Representations of molecules and materials for interpolation of quantum-mechanical simulations via machine learning

Marcel F. Langer [*] Alex Goeßmann [*†] Matthias Rupp [*‡]

[*] Theory Department, Fritz Haber Institute of the Max Planck Society, Berlin, Germany; [†] Institute of Mathematics, Technical University Berlin, Germany; [‡] Citrine Informatics, Redwood City, CA, USA

Abstract

Computational study of molecules and materials from first principles is a cornerstone of physics, chemistry and materials science, but limited by the cost of accurate and precise simulations. In settings involving many simulations, machine learning can reduce these costs, sometimes by orders of magnitude, by interpolating between reference simulations. This requires *representations* that describe any molecule or material and support interpolation. We review, discuss and benchmark state-of-the-art representations and relations between them, including smooth overlap of atomic positions, many-body tensor representation, and symmetry functions. For this, we use a unified mathematical framework based on many-body functions, group averaging and tensor products, and compare energy predictions for organic molecules, binary alloys and Al-Ga-In sesquioxides in numerical experiments controlled for data distribution, regression method and hyper-parameter optimization.

Contents

Glossary

Acronym	Meaning
BoB	bag of bonds
BS	bispectrum
CM	Coulomb matrix
FCHL	Faber-Christensen-Huang-von Lilienfeld
HDAD	histograms of distances, angles and dihedral angles
MBTR	many-body tensor representation
MTP	moment tensor potential
SF	symmetry function
SOAP	smooth overlap of atomic positions
GPR	Gaussian process regression
HP	hyperparameter (free parameter)
KRR	kernel ridge regression
ML	machine learning
QM	quantum mechanics
QM/ML	ML model for accurate prediction of QM data
RMSE	root mean squared error
system	poly-atomic system, e.g., a molecule or crystal
SI	supplementary information

1 Introduction

Quantitative understanding of atomic-scale phenomena is central for scientific insights and technological innovations in many areas of physics, chemistry and materials science. Such understanding is obtained by solving the equations that govern quantum mechanics (QM), such as Schrödinger's or Dirac's equations, which allow to calculate properties of molecules, clusters, bulk crystals, surfaces and other poly-atomic systems. For this, numerical simulations of the electronic structure of matter are used, with tremendous success in explaining observations and quantitative predictions.

The high computational cost of these ab initio methods, (SI 1) however, often only allows to investigate from tens of thousands of small systems with a few dozen atoms to a few large systems with thousands of atoms, in particular for periodic structures. In contrast, the number of possible molecules and materials grows combinatorially with the number of atoms: 13 or fewer C, N, O, S, Cl atoms can form a billion possible molecules,[1] and for 5-component alloys there are more than a billion possible compositions when choosing from 30 elements. (SI 2) This limits systematic computational study and exploration of molecular and materials spaces. Similar considerations hold for ab initio dynamics simulations, which are typically restricted to systems with a few hundred atoms and sub-nanosecond timescales.

Such situations require many simulations of systems that are correlated in structure, implying a high degree of redundancy. Machine learning[2,3] (ML) can exploit this redundancy to accurately interpolate between reference simulations[4–6] (Figure 1). Most ab initio simulations can thus be replaced by ML predictions based on a small set of reference simulations. Effectively, the problem of repeatedly solving a QM equation for many related systems is mapped onto a regression problem. This approach has been demonstrated in benchmark settings,[4,7,8] with reported speed-ups anywhere between zero to six orders of magnitude.[9–11] It is currently regarded as a highly promising avenue towards extending the scope of ab initio methods.

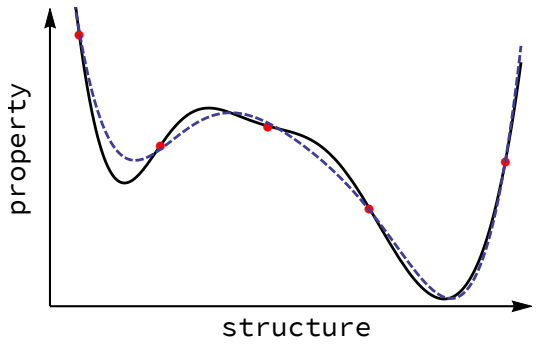


Figure 1: *Sketch illustrating accurate interpolation of quantum-mechanical simulations by machine learning.* The horizontal axis represents chemical or materials space, the vertical axis the predicted property. Instead of conducting many computationally expensive ab initio simulations (solid line), machine learning (dashed line) interpolates between a few reference simulations (dots).

The main aspect of ML models for accurate interpolation of QM simulations (QM/ML models) after data quality (SI 3) is the definition of a suitable *representation* for atomistic systems. It defines how these relate to each other for the purpose of regression, and is the subject of this review.

Scope and structure

QM/ML models require a space in which interpolation takes place. Such spaces can be defined explicitly, often as vector spaces, or implicitly, for example, via the kernel function in kernel-based machine learning.[12,13] *This work reviews and compares Hilbert-space representations of finite and periodic poly-atomic systems for accurate interpolation of QM observables via ML*, with particular focus on "exact" representations that satisfy the requirements in Section 3.

This excludes coarse-graining features such as descriptors or fingerprints used in cheminformatics and materials informatics to interpolate between experimental outcomes,[14] and, deep neural networks, which can learn (internal) representations, but require considerably more data for this. The latter can be avoided by designing the network architecture to incorporate physical constraints.[15–18] (SI 4)

General characteristics and requirements of representations are discussed in Sections 2 and 3. Section 4 describes a unified mathematical framework for representations. Specific representations are then delineated (Sections 5 and 6), qualitatively compared (Section 7) and empirically benchmarked (Section 8). We conclude with an outlook on open problems and possible directions for future research.

Related work

Studies of QM/ML models often compare performance estimates to those of other models reported in the literature. While such comparisons have value, they entertain considerable uncertainty due to differences in data, learning algorithms, including choice of hyperparameters (HPs, free parameters), sampling, validation procedures and reported quantities. Accurate reliable performance estimates require a systematic comparison that controls for above factors, which we perform in this work.

Several recent studies systematically measured and compared prediction errors of exact representations (Table 1). We distinguish between automated optimization of numerical HPs of representations, for example, the width of a normal distribution; structural HPs of representations, for example, choice of basis functions; and, HPs of the regression method, for example, regularization strength.

Faber et al.[19] compare combinations of representations and regression methods for atomization energies of organic molecules (`qm9` dataset, see Section 8). Only some of the tested representations are exact; their HPs are not optimized.

Himanen et al.[20] investigate the representations in Section 5, also using kernel regression, to predict ionic charges of molecules from the `qm9` dataset, as well as formation energies in a custom dataset of inorganic crystals obtained from the Open Quantum Materials Database. They optimize numerical HPs of representations and regression method, but not structural ones.

Table 1: *Related work.* Molecules = molecular datasets, Materials = materials datasets, Other pts. = include properties other than energy or its derivatives, HP num. = optimize numerical representation hyperparameters algorithmically, HP struct. = optimize structural representation hyperparameters algorithmically, HP regr. = optimize hyperparameters of regression method algorithmically.

	Reference						
	19	20	21	22	23	24	here
Molecules	✓	✓	×	✓	×	✓	✓
Materials	×	✓	✓	×	✓	×	✓
Other pts.	✓	✓	×	×	×	✓	×
HP num.	×	✓	✓	×	×	✓	✓
HP struct.	×	×	×	×	×	×	✓
HP regr.	✓	✓	✓	✓	✓	✓	✓
Timings	×	×	✓	✓	×	×	✓

Zuo et al.[21] focus on dynamics simulations, and therefore include forces and stresses in training and evaluation. They also evaluate predictions of derived physical quantities, such as elastic constants or equation-of-state curves. Different combinations of representation, regression method, and HP tuning are evaluated on a dataset of elemental solids. Timings are discussed.

Schmitz et al.[22] compare regression methods for potential energy surfaces of 15 small organic molecules, using non-redundant internal coordinates as features. HPs of the representation are not optimized.

Nyshadham et al.[23] compare selected combinations of representations and regression methods on binary alloys (`ba10` dataset, see Section 8). HPs of the representations are not optimized.

Stuke et al.[24] evaluate prediction of molecular orbital energies with kernel regression on three datasets: organic molecules (`qm9` dataset, see Section 8), amino acids and dipeptides, as well as opto-electronically active molecules. Numerical HPs of representations and regression method are optimised via local grid search.

2 Role and types of representations

An N-atom system formally has $3N-6$ degrees of freedom. Covering those with M samples per dimension requires M^{3N-6} reference calculations, which is infeasible but for the smallest systems. How then is it possible to learn high-dimensional energy surfaces?

Part of the answer is that learning the whole energy surface is not necessary as configurations high in energy become exponentially unlikely—it is sufficient to learn low-energy regions. Another part of the answer is that the formal dimensionality of the regression space is less important than *how the data are distributed* in this space. (SI 5) Exact representations can have thousands of dimensions, but these are highly correlated, and their effective dimensionality[25] is much lower. The role of representations is therefore to map atomistic systems to a space amenable to regression. The spaces they define, together with the distribution of the data, determine the efficiency of learning.

We classify representations (Table 2) according to whether they represent parts of an atomistic system, such as atoms in their environment[26] (*local*) or the whole system (*global*), and, whether represented systems are *finite*, such as molecules and clusters, or *periodic*, such as bulk crystals and surfaces.

Table 2: *Types of representations.* We distinguish between local (atoms in their environment) and global (holistic, whole system) representations, as well as between representations for finite (molecules, clusters) and periodic systems (bulk crystals, surfaces). Local representations have finite support, and thus do not need to distinguish between finite and periodic systems. See Glossary for abbreviations.

	finite	periodic
local	BS, FCHL, MTP, SF, SOAP	
global	CM, BoB, HDAD, MBTR	MBTR

Local representations are directly suitable for local properties, such as forces, nuclear magnetic resonance shifts, or core level excitations,[27] which depend only on a finite-size environment of an atom. Extensive global properties (SI 6) such as energies can be modeled with local representations via additive approximations, summing over atomic contributions (SI 7). Since local representations require only finite support, it does not matter whether the surrounding system is finite or periodic. Global representations are suited for properties of the whole system, such as energy, band gap, or the polarizability tensor. Since periodic systems are infinitely large, global representations usually need to be designed for or adapted to these. Trade-offs between local and global representations are discussed in Section 7.

Historically, interpolation has been used to reduce the effort of numerical solutions to quantum problems from the beginning. Early works employing ML techniques such as Tikhonov regularization and reproducing kernel Hilbert spaces in the late 1980s and throughout the 1990s were limited to small systems.[28–31] Representations for high-dimensional systems appeared a decade later,[7,8,32] underwent rapid development and constitute an active area of research today.[26,33–35] Table 3 presents an overview.

3 Requirements

The figure of merit of ML models for fast accurate interpolation of ab initio properties is sample efficiency: The number of reference simulations required to reach a target accuracy. These demands—speed, accuracy, and sample efficiency—give rise to specific requirements, some of which depend on the predicted property. Imposing physical constraints on representations improves their sample efficiency by removing the need to learn these constraints from the training data.

(i) Invariance to transformations that preserve the predicted property, including (a) changes in atom indexing (input order, permutation of like atoms), and often (b) translations,

Table 3: *Overview and selected references.* For each representation (Repr.), year of publication (Year), original reference (Orig.), references for further methodological development (Dev.) and availability of implementations (Avail.) are shown. See Glossary for abbreviations.

		References		
Year	Repr.	Orig.	Dev.	Avail.
2007	SF	7	36–40	41
2010	BS	8	42	43
2012	CM	4	27, 44–46	47
2013	SOAP	26	6, 8, 48–50	51
2015	BoB	52	—	53
2016	MTP	33	54–57	58
2017	MBTR	34	23	47
2017	HDAD	19	—	—
2018	FCHL	35	59	53

(c) rotations, and (d) reflections. Predicting tensorial properties requires (e) *covariance* with rotations.[6,60,61]

Dependencies on a global frame of reference can affect variance requirements, for example through the presence of a non-isotropic external field.

(ii) Uniqueness, that is, variance against all transformations that change the predicted property: The map from atomistic systems to representations should be injective modulo the property.

Systems with identical representations that differ in property introduce errors.[44,62] As the ML model can not distinguish them, it predicts the same value for both, resulting in at least one erroneous prediction. Uniqueness is necessary and sufficient for reconstruction, up to invariant transformations, of an atomistic system from its representation.

(iii) (a) *Continuity,* and ideally (b) *differentiability*, with respect to atomic coordinates.

Discontinuities work against the regularity assumption of ML models, which try to find the least complex function compatible with the training data. Intuitively, continuous functions require less training data than functions with jumps. Differentiable representations enable differentiable ML models. Reference gradients, if available, can then constrain the interpolation function further ("force matching"), improving sample efficiency.[63,64]

(iv) Computational efficiency, relative to the reference simulations.

For substantial advantage over simulations alone (without ML), overall computational costs must be reduced, ideally by one or more orders of magnitude to justify the effort. Costs are usually dominated by the difference between running reference simulations and computing representations. (SI 8) Results of computationally sufficiently cheaper simulations at a lower level of theory can therefore be used for representations to predict properties at a higher level of theory ("Δ-learning").[46,65]

(v) Structure of representations and resulting distribution of the data should be suitable for regression. (SI 5 and 9) Useful properties include constant size.[36,66]

Exact representations often have Hilbert space structure, featuring constant size, an inner product, completeness, projections and other advantages. In the formal space defined by the representation, the structure of the subspace spanned by the data is important as well. This requirement is currently less well understood than *(i)–(iv)* and evaluated mostly empirically (see Section 8).

(vi) Generality, in the sense of being able to encode any atomistic system.

While current representations handle finite and periodic systems, less work was done on charged systems, excited states, continuous spin systems, isotopes, and systems subjected to external fields.

Albeit hard to quantify, we feel that *simplicity*, both conceptually and in terms of implementation, is a desirable quality of representations.

Above requirements preclude direct use of Cartesian coordinates, which violate requirement *(i)*, and internal coordinates, which satisfy (i.b)–(i.d) but are still system-specific, violating *(v)* and possibly (i.a) if not defined uniquely. Early representations such as the Coulomb matrix (Section 6) suffered from either coarse-graining, violating *(ii)*, or discontinuities, violating (iii.a). In practice, representations do not satisfy all requirements exactly (Section 7).

4 A unified framework

Based on recent work[6,67] we describe concepts and notation towards a unified treatment of representations. For this, we successively build up Hilbert spaces of atoms, k-atom tuples, local environments and global structures, using group averaging and tensor products to ensure invariants while retaining desired information.

Representing atoms, environments and systems

Information about a single atom, such as position and proton number, is represented as an abstract ket $|\alpha\rangle$ in a Hilbert space $\mathcal{H}_\alpha$. Relations between k atoms, where order can matter, are encoded as k-body functions $g_k : \mathcal{H}_\alpha^{\times k} \to \mathcal{H}_g$. These functions can be purely geometric, such as distances or angles, but could also be of (al)chemical or mixed nature. (SI 10) Tuples of atoms and associated many-body properties are thus encoded as elementary tensors of a space $\mathcal{H} \equiv \mathcal{H}_\alpha^{\otimes k} \otimes \mathcal{H}_g$,

$$|\mathcal{A}_{\alpha_1 \ldots \alpha_k}\rangle \equiv |\alpha_1\rangle \otimes \ldots \otimes |\alpha_k\rangle \otimes g_k(|\alpha_1\rangle, \ldots, |\alpha_k\rangle).$$

A local environment of an atom $|\alpha\rangle$ is represented via the relations to its $k-1$ neighbours by keeping $|\alpha\rangle$ fixed:

$$|\mathcal{A}_\alpha\rangle \equiv \sum_{\alpha_1,\ldots,\alpha_{k-1}} |\mathcal{A}_{\alpha,\alpha_1,\ldots,\alpha_{k-1}}\rangle .$$

Weighting functions are used to reduce influence of atoms far from $|\alpha\rangle$; these are included in g_k. Atomistic systems as a whole are represented by summing over the local environments of all its atoms:

$$|\mathcal{A}\rangle = \sum_{\alpha_i} |\mathcal{A}_{\alpha_i}\rangle = \sum_{\alpha_1,\ldots,\alpha_k} |\mathcal{A}_{\alpha_1,\ldots,\alpha_k}\rangle\,.$$

For periodic systems, this sum diverges, which requires either exploiting periodicity, for example, by working in reciprocal space, or, employing strong weighting functions and keeping one index constrained to the unit cell.[34]

Symmetries, tensor products and projections

Symmetry constraints (Section 3) have been incorporated in two ways: Via invariant many-body functions g_k, such as distances or angles, and by explicit symmetrization via group averaging.[67] For the latter, a tensor $|T\rangle$ is transformed by integrating over a symmetry group $\mathcal{S}$ with right-invariant Haar measure dS,

$$|T\rangle_{\mathcal{S}} \equiv \int_{\mathcal{S}} S\,|T\rangle\,dS,$$

where symmetry transformations $S \in \mathcal{S}$ act separately on each subspace $\mathcal{H}$ or parts thereof. For example, for rotational invariance only the atomic positions in $\mathcal{H}_\alpha$ change.

Sometimes group averaging can integrate out desired information encoded in $|T\rangle$. To counter this, one can perform tensor products of $|T\rangle$ with itself, effectively replacing $\mathcal{H}$ by $\mathcal{H}^{\otimes\nu}$. Together, this results in a generalized transform

$$|T^\nu\rangle_{\mathcal{S}} \equiv \int_{\mathcal{S}} (S\,|T\rangle)^{\otimes\nu} dS.$$

For distances it is sometimes practical to retain only part of the information. This can be achieved by projecting onto orthogonal elements $\{|h_l\rangle\}_{l=1}^m$ in $\mathcal{H}$ via an associated projection operator $\mathcal{P} = \sum_l |h_l\rangle\langle h_l|$. Inner products and induced distances between representations are then given by

$$\langle\mathcal{A}|\mathcal{P}|\mathcal{A}'\rangle \text{ and } d_{\mathcal{P}}(|\mathcal{A}\rangle, |\mathcal{A}'\rangle) = ||\mathcal{P}\,|\mathcal{A}\rangle - \mathcal{P}\,|\mathcal{A}'\rangle\,||_{\mathcal{H}}. \quad (1)$$

5 Representations

We discuss selected representations that fulfill the requirements in Section 3.

