nature communications

Article

https://doi.org/10.1038/s41467-024-55449-7

Automatic feature selection and weighting in molecular systems using Differentiable Information Imbalance

Received: 21 May 2024

Accepted: 12 December 2024

Published online: 02 January 2025

Check for updates

Romina Wild[1,5], Felix Wodaczek[2,5], Vittorio Del Tatto[1,5], Bingqing Cheng[2,3] & Alessandro Laio[1,4]

Feature selection is essential in the analysis of molecular systems and many other fields, but several uncertainties remain: What is the optimal number of features for a simplified, interpretable model that retains essential information? How should features with different units be aligned, and how should their relative importance be weighted? Here, we introduce the Differentiable Information Imbalance (DII), an automated method to rank information content between sets of features. Using distances in a ground truth feature space, DII identifies a low-dimensional subset of features that best preserves these relationships. Each feature is scaled by a weight, which is optimized by minimizing the DII through gradient descent. This allows simultaneously performing unit alignment and relative importance scaling, while preserving interpretability. DII can also produce sparse solutions and determine the optimal size of the reduced feature space. We demonstrate the usefulness of this approach on two benchmark molecular problems: (1) identifying collective variables that describe conformations of a biomolecule, and (2) selecting features for training a machine-learning force field. These results show the potential of DII in addressing feature selection challenges and optimizing dimensionality in various applications. The method is available in the Python library DADApy.

Data sets are growing in number, in width, and in length. This abundance in data is generally used for two purposes: Predicting and understanding; likewise, feature selection has two essential aims: Model improvement and interpretability. Very often, most of the features defining a data point are redundant, irrelevant, or affected by large noise, and have to be discarded or combined, yet not many user-friendly, reliable feature selection packages exist. For predictive modeling, feature selection is an important preprocessing step, as it helps to prevent overfitting and increases performance and efficiency[1]. In a study on leukemia cancer, for example, it was demonstrated that the disease can be best identified using just 19 out of more than 7000 genes[2]. The other aim of feature selection is finding interpretable low-dimensional representations of high-dimensional or complex feature spaces[1], such as those generated by molecular dynamics (MD) simulations, or learned by neural networks[3], UMAP[4,5] or stochastic neighbor embedding methods[6]. For example, MD trajectories produce an enormous number of variables, yet within one graph one can only visualize the free energy landscape in two or three dimensions that are preferably interpretable[7]. In fields like finance and medicine, finding a small number of interpretable variables is especially important for understanding the mechanisms of stock markets[8] or diseases[9–11] and can improve predictions[12].

[1]International School for Advanced Studies (SISSA), Trieste, Italy. [2]The Institute of Science and Technology Austria (ISTA), Klosterneuburg, Austria. [3]Department of Chemistry, University of California, Berkeley, CA, USA. [4]The Abdus Salam International Centre for Theoretical Physics (ICTP), Trieste, Italy. [5]These authors contributed equally: Romina Wild, Felix Wodaczek, Vittorio Del Tatto. e-mail: laio@sissa.it

Feature selection methods can be broadly divided into wrapper, embedded, and filter methods[13]. Wrapper methods use a downstream task, such as a prediction, as the feature selection criterion, but suffer from combinatorial explosion problems. If the downstream task is akin to a classification problem, then embedded methods can perform well because they incorporate feature subset selection into the training[13]. These algorithms are often based on regression[14,15] or on support vector machines[16,17]. Filter methods, on the other hand, are independent of a downstream task and make use of a separate criterion to rank features. They are chosen if the downstream tasks cannot be modeled easily or involve several different models. While most wrapper and embedded methods are supervised by definition, filter methods include both, supervised and unsupervised formulations. Instead of using target data, unsupervised filter techniques exploit the topology of the original data manifold in various ways[18–22]. The classic supervised filters include correlation coefficient scores, mutual information[23], chi-square tests, and ANOVA methods[24], which are efficient but typically consider one feature at a time, resulting in selected subsets with redundant information[1]. Specific supervised feature subset evaluation filters like FOCUS rely on enumerating all possible subsets[25,26], similarly to wrapper methods, and they are affected by the same combinatorial problems. The relief algorithm and its variants[26,27] are more efficient as they do not explicitly evaluate the feature subsets. Instead, they employ nearest neighbor information to weight features, but the identified subsets can still include redundant features[26]. A review of feature selection filter methods can be found in ref. 28. Overall, the field of feature selection is clearly lacking the numerous powerful and out-of-the-box tools that are available in related fields such as dimensionality reduction.

