo molecules *Sci. Data* **1** 140022
[57] Mahalanobis P C 1936 On the generalized distance in statistics *Proc. Indian National Sci. Acad.* **2** 49–55
[58] Welling M 2019 *Kernel ridge regression* (available at: https://web2.qatar.cmu.edu/~gdicaro/10315-Fall19/additional/welling-notes-on-kernel-ridge.pdf) (Accessed 12 September 2022)
[59] De Vazelhes W, Carey C, Tang Y, Vauquier N and Bellet A 2020 Metric-learn: metric learning algorithms in Python *J. Mach. Learn. Res.* **21** 1–6
[60] Christensen A S, Faber F A, Huang B, Bratholm L A, Tkatchenko A, Müller K R and von Lilienfeld O A 2017 QML: a Python toolkit for quantum machine learning (available at: https://github.com/qmlcode/qml)
[61] Virtanen P *et al* SciPy 10 Contributors 2020 SciPy 1.0: fundamental algorithms for scientific computing in Python *Nat. Methods* **17** 261–72
[62] Boyd S, Boyd S P and Vandenberghe L 2004 *Convex Optimization* (Cambridge: Cambridge University Press)
[63] Willatt M J, Musil F and Ceriotti M 2018 Feature optimization for atomistic machine learning yields a data-driven construction of the periodic table of the elements *Phys. Chem. Chem. Phys.* **20** 29661–8
[64] Brazdil P, Carrier C G, Soares C and Vilalta R 2009 *Metalearning: Applications to Data Mining Cognitive Technologies* ed D M Gabbay and J Siekmann (Berlin: Springer Science & Business Media) (https://doi.org/10.1007/978-3-540-73263-1)