Symmetry functions

Symmetry functions[7,36] (SFs) describe k-body relations between a central atom and the atoms in a local environment around it. (SI 11) They are typically based on distances (*radial* SFs) and angles (*angular* SFs). Each SF encodes a local feature of an atomic environment, for example the number of H atoms at a given distance from a central C atom.

For each SF and k-tuple of chemical elements, contributions are summed. Sufficient resolution is achieved by varying the HPs of a SF. For continuity (and differentiability), a *cut-off* function ensures that SFs decay to zero at the cut-off radius. Two examples of SFs from Reference 36 (see Table 3 and SI 22 for further references and SFs) are

$$G_i^2 = \sum_j \exp\left(-\eta(d_{ij}-\mu)^2\right) f_c(d_{ij})$$
$$G_i^4 = 2^{1-\zeta} \sum_{j,k\neq i} (1+\lambda\cos\theta_{ijk})^\zeta \cdot$$
$$\exp\left(-\eta(d_{ij}^2+d_{ik}^2+d_{jk}^2)\right) f_c(d_{ij})\, f_c(d_{ik})\, f_c(d_{jk})$$

where η, μ, ζ are HPs, d_{ij} is distance and θ_{ijk} is angle between atoms i, j, k, and f_c is a cut-off function. Figure 2 illustrates the radial SFs in Section 5. Variants of SFs include partial radial distribution functions[68] and reparametrizations for improved scaling with number of chemical species[38–40] .

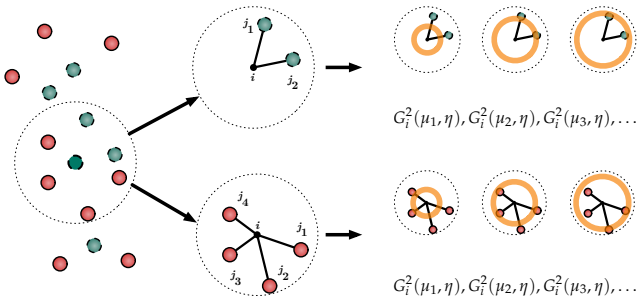


Figure 2: *Symmetry functions.* Shown are radial functions $G_i^2(\mu,\eta)$ (Section 5) for increasing values of μ. The local environment of a central atom is described by summing contributions from neighboring atoms separately by element.

Many-body tensor representation

The global many-body tensor representation[34] (MBTR) consists of broadened distributions of k-body terms, arranged by element combination. For a given k-body function and k-tuple of elements, all corresponding terms (for example, all distances between C and H atoms) are computed, broadened and summed up (Figure 3). This results in a collection of distributions describing the geometric features of an atomistic system:

$$f_k(x, z_1, \ldots, z_k) = \sum_{i_1,\ldots,i_k} w_k\, \mathcal{N}(x|g_k) \prod_{j=1}^{k} \delta_{z_j, Z_{i_j}}, \quad (2)$$

where w_k is a weighting function that reduces influence of tuples with atoms far from each other, and g_k is a k-body function; both w_k and g_k depend on atoms $i_1, \ldots, i_k$. $\mathcal{N}(x|\mu)$ denotes a normal distribution with mean μ, evaluated at x. The product of Kronecker δ-functions restricts to the given element combination $z_1, \ldots, z_k$.

Periodic systems can be treated by using strong weighting functions and constraining one index to the unit cell. In practice, Equation 2 is discretized, approximating overlap integrals between two MBTR representations via inner products of histograms. HPs include choice of w_k, g_k, and variance of normal distributions.

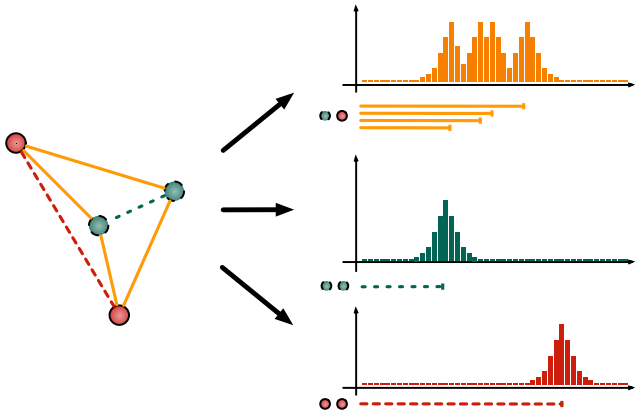


Figure 3: *Many-body tensor representation.* Shown are broadened distances (no weighting) arranged by element combination

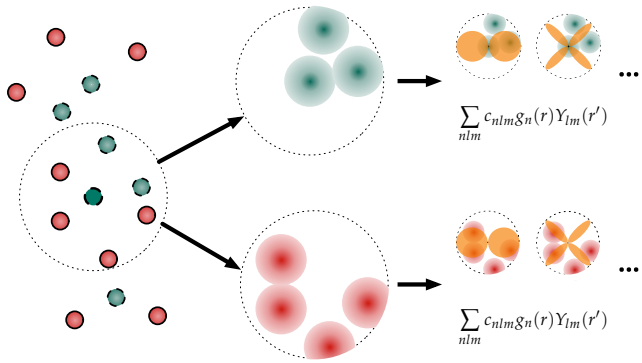


Figure 4: *Smooth overlap of atomic positions.* The local density around a central atom is modeled by atom-centered normal distributions and expanded into radial and spherical harmonics basis functions.

Smooth overlap of atomic positions

Smooth overlap of atomic positions[26] (SOAP) representations expand the local neighbourhood density around a central atom, approximated by normal distributions located at atom positions, in terms of orthogonal radial and spherical harmonics basis functions (Figure 4):

$$\rho(\boldsymbol{r}) = \sum_{n,l,m} c_{n,l,m}\, g_n(\boldsymbol{r})\, Y_{l,m}(\boldsymbol{r}), \tag{3}$$

where $c_{n,l,m}$ are expansion coefficients, g_n are radial and $Y_{l,m}$ are (angular) spherical harmonics basis functions. From the coefficients rotationally invariant quantities can be constructed, such as the power spectrum

$$p_{n,n',\ell} = \sum_{m} c_{n,\ell,m} c^{*}_{n',\ell,m}, \tag{4}$$

which is equivalent to a radial and angular distribution function,[50] and therefore captures up to three-body interactions. HPs are the maximal number of radial and angular basis functions, the broadening width, and the cut-off radius.

An alternative to the power spectrum is the *bispectrum*[8] (BS), a set of invariants that couples multiple angular momentum and radial channels. The BS has been extended to also include quadratic terms.[42]

6 Other representations

Many other representations were proposed.

The Coulomb matrix[4] (CM) globally describes an atomistic system via inverse distances, but does not contain higher-order terms. It is fast to compute, easy to implement, and in the commonly used sorted version (footnote reference 25 in Reference 4) allows reconstruction of the atomistic system via a least-squares problem. However, its direct use of atomic numbers to encode elements is problematic, and it suffers from discontinuities in the sorted version, or, from information loss in the diagonalized version as its eigenspectrum is not unique.[44,45]

The bag-of-bonds[52] (BoB) representation uses the same inverse distance terms as the CM, but arranges them by element pair instead of by atom pair. The "BA-representation"[69] extends this to higher-order interactions by using bags of dressed atoms, distances, angles and torsions. Other work[70] employs higher powers of inverse distances and separation by element combinations.

Histograms of distances, angles, and dihedral angles (HDAD)[19] are (coarsely binned) histograms of geometric features organised by element combination. This global representation is similar to MBTR, but typically uses fewer (15–25) bins, without broadening or explicit weighting.

The Faber-Christensen-Huang-von Lilienfeld representation (FCHL)[35,59] describes local atomic environments using functions of distances and angles, structurally similar to SFs. It includes an 'alchemical' distance between chemical elements based on their row and column in the periodic table.

Moment-tensor potentials[33] (MTP) describe local atomic environments in an efficiently computable basis of rotationally and permutationally invariant polynomials.

7 Analysis

We discuss relationships between specific representations, local and global ones, and to which degree they satisfy the requirements in Section 3.

Relationships between representations

All representations in Section 5 are related through the concepts of Section 4, but some share more specific connections.

MBTR and an evenly-spaced grid of SFs can both be seen as histograms of distances, angles, or higher-order terms. From this, a local variant of MBTR could be constructed by restricting summation to atomic environments,[71] and global SFs by summing over the whole system. A difference is that

MBTR explicitly broadens k-body terms, whereas SFs implicitly broaden them via the exponential functions in Section 5. In the original formulation of SFs, separate regression models are trained for each chemically distinct central atom, whereas for MBTR, each (unique) tuple of k elements is represented in separate tensor components. Both approaches correspond to insertion of Kronecker δ functions on element types in Equation 1.

SFs and MBTR use invariant k-body functions, whereas SOAP explicitly constructs (rotationally) invariant quantities (Equation 4) from variant ones (Equation 3) via symmetry integration.

Requirements

The representations in Section 5 fulfill some of the requirements in Section 3 only in the limit, that is, absent practical constraints such as truncation of infinite sums, short cut-off radii and restriction to low-order interaction terms. The degree to which these requirements are fulfilled often depends on a HP, such as when an infinite expansion is truncated, the length of a cut-off radius, or highest interaction order k used. Effects can be antagonistic; for example, in Equation 3 both *(ii)* uniqueness and *(iv)* computational effort increase with n, l, m. In addition, not all invariances of a property might be known, or require additional effort to model, for example, symmetries.[60]

Mathematical proof or systematic empirical verification that a representation satisfies a requirement or related property are sometimes provided: For MTP, Shapeev[33] shows that the employed basis can represent any permutationally and rotationally invariant polynomial. For SOAP, Bartók et al.[26] perform systematic reconstruction experiments to demonstrate uniqueness and its dependence on parametrization. While *(ii)* uniqueness guarantees that reconstruction is possible in principle, accuracy and complexity of this task vary with representation and parametrization. For example, reconstruction is a simple least squares problem for the CM, but is more involved for local representations.

Global versus local representations

Local representations can be used to model global properties by assuming that these can be decomposed into atomic contributions. In terms of prediction errors, this tends to work well for energies. (SI 6) Learning with atomic contributions adds technical complexity to the regression model, and is equivalent to pairwise-sum kernels on whole systems, (SI 7) with favorable computational scaling for large systems (see SI 8 and 27 and Table 4). Other approaches to create global kernels from local ones exist.[48]

Conversely, using global representations for local properties can require modifying the representation to incorporate locality and directionality of the property.[20,27] A general recipe to construct local representations from global ones is to use a central "ghost atom" (for example, of charge $Z = 0$; its position does not need to coincide with an actual atom), and require interactions to include it, starting from $k = 2$.[71]

8 Empirical comparison

We benchmark prediction errors of the representations from Section 5 on three benchmark datasets. In this, our focus is exclusively on the representations. We therefore control for other factors, in particular for data distribution, regression method and HP optimization.

Data

The `qm9` dataset is a consensus benchmarking dataset of 133 885 organic molecules composed of H, C, N, O, F with up to 9 non-H atoms.[72,73] (SI 12) Ground state geometries and energies are given at DFT/B3LYP/6-31G(2df,p) level of theory. We predict U_0, the energy of atomization at 0 K.

The `ba10` dataset[23,72] (SI 13) contains 10 binary alloys: AgCu, AlFe, AlMg, AlNi, AlTi, CoNi, CuFe, CuNi, FeV, NbNi. For each alloy system, all structures with up to 8 atoms are given for face-centered cubic (FCC), body-centered cubic (BCC) and hexagonal close-packed (HCP) crystal types, 15 950 structures in total. Formation energies of unrelaxed structures are provided at the DFT/PBE level of theory.

The `nmd18` challenge[74] dataset[75] (SI 14) contains 3 000 ternary $(Al_x\text{-}Ga_y\text{-}In_z)_2O_3$ oxides, $x + y + z = 1$, of potential interest as transparent conducting oxides. Formation and band-gap energies of relaxed structures are provided at the DFT/PBE level of theory. The dataset contains both relaxed (`nmd18r`, used here) and approximate (`nmd18u`, see SI 15) structures as input. In the challenge, energies of relaxed structures were predicted from approximate structures.

Together, these datasets cover finite and periodic systems, organic and inorganic chemistry, ground state and off-equilibrium structures. See SI 12 to 15 for details.

Method

We estimate prediction errors as a function of training set size ("learning curves"). (SI 16 and 17) To ensure that subsets are representative, we control for distribution of elemental composition, size and energy. (SI 18) This reduces variance of performance estimates and ensures validity of the independent-and-identically-distributed data assumption inherent in ML. All predictions are on data never seen during training.

We use kernel ridge regression[76] (KRR; predictions are equivalent to those of Gaussian process regression,[77] GPR) as ML model. (SI 19) KRR is a widely-used non-parametric non-linear regression method. In this work, training is exclusively on energies; in particular, derivatives are not used. All HPs, including numerical ones (e.g., a weight in a weighting function) and structural ones (e.g., which weighting function to use), are optimized using a consistent and fully automatic scheme based on sequential model-based optimization with tree-structured Parzen estimators.[78,79] (SI 20) This setup ensures that all representations are treated on equal footing. See SI 21 to 24 for details on optimized HPs.

Results

Figure 5 presents learning curves for SF, MBTR and SOAP on datasets `qm9`, `ba10` and `nmd18r` (see SI 25 for tabulated values). For each dataset, representation and training set size, a KRR model is trained and its predictions evaluated on a separate hold-out validation set of size 10 k (`qm9`), 1 k (`ba10`) and 0.6 k (`nmd18r`). This is repeated 10 times to estimate the variance in these performance estimates.

Boxes, whiskers, horizontal bars and crosses show interquartile ranges, minimum / maximum value, median, and mean, respectively, of the root mean squared error (RMSE) of hold-out-set predictions for each repetition. We show RMSE as it is the loss minimized by least-squares regression such as KRR, and thus a "natural" choice. For other loss functions, see SI 26. From statistical learning theory, RMSE decays as a negative power of training set size (a reason why learning curves are preferably shown as log-log plots).[80–82] Lines show corresponding fits of mean RMSE weighted by standard deviation for each training set size.

Figure 6 reveals dependencies between the time to compute representations in a training set (horizontal axis) and RMSE (vertical axis). When comparing observations in two dimensions, here time t and error e, there is no unique ordering $<$, and we resort to the usual notion of dominance: Let $\boldsymbol{x}, \boldsymbol{x'} \in \mathbb{R}^d$; then $\boldsymbol{x}$ dominates $\boldsymbol{x'}$ if $x_i \leq x'_i$ for all dimensions i and $x_i < x'_i$ for some i. The set of all non-dominated points is called the Pareto frontier. Lines indicate points on the Pareto frontier, with numbers indicating corresponding training set sizes. Table 4 presents compute times for representations (see SI 27 for kernel matrices).

Table 4: *Computational cost of calculating representations* in milliseconds of processor time. Shown are mean ± standard deviation over all training set sizes of a dataset of time to compute representation of a single molecule or unit cell. See SI 27 for details.

	Dataset		
Representation	`qm9`	`ba10`	`nmd18`
MBTR $k=2$	0.76 ± 0.32	13 ± 5.1	340 ± 99
SF $k=2$	1.40 ± 0.18	3.3 ± 1.4	8.2 ± 1.1
MBTR $k=2,3$	12.0 ± 6.9	290 ± 140	$28\,\text{k} \pm 4.4\,\text{k}$
SF $k=2,3$	2.80 ± 0.85	27 ± 12	98 ± 89
SOAP	1.90 ± 0.54	9.1 ± 4.8	19.0 ± 8.6

Discussion

Asymptotically, observed prediction errors for all representations on all datasets are related as

$$\text{SF-2,3} \prec \text{SF-2}, \qquad \text{MBTR-2,3} \preceq \text{MBTR-2},$$
$$\text{SOAP} \prec \text{SF-2,3}, \qquad \text{SOAP} \prec \text{MBTR-2,3},$$
$$\text{SF-2,3} \preceq \text{MBTR-2,3}, \qquad \text{SF-2} \prec \text{MBTR-2},$$

where $A \prec B$ ($A \preceq B$) indicates that A has lower (or equal) estimated error than B asymptotically. With the exception of MBTR-2,3 $\not\preceq$ SF-2 on dataset `nmd18r`,

$$\text{SOAP} \prec \text{SF-2,3} \preceq \text{MBTR-2,3} \prec \text{SF-2} \prec \text{MBTR-2}.$$

From this *we conclude that for energy predictions, accuracy improves with modelled interaction order and for local representations over global ones.* The magnitude of, and between, these effects varies across datasets.

Dependence on interaction order has also been observed by others,[20,35,42,70,83] and might in part be due to finer resolution of structural features. The latter would only show for sufficient training data, such as for dataset `ba10` in Figure 5. We do not observe this for dataset `qm9`, possibly because angular terms might be immediately relevant for characterizing carbon scaffolds of organic molecules.[70]

Better performance of local representations might be due to higher resolution and better generalization (both from having to represent only a small part of the whole structure). The impact of assuming additivity is unclear, but likely depends on the structure of the modeled property. (SI 6) As our observations are based on only a single global representation (MBTR), further study of the locality aspect is warranted.

Computational costs (Table 4) tend to increase with predictive accuracy. Representations should therefore be selected based on target accuracy, constrained by available computing resources. Additional analysis details can be found in SI 28.