The first, shared challenge in most of these feature selection approaches is related to the choice of the number of variables that are actually necessary to describe the system. A lower bound to such a number is provided by the intrinsic dimension[29], which is the dimension of the manifold containing the data. However, this number is often scale-dependent[30] and position-dependent[31]. Moreover, if one wants to visualize the data within a single graph, the number of variables is necessarily limited to two or three. One could show several low-dimensional projections of a high-dimensional distribution, but this comes at the cost of readability, and a single plot is often preferable. This typically implies neglecting part of the information, and poses the problem of choosing which variables should be retained for visualization.

A second complication arises when the variables are heterogeneous; in many cases, a data point is defined by features with different nature and units of measures, sometimes referred to as multi-view features[32]. For example, in atomistic simulations, one can describe a molecule in water solution by providing the value of all the distances between the atoms of the molecule, which are measured in nanometers, together with the number of hydrogen bonds that they form with the solvent, which are dimensionless. In the clinical context, the features associated with a patient may include blood exams, gene expression data, and many others[33]. In order to mix heterogeneous variables in a low-dimensional description, feature selection algorithms should enable the automatic learning of feature-specific weights to correct for units of measure[32] and information content[34].

In this work, we propose a feature selection filter algorithm which mitigates many of the aforementioned problems. Our approach aims to find a small subset of features that can best reproduce the neighbors of the data points based on a target feature space that is assumed to be fully informative. The algorithm finds, for each input feature, an optimal *weight* that accounts for different units of measure and different importance of the features. It also provides information on the optimal number of features.

The approach builds on a measure called Information Imbalance (Δ), which allows comparing the information content of distances in two feature spaces[35]. Informally, the Information Imbalance quantifies how well pairwise distances in the first space allow for predicting pairwise distances in the second space, in terms of a score between 0 (optimal prediction) and 1 (random prediction). This measure has been applied to find the most informative mix of containment measures for the COVID-19 pandemic[35], compare the information content of different machine learning interatomic potentials[36], assess the information content of chemical order parameters[37], measure the relative information content of Smooth Overlap of Atomic Orbitals (SOAP) descriptors[38], and recently, to infer the presence of causal links in high-dimensional time series[39]. In all these works, the distance space maximizing the prediction quality has been constructed by means of strategies including full combinatorial search of the optimal features[37], greedy search approaches[33], and grid search optimization of scaling parameters[39], with drawbacks related to the algorithm efficiency.

Here we make a major step forward by introducing the Differentiable Information Imbalance *DII*, which allows learning the most predictive feature weights by using gradient-based optimization techniques. The input feature space, as well as the ground truth feature space (targets, labels), can have any number of features. This provides a data analysis framework for feature selection where the optimal features and their weights are identified automatically. Moreover, carrying out the optimization with a sparsity constraint, such as L_1 regularization, allows finding representations of a data set formed by a small set of interpretable features. If the full input feature set is used as ground truth, then the approach can be used as an unsupervised feature selector, whereas it acts in a supervised fashion if a separate ground truth is employed. To our knowledge, there is no other feature selection filter algorithm implemented in any available software package which has above mentioned capabilities. The *DII* algorithm is publicly available in the Python package DADApy[40] and a comprehensive description can be found in the according documentation[41], which includes a dedicated tutorial.

In the following, we will first show the effectiveness of our method on artificial examples in which the optimal set of features is known. Then we move to a real-world application and show that our approach allows addressing one of the most important challenges in molecular modeling and solid state physics: Identifying the optimal set of collective variables (CVs) for describing the configuration space of a molecular system. As a second application, we use our method to select a subset of Atom Centered Symmetry Functions (ACSFs), descriptors of atomic environments, as input for a Behler-Parrinello machine learning potential[42], which learns energies and forces in systems of liquid water. In the same application, we show that SOAP[43,44] descriptors can be used as ground truth to choose informative subsets of ACSF descriptors.