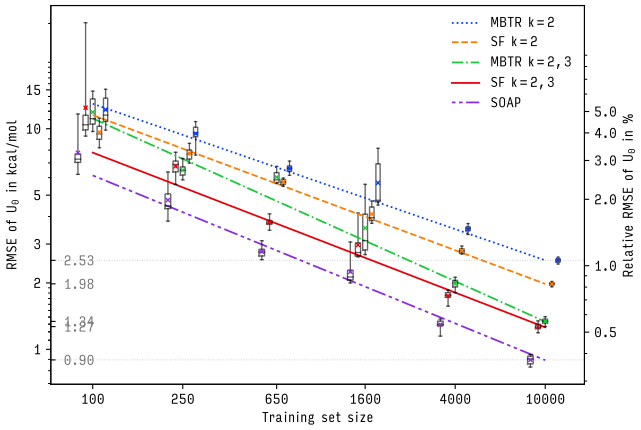
Converged prediction errors are in reasonable agreement with the literature considering lack of standardized conditions such as sampling, regression method, HP optimization and reported performance statistics. (SI 29) In absolute terms, prediction errors of models trained on 10 k samples are closer to the differences between different DFT codes than to the (systematic) differences between the underlying DFT reference and experimental measurements. (SI 30)

9 Outlook

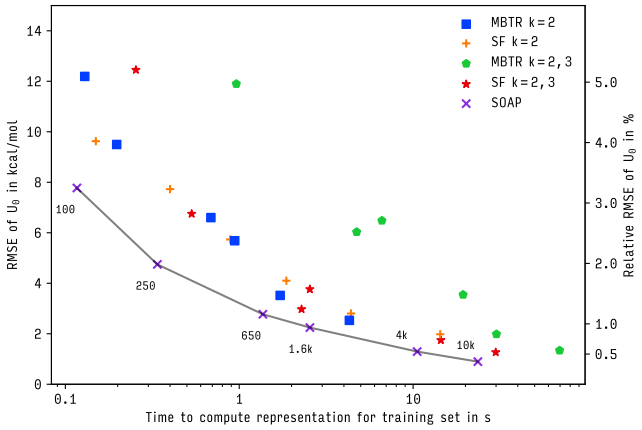
We review representations of atomistic systems, such as molecules and crystalline materials, for machine learning of ab initio quantum-mechanical simulations. Despite their apparent diversity, these representations can be formulated in a single mathematical framework based on k-atom terms, symmetrization and tensor products. Empirically we observe that when controlling for other factors, including distribution of training and validation data, regression method and HP optimization, both prediction errors and compute time of SFs, MBTR and SOAP improve with interaction order k, and for local representations over global ones.

Our findings suggest the following guidance:

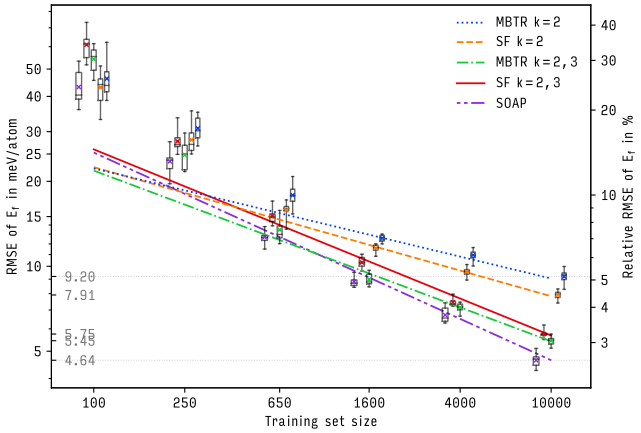
- If their prediction errors are sufficient for an application, we recommend two-body versions of simple representations such as SF and MBTR as they are fastest to compute.
- For large systems, local representations should be used.
- For strong noise or bias on input structures, as in dataset `nmd18u`, performance differences between representations vanish, and computationally cheaper features not satisfying requirements in Section 3 ("descriptors") suffice.



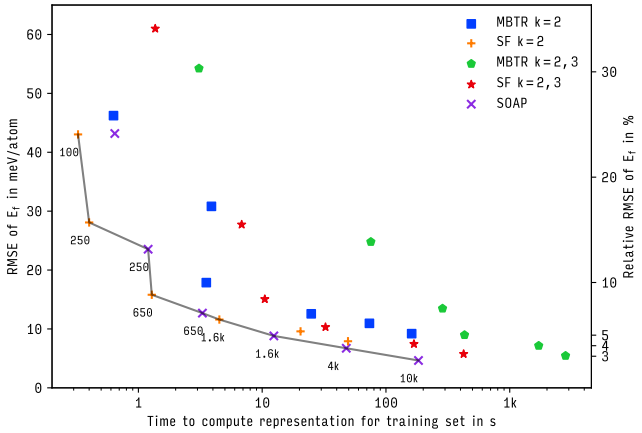
Dataset qm9.



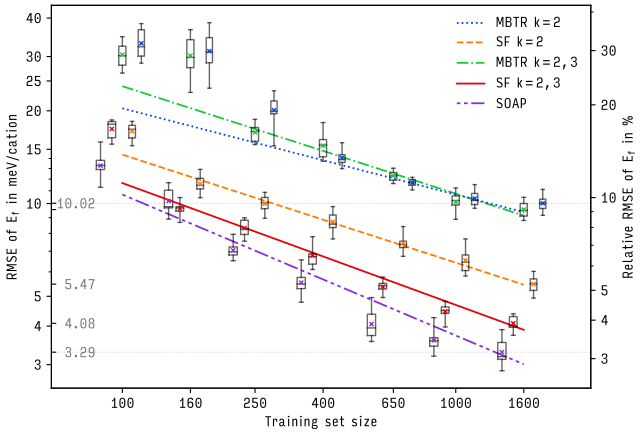
Dataset qm9.



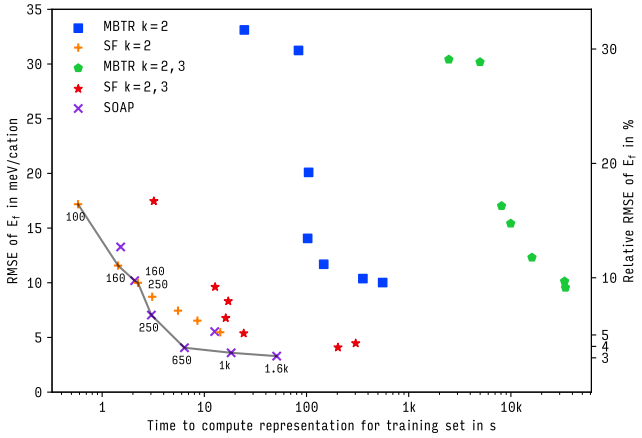
Dataset ba10.



Dataset ba10.



Dataset nmd18r.



Dataset nmd18r.

Figure 5: *Learning curves* for representations in Section 5 on datasets qm9 (top), ba10 (middle), and nmd18r (bottom). Shown is root mean squared error (RMSE) of energy predictions on out-of-sample-data as a function of training set size. Boxes, whiskers, bars, crosses show interquartile range, total range, median, mean, respectively. Lines are fits to theoretical asymptotic RMSE. (SI 16) See Glossary for abbreviations.

Figure 6: *Compute times* of representations in Section 5 for datasets qm9 (top), ba10 (middle), and nmd18r (bottom). Shown is root mean squared error (RMSE) of energy predictions on out-of-sample-data as a function of time needed to compute all representations in a training set. Lines indicate Pareto frontiers, inset numbers show training set sizes. See Glossary for abbreviations.

We hope that our work contributes to understanding, development, assessment and application of representations. All datasets, HP search spaces, ML models, program code and results are publicly available.[84] A tutorial introduction to the `cmlkit` Python framework developed for this work is provided as part of the Nomad Analytics Toolkit.[85] We conclude by providing related open research problems, grouped by topic.

Related to benchmarking of representations:

- *Extended scope.* We numerically compare one global and two local representations on three datasets for prediction of energies using KRR with a Gaussian kernel. For a more systematic coverage, other state-of-the-art representations (Section 6), further datasets, the effect of training with forces,[63,64] and more properties should be included while maintaining control over regression method, data distribution and HP optimization. Deep neural networks[86–89] could be included via representation learning.

- *Improved optimization of HPs:* The stochastic optimizer used in this work required multiple restarts in practice to avoid sub-optimal results, and reached its limits for large HP search spaces. It would be desirable to reduce influence and computational cost of HP optimization by reducing number of HPs in representations, by employing more systematic and thus robust optimization methods, and by providing reliable heuristics for HPs as starting values.

- *Multi-objective optimization.* We optimize HPs for predictive accuracy on a single property. However, in practice parametrizations of similar accuracy but lower computational cost would be preferable. HPs should therefore be optimized for multiple properties and criteria, including computational cost and predictive uncertainties (see below). How to balance these is part of the problem.[90]

- *Predictive uncertainties.* While prediction errors are frequently analyzed and reasonable guidelines exist, this is not the case for predictive uncertainties. These are becoming increasingly important as applications of ML mature, for example, for human assessment and decisions, learning on the fly[91] and active learning. Beyond global characterization of uncertainty estimates, locality (in input or feature space) of prediction errors is relevant as well.[90,92]

Directly related to representations:

- *Systematic development of representations* via extending the mathematical framework (Section 4) to include more state-of-the-art representations. This would enable derivation of "missing" variants of representations (see Table 2), such as a global SOAP[48] and local MBTR,[71] on a principled basis, as well as understanding and reformulation of existing representations in a joint framework, perhaps to the extent of an efficient general implementation.

- *Representing more systems.* Develop or extend representations for atomistic systems currently not representable, or only to a limited extent, such as charged atoms and systems, excited states, spin systems, isotopes, and systems under an applied external field.

- *Alchemical learning.* Further understand and develop alchemical representations, that is, representations incorporating similarity between chemical species to improve sample efficiency. What are the salient features of chemical elements that need to be considered, also with respect to charges, excitations, spins and isotopes?

- *Analysis of representations* to better understand structure and data distribution in feature spaces, and how they relate to concepts in physics and chemistry. Possible approaches include quantitative measures of structure and distribution of datasets in these spaces, dimensionality reduction methods, and analysis of data-driven representations from deep neural networks.

Related through context:

- *Long-range interactions.* ML models are thought to be well-suited for short- and medium-ranged interactions, but to be problematic for long-ranged interactions due to increasing degrees of freedom of larger systems and larger necessary cut-off radii of atomic environments. Integration with fast models for long-ranged interactions would be desirable, but best approaches for this have not been established yet; role of and requirements on representations for this purpose are not well understood.

- *Relationships between QM and ML.* While ML has become a useful tool for QM simulations, a deeper understanding of relationships between QM and kernel-based ML could lead to insights and technical progress in both fields. As both share concepts from linear algebra, such relationships could be formal mathematical ones. For example, QM concepts such as matrix product states can parameterize non-linear kernel models.[93]

Acknowledgments

This work received funding from the European Union's Horizon 2020 Research and Innovation Programme, Grant Agreements No. 676580, the NOMAD Laboratory CoE, and No. 740233, ERC: TEC1P.

The authors thank Profs. Matthias Scheffler, Klaus-Robert Müller, Jörg Behler, Gábor Csányi, Carsten Baldauf, as well as Emre Ahmetcik, Lauri Himanen, Yair Litman, Dmitrii Maksimov, Felix Mocanu, Wiktor Pronobis, and Christopher Sutton for constructive discussions.

References

[1] Lorenz C. Blum and Jean-Louis Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database GDB-13. *Journal of the American Chemical Society*, 131(25):8732–8733, 2009.

[2] Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.

[3] Michael I. Jordan and Tom M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[4] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*, 108(5):058301, 2012.

[5] Jörg Behler. First principles neural network potentials for reactive simulations of large molecular and condensed systems. *Angewandte Chemie International Edition*, 56(42):12828–12840, 2017.

[6] Michele Ceriotti, Michael J. Willatt, and Gábor Csányi. Machine learning of atomic-scale properties based on physical principles. In Wanda Andreoni and Sidney Yip, editors, *Handbook of Materials Modeling. Methods: Theory and Modeling*. Springer, 2018.

[7] Jörg Behler and Michele Parrinello. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical Review Letters*, 98(14):146401, 2007.

[8] Albert P. Bartók, Mike C. Payne, Risi Kondor, and Gábor Csányi. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. *Physical Review Letters*, 104(13):136403, 2010.

[9] Shin Kiyohara, Hiromi Oda, Koji Tsuda, and Teruyasu Mizoguchi. Acceleration of stable interface structure searching using a kriging approach. *Japanese Journal of Applied Physics*, 55(4):045502, 2016.

[10] Albert P. Bartók, James Kermode, Noam Bernstein, and Gábor Csányi. Machine learning a general-purpose interatomic potential for silicon. *Physical Review X*, 8(4):041048, 2018.

[11] Austin D. Sendek, Ekin D. Cubuk, Evan R. Antoniuk, Gowoon Cheon, Yi Cui, and Evan J. Reed. Machine learning-assisted discovery of solid Li-ion conducting materials. *Chemistry of Materials*, 31(2):342–352, 2018.

[12] Bernhard Schölkopf and Alexander Smola. *Learning with Kernels*. MIT Press, Cambridge, 2002.

[13] Thomas Hofmann, Bernhard Schölkopf, and Alexander Smola. Kernel methods in machine learning. *Annals of Statistics*, 36(3):1171–1220, 2008.

[14] Roberto Todeschini and Viviana Consonni. *Handbook of Molecular Descriptors*. Wiley, Weinheim, Germany, 2nd edition, 2009.

[15] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning (ICML 2017), Sydney, Australia, August 6–11*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. Proceedings of Machine Learning Research, 2017.

[16] Kristof T. Schütt, Pieter-Jan Kindermans, Huziel E. Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna Wallach, Rob Fergus, S.V.N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30 (NIPS 2017), Los Angeles, California, December 4–9*. Curran Associates, 2017.

[17] Kristof T. Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus-Robert Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, 8:13890, 2017.

[18] Risi Kondor. n-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. *arXiv*, 1803.01588, 2018.

[19] Felix A. Faber, Luke Hutchison, Bing Huang, Justin Gilmer, Samuel S. Schoenholz, George E. Dahl, Oriol Vinyals, Steven Kearnes, Patrick F. Riley, and O. Anatole von Lilienfeld. Prediction errors of molecular machine learning models lower than hybrid DFT error. *Journal of Chemical Theory and Computation*, 13(11):5255–5264, 2017.

[20] Lauri Himanen, Marc O.J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Yashasvi S. Ranawat, David Z. Gao, Patrick Rinke, and Adam S. Foster. DScribe: Library of descriptors for machine learning in materials science. *Computer Physics Communications*, 247:106949, 2020.

[21] Yunxing Zuo, Chi Chen, Xiangguo Li, Zhi Deng, Yiming Chen, Jörg Behler, Gábor Csányi, Alexander V. Shapeev, Aidan P. Thompson, Mitchell A. Wood, and Shyue Ping Ong. Performance and cost assessment of machine learning interatomic potentials. *Journal of Physical Chemistry A*, 124(4):731–745, 2020.

[22] Gunnar Schmitz, Ian Heide Godtliebsen, and Ove Christiansen. Machine learning for potential energy surfaces: An extensive database and assessment of methods. *Journal of Chemical Physics*, 150(24):244113, 2019.

[23] Chandramouli Nyshadham, Matthias Rupp, Brayden Bekker, Alexander V. Shapeev, Tim Mueller, Conrad W. Rosenbrock, Gábor Csányi, David W. Wingate, and Gus L.W. Hart. Machine-learned multi-system

surrogate models for materials prediction. *Nature Partner Journal Computational Materials*, 5:51, 2019.

[24] Annika Stuke, Milica Todorović, Matthias Rupp, Christian Kunkel, Kunal Ghosh, Lauri Himanen, and Patrick Rinke. Chemical diversity in molecular orbital energy predictions with kernel ridge regression. *Journal of Chemical Physics*, 150:in press, 2019.

[25] Mikio L. Braun, Joachim M. Buhmann, and Klaus-Robert Müller. On relevant dimensions in kernel feature spaces. *Journal of Machine Learning Research*, 9(Aug):1875–1908, 2008.

[26] Albert P. Bartók, Risi Kondor, and Gábor Csányi. On representing chemical environments. *Physical Review B*, 87(18):184115, 2013.

[27] Matthias Rupp, Raghunathan Ramakrishnan, and O. Anatole von Lilienfeld. Machine learning for quantum mechanical properties of atoms in molecules. *Journal of Physical Chemistry Letters*, 6(16):3309–3313, 2015.

[28] Joel M. Bowman, Joseph S. Bittman, and Lawrence B. Harding. *Ab initio* calculations of electronic and vibrational energies of HCO and HOC. *Journal of Chemical Physics*, 85(2):911–921, 1986.

[29] Jerry A. Darsey, Donald W. Noid, and Belle R. Upadhyaya. Application of neural network computing to the solution for the ground-state eigenenergy of two-dimensional harmonic oscillators. *Chemical Physics Letters*, 177(2):189–194, 1991.

[30] Hoon Heo, Tak-San Ho, Kevin K. Lehmann, and Herschel Rabitz. Regularized inversion of diatomic vibration-rotation spectral data: A functional sensitivity analysis approach. *Journal of Chemical Physics*, 97(2):852–861, 1992.

[31] Timothy Hollebeek, Tak-San Ho, and Herschel Rabitz. Constructing multidimensional molecular potential energy surfaces from ab initio data. *Annual Review of Physical Chemistry*, 50:537–570, 1999.

[32] Genyuan Li, Jishan Hu, Sheng-Wei Wang, Panos G. Georgopoulos, Jacqueline Schoendorf, and Herschel Rabitz. Random sampling-high dimensional model representation (RS-HDMR) and orthogonality of its different order component functions. *Journal of Physical Chemistry A*, 110(7):2474–2485, 2006.

[33] Alexander V. Shapeev. Moment tensor potentials: A class of systematically improvable interatomic potentials. *Multiscale Modeling and Simulation*, 14(3):1153–1173, 2016.

[34] Haoyan Huo and Matthias Rupp. Unified representation for machine learning of molecules and materials. *arXiv*, 1704.06439, 2017.

[35] Felix A. Faber, Anders S. Christensen, Bing Huang, and O. Anatole von Lilienfeld. Alchemical and structural distribution based representation for universal quantum machine learning. *Journal of Chemical Physics*, 148(24):241717, 2018.

[36] Jörg Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *Journal of Chemical Physics*, 134(7):074106, 2011.

[37] Jörg Behler, Sönke Lorenz, and Karsten Reuter. Representing molecule-surface interactions with symmetry-adapted neural networks. *Journal of Chemical Physics*, 127(1):014705, 2007.

[38] Michael Gastegger, Ludwig Schwiedrzik, Marius Bittermann, Florian Berzsenyi, and Philipp Marquetand. WACSF—weighted atom-centered symmetry functions as descriptors in machine learning potentials. *Journal of Chemical Physics*, 148(24):241709, 2018.

[39] Samare Rostami, Maximilian Amsler, and S. Alireza Ghasemi. Optimized symmetry functions for machine-learning interatomic potentials of multicomponent systems. *Journal of Chemical Physics*, 149(12):124106, 2018.

[40] Nongnuch Artrith, Alexander Urban, and Gerbrand Ceder. Constructing first-principles phase diagrams of amorphous Li_xSi using machine-learning-assisted sampling with an evolutionary algorithm. *Journal of Chemical Physics*, 148(24):241711, 2018.

[41] Available as part of the software `RuNNer` (`http://www.uni-goettingen.de/de/560580.html`, GPL license, per email request).

[42] Mitchell A. Wood and Aidan P. Thompson. Extending the accuracy of the SNAP interatomic potential form. *Journal of Chemical Physics*, 148(24):241721, 2018.

[43] Available as part of the software `LAMMPS` (large-scale atomic/molecular massively parallel simulator, `http://lammps.sandia.gov`, GPL license, publicly accessible).

[44] Jonathan E. Moussa. Comment on "Fast and accurate modeling of molecular atomization energies with machine learning". *Physical Review Letters*, 109(5):059801, 2012.

[45] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. Reply to the comment by J.E. Moussa. *Physical Review Letters*, 109(5):059802, 2012.

[46] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Big data meets quantum chemistry approximations: The Δ-machine learning approach. *Journal of Chemical Theory and Computation*, 11(5):2087–2096, 2015.

[47] Available as part of the software qmmlpack (quantum mechanics machine learning package) at https://gitlab.com/qmml/qmmlpack, Apache 2.0 license, publicly accessible).

[48] Sandip De, Albert P. Bartók, Gábor Csányi, and Michele Ceriotti. Comparing molecules and solids across structural and alchemical space. *Physical Chemistry Chemical Physics*, 18(20):13754–13769, 2016.

[49] Albert P. Bartók, Sandip De, Carl Poelking, Noam Bernstein, James R. Kermode, Gábor Csányi, and Michele Ceriotti. Machine learning unifies the modelling of materials and molecules. *Science Advances*, 3(12):e1701816, 2017.

[50] Ryosuke Jinnouchi, Ferenc Karsai, and Georg Kresse. On-the-fly machine learning force field generation: Application to melting points. *Physical Review B*, 100(1):014105, 2019.

[51] Available as part of the software libAtoms (http://www.libatoms.org, custom license, per webform request).

[52] Katja Hansen, Franziska Biegler, Raghunathan Ramakrishnan, Wiktor Pronobis, O. Anatole von Lilienfeld, Klaus-Robert Müller, and Alexandre Tkatchenko. Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space. *Journal of Physical Chemistry Letters*, 6(12):2326–2331, 2015.

[53] Available as part of the software QML (quantum machine learning, https://www.qmlcode.org/, MIT license, publicly accessible).

[54] Evgeny V. Podryabinkin and Alexander V. Shapeev. Active learning of linearly parametrized interatomic potentials. *Computational Materials Science*, 140:171–180, 2017.

[55] Konstantin Gubaev, Evgeny V. Podryabinkin, and Alexander V. Shapeev. Machine learning of molecular properties: Locality and active learning. *Journal of Chemical Physics*, 148(24):241727, 2018.

[56] Ivan S. Novikov and Alexander V. Shapeev. Improving accuracy of interatomic potentials: more physics or more data? A case study of silica. *Materials Today Communications*, 18:74–80, 2018.

[57] Alexander V. Shapeev. Applications of machine learning for representing interatomic interactions. In Artem R. Oganov, Gabriele Saleh, and Alexander G. Kvashnin, editors, *Computational Materials Discovery*, chapter 3, pages 66–86. Royal Society of Chemistry, Croydon, United Kingdom, 2019.

[58] Code for single-component systems is available as part of the software MLIP (machine learning of interatomic potentials, https://mlip.skoltech.ru/, BSD license, publicly accessible). Code for multi-component systems is available on request from Alexander Shapeev (a.shapeev@skoltech.ru).

[59] Anders S. Christensen, Lars A. Bratholm, Felix A. Faber, and O. Anatole von Lilienfeld. FCHL revisited: Faster and more accurate quantum machine learning. *Journal of Chemical Physics*, 152(4):044107, 2020.

[60] Aldo Glielmo, Peter Sollich, and Alessandro De Vita. Accurate interatomic force fields via machine learning with covariant kernels. *Physical Review B*, 95(21):214302, 2017.

[61] Andrea Grisafi, David M. Wilkins, Gábor Csányi, and Michele Ceriotti. Symmetry-adapted machine-learning for tensorial properties of atomistic systems. *Physical Review Letters*, 120(3):036002, 2017.

[62] O. Anatole von Lilienfeld, Raghunathan Ramakrishnan, Matthias Rupp, and Aaron Knoll. Fourier series of atomic radial distribution functions: A molecular fingerprint for machine learning models of quantum chemical properties. *International Journal of Quantum Chemistry*, 115(16):1084–1093, 2015.

[63] Albert P. Bartók and Gábor Csányi. Gaussian approximation potentials: A brief tutorial introduction. *International Journal of Quantum Chemistry*, 116(13):1051–1057, 2015.

[64] Stefan Chmiela, Huziel E. Sauceda, Klaus-Robert Müller, and Alexandre Tkatchenko. Towards exact molecular dynamics simulations with machine-learned force fields. *Nature Communications*, 9:3887, 2018.

[65] Matthew Welborn, Lixue Cheng, and Thomas F. Miller, III. Transferability in machine learning for electronic structure via the molecular orbital basis. *Journal of Chemical Theory and Computation*, 14(9):4772–4779, 2018.

[66] Christopher R. Collins, Geoffrey J. Gordon, O. Anatole von Lilienfeld, and David J. Yaron. Constant size descriptors for accurate machine learning models of molecular properties. *Journal of Chemical Physics*, 148(24):241718, 2018.

[67] Michael J. Willatt, Félix Musil, and Michele Ceriotti. Atom-density representations for machine learning. *Journal of Chemical Physics*, 150(15):154110, 2019.

[68] Kristof T. Schütt, Henning Glawe, Felix Brockherde, Antonio Sanna, Klaus-Robert Müller, and Eberhard K.U. Gross. How to represent crystal structures for machine learning: Towards fast prediction of electronic properties. *Physical Review B*, 89(20):205118, 2014.

[69] Bing Huang and O. Anatole von Lilienfeld. Communication: Understanding molecular representations in machine learning: the role of uniqueness and target similarity. *Journal of Chemical Physics*, 145(16):161102, 2016.

[70] Wiktor Pronobis, Alexandre Tkatchenko, and Klaus-Robert Müller. Many-body descriptors for predicting molecular properties with machine learning: Analysis of pairwise and three-body interactions in molecules. *Journal of Chemical Theory and Computation*, 14(6):2991–3003, 2018.

[71] The `DScribe` code contains a local MBTR example of this. See `https://github.com/SINGROUP/dscribe`.

[72] Available at the `QM/ML` website (quantum mechanics/machine learning, `https://qmml.org`, publicly accessible).

[73] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1:140022, 2014.

[74] Nomad2018 Predicting Transparent Conductors. Predict the key properties of novel transparent semiconductors. Available at `https://www.kaggle.com/c/nomad2018-predict-transparent-conductors`.

[75] Christopher Sutton, Luca M. Ghiringhelli, Takenori Yamamoto, Yury Lysogorskiy, Lars Blumenthal, Thomas Hammerschmidt, Jacek R. Golebiowski, Xiangyue Liu, Angelo Ziletti, and Matthias Scheffler. Crowd-sourcing materials-science challenges with the NOMAD 2018 Kaggle competition. *Nature Partner Journal Computational Materials*, 5:111, 2019.

[76] Matthias Rupp. Machine learning for quantum mechanics in a nutshell. *International Journal of Quantum Chemistry*, 115(16):1058–1073, 2015.

[77] Carl Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, 2006.

[78] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C.N. Pereira, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24 (NIPS 2011), Granada, Spain, December 12–15*, pages 2546–2554, 2011.

[79] James S. Bergstra, Daniel Yamins, and David D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML 2013), Atlanta, Georgia, USA, June 16–21*, pages 115–123, 2013.

[80] Corinna Cortes, Lawrence D. Jackel, Sara A. Solla, Vladimir Vapnik, and John S. Denker. Learning curves: Asymptotic values and rate of convergence. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems 6 (NIPS 1993), Denver, Colorado, USA, November 29–December 2*. Morgan Kaufmann, 1993.

[81] Klaus-Robert Müller, Michael Finke, Noboru Murata, Klaus Schulten, and Shun-ichi Amari. A numerical study on learning curves in stochastic multilayer feedforward networks. *Neural Computation*, 8(5):1085–1106, 1996.

[82] Bing Huang, Nadine O. Symonds, and O. Anatole von Lilienfeld. Quantum machine learning in chemistry and materials. In Wanda Andreoni and Sidney Yip, editors, *Handbook of Materials Modeling. Methods: Theory and Modeling*. Springer, 2018.

[83] Amit Samanta. Representing local atomic environment using descriptors based on local correlations. *Journal of Chemical Physics*, 149(24):244102, 2018.

[84] Available at `https://marcel.science/repbench` and `https://qmml.org`.

[85] Analytics Toolkit of the Novel Materials Discovery (NOMAD) Laboratory, `https://analytics-toolkit.nomad-coe.eu`.

[86] Kristof T. Schütt, Huziel E. Sauceda, Pieter-Jan Kindermans, Alexandre Tkatchenko, and Klaus-Robert Müller. SchNet—a deep learning architecture for molecules and materials. *Journal of Chemical Physics*, 148(24):241722, 2018.

[87] Benjamin Nebgen, Nicholas Lubbers, Justin S. Smith, Andrew E. Sifain, Andrey Lokhov, Olexandr Isayev, Adrian E. Roitberg, Kipton Barros, and Sergei Tretiak. Transferable dynamic molecular charge assignment using deep neural networks. *Journal of Chemical Theory and Computation*, 14(9):4687–4698, 2018.

[88] Kristof T. Schütt, Michael Gastegger, Alexandre Tkatchenko, and Klaus-Robert Müller. Quantum-chemical insights from interpretable atomistic neural networks. In Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller, editors, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 311–330. Springer, 2019.

[89] Kristof T. Schütt, Michael Gastegger, Alexandre Tkatchenko, Klaus-Robert Müller, and Reinhard J. Maurer. Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions. *Nature Communications*, 10:5024, 2019.

[90] Zachary del Rosario, Matthias Rupp, Yoolhee Kim, Erin Antono, and Julia Ling. Assessing the frontier: Active learning, model accuracy, and multi-objective materials discovery and optimization. *arXiv*, 1911.03224, 2019.

[91] Gábor Csányi, Tristan Albaret, Mike C. Payne, and Alessandro De Vita. "learn on the fly": A hybrid classical and quantum-mechanical molecular dynamics simulation. *Physical Review Letters*, 93(17):175503, 2004.

[92] Christopher Sutton, Mario Boley, Luca M. Ghiringhelli, Matthias Rupp, Jilles Vreeken, and Matthias Scheffler. Identifying domains of applicability of machine learning models for materials science. *ChemRxiv*, 9778670, 2019.

[93] Edwin Miles Stoudenmire and David J. Schwab. Supervised learning with tensor networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29 (NIPS 2016), Barcelona, Spain, December 5–10*, pages 4799–4807. Curran Associates, 2016.

Supplementary material for

Representations of molecules and materials for interpolation of quantum-mechanical simulations via machine learning

Marcel F. Langer [*] Alex Goeßmann [*†] Matthias Rupp [*‡]

[*] Theory Department, Fritz Haber Institute of the Max Planck Society, Berlin, Germany; [†] Institute of Mathematics, Technical University Berlin, Germany; [‡] Citrine Informatics, Redwood City, CA, USA

Introduction

1 | Cost of electronic structure calculations Although the computational cost of ab initio methods scales only polynomially in system size N (measured, for example, in number of electrons or orbitals), it remains a strongly limiting factor. For example, the currently most-widely used approach, Kohn-Sham density functional theory, scales as $O(N^3)$ for (semi)local and $O(N^4)$ for hybrid functionals: Doubling N thus increases compute time by roughly an order of magnitude, and a few such doublings will exhaust any computational resource. Advances in large-scale computing facilities, such as current exascale computing initiatives, will move this "computational wall" to larger systems, but cannot remove it. In practice, the large prefactor hidden in the asymptotic runtimes is relevant as well.

2 | Size of molecular and materials spaces Various estimates of the size of chemical compound spaces exist, popular ones[1,2] including 10^{33} and 10^{60} molecules. Reymond et al.[3–5] systematically enumerate all small organic molecules with up to 11 C, N, O, F atoms, 13 C, N, O, S, Cl atoms, and 17 C, N, O, S, F, Cl, Br, I atoms, yielding 26 million, 970 million and 166 billion molecules, respectively. Following Cantor,[6] we estimate the number of possible compositions (not considering unit cell size or symmetry) for an alloy system to be the multinomial coefficient $(n-1,k)! = \binom{n-1+k}{k}$, where n is number of components and $k = 100/x$ is determined by the tolerance $x\,\%$ to which the amount of a species is specified. For $n = 5$ and a very conservative choice of $x = 5\,\%$, removing combinations that contain only 4 or fewer components and multiplying by all ways to choose 5 out of 30 elements yields $\left(\binom{5-1+20}{20} - \binom{4-1+20}{20}\right)\binom{30}{5} \approx 1.5 \cdot 10^9$.

3 | The role of data quality for QM/ML models Data are the basis for data-driven models, and errors in them can only be corrected to a limited extent ("garbage in, garbage out"). Even dealing with simple errors like independent identically distributed noise requires additional data, and more severe errors lead to qualitative problems such as outliers. Conversely, problems in fitting a ML model can be indicative of problems in the data.

4 | Explicit and implicit features Features used for regression can be defined explicitly via representations, or implicitly, for example, via kernels or deep neural networks.

In this work, we focus on explicit Hilbert-space representations in conjunction with kernel-based regression with a Gaussian kernel. Technically, the features used for regression are the components of the kernel feature space, that is, the non-linear transformations of the representations' components via the Gaussian kernel. While used implicitly in this sense, the representations are still defined explicitly.

This is in contrast to implicitly defined representations, for example, feature spaces of kernels defined directly on "raw inputs" such as atomic coordinates and numbers, without an intermediate explicit Hilbert-space representation, or, the layers of deep neural networks (end-to-end learning). For the latter, the requirements in Section 3 can be imposed via the network architecture, which can be seen as the conceptual analog to explicitly conformant representations or kernels.

Role and types of representations

5 | Structure and distribution of data Figure S1 illustrates the importance of representation space structure for regression with a toy example. Low-dimensional (here, essentially one-dimensional) data is embedded into a high-dimensional (here, two-dimensional) space. The spiral embedding is not suited for linear regression, whereas the linear embedding is.

6 | Extensive and intensive properties A property whose magnitude is additive in the size (extent or mass) of an object is called *extensive*; a property whose magnitude is independent of the size of an object is called *intensive*. For example, internal energy is an extensive property, band gap energy an intensive one.

Originating from thermodynamics,[7,8] the application of these terms to microscopic quantities is limited by allowed changes in "size" of a system: For finite systems such as molecules, a property p is extensive if for any two *non-interacting* systems A and B, $p(A+B) = p(A) + p(B)$,[9] and intensive if $p(A) = p(A+A)$. For periodic systems such as bulk crystals, we take A and B to be supercells of

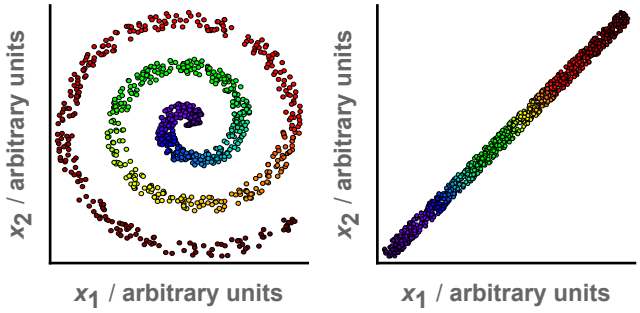


Figure S1: *Structure of representation determines suitability for regression.* Almost one-dimensional data is embedded into a two-dimensional space. The spiral embedding (left) is not suited for linear regression, but the "unrolled" embedding (right) is.

the same unit cell. In this minimal sense, total and atomization energy of atomistic systems are extensive.

However, energies are not additive for general changes in a system, such as changes in atomic position, and addition or removal of atoms. With respect to the requirements in Section 3, *ML models for energies should be size-extensive in the (minimal) sense above.* For global representations, this can be achieved via normalization in conjunction with the linear kernel,[9] whereas local representations as described in SI 7 automatically satisfy this requirement.

7 | Learning with atomic contributions One ansatz to scale prediction of global properties to large atomistic systems is to predict atomic contributions. This assumes additivity, as the predicted property is a sum of predicted atomic contributions, and locality, as efficient scaling requires representations of atoms in their environment to have local support, often achieved through a finite-radius cut-off function.