Differentiable Information Imbalance

Given a data set where each point i can be expressed in terms of two feature vectors, $\mathbf{X}_i^A \in \mathbb{R}^{D_A}$ and $\mathbf{X}_i^B \in \mathbb{R}^{D_B}$ ($i = 1, \ldots, N$), the standard Information Imbalance $\Delta(d^A \to d^B)$ provides a measure of the prediction power which a distance built with features A carries about a distance built with features B. The Information Imbalance is proportional to the average distance rank according to d^B, restricted to the nearest neighbors according to d^A[35]:

$$\Delta\left(d^A \to d^B\right) := \frac{2}{N^2} \sum_{i,j:\, r_{ij}^A = 1} r_{ij}^B. \tag{1}$$

Here, r_{ij}^A (resp. r_{ij}^B) is the distance rank of data point j with respect to data point i according to the distance metric d^A (resp. d^B). For example, $r_{ij}^A = 7$ if j is the 7th neighbor of i according to d^A. $\Delta(d^A \to d^B)$ will be close to 0 if d^A is a good predictor of d^B, since the nearest neighbors according to d^A will be among the nearest neighbors according to d^B. If d^A provides no information about d^B, instead, the ranks r_{ij}^B in Eq. (1) will

be uniformly distributed between 1 and $N-1$, and $\Delta(d^A \to d^B)$ will be close to 1. As shown in ref. 39, the estimation of Eq. ((1)) can potentially be improved by considering k neighbors for each point. Considering d^B as the ground truth distance, the goal is identifying the best features in space A to minimize $\Delta(d^A \to d^B)$. If the features in A and the distances d^A are chosen in such a way that they depend on a set of variational parameters $\boldsymbol{w}$, finding the optimal feature space A requires optimizing $\Delta\left(d^A(\boldsymbol{w}) \to d^B\right)$ with respect to $\boldsymbol{w}$. However, Δ is defined as a conditional average of ranks, which cannot be minimized by standard gradient-based techniques.

Here we extend Eq. (1) to a differentiable version that we call Differentiable Information Imbalance (*DII*) in order to automatically learn the optimal distance $d^A(\boldsymbol{w})$. We approximate the non-differentiable, rank-dependent sum in Eq. (1) by introducing the softmax coefficients c_{ij}:

$$DII\left(d^A(\boldsymbol{w}) \to d^B\right) := \frac{2}{N^2} \sum_{\substack{i,j=1 \\ (j \neq i)}}^{N} c_{ij}(\lambda, d^A(\boldsymbol{w}))\, r_{ij}^B, \tag{2}$$

where

$$c_{ij}(\lambda, d^A(\boldsymbol{w})) := \frac{e^{-d_{ij}^A(\boldsymbol{w})/\lambda}}{\sum_{m(\neq i)} e^{-d_{im}^A(\boldsymbol{w})/\lambda}}. \tag{3}$$

The coefficients c_{ij} in Eq. (2) approximate the constraint $r_{ij}^A = 1$, such that $c_{ij} \to \delta_{1, r_{ij}^A}$ as $\lambda \to 0$ (δ denotes the Kronecker delta). Therefore, as illustrated in the tutorial `Differentiable Information Imbalance` in ref. 41, in the limit of small λ the *DII* converges to Δ:

$$\lim_{\lambda \to 0} DII\left(d^A(\boldsymbol{w}) \to d^B\right) = \Delta\left(d^A(\boldsymbol{w}) \to d^B\right). \tag{4}$$

For any positive and small λ, the quantity $DII(d^A \to d^B)$ can be seen as a continuous version of the Information Imbalance, where the coefficients c_{ij} assign, for each point i, a non-zero and exponentially decaying weight to points j ranked after the nearest neighbor in space d^A. The parameter λ is chosen according to the average and minimum nearest neighbor distances (see "Methods").

The *DII* is differentiable with respect to the parameters $\boldsymbol{w}$ for any distance d^A which is a differentiable function of $\boldsymbol{w}$. In this work, we assume that the variational parameters are weights, $\boldsymbol{w} = (w^1, \ldots, w^{D_A})$, scaling the features in space A as $\boldsymbol{w} \odot \mathbf{X}_i^A = (w^1 X_i^1, \ldots, w^{D_A} X_i^{D_A})$ (the symbol $\odot$ denotes the element-wise product). We construct $d^A(\boldsymbol{w})$ as the Euclidean distance between these scaled data points, $d_{ij}^A(\boldsymbol{w}) = \| \boldsymbol{w} \odot (\mathbf{X}_i^A - \mathbf{X}_j^A) \|$. In this case, the coefficients c_{ij} can be written as

$$c_{ij} = \frac{e^{-\left|\left|\boldsymbol{w} \odot \left(\mathbf{x}_i^A - \mathbf{x}_j^A\right)\right|\right|/\lambda}}{\sum_{m(\neq i)} e^{-\left|\left|\boldsymbol{w} \odot (\mathbf{x}_i^A - \mathbf{x}_m^A)\right|\right|/\lambda}}, \tag{5}$$