Predicting atomic contributions requires a modification of the basic kernel regression scheme, which we derive here building on References 10 and 11:

Let $\mathcal{A}_1, \ldots, \mathcal{A}_a$ denote atoms of systems $\mathcal{M}_1, \ldots, \mathcal{M}_m$ and let $\boldsymbol{D} \in \{0,1\}^{m \times a}$ be their incidence matrix, that is $D_{i,j} = 1$ if $\mathcal{A}_j$ belongs to $\mathcal{M}_i$ and 0 otherwise. Let $\tilde{k}$ denote a kernel function on atoms. The prediction for the i-th system is the sum of its predicted atomic contributions,

$$f(\mathcal{M}_i) = \sum_{j=1}^{a} f(\mathcal{A}_j) D_{i,j} = \sum_{j,\ell=1}^{a} \tilde{\alpha}_\ell \tilde{k}(\mathcal{A}_\ell, \mathcal{A}_j) D_{i,j}.$$

Minimizing quadratic loss yields

$$\begin{aligned} &\arg\min_{\tilde{\boldsymbol{\alpha}} \in \mathbb{R}^a} \sum_{i=1}^{m} \Big(y_i - \sum_{j,\ell=1}^{a} \tilde{\alpha}_\ell \tilde{k}(\mathcal{A}_\ell, \mathcal{A}_j) D_{i,j} \Big)^2 + \lambda ||f||^2_{\mathcal{H}} \\ &= \arg\min_{\tilde{\boldsymbol{\alpha}} \in \mathbb{R}^a} \Big\langle \boldsymbol{y} - \boldsymbol{D}\tilde{\boldsymbol{K}}\tilde{\boldsymbol{\alpha}} \Big| \boldsymbol{y} - \boldsymbol{D}\tilde{\boldsymbol{K}}\tilde{\boldsymbol{\alpha}} \Big\rangle + \lambda\, \tilde{\boldsymbol{\alpha}}^T \tilde{\boldsymbol{K}} \tilde{\boldsymbol{\alpha}}. \end{aligned}$$

Since this is a quadratic form, it suffices to set its gradient to zero and solve for $\tilde{\boldsymbol{\alpha}}$:

$$\begin{aligned} \nabla_{\tilde{\boldsymbol{\alpha}}} &= -2\tilde{\boldsymbol{\alpha}}^T \tilde{\boldsymbol{K}} \boldsymbol{D}^T \boldsymbol{y} + 2\tilde{\boldsymbol{K}} \boldsymbol{D}^T \boldsymbol{D} \tilde{\boldsymbol{K}} \tilde{\boldsymbol{\alpha}} + 2\lambda \tilde{\boldsymbol{K}} \tilde{\boldsymbol{\alpha}} = \boldsymbol{0} \\ &\Leftrightarrow\ \tilde{\boldsymbol{\alpha}} = (\boldsymbol{D}^T \boldsymbol{D} \tilde{\boldsymbol{K}} + \lambda \boldsymbol{I})^{-1} \boldsymbol{D}^T \boldsymbol{y} \\ &\Leftrightarrow\ \tilde{\boldsymbol{\alpha}} = \boldsymbol{D}^T (\boldsymbol{D} \tilde{\boldsymbol{K}} \boldsymbol{D}^T + \lambda \boldsymbol{I})^{-1} \boldsymbol{y}, \end{aligned} \tag{5}$$

where the last expression is preferable for numerical evaluation. Predictions for m' new systems $\mathcal{M}'$ with a' atoms $\mathcal{A}'$ can be expressed efficiently as

$$\boldsymbol{y'} = \Big(\sum_{j=1}^{a'} D'_{i,j} \sum_{\ell=1}^{a} \tilde{\alpha}_\ell \tilde{k}(\mathcal{A}_\ell, \mathcal{A}'_j) \Big)_{i=1,\ldots,m'} = \boldsymbol{D'} \tilde{\boldsymbol{L}}^T \tilde{\boldsymbol{\alpha}}, \tag{6}$$

where $\boldsymbol{D'}$ is the incidence matrix for the predicted systems and $\tilde{\boldsymbol{L}}$ is the $a \times a'$ kernel matrix between atoms $\mathcal{A}$ and $\mathcal{A}'$.

This approach is equivalent to kernel regression (SI 19) on whole systems with a kernel k given by the sum of the atom kernel $\tilde{k}$ over all pairs of atoms in two systems,

$$k(\mathcal{M}_i, \mathcal{M}_j) = \sum_{p,q=1}^{a} \boldsymbol{D}_{i,p} \tilde{\boldsymbol{K}}_{p,q} \boldsymbol{D}^T_{q,j}.$$

This follows from $\boldsymbol{K} = \boldsymbol{D}\tilde{\boldsymbol{K}}\boldsymbol{D}^T$, $\boldsymbol{L} = \boldsymbol{D}\tilde{\boldsymbol{L}}\boldsymbol{D'}^T$ and $\tilde{\boldsymbol{\alpha}} = \boldsymbol{D}^T\boldsymbol{\alpha}$ (Equation 5): Predictions for whole systems using k are identical to predictions using $\tilde{k}$: $\boldsymbol{y'} = \boldsymbol{L}^T\boldsymbol{\alpha} = \boldsymbol{D'}\tilde{\boldsymbol{L}}^T\boldsymbol{D}^T\boldsymbol{\alpha} = \boldsymbol{D'}\tilde{\boldsymbol{L}}\tilde{\boldsymbol{\alpha}}$. In particular, atomic weights $\tilde{\boldsymbol{\alpha}}$ are blocks of system weights $\boldsymbol{\alpha}$.

Computing full atom kernel matrices $\tilde{\boldsymbol{K}}$ and incidence matrices $\boldsymbol{D}$ can require large amounts of memory. In practice, we compute blocks of $\tilde{\boldsymbol{K}}$ on the fly and directly sum over its entries. Learning with atomic contributions is extensive (SI 6).

Requirements

8 | Computational cost Let f^{QM} and f^{ML} denote the total computational cost when using only ab initio simulations and a ML-augmented model, respectively:

$$\begin{aligned} f^{\text{QM}}(n,m) &= (n+m)\,\text{ref} \\ f^{\text{ML}}(n,m) &= n\,\text{ref} + (n+m)\,\text{repr} + \text{train}(n) + m\,\text{pred}, \end{aligned}$$

where n and m are number of training and predicted systems, ref, repr, pred are the cost of one simulation, representation calculation and prediction, and train is the cost of training the ML model. If $n \approx \epsilon\, m$ for small ϵ, and costs of training and prediction are negligible, total savings in compute time are

$$f^{\text{QM}}(n,m) - f^{\text{ML}}(n,m) \approx m\,(\text{ref} - \text{repr}).$$

Both ref and repr depend on system size, often polynomially, with differences in asymptotic runtime as well as constant factors relevant in practice. Local representations require computing more kernel matrix entries (SI 7) than global representations, which can be noticeable (Table S5), but enable scaling with system size:

Let c denote the (average) number of atoms per system, and d the (average) number of atoms in a local environment. In the following, we assume d to be constant (bounded from above), and representations to have constant size. Total computational effort to compute representations (first term) and kernel matrices (second term) is then given by

$$\mathcal{O}\big((n+m)\,c^k + nm\big) \quad \text{and} \quad \mathcal{O}\big((n+m)\,c\,d^k + nmc^2\big)$$

for global (left) and local (right) representations, where d^k is constant. For small systems $c \approx d$, and the additional overhead in computing kernel matrices will dominate runtime for small k. In the limit $c \to \infty$ of large systems, the c^k term will dominate for global representations, while local representations enjoy quadratic scaling. This can be observed to some extent in Tables S4 to S6.

9 | Role of representations The role of the representation is to map atomistic systems into a space amenable to regression (linear interpolation). Strictly speaking, for kernel regression this is the kernel feature space, that is, representation space transformed by the kernel. We limit our discussion to the representation itself—for the linear kernel this is exact as the transformation is the identity, and many non-linear kernels like the Gaussian kernel act on the representation space, relying on its structure and implied metric.

A unified framework

10 | k-body functions A k-body function maps information about k atoms $|\alpha_1\rangle, \ldots, |\alpha_k\rangle$, where order can matter, to an output space, here the real numbers, or a distribution on them. Atom information $|\alpha\rangle$ typically includes coordinates and proton number, but is not limited to those; for example, it could include neutron number to model isotopes.

Typical k-body functions include atomic number counts ($k = 1$), distances, sometimes inverted or squared ($k = 2$), angles or their cosine ($k = 3$), dihedral or torsional angles, volume-related terms ($k = 4$). Less common, (al)chemical relationships can be included, for example, based on atoms' period and group in the periodic table.[12]

In this work, we do not model $k = 4$ or higher-order interactions due to the computational cost from combinatorial growth of number of terms, which becomes a limiting factor for larger systems, such as in the `nmd18` dataset.

Representations

11 | Local atomic neighbourhoods Local representations are computed for a local neighbourhood of a central atom, usually defined as $\{\,|\alpha_i\rangle \,|\, d_i \leq r_c\}$, where $|\alpha_i\rangle$ denotes atom i, d_i is distance of $|\alpha_i\rangle$ to the central atom, and $r_c \geq 0$ is a cut-off radius.

Both the `quippy` and `DScribe` implementations of SOAP include the central atom in the neighbourhood, and thus in the neighbourhood density,[13] in contrast to the original definition.[14] SFs do not take the central atom into account explicitly.

In periodic systems, the unit cell is replicated up to the cut-off radius to ensure that all interactions within the neighbourhood are included. In practice, some implementations may internally use a modified *effective* cut-off radius. For instance, `DScribe` ensures that atoms up to the tail of the radial basis function are taken into account.

Empirical comparison

12 | `qm9` dataset The `qm9` dataset,[15,16] also known as `gdb9-14`, contains 133 885 small organic molecules composed of H, C, N, O, F with up to 9 non-H atoms. It is a subset of the "generated database 17" (GDB-17).[5] Molecular ground state geometries and properties, including energetics, are computed at density functional level of theory using the Becke 3-parameter Lee-Yang-Parr (B3LYP)[17] hybrid functional with 6-31G(2df,p) basis set.

We use the version available at `qmml.org`, which offers a convenient format for parsing, and exclude all structures in the `uncharacterized.txt` file and those listed in the `readme.txt` file as "difficult to converge", as those are potentially problematic. Total energies were converted to energies of atomization by subtracting the atomic contributions given in file `atomref.txt`.

13 | `ba10` dataset The `ba10` dataset,[18] also known as `dft-10b`, contains unrelaxed geometries and their enthalpies of formation for the 10 binary alloys AgCu, AlFe, AlMg, AlNi, AlTi, CoNi, CuFe, CuNi, FeV, and NbNi. For each alloy system, unrelaxed geometries with lattice parameters from Vegard's rule[19,20] and energies are computed for all possible unit cells[21] with 1–8 atoms for FCC and BCC lattices, and 2–8 atoms for HCP lattices, using the generalized gradient approximation (GGA) of Perdew, Burke and Ernzerhof (PBE) with projector-augmented wave (PAW) potentials and generalized regular k-point grids.[22,23] The dataset contains 631 FCC, 631 BCC, and 333 HCP structures per alloy system, yielding 15 950 structures in total. We use the version available at `qmml.org`.

14 | `nmd18` dataset The `nmd18` dataset[24] is a Kaggle challenge[25] dataset containing 3 000 ternary $(Al_x\text{-}Ga_y\text{-}In_z)_2O_3$ oxides, $x + y + z = 1$, of potential interest as transparent conducting oxides. We predict formation and band-gap energies of relaxed structures, using either relaxed (`nmd18r`) or approximate (`nmd18u`) structures from Vegard's rule as input. Geometries and energies are computed at the density functional level of theory using the PBE functional as implemented in the all-electron code `FHI-aims`[26] with `tight` settings.

The challenge scenario is to predict formation and band-gap energies of relaxed structures from unrelaxed geometries obtained via Vegard's rule. This is equivalent to strong noise or bias in the inputs. Unlike pure benchmarking scenarios, where computationally expensive relaxed geometries are given, the challenge scenario is closer to a virtual screening application in that Vegard's rule geometries are computationally inexpensive to obtain.

The dataset contains all structures from the challenge training and leaderboard data. Unless otherwise noted, we report RMSE, not the root mean square logarithmic error used in the challenge.

15 | `nmd18u` dataset Figures S2 and S3, and, Tables S2 to S6 present results for energy predictions on the `nmd18u` dataset, that is, the `nmd18` dataset with approximate geometries obtained from Vegard's rule. In contrast to relaxed structures, such geometries can be obtained at almost no cost, and could be used in virtual screening campaigns.

We observe (i) a strong increase in prediction errors (14–21 % for rRMSE), (ii) collapse of all representations to similar performance, (iii) large differences between MAE and RMSE, indicating significant outliers. From this, we conclude that the map from unrelaxed structures to ground-state energies is harder to learn than the map from relaxed structures to their energies, and, that here the representation is not the limiting factor, and other sources of error dominate.

16 | Learning curves Plots of empirical prediction error ϵ as a function of training set size n are called "learning curves". Asymptotically, we assume the error to decay as a negative power,[27] $\epsilon = a' n^{-b}$. On a log-log plot, ϵ is therefore linear, $\log \epsilon = a - b \log(n)$, and the offset a and slope b can be used to characterize predictive performance of models.[28] For QM/ML models the estimated quantities are noise-free (except for numerical noise, which is negligible for converged calculations) and representations are unique. For asymptotic fits we weight training set sizes by the standard deviation over their respective splits to attenuate for small sample effects.

17 | Subsets For training and validation, data subsets were sampled as follows: An *outer validation set*[1] was randomly drawn (10 k molecules for `qm9`, 1 k structures for `ba10`, 600 structures for `nmd18`). From the remaining entries, *outer training sets* of sizes 100, 250, 650, 1 600, 4 000 and 10 000 for datasets `qm9`, `ba10` and 100, 160, 250, 400, 650, 1 000 and 1 600 for dataset `nmd18` were randomly drawn. These sizes were chosen to be equidistant in log-space. Each outer training set was then split into an *inner training set* and an *inner validation set* by randomly drawing the latter. We used an 80 / 20 % split, yielding inner validation sets of size 20, 50, 130, 320, 800, 2 000 for datasets `qm9`, `ba10` and 20, 32, 50, 80, 130, 200, 320 for `nmd18`. The whole procedure was repeated 10 times. We excluded structures with few atoms (6 or fewer non-H atoms for `qm9`, 5 or fewer atoms per unit cell for `ba10`, 10 atoms per unit cell for `nmd18`) as there are not enough of these for statistical learning.

18 | Sampling To reduce variance, remove bias and ensure that subsets faithfully represent the distribution of the whole dataset, subsets were drawn using Monte-Carlo sampling such that differences to the parent dataset in selected statistics were below pre-defined fractional thresholds.

For dataset `qm9`, these were number of N, O and F atoms, number of molecules with 7, 8 and 9 non-H atoms, binned number of atoms (with H), and binned energy. For dataset `ba10`, these were number of all constituting elements, unit cells with 6, 7, 8, and 9 atoms, binned sizes and energies. For dataset `nmd18`, these were number of Al, Ga, In, O atoms, unit cells with 20, 30, 40, 60, 80 atoms, and binned energies.

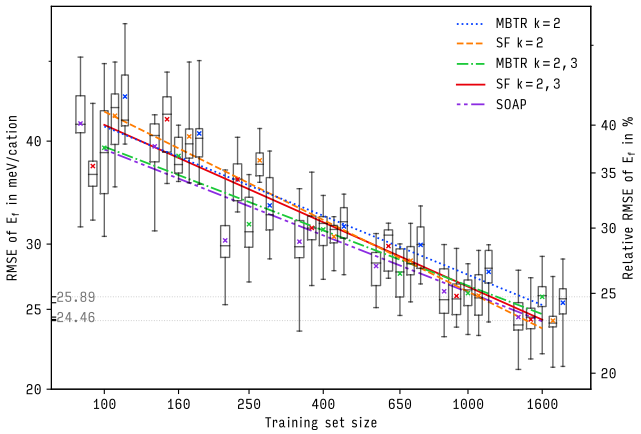
19 | Kernel regression We use kernel ridge regression[29] or Gaussian process regression[30] (the two are equivalent in terms of predictions). A detailed derivation can be found in Reference 31. In summary, predictions are basis set expansions of the form $f(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i k(\boldsymbol{x_i}, \boldsymbol{x})$, where $\boldsymbol{x}$ is the system to predict, $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$ are the training systems, and k is a symmetric positive definite function (kernel). The regression coefficients $\boldsymbol{\alpha}$ are obtained by minimizing the regularized quadratic loss $\sum_{i=1}^{n} (y_i - f(\boldsymbol{x_i}))^2 + \lambda ||f||^2_{\mathcal{H}}$. Here, $\boldsymbol{y}$ are property values of the training data and the regularization strength λ is a HP that controls the smoothness of the predictor. We used the `qmmlpack`[31,32] implementation.

20 | HP optimization For model selection we optimized the HPs of representations, kernel and regression method, including structural HPs (for example, which k-body functions to use) and numerical HPs (for example, the Gaussian kernel length scale). Specifically, the RMSE of an inner validation set was minimized using tree-structured Parzen estimators[33,34] in combination with local grid search. The same optimization scheme was used for all representations, using consistent grid spacings and parameter ranges to reduce human bias. Our corresponding `cmlkit` package[35] provides interfaces to the `hyperopt` optimization package[33] and to each representation's implementation(s); it is freely available under an open source license.

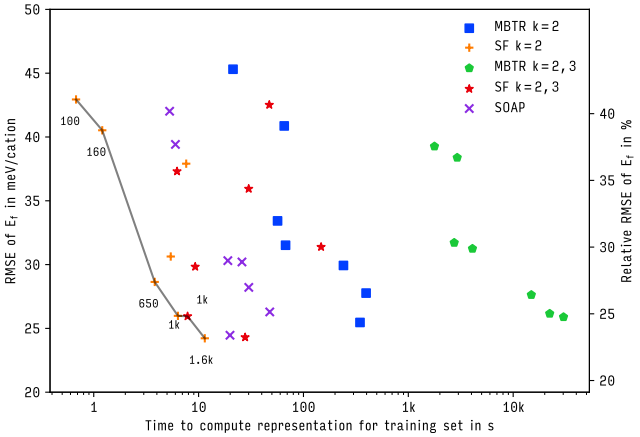
The space of possible models ("HP search space") is a tree-structured set of choices, for instance, between different k-body functions, or different values of a numerical HP. Tree-structured Parzen estimators treat this search space as a prior distribution over HPs, updated every time a loss is computed to increase prior weight around HP settings with better loss. We use uniform priors throughout, discretizing numerical HPs on logarithmic or linear grids as necessary. Once a HP search space has been defined, model selection is fully automatic.

HPs were optimized for each training set size as follows: For each trial, representation HPs and starting values for regression method HPs (Gaussian kernel length scale and regularization strength) were drawn from the prior. The latter were then refined through a randomized local grid search and the resulting HP values used to update the prior. All optimizations were run for 2 000 steps, and rerun three times, to minimize variance from stochastic optimization. To reduce computational cost, HPs were optimized on only one outer split; reported values are averages over all ten outer splits.

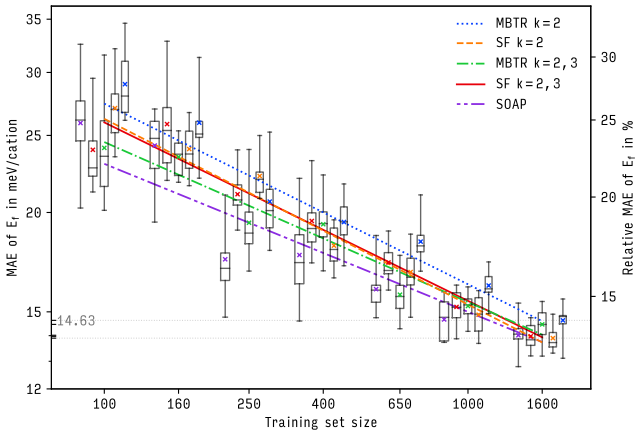
[1] In the literature, the terms "test set" and "validation set" are sometimes used with different meaning. To avoid confusion, we use "outer" for the subset employed to measure performance, and "inner" for the subset employed to optimize HPs.



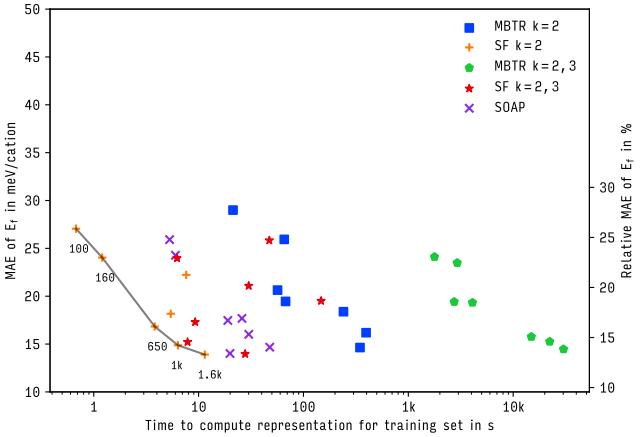
Learning curves (RMSE).



Prediction errors (RMSE) versus compute times.



Learning curves (MAE).



Prediction errors (MAE) versus compute times.

Figure S2: *Learning curves for dataset* `nmd18u` of the representations in Section 5. Shown are root mean squared error (RMSE, top) and mean absolute error (MAE, bottom) of energy predictions on out-of-sample-data as a function of training set size. Boxes, whiskers, bars, crosses show interquartile range, total range, median, mean. Lines are fits to theoretical asymptotic error. See Glossary for abbreviations.

Figure S3: *Compute times for dataset* `nmd18u` of the representations in Section 5. Shown are root mean squared error (RMSE, top) and mean absolute error (MAE, bottom) of energy predictions on out-of-sample-data as a function of time needed to compute representations. Lines indicate Pareto frontiers, inset numbers show training set sizes. See Glossary for abbreviations.

21 | Kernel regression HPs We used KRR with a single Gaussian kernel, a frequently used combination in the literature. Note that due to Requirement (iii.a), the Gaussian kernel is better suited than less smooth kernels such as the Laplacian kernel.[36]

No post-processing of the kernel was performed. In particular, centering of kernel and labels, which together is equivalent to having an explicit bias term b in the regression, were not performed, as this is not necessary for the Gaussian kernel.[37] Depending on the representation used, labels were normalized for training as needed to either represent values per atom or per entire system.

This setup entails two HPs: The width σ of the Gaussian kernel, and the regularization strength λ. Search spaces for these two HPs (Table S1) were held constant across all representations and learning curves. See Reference 38 for HP search spaces and optimized model HPs.

Table S1: *Kernel regression hyperparameter search space.* Both parameters optimized on a base-2 logarithmic grid. TPE = tree-structured Parzen estimators; LGS = local grid search.

Hyper-	TPE			LGS		
parameter	min	max	step	min	max	step
$\log_2 \lambda$	-18	0	1.0	-20	2	0.5
$\log_2 \sigma$	-13	13	1.0	-15	15	0.5

22 | Symmetry function HPs We consider the five SFs proposed in Reference 39:

$$G_i^1 = \sum_j f_c(d_{ij}) \tag{7}$$

$$G_i^2 = \sum_j \exp\left(-\eta(d_{ij}-\mu)^2\right) f_c(d_{ij}) \tag{8}$$

$$G_i^3 = \sum_j \cos(\kappa\, d_{ij})\, f_c(d_{ij}) \tag{9}$$

$$\begin{aligned} G_i^4 = {} & 2^{1-\zeta} \sum_{j,k\neq i} (1+\lambda\cos\theta_{ijk})^\zeta \\ & \exp\left(-\eta(d_{ij}^2+d_{ik}^2+d_{jk}^2)\right) \\ & f_c(d_{ij})\, f_c(d_{ik})\, f_c(d_{jk}) \end{aligned} \tag{10}$$

$$\begin{aligned} G_i^5 = {} & 2^{1-\zeta} \sum_{j,k\neq i} (1+\lambda\cos\theta_{ijk})^\zeta \\ & \exp\left(-\eta(d_{ij}^2+d_{ik}^2)\right) f_c(d_{ij})\, f_c(d_{ik})\,, \end{aligned} \tag{11}$$

with the cut-off function

$$f_c(d_{ij}) = \begin{cases} 0.5\cos\left(\pi d_{ij}/c\right) & \text{for} \quad d_{ij} \leq c \\ 0 & \text{for} \quad d_{ij} > c\,. \end{cases} \tag{12}$$

In Equations 7 to 12, index i is the central atom, j, k run over all atoms in the local environment around i with cut-off radius c, d_{lm} indicates pairwise distance, θ_{lmn} the angle between three atoms; η and κ are broadening parameters, μ a shift, ζ determines angular resolution. $\lambda = \pm 1$ determines whether the angular part of G_i^4 and G_i^5 peaks at $0°$ or $180°$.

We utilize the `RuNNer`[40] software to compute SFs and restrict ourselves to the radial SFs G_i^2 and angular SFs G_i^4 (`RuNNer` functions 2 and 3). We use the same SFs for all element combinations to minimize size of HP search space. Similarly, we use an empirical parametrization scheme[41] to choose HPs μ and η for G_i^2 and HPs η and ζ for G_i^4.

For radial SFs we use two schemes, *shifted* and *centered*. For *shifted*, μ is chosen on a linear grid while η is held fixed. For *centered*, $\mu = 0$ and η is chosen such that the standard deviation of each SF lies on the same grid points. For $i \in \{0, 1, \ldots, n\}$ a point on a one-dimensional grid, $\Delta = \frac{c-1.5}{n-1}$, and $r_i = 1 + \Delta i$, in the *centered* scheme, $\mu_i = 0$ and $\eta_i = \frac{1}{2r_i^2}$, and in the *shifted* scheme, $\mu_i = r_i$ and $\eta_i = (2\Delta^2)^{-1}$. In this setting, the only HP is the number of grid points $n+1$, which we allow to vary from 2 to 10 for each scheme.

For angular SFs, we choose $\lambda = \pm 1$ and $\zeta = 1, 2, 4$. The only HP remaining is the broadening η, optimized on a $\log_2$ grid between -20 and 1 with spacing 0.5. The radial SFs and two angular SFs with $\lambda = \pm 1$ and $\zeta = 1$ are always included, but the optimizer can enable or disable any of the remaining $k = 3$ SFs with $\lambda = \pm 1$ and $\zeta = 2, 4$. Cut-off radii are varied in integer steps, starting from the integer above the smallest distance found in the dataset.

The output of `RuNNer` is post-processed to be suitable for KRR, placing all SFs for a given type of central atom in separate blocks of an atomic feature vector with # elements × # SFs components. For the Gaussian kernel, this leads to negligible kernel values between representations belonging to different elements. As SFs are local representations, labels were normalized to (extensive) per-system values.

See Reference 38 for HP search spaces and optimized model HPs.

23 | Many-body tensor representation HPs We employed the MBTR implementation in `qmmlpack`,[32] adding optional normalization by ℓ_1 or ℓ_2 norm. For $k = 2, 3$, representations for $k = 2$ and $k = 3$ were concatenated. MBTR exhibits several categorical HPs, with subsequent numerical HPs conditional on prior choices.

We used the k-body functions `1/distance`, `1/dot` (k=2), and `angle`, `cos_angle`, `dot/dotdot` (k=3). No one-body terms were used as atomization and formation energies already contain linear contributions of element counts. Histogram ranges were chosen based on the whole dataset, as inter-atomic distance ranges are similar for all subsets. 100 discretization bins were used throughout. Broadening parameters were restricted to at least a single bin and at most a quarter of the range of the corresponding geometry function.

From the weighting functions, we used `identity^2`, `exp_-1/identity`, `exp_-1/identity^2` (k=2), and `1/dotdotdot`, `exp_-1/normnormnorm`, `exp_-1/norm+norm+norm` (k=3). The latter two in each set introduce conditional HPs. For periodic systems, in particular the `nmd18` dataset, the ranges of these parameters were manually restricted to avoid excessive computation times (above 30 s for one trial). The convergence threshold was set to 0.001.

We used the `full` indexing scheme, which generates all permutations of elements (as opposed to `noreversals`, which does not double-count element combinations, for example, CH and HC). This seems to lead to more consistent behaviour and higher predictive accuracy for supercells, or unit cells of different sizes, and similar accuracy for molecules, at the expense of higher computational cost. We used per-system energies for the `qm9` dataset and per-atom energies for datasets `ba10` and `nmd18`.

24 | Smooth Overlap of Atomic Positions HPs We used the `DScribe` implementation of SOAP with Gaussian-type orbitals,[42,43] which we found to provide more accurate predictions at lower computational cost than the `quippy`[44] implementation. Results are already structured by element types; no post-processing was applied. HPs l_{max} and n_{max} were chosen between 2 and 8. Cut-off radii were chosen as for SFs, and the broadening adapted to the resulting ranges (53 steps from -20 to 6 on a $\log_2$ grid). We report results for the `gto` basis set, which resulted in lower prediction errors than the `polynomial` one, and was faster to compute. Labels were normalized to per-system values. See Reference 38 for HP search spaces and optimized model HPs.

25 | Prediction errors Tables S2 and S3 present numerical values underlying the learning curves for RMSE (Table S2) and MAE (Table S3). For rRMSE (SI 26), standard deviations of $239.31\,\text{kcal}\,\text{mol}^{-1}$, $178.86\,\text{meV}$, $104.57\,\text{meV}$, were used for datasets `qm9`, `ba10`, `nmd18`, computed over the whole dataset (differences to standard deviations over validation sets were around 1 % or less in all cases).

26 | Error metrics We measure predictive performance by two metrics, an absolute one and a relative one that facilitates comparison across datasets. In addition, we also provide a metric for qualitative comparison with the literature.

Let y_i, f_i, $e_i = f_i - y_i$ denote i-th observed label, prediction and residual. Root mean squared error (RMSE) and mean absolute error (MAE) are given by

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2}, \quad \text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |e_i|.$$

The canonical loss for least-squares regression is RMSE (as it is optimized by the regression). We also provide MAE since it is often reported in the literature (Figures S4 and S5).

RMSE and MAE are scale-dependent, and thus not suited for comparison across different datasets. We therefore also report the scale-independent relative RMSE (rRMSE),

$$\text{rRMSE} = \frac{\text{RMSE}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2}} = \frac{\text{RMSE}}{\sigma_y} = \frac{\text{RMSE}}{\text{RMSE}^*},$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ is the mean of the observed labels and σ_y is their standard deviation. The rRMSE can be seen as RMSE relative to the RMSE of a baseline model RMSE* that always predicts the mean of the labels. While the latter is more naturally computed using training labels and the former using validation labels, as long as as the assumption of independent and identically distributed data holds, the number of samples is more important.

See References 45, 46 and references therein for an extended discussion of error metrics.

27 | Compute times Tables S4 to S6 present empirical computational costs, measured by processor wall-time, for calculating representations and kernel matrices, respectively. Experiments were run on a single core of an Intel Xeon E5-2698v4 2.2 GHz processor.

For Table S4, representations of the 10 k, 1 k, 600 systems (datasets `qm9`, `ba10`, `nmd18`) in the first outer validation set were computed en bloc and the result divided by number of systems; this was repeated three times.

Similarly, for Table S5 kernel matrices between the representations of these systems were computed, also over three repetitions. The results were divided by the number of entries in the respective kernel matrices, yielding average kernel evaluation times.

Table S6 presents a summary overview of compute times for representations and kernel matrices.

28 | Analysis details Predictive accuracy as measured by rRMSE is worse for solid-state datasets compared to the molecular `qm9` one. This might indicate that periodic systems pose harder learning tasks than molecules.

MBTR performs worse for solid-state datasets than for the `qm9` one. This might be due to increasing difficulty of the learning problem with system size (see discussion in Section 8) and lack of intrinsic scaling with number of atoms, impeding interpolation between unit cells of different size.

For the `qm9` dataset at 1 600 training samples, we observe an increase in RMSE standard deviation compared to neighbouring training set sizes for most methods. Comparing to MAE, which exhibits no such effect, and investigating errors individually, revealed that this is due to outliers, that is, few predictions with high error in some, but not all, outer splits. The problematic structures are ring molecules, and are not present in the outer training split used for HP optimization. This stresses the importance of carefully stratifying benchmark datasets.

29 | Comparison with literature-reported errors Due to different conditions, such as sampling, regression and HP optimization methods, comparisons with performance estimates reported in the literature must remain qualitative. Frequently, only MAE is reported, which tends to result in lower absolute values and to de-emphasize outliers. Table S7 presents selected performance estimates from the literature. Overall, errors in this work appear to be compatible with reported ones.

Table S2: *Prediction errors (RMSE)* for representations of Section 5. Shown is mean ± standard deviation over ten outer splits for energy predictions, measured on an out-of-sample validation set.

(a) Dataset `qm9`.

	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k=2$	12.19 ± 1.83	9.50 ± 0.92	6.60 ± 0.29	5.68 ± 1.33	3.52 ± 0.12	2.53 ± 0.06
SF $k=2$	9.63 ± 0.94	7.73 ± 0.54	5.74 ± 0.17	4.10 ± 0.36	2.80 ± 0.08	1.98 ± 0.04
MBTR $k=2,3$	11.90 ± 1.86	6.48 ± 0.46	6.03 ± 0.40	3.55 ± 0.92	1.98 ± 0.10	1.34 ± 0.05
SF $k=2,3$	12.45 ± 5.98	6.75 ± 0.60	3.76 ± 0.16	2.97 ± 0.48	1.75 ± 0.09	1.27 ± 0.05
SOAP	7.77 ± 1.53	4.75 ± 0.75	2.77 ± 0.16	2.25 ± 0.31	1.29 ± 0.07	0.90 ± 0.05

(b) Dataset `ba10`.

	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k=2$	46.22 ± 6.76	30.81 ± 2.90	17.87 ± 1.69	12.57 ± 0.33	10.95 ± 0.45	9.20 ± 0.46
SF $k=2$	43.03 ± 5.47	28.08 ± 3.09	15.80 ± 0.98	11.60 ± 0.34	9.59 ± 0.37	7.91 ± 0.27
MBTR $k=2,3$	54.24 ± 5.10	24.79 ± 2.70	13.47 ± 1.22	8.98 ± 0.42	7.15 ± 0.24	5.45 ± 0.20
SF $k=2,3$	60.99 ± 6.66	27.72 ± 2.34	15.07 ± 0.89	10.32 ± 0.48	7.45 ± 0.24	5.75 ± 0.17
SOAP	43.18 ± 5.95	23.54 ± 2.06	12.69 ± 0.64	8.82 ± 0.37	6.72 ± 0.42	4.64 ± 0.25

(c) Dataset `nmd18r`.

	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k=2$	33.12 ± 3.55	31.24 ± 4.51	20.09 ± 2.08	14.06 ± 0.82	11.69 ± 0.33	10.38 ± 0.58	10.02 ± 0.53
SF $k=2$	17.18 ± 0.99	11.57 ± 0.73	10.00 ± 0.61	8.72 ± 0.66	7.45 ± 0.46	6.54 ± 0.57	5.47 ± 0.32
MBTR $k=2,3$	30.41 ± 2.78	30.18 ± 4.45	17.03 ± 1.10	15.42 ± 1.34	12.31 ± 0.47	10.13 ± 0.73	9.56 ± 0.57
SF $k=2,3$	17.47 ± 1.20	9.62 ± 0.44	8.31 ± 0.51	6.78 ± 0.55	5.37 ± 0.24	4.47 ± 0.24	4.08 ± 0.22
SOAP	13.28 ± 1.19	10.19 ± 0.91	7.05 ± 0.38	5.54 ± 0.47	4.06 ± 0.43	3.60 ± 0.30	3.29 ± 0.34

(d) Dataset `nmd18u`.

	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k=2$	45.31 ± 5.43	40.87 ± 3.93	33.42 ± 3.08	31.52 ± 2.14	29.93 ± 2.07	27.77 ± 1.87	25.46 ± 2.01
SF $k=2$	42.94 ± 4.61	40.53 ± 3.83	37.91 ± 1.95	30.63 ± 1.43	28.64 ± 1.82	25.98 ± 1.95	24.22 ± 1.52
MBTR $k=2,3$	39.27 ± 5.65	38.38 ± 2.16	31.71 ± 3.01	31.25 ± 2.07	27.63 ± 1.92	26.16 ± 1.43	25.89 ± 1.76
SF $k=2,3$	37.31 ± 3.64	42.52 ± 4.21	35.94 ± 2.28	31.39 ± 2.53	29.83 ± 1.78	25.95 ± 1.65	24.30 ± 1.46
SOAP	42.02 ± 5.04	39.42 ± 3.64	30.31 ± 2.92	30.21 ± 3.09	28.21 ± 1.81	26.29 ± 2.11	24.46 ± 1.88

Table S3: *Prediction errors (MAE)* for representations of Section 5. Shown is mean ± standard deviation over ten outer splits for energy predictions, measured on an out-of-sample validation set.

(a) Dataset `qm9`.