and the derivatives of $DII(d^A(\boldsymbol{w}) \to d^B)$ with respect to the parameters w^α can be computed:

$$\frac{\partial}{\partial w^\alpha} DII\left(d^A(\boldsymbol{w}) \to d^B\right) = \frac{2\, w^\alpha}{\lambda N^2} \sum_{\substack{i,j \\ (i \neq j)}} c_{ij}\, r_{ij}^B \left(-\frac{(X_i^\alpha - X_j^\alpha)^2}{\| \boldsymbol{w} \odot (\mathbf{X}_i^A - \mathbf{X}_j^A) \|} + \sum_{m(\neq i)} c_{im} \frac{(X_i^\alpha - X_m^\alpha)^2}{\| \boldsymbol{w} \odot (\mathbf{X}_i^A - \mathbf{X}_m^A) \|} \right). \tag{6}$$

These derivatives can be used in gradient-based methods to minimize the *DII* with respect to the variational weights.

If one aims at a low-dimensional representation of the feature space A, as in the case of feature selection, it is desirable that several of the weights are set to zero. While for up to $D_A \sim 10$ a full combinatorial search of all feature subsets can be carried out, optimizing the *DII* over each subset, for larger feature spaces a sparsification heuristic becomes necessary. We complement the *DII* optimization with two approaches for learning sparse features: Greedy backward selection and L_1 (lasso) regularization. Greedy selection removes one feature at a time from the full set, according to the lowest weight. L_1 regularization selects the subset of features that optimizes the *DII* while simultaneously keeping the L_1 norm of the weights small (see "Methods"). While greedy backward selection gives reliable results for up to ≈100 features, in larger feature spaces this algorithm becomes computationally demanding, and it is advisable to use L_1 regularization to find sparse solutions.

Results

Benchmarking the approach: Gaussian random variables and their monomials

We first test the *DII* approach using two illustrative examples where the distances $d^A(\boldsymbol{w})$ and d^B are built with the same features, so that the target weights minimizing Eq. (2) are known. In particular, we take as ground truth distance d^B the Euclidean distance in the space of the scaled data points $\boldsymbol{w}_{GT} \odot \mathbf{X}_i$, where the weights $\boldsymbol{w}_{GT}$ are fixed and known. We aim at recovering the target weights by scaling the unscaled input features, $\boldsymbol{w} \odot \mathbf{X}_i$, with the proposed *DII*-minimization.

In each example, we carry out several optimizations, both without any regularization term and in presence of a L_1 penalty, which induces sparsity in the learned weights. For each optimization, we employ a standard gradient descent algorithm, initializing the parameters $\boldsymbol{w}$ with the inverse of the features' standard deviations (see "Methods" for further details). In order to judge the quality of the recovered weights in the various settings, we calculate the cosine similarity between the vector of the optimized weights and $\boldsymbol{w}_{GT}$. This evaluation metric, which is bounded between 0 (minimum overlap) and 1 (maximum overlap), only depends on the relative angle between the two vectors, reflecting the fact that the *DII* allows to recover the target weights up to a uniform scaling factor ("Methods").

In the first example, we use a data set of 1500 points drawn from a 10-dimensional Gaussian with unit variance in each dimension, and we construct a ground truth distance d^B by assigning non-zero weights w_{GT}^α to all its 10 components (Table 1). The target weights w_{GT}^6 to w_{GT}^{10} are close to zero, such that these features carry almost no information.

The optimization without any L_1 regularization yields a very good result in terms of *DII* and overlap (blue in Fig. 1A I and II). If a soft L_1 regularization strength is employed, the results are qualitatively the same, but the irrelevant features $\alpha = 6$–10 receive zero weights, inducing sparsity and leading to an effective feature selection (green in Fig. 1A II). Table 1 shows the learned weights for different strengths of the L_1 penalty, scaled in such a way that the largest weight is identical to the largest component of $\boldsymbol{w}_{GT}$. Since in *DII* only the relative weights are important, this scaling is permissible and helps illustrating the comparison. By increasing the regularization strength, more features are set to zero following the order of their ground truth weights. When features of higher importance, namely with higher ground truth weights, are forced to zero by the regularization, then the resulting *DII* increases and the cosine similarity decreases, showcasing the loss of information (Fig. 1A II, Table 1).