	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k=2$	8.54 ± 0.85	5.93 ± 0.26	4.66 ± 0.17	3.28 ± 0.14	2.33 ± 0.03	1.67 ± 0.03
SF $k=2$	6.72 ± 0.78	5.34 ± 0.28	3.86 ± 0.12	2.64 ± 0.05	1.87 ± 0.03	1.34 ± 0.02
MBTR $k=2,3$	8.25 ± 0.87	4.28 ± 0.18	3.88 ± 0.12	1.91 ± 0.09	1.21 ± 0.03	0.87 ± 0.02
SF $k=2,3$	7.34 ± 1.35	4.18 ± 0.23	2.49 ± 0.06	1.80 ± 0.05	1.09 ± 0.02	0.78 ± 0.02
SOAP	4.93 ± 0.59	2.79 ± 0.20	1.70 ± 0.05	1.26 ± 0.04	0.73 ± 0.02	0.49 ± 0.01

(b) Dataset `ba10`.

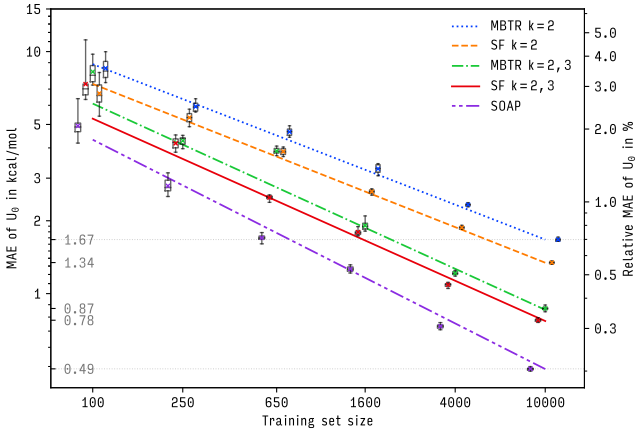
	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k=2$	27.01 ± 1.99	18.22 ± 1.21	10.74 ± 0.60	7.49 ± 0.19	6.35 ± 0.19	5.16 ± 0.19
SF $k=2$	28.02 ± 2.21	18.23 ± 1.24	9.76 ± 0.56	6.98 ± 0.16	5.52 ± 0.18	4.49 ± 0.12
MBTR $k=2,3$	36.93 ± 2.62	15.49 ± 1.32	8.47 ± 0.42	5.56 ± 0.16	4.34 ± 0.07	3.26 ± 0.07
SF $k=2,3$	40.67 ± 2.22	18.18 ± 1.20	9.43 ± 0.48	6.38 ± 0.23	4.43 ± 0.07	3.45 ± 0.08
SOAP	27.68 ± 2.06	14.82 ± 0.78	7.89 ± 0.34	5.43 ± 0.19	3.96 ± 0.09	2.78 ± 0.11

(c) Dataset `nmd18r`.

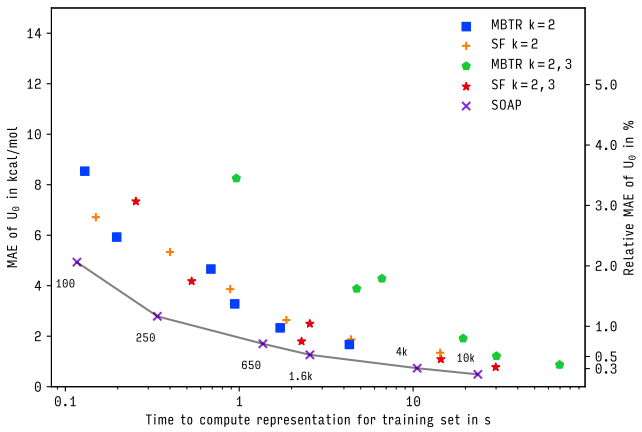
	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k=2$	21.06 ± 2.10	18.94 ± 2.05	11.73 ± 0.91	8.24 ± 0.35	6.73 ± 0.19	5.69 ± 0.27	5.63 ± 0.17
SF $k=2$	11.10 ± 0.64	7.43 ± 0.42	6.10 ± 0.48	4.99 ± 0.35	4.30 ± 0.14	3.52 ± 0.19	2.98 ± 0.12
MBTR $k=2,3$	19.77 ± 1.41	18.30 ± 1.99	10.51 ± 0.66	9.49 ± 0.60	7.23 ± 0.30	5.60 ± 0.25	5.52 ± 0.14
SF $k=2,3$	11.49 ± 0.75	6.10 ± 0.22	5.07 ± 0.29	3.93 ± 0.27	3.07 ± 0.10	2.50 ± 0.13	2.21 ± 0.08
SOAP	8.38 ± 1.05	6.18 ± 0.43	4.24 ± 0.21	3.19 ± 0.30	2.29 ± 0.18	1.89 ± 0.14	1.70 ± 0.11

(d) Dataset `nmd18u`.

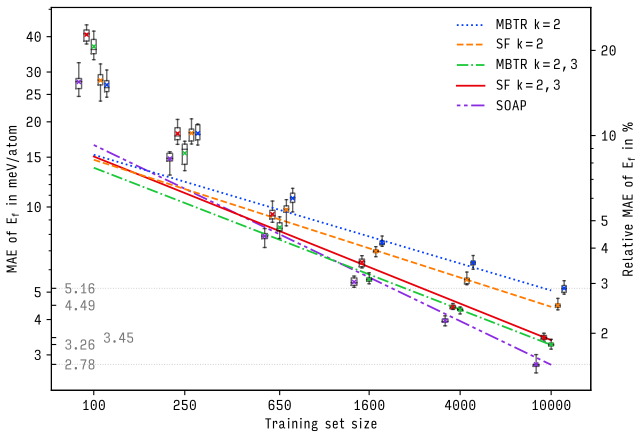
	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k=2$	29.00 ± 2.65	25.94 ± 2.79	20.64 ± 2.24	19.46 ± 1.30	18.38 ± 1.16	16.19 ± 0.67	14.63 ± 0.62
SF $k=2$	27.06 ± 2.49	24.03 ± 1.71	22.23 ± 1.32	18.17 ± 1.04	16.83 ± 1.16	14.87 ± 0.78	13.90 ± 0.48
MBTR $k=2,3$	24.11 ± 3.47	23.49 ± 1.19	19.41 ± 2.12	19.33 ± 1.40	15.76 ± 0.93	15.26 ± 0.53	14.46 ± 0.64
SF $k=2,3$	23.98 ± 2.80	25.83 ± 3.34	21.09 ± 1.29	19.51 ± 1.76	17.30 ± 0.97	15.21 ± 0.73	13.97 ± 0.49
SOAP	25.91 ± 3.32	24.28 ± 2.29	17.47 ± 1.77	17.68 ± 2.10	16.02 ± 1.08	14.67 ± 0.82	14.02 ± 0.68



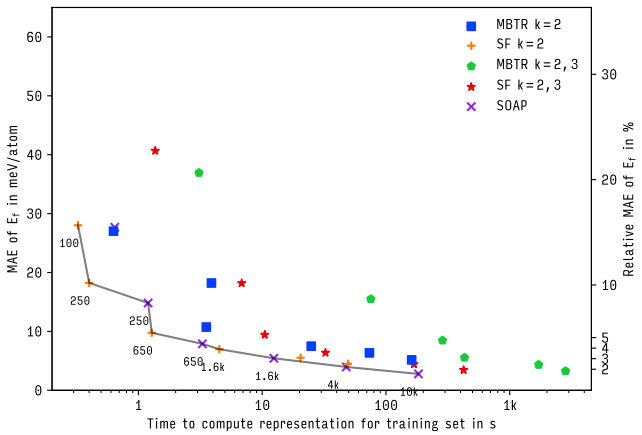
Dataset qm9.



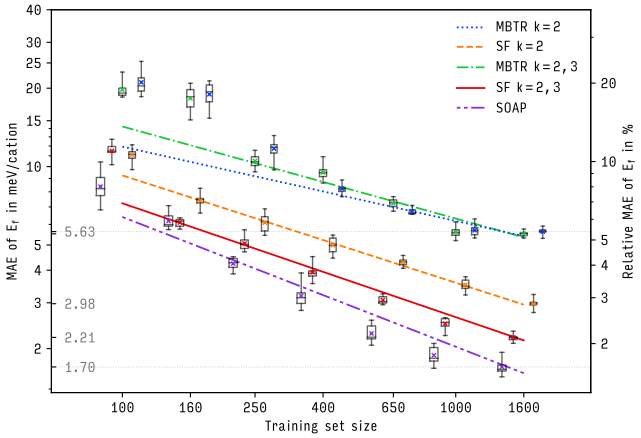
Dataset qm9.



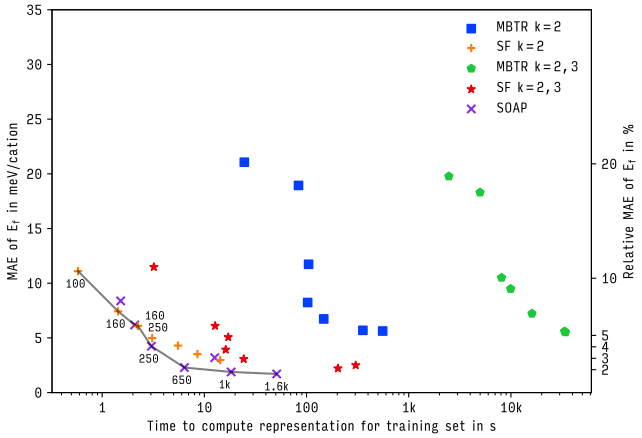
Dataset ba10.



Dataset ba10.



Dataset nmd18r.



Dataset nmd18r.

Figure S4: *Learning curves for mean absolute error* (MAE) of representations in Section 5 on datasets qm9 (top), ba10 (middle), and nmd18r (bottom). Shown is MAE of energy predictions on out-of-sample-data as a function of training set size. Boxes, whiskers, bars, crosses show interquartile range, total range, median, mean, respectively. Lines are fits to theoretical asymptotic error. See Glossary for abbreviations.

Figure S5: *Compute times* for representations in Section 5 on datasets qm9 (top), ba10 (middle), and nmd18r (bottom). Shown is mean absolute error (MAE) of energy predictions on out-of-sample-data as a function of time needed to compute representations. Lines indicate Pareto frontiers, inset numbers show training set sizes. See Glossary for abbreviations.

Table S4: *Computational cost of calculating representations* in milliseconds of processor wall-time on a single core. Shown are mean ± standard deviation over three repetitions of the time to compute a single system (molecule or unit cell).

(a) Dataset `qm9`.

	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k = 2$	1.3 ± 0.1	0.8 ± 0.1	1.1 ± 0.1	0.6 ± 0.1	0.4 ± 0.1	0.4 ± 0.1
SF $k = 2$	1.5 ± 0.1	1.6 ± 0.1	1.4 ± 0.1	1.2 ± 0.1	1.1 ± 0.1	1.4 ± 0.1
MBTR $k = 2, 3$	9.6 ± 0.1	26.5 ± 0.1	7.3 ± 0.1	12.1 ± 0.1	7.5 ± 0.1	7.0 ± 0.1
SF $k = 2, 3$	2.5 ± 0.2	2.1 ± 0.1	3.9 ± 0.2	1.4 ± 0.1	3.6 ± 0.1	3.0 ± 0.1
SOAP	1.2 ± 0.1	1.4 ± 0.1	2.1 ± 0.1	1.6 ± 0.1	2.6 ± 0.1	2.4 ± 0.1

(b) Dataset `ba10`.

	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k = 2$	6.3 ± 0.3	15.6 ± 0.1	5.4 ± 0.1	15.6 ± 0.1	18.4 ± 0.1	16.1 ± 0.1
SF $k = 2$	3.3 ± 0.7	1.6 ± 0.1	2.0 ± 0.1	2.8 ± 0.2	5.1 ± 1.5	4.9 ± 0.1
MBTR $k = 2, 3$	30.9 ± 0.4	302.0 ± 0.1	440.8 ± 0.1	269.6 ± 0.1	428.5 ± 0.2	282.5 ± 0.1
SF $k = 2, 3$	13.7 ± 0.1	27.3 ± 0.8	16.1 ± 0.2	20.3 ± 0.2	42.1 ± 0.1	42.4 ± 0.3
SOAP	6.4 ± 0.1	4.8 ± 0.1	5.1 ± 0.1	7.8 ± 0.1	12.0 ± 0.1	18.3 ± 0.1

(c) Dataset `nmd18r`.

	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k = 2$	246 ± 1	522 ± 1	419 ± 1	256 ± 1	227 ± 1	356 ± 1	348 ± 1
SF $k = 2$	6 ± 1	9 ± 1	9 ± 1	8 ± 1	8 ± 1	9 ± 1	9 ± 1
MBTR $k = 2, 3$	$24\,688 \pm 2$	$31\,168 \pm 3$	$32\,408 \pm 1$	$24\,864 \pm 2$	$24\,728 \pm 3$	$33\,518 \pm 1$	$21\,377 \pm 5$
SF $k = 2, 3$	32 ± 1	80 ± 1	69 ± 1	40 ± 1	37 ± 1	303 ± 2	127 ± 1
SOAP	15 ± 1	13 ± 1	12 ± 1	32 ± 1	10 ± 1	18 ± 1	32 ± 1

(d) Dataset `nmd18u`.

	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k = 2$	213 ± 2	410 ± 1	226 ± 1	169 ± 1	370 ± 1	396 ± 1	216 ± 1
SF $k = 2$	7 ± 1	8 ± 1	30 ± 1	14 ± 1	6 ± 1	6 ± 1	7 ± 1
MBTR $k = 2, 3$	$17\,757 \pm 8$	$18\,286 \pm 2$	$10\,959 \pm 1$	$10\,225 \pm 1$	$22\,921 \pm 1$	$22\,296 \pm 2$	$18\,878 \pm 1$
SF $k = 2, 3$	62 ± 1	295 ± 3	120 ± 2	368 ± 2	14 ± 1	8 ± 1	17 ± 1
SOAP	53 ± 1	38 ± 1	76 ± 1	65 ± 1	46 ± 1	48 ± 1	12 ± 1

Table S5: *Computational costs of calculating kernel matrices* in microseconds of processor wall-time on a single core. Shown are mean ± standard deviation over three repetitions of the time to compute a single kernel matrix entry.

(a) Dataset `qm9`.

	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k=2$	0.16 ± 0.01	0.16 ± 0.01	0.16 ± 0.01	0.16 ± 0.01	0.16 ± 0.01	0.16 ± 0.01
SF $k=2$	13.74 ± 0.04	14.86 ± 0.01	12.17 ± 0.04	10.14 ± 0.01	9.80 ± 0.14	12.22 ± 0.01
MBTR $k=2,3$	0.90 ± 0.01	0.90 ± 0.01	0.90 ± 0.01	0.90 ± 0.01	0.90 ± 0.01	0.90 ± 0.01
SF $k=2,3$	15.16 ± 0.04	14.35 ± 0.01	21.75 ± 0.03	14.27 ± 0.01	21.89 ± 0.09	17.02 ± 0.01
SOAP	14.97 ± 0.03	28.80 ± 0.01	71.30 ± 0.03	31.81 ± 0.01	120.09 ± 0.10	91.74 ± 0.01

(b) Dataset `ba10`.

	Training set size					
Representation	100	250	650	1600	4000	10 000
MBTR $k=2$	0.64 ± 0.02	0.63 ± 0.01	0.63 ± 0.01	0.63 ± 0.01	0.64 ± 0.01	0.63 ± 0.01
SF $k=2$	9.73 ± 0.11	7.19 ± 0.01	8.22 ± 0.01	11.38 ± 0.02	9.13 ± 0.01	7.74 ± 0.01
MBTR $k=2,3$	7.36 ± 0.62	6.91 ± 0.02	6.95 ± 0.01	6.95 ± 0.01	6.94 ± 0.01	6.94 ± 0.01
SF $k=2,3$	15.14 ± 0.22	15.09 ± 0.02	20.92 ± 0.01	22.50 ± 0.14	20.03 ± 0.08	26.49 ± 0.03
SOAP	108.49 ± 1.24	66.37 ± 2.01	74.27 ± 0.45	43.86 ± 0.07	80.48 ± 0.47	62.11 ± 0.01

(c) Dataset `nmd18r`.

	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k=2$	0.2 ± 0.2	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1
SF $k=2$	74.7 ± 0.1	66.7 ± 0.1	66.7 ± 0.1	70.7 ± 0.1	78.6 ± 0.1	78.7 ± 0.1	74.7 ± 0.2
MBTR $k=2,3$	3.3 ± 3.8	0.5 ± 0.1	0.5 ± 0.1	0.6 ± 0.1	0.6 ± 0.1	0.5 ± 0.1	0.5 ± 0.1
SF $k=2,3$	84.0 ± 0.3	97.4 ± 0.1	104.7 ± 0.6	118.9 ± 0.1	95.7 ± 0.3	111.4 ± 0.1	132.2 ± 0.2
SOAP	89.7 ± 0.3	142.9 ± 0.1	308.6 ± 0.2	907.9 ± 2.5	174.2 ± 0.1	198.0 ± 0.1	252.3 ± 0.1

(c) Dataset `nmd18u`.

	Training set size						
Representation	100	160	250	400	650	1000	1600
MBTR $k=2$	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.1 ± 0.1
SF $k=2$	78.9 ± 0.4	70.8 ± 0.4	66.3 ± 0.1	83.8 ± 0.1	70.2 ± 0.2	66.4 ± 0.1	87.7 ± 0.1
MBTR $k=2,3$	0.7 ± 0.3	0.8 ± 0.1	0.5 ± 0.1	1.1 ± 0.1	0.5 ± 0.1	0.5 ± 0.1	0.5 ± 0.1
SF $k=2,3$	117.2 ± 0.9	91.7 ± 0.2	128.2 ± 1.2	124.1 ± 0.1	105.5 ± 0.1	88.4 ± 0.1	99.3 ± 0.1
SOAP	294.5 ± 2.7	79.3 ± 0.1	607.1 ± 0.4	199.3 ± 0.1	542.0 ± 3.8	382.4 ± 0.7	110.4 ± 0.1

Table S6: *Overview of computational costs for calculating representations and kernel matrices.* Shown are computational cost estimates for (a) training on 10 k training samples and (b) prediction of 10 k validation samples. Based on mean observed compute times t_{rep} for representations and t_{kernel} for kernel matrices from Tables S4 and S5, we estimate total training times as $N_{\text{train}} \cdot t_{\text{rep}} + N_{\text{train}}^2 \cdot t_{\text{kernel}}/2$ and prediction times as $N_{\text{test}} \cdot t_{\text{rep}} + N_{\text{train}} \cdot N_{\text{test}} \cdot t_{\text{kernel}}$. Training times do not include time to calculate regression weights. All times are rounded to the nearest second, minute, or hour.