Secondly, to test the method in a high-dimensional setting, we created a data set with 285 features including all the products up to order three of the 10 Gaussian random variables used in the previous example. Products of Gaussian random variables are distributed according to Meijer G-functions, which may not be Gaussian[45]. The ground truth distance d^B is here built by only selecting ten of these

https://doi.org/10.1038/s41467-024-55449-7

Table 1 | Ground truth weights, optimized weights, and optimization details for the 10 Gaussian random variables corresponding to Fig. 1A

Features										L_1 reg.	Nnz.	DII
X^1	X^2	X^3	X^4	X^5	X^6	X^7	X^8	X^9	X^{10}			
Ground truth weights												
5.0	2.0	1.0	1.0	0.5	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}			
DII optimized weights												
5.0	2.3	1.2	1.2	0.6	10^{-9}	10^{-12}	10^{-9}	10^{-9}	10^{-9}	None	10	0.003
5.0	2.1	1.1	1.1	0.5	0	0	0	0	0	0.0001	5	0.002
5.0	2.0	1.1	1.1	0	0	0	0	0	0	0.0002	4	0.005
5.0	0.5	0	0.2	0	0	0	0	0	0	0.0068	3	0.039
5.0	0.6	0	0	0	0	0	0	0	0	0.01	2	0.085

The feature space consists of ten independent and identically distributed Gaussian random variables, X^1–X^{10}. The same features are used as ground truth, but scaled. Optimized weights are shown at selected L_1 regularization strengths (L_1 reg.), and the resulting number of nonzero features (Nnz.) and Differentiable Information Imbalance (*DII*) are provided.

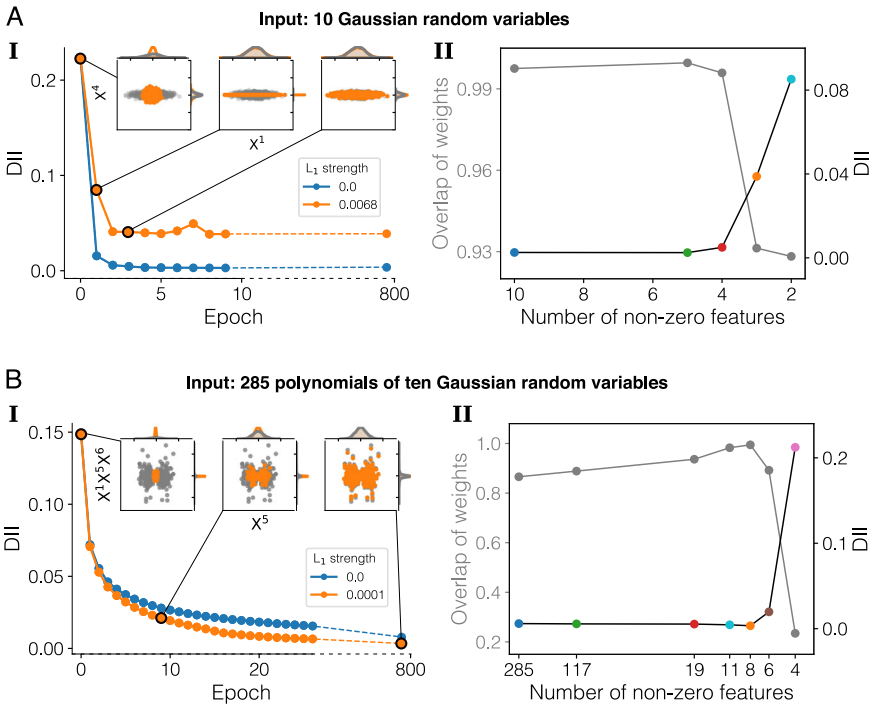

Fig. 1 | *DII* feature selection applied to Gaussian random variables and their monomials. A The input features are ten independent and identically distributed Gaussian random variables, X^1–X^{10}. The same features are used as ground truth, but scaled. **I** Differentiable Information Imbalance (*DII*), with (orange) and without (blue) L_1 regularization in the optimization. The insets show two exemplary features, with the weights during optimization (orange) and the ground truth weights (gray). **II** Cosine similarity (overlap) of the ground truth and optimized weights in gray, and *DII*s in black with colored markers, for several L_1 strengths and associated numbers of non-zero features. Table 1 provides the ground truth and optimized weights for points in this graph. **B** The feature space consists of the 285 monomials up to order three of the ten Gaussian random variables from (**A**). As ground truth, ten features were selected at random and scaled, while all the other feature weights are zero. **I**, **II** Analogous to (**A**). Table 2 provides the ground truth and optimized weights for points in this graph. Source data are provided as a Source Data file.

monomials, with various weights (Table 2). All other feature weights in the ground truth can be considered zero.

Since in this case the correct solution is very sparse in the full feature space, an appropriate sparsity-inducing regularization becomes essential to obtain good results. Without any L_1 regularization, all the 285 features receive a non-zero weight. Even if, in this case, the ground truth features are assigned the highest weights, there might not be a clear cut-off in the weight spectrum to distinguish them from the less-informative features.

As shown in Fig. 1B, the correct level of regularization can be identified by computing the *DII* as a function of the non-zero features or regularization strength. The intermediate L_1 strength of 0.0001 results in the best performance, as it coincides with the lowest *DII* and the largest weight overlap (orange in Fig. 1B I and II). The eight most relevant ground truth features are correctly identified, with an overlap between the learned and the ground truth weights which is remarkably close to 1.

Furthermore, panel I in Fig. 1B shows that weights found with L_1 regularization have a lower *DII* than the ones without L_1 regularization in the same optimization time, which means that the weights resulting from a certain level of regularization are effectively better than the unregulated ones. As in the previous example, when the regularization is too strong, some of the relevant features are discarded, resulting in a drop in the weight overlap and an increase in the *DII* (Fig. 1B II, Table 2).

https://doi.org/10.1038/s41467-024-55449-7

Table 2 | Ground truth weights, optimized weights and optimization details for the 285 monomials corresponding to Fig. 1B

Features											L_1 reg.	Nnz.	DII
X^5	$X^1X^5X^6$	X^3	$(X^2)^2$	X^6	X^{10}	X^1X^2	$X^8(X^{10})^2$	X^8	X^5X^8	Other			
Ground truth weights										**Sum**			
10.0	7.0	6.0	5.0	5.0	4.0	3.0	2.0	1. 0	1.0	0			
DIIoptimized weights													
10.0	6.2	6.9	2.9	6.1	5.1	2.2	2.5	0.8	0.7	72.4	None	285	0.006
10.0	6.2	6.9	2.9	6.1	5.1	2.2	2.4	0.8	0	53.0	4×10^{-6}	117	0.006
10.0	4.5	6.0	1.8	5.1	4.0	1.6	1.6	0	0	8.9	5×10^{-4}	19	0.005
10.0	6.2	6.5	3.7	5.6	4.5	3.1	2.1	0.7	0	2.6	5×10^{-5}	11	0.005
10.0	6.3	6.3	5.0	5.4	4.3	3.0	2.1	0	0	0	0.0001	8	0.003
10.0	6.9	6.2	0	5.7	3.1	0	0	0	0	3.8	0.0014	6	0.020
2.3	1. 1	1.4	0	0	0	0	0	0	0	10	0.0038	4	0.212
10.0	0	0	0	0	0	0	0	0	0	0	0.0023	1	0.606

The feature space consists of the 285 monomials up to order three of ten Gaussian random variables. As ground truth, ten features were selected at random and scaled, while all the other feature weights are zero. Optimized weights are shown at selected L_1 regularization strengths (L_1 reg.), and the resulting number of nonzero features (Nnz.) and Differentiable Information Imbalance (*DII*) are provided. The sum of the remaining 275 non-ground-truth weights is shown (Sum).

We then benchmarked the *DII* method against other feature selection methods. We perform the benchmark on the example with 285 monomials, in which the ground truth is known.

There are very few methods available in software packages which can be applied to the specific task we are considering, which is selecting and scaling features from a high-dimensional input space to be maximally informative about a multi-dimensional continuous ground truth, defining a pairwise *distance*. Considering filter methods, we compare *DII* to relief-based algorithms (RBAs), specifically RReliefF and MultiSURF, implemented in scikit-rebate[46], which support a continuous ground truth[26]. RBAs are filter methods that weight features, but importantly only work with a one-dimensional ground truth. This poses a problem for all use cases in this paper because the ground truth is always defined by the multi-dimensional vector of features used to compute the target distance. RBAs extended to the multi-label case[47,48] but, to our knowledge, are not implemented in software packages. We apply scikit-rebate RReliefF and MultiSURF for each ground truth dimension individually, and sum the resulting weights with and without prior importance cutoff (see Supplementary Information). The methods detect the most important input feature in most cases, leading to overall cosine similarities ranging from 0.56 to 0.84 for the various settings (Supplementary Fig. 1).

As a second benchmark we use a method from scikit-learn[49], which can handle the task's requirements: The decision tree regressor (