(a) Training times.

	Dataset								
	qm9			ba10			nmd18		
Representation	t_{rep}	t_{kernel}	total	t_{rep}	t_{kernel}	total	t_{rep}	t_{kernel}	total
MBTR $k = 2$	8s	+ 8s	= 15s	2m	+ 32s	= 3m	57m	+ 6s	= 57m
SF $k = 2$	14s	+ 10m	= 10m	33s	+ 7m	= 8m	1m	+ 1h	= 1h
MBTR $k = 2, 3$	2m	+ 45s	= 3m	49m	+ 6m	= 55m	76h	+ 46s	= 77h
SF $k = 2, 3$	28s	+ 15m	= 15m	4m	+ 17m	= 21m	16m	+ 1h	= 2h
SOAP	19s	+ 50m	= 50m	2m	+ 1h	= 1h	3m	+ 4h	= 4h

(b) Prediction times.

	Dataset								
	qm9			ba10			nmd18		
Representation	t_{rep}	t_{kernel}	total	t_{rep}	t_{kernel}	total	t_{rep}	t_{kernel}	total
MBTR $k = 2$	8s	+ 16s	= 23s	2m	+ 1m	= 3m	57m	+ 13s	= 57m
SF $k = 2$	14s	+ 20m	= 20m	33s	+ 15m	= 15m	1m	+ 2h	= 2h
MBTR $k = 2, 3$	2m	+ 1m	= 3m	49m	+ 12m	= 1h	76h	+ 2m	= 77h
SF $k = 2, 3$	28s	+ 29m	= 29m	4m	+ 33m	= 38m	16m	+ 3h	= 3h
SOAP	19s	+ 2h	= 2h	2m	+ 2h	= 2h	3m	+ 8h	= 8h

30 | Comparison with DFT and experimental errors The error of DFT simulations against experimentally measured observations depends on system and property, as well as choice of density functional and other parameters, such as convergence thresholds and k-point density. For heats of formation and the Becke 3-parameter Lee-Yang-Parr (B3LYP) functional used for the `qm9` dataset, (systematic) MAEs relative to experiment of $\approx 2.6\,\mathrm{kcal\,mol^{-1}}$ have been reported for small organic molecules containing only C, H, N, O.[47] For cohesive energies and the Perdew-Burke-Ernzerhof (PBE) functional used for the `ba10` and `nmd18` datasets, values of approximately 200 to $300\,\mathrm{meV/atom}$ have been reported.[48–50]

For the PBE functional, reported MAEs in computed energies between different parametrizations of DFT codes and RMSEs between 20 different DFT codes on 71 elements in bulk crystalline form were approximately 2 meV/atom and 1.7 meV/atom, respectively;[51] the latter reduces to 0.6 meV/atom for all-electron codes only. The best models for bulk crystal reported here have RMSEs of 4.6 meV/atom and 3.3 meV/cation on the `ba10` and `nmd18` datasets. However, the former benchmark values are integrated over a $\pm 6\,\%$ interval around the equilibrium volume, whereas the values reported here are computed at the minima themselves and therefore measure related but distinct quantities. This suggests that prediction errors are at least $\approx$ 2–6 times larger than DFT-intrinsic variations.

Table S7: *Performance estimates from the literature.* Ref. = Reference, MAE = mean absolute error, RMSE = root mean square error, N = training set size.

(a) `qm9` dataset.

	Error / $\mathrm{kcal\,mol^{-1}}$			
Ref.	MAE	RMSE	N	Method
52	1.5	2.8	5 k	IDMBR [a]
53	0.72	—	10 k	SOAP
54	1.27	—	10 k	SchNet
12	0.44	—	10 k	FCHL
53	0.66	—	10 k	FCHL [b]
55	0.14	—	100 k	SOAP [c]
56	0.35	0.94	100 k	SchNet
57	0.58	—	118 k	HDAD
here	0.49	0.90	10 k	SOAP

[a] inverse-distance many-body representation
[b] revised FCHL19 version
[c] radial-scaling modification

(b) `ba10` dataset.

	Error / $\mathrm{meV\,atom^{-1}}$			
Ref.	MAE	RMSE	N	Method
18	5.3	—	10 k	MBTR
18	3.4	—	10 k	MTP
here	2.8	4.6	10 k	SOAP

(c) `nmd18u` dataset. Here, all representations performed roughly equally. At the time of printing, no published results existed for the relaxed `nmd18r` version.

	Error / $\mathrm{meV\,cation^{-1}}$			
Ref.	MAE	RMSE	N	Method
24	13	—	2 400	SOAP
here	14–15	24–26	1 600	all

References

[1] Pavel G. Polishchuk, Timur I. Madzhidov, and Alexandre Varnek. Estimation of the size of drug-like chemical space based on GDB-17 data. *Journal of Computer-Aided Molecular Design*, 27(8):675–679, 2013.

[2] Regine S. Bohacek, Colin McMartin, and Wayne C. Guida. The art and practice of structure-based drug design: A molecular modeling perspective. *Medicinal Research Reviews*, 16(1):3–50, 1996.

[3] Tobias Fink and Jean-Louis Reymond. Virtual exploration of the chemical universe up to 11 atoms of C, N, O, F: Assembly of 26.4 million structures (110.9 million stereoisomers) and analysis for new ring systems, stereochemistry, physicochemical properties, compound classes, and drug discovery. *Journal of Chemical Information and Modeling*, 47(2):342–353, 2007.

[4] Lorenz C. Blum and Jean-Louis Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database GDB-13. *Journal of the American Chemical Society*, 131(25):8732–8733, 2009.

[5] Lars Ruddigkeit, Ruud van Deursen, Lorenz C. Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. *Journal of Chemical Information and Modeling*, 52(11):2864–2875, 2012.

[6] Brian Cantor. Multicomponent and high entropy alloys. *Entropy*, 16(9):4749–4768, 2014.

[7] Richard C. Tolman. The measurable quantities of physics. *Physical Review*, 9(3):237–253, 1917.

[8] George N. Hatsopoulos and Joseph H. Keenan. *Principles of General Thermodynamics*. Wiley, New York, 1965.

[9] Hyunwook Jung, Sina Stocker, Christian Kunkel, Harald Oberhofer, Byungchan Han, Karsten Reuter, and Johannes T. Margraf. Size-extensive molecular machine learning with global representations. *ChemSystemsChem*, in press, 2020.

[10] Albert P. Bartók and Gábor Csányi. Gaussian approximation potentials: A brief tutorial introduction. *International Journal of Quantum Chemistry*, 116(13):1051–1057, 2015.

[11] Sonja Mathias. A kernel-based learning method for an efficient approximation of the high-dimensional Born-Oppenheimer potential energy hypersurface. Master's thesis, Institute for Numerical Simulation, Mathematisch-Naturwissenschaftliche Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn, Germany, 2015.

[12] Felix A. Faber, Anders S. Christensen, Bing Huang, and O. Anatole von Lilienfeld. Alchemical and structural distribution based representation for universal quantum machine learning. *Journal of Chemical Physics*, 148(24):241717, 2018.

[13] Sandip De, Albert P. Bartók, Gábor Csányi, and Michele Ceriotti. Comparing molecules and solids across structural and alchemical space. *Physical Chemistry Chemical Physics*, 18(20):13754–13769, 2016.

[14] Albert P. Bartók, Risi Kondor, and Gábor Csányi. On representing chemical environments. *Physical Review B*, 87(18):184115, 2013.

[15] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1:140022, 2014.

[16] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Big data meets quantum chemistry approximations: The Δ-machine learning approach. *Journal of Chemical Theory and Computation*, 11(5):2087–2096, 2015.

[17] Philip J. Stephens, Frank J. Devlin, Cary F. Chabalowski, and Michael J. Frisch. *Ab Initio* calculation of vibrational absorption and circular dichroism spectra using density functional force fields. *Journal of Physical Chemistry*, 98(45):11623–11627, 1994.

[18] Chandramouli Nyshadham, Matthias Rupp, Brayden Bekker, Alexander V. Shapeev, Tim Mueller, Conrad W. Rosenbrock, Gábor Csányi, David W. Wingate, and Gus L.W. Hart. Machine-learned multi-system surrogate models for materials prediction. *Nature Partner Journal Computational Materials*, 5:51, 2019.

[19] Lars Vegard. Die Konstitution der Mischkristalle und die Raumfüllung der Atome. *Zeitschrift für Physik*, 5(1):17–26, 1921.

[20] Alan R. Denton and Neil W. Ashcroft. Vegard's law. *Physical Review A*, 43(6):3161, 1991.

[21] Gus L. W. Hart and Rodney W. Forcade. Algorithm for generating derivative structures. *Physical Review B*, 77(22):224115, 2008.

[22] Pandu Wisesa, Kyle A. McGill, and Tim Mueller. Efficient generation of generalized Monkhorst-Pack grids through the use of informatics. *Physical Review B*, 93(15):155109, 2016.

[23] Wiley S. Morgan, Jeremy J. Jorgensen, Bret C. Hess, and Gus L.W. Hart. Efficiency of generalized regular k-point grids. *Computational Materials Science*, 153:424–430, 2018.

[24] Christopher Sutton, Luca M. Ghiringhelli, Takenori Yamamoto, Yury Lysogorskiy, Lars Blumenthal, Thomas Hammerschmidt, Jacek R. Golebiowski, Xiangyue Liu, Angelo Ziletti, and Matthias Scheffler. Crowd-sourcing materials-science challenges with the NOMAD 2018 Kaggle competition. *Nature Partner Journal Computational Materials*, 5:111, 2019.

[25] Nomad2018 Predicting Transparent Conductors. Predict the key properties of novel transparent semiconductors. Available at `https://www.kaggle.com/c/nomad2018-predict-transparent-conductors`.

[26] Volker Blum, Ralf Gehrke, Felix Hanke, Paula Havu, Ville Havu, Xinguo Ren, Karsten Reuter, and Matthias Scheffler. *Ab initio* molecular simulations with numeric atom-centered orbitals. *Computer Physics Communications*, 180(11):2175–2196, 2009.

[27] Shun-ichi Amari, Naotake Fujita, and Shigeru Shinomoto. Four types of learning curves. *Neural Computation*, 4(4):605–618, 1992.

[28] Bing Huang and O. Anatole von Lilienfeld. Communication: Understanding molecular representations in machine learning: the role of uniqueness and target similarity. *Journal of Chemical Physics*, 145(16):161102, 2016.

[29] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer, New York, 2 edition, 2009.

[30] Carl Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, 2006.

[31] Matthias Rupp. Machine learning for quantum mechanics in a nutshell. *International Journal of Quantum Chemistry*, 115(16):1058–1073, 2015.

[32] The `qmmlpack` (quantum mechanics machine learning package) library is publicly available at `https://gitlab.com/qmml/qmmlpack` under the Apache-2.0 license.

[33] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C.N. Pereira, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24 (NIPS 2011), Granada, Spain, December 12–15*, pages 2546–2554, 2011.

[34] James S. Bergstra, Daniel Yamins, and David D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML 2013), Atlanta, Georgia, USA, June 16–21*, pages 115–123. Proceedings of Machine Learning Research 28, 2013.

[35] The `cmlkit` Python package is publicly available at `https://marcel.science/cmlkit` under an MIT license and as part of the Nomad Analytics Toolkit (`https://analytics-toolkit.nomad-coe.eu/`).

[36] Haoyan Huo and Matthias Rupp. Unified representation for machine learning of molecules and materials. *arXiv*, 1704.06439, 2017.

[37] Tomaso Poggio, Sayan Mukherjee, Ryan Rifkin, Alexander Rakhlin, and Alessandro Verri. b. Technical Report AI Memo 2001-011, CBCL Memo 198, Massachusetts Institute of Technology, 2001.

[38] HP search spaces and HP values for optimized models are available at `https://marcel.science/repbench` and `https://qmml.org`.

[39] Jörg Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *Journal of Chemical Physics*, 134(7):074106, 2011.

[40] The `RuNNer` software (`https://www.uni-goettingen.de/de/560580.html`, GPL license) is available from its author Jörg Behler (`joerg.behler@uni-goettingen.de`) on request.

[41] Michael Gastegger, Ludwig Schwiedrzik, Marius Bittermann, Florian Berzsenyi, and Philipp Marquetand. WACSF—weighted atom-centered symmetry functions as descriptors in machine learning potentials. *Journal of Chemical Physics*, 148(24):241709, 2018.

[42] The `DScribe` software library is publicly available at `https://github.com/SINGROUP/dscribe` under the Apache-2.0 license.

[43] Lauri Himanen, Marc O.J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Yashasvi S. Ranawat, David Z. Gao, Patrick Rinke, and Adam S. Foster. DScribe: Library of descriptors for machine learning in materials science. *Computer Physics Communications*, 247:106949, 2020.

[44] The `quippy` software is publicly available at `http://libatoms.github.io/QUIP/` under the GNU General Public license 2.

[45] Chao Chen, Jamie Twycross, and Jonathan M. Garibaldi. A new accuracy measure based on bounded relative error for time series forecasting. *PLoS ONE*, 12(3):e0174202, 2017.

[46] Alexei Botchkarev. A new typology design of performance metrics to measure errors in machine learning regression algorithms. *Interdisciplinary Journal of Information, Knowledge, and Management*, 14:45–76, 2019.

[47] Julian Tirado-Rives and William L. Jorgensen. Performance of B3LYP density functional methods for a large set of organic molecules. *Journal of Chemical Theory and Computation*, 4(2):297–306, 2008.

[48] Stephan Lany. Semiconductor thermochemistry in density functional calculations. *Physical Review B*, 78(24):245207, 2008.

[49] Kurt Lejaeghere, Veronique Van Speybroeck, Guido Van Oost, and Stefaan Cottenier. Error estimates for solid-state density-functional theory predictions: An overview by means of the ground-state elemental crystals. *Critical Reviews in Solid State and Materials Sciences*, 39(1):1–24, 2014.

[50] Guo-Xu Zhang, Anthony M. Reilly, Alexandre Tkatchenko, and Matthias Scheffler. Performance of various density-functional approximations for cohesive properties of 64 bulk solids. *New Journal of Physics*, 20(6):063020, 2018.

[51] Kurt Lejaeghere, Gustav Bihlmayer, Torbjörn Björkman, Peter Blaha, Stefan Blügel, Volker Blum, Damien Caliste, Ivano E. Castelli, Stewart J. Clark, Andrea Dal Corso, Stefano de Gironcoli, Thierry Deutsch, John Kay Dewhurst, Igor Di Marco, Claudia Draxl, Marcin Dułak, Olle Eriksson, José A. Flores-Livas, Kevin F. Garrity, Luigi Genovese, Paolo Giannozzi, Matteo Giantomassi, Stefan Goedecker, Xavier Gonze, Oscar Grånäs, Eberhard K. U. Gross, Andris Gulans, François Gygi, Donald R. Hamann, Phil J. Hasnip, Natalie A. W. Holzwarth, Diana Iuşan, Dominik B. Jochym, François Jollet, Daniel Jones, Georg Kresse, Klaus Koepernik, Emine Küçükbenli, Yaroslav O. Kvashnin, Inka L. M. Locht, Sven Lubeck, Martijn Marsman, Nicola Marzari, Ulrike Nitzsche, Lars Nordström, Taisuke Ozaki, Lorenzo Paulatto, Chris J. Pickard, Ward Poelmans, Matt I. J. Probert, Keith Refson, Manuel Richter, Gian-Marco Rignanese, Santanu Saha, Matthias Scheffler, Martin Schlipf, Karlheinz Schwarz, Sangeeta Sharma, Francesca Tavazza, Patrik Thunström, Alexandre Tkatchenko, Marc Torrent, David Vanderbilt, Michiel J. van Setten, Veronique Van Speybroeck, John M. Wills, Jonathan R. Yates, Guo-Xu Zhang, and Stefaan Cottenier. Reproducibility in density functional theory calculations of solids. *Science*, 351(6280):aad3000, 2016.

[52] Wiktor Pronobis, Alexandre Tkatchenko, and Klaus-Robert Müller. Many-body descriptors for predicting molecular properties with machine learning: Analysis of pairwise and three-body interactions in molecules. *Journal of Chemical Theory and Computation*, 14(6):2991–3003, 2018.

[53] Anders S. Christensen, Lars A. Bratholm, Felix A. Faber, and O. Anatole von Lilienfeld. FCHL revisited: Faster and more accurate quantum machine learning. *Journal of Chemical Physics*, 152(4):044107, 2020.

[54] Kristof T. Schütt, Huziel E. Sauceda, Pieter-Jan Kindermans, Alexandre Tkatchenko, and Klaus-Robert Müller. SchNet—a deep learning architecture for molecules and materials. *Journal of Chemical Physics*, 148(24):241722, 2018.

[55] Michael J. Willatt, Félix Musil, and Michele Ceriotti. Feature optimization for atomistic machine learning yields a data-driven construction of the periodic table of the elements. *Physical Chemistry Chemical Physics*, 20(47):29661–29668, 2018.

[56] Kristof T. Schütt, Michael Gastegger, Alexandre Tkatchenko, and Klaus-Robert Müller. Quantum-chemical insights from interpretable atomistic neural networks. In Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller, editors, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 311–330. Springer, 2019.

[57] Felix A. Faber, Luke Hutchison, Bing Huang, Justin Gilmer, Samuel S. Schoenholz, George E. Dahl, Oriol Vinyals, Steven Kearnes, Patrick F. Riley, and O. Anatole von Lilienfeld. Prediction errors of molecular machine learning models lower than hybrid DFT error. *Journal of Chemical Theory and Computation*, 13(11):5255–5264, 2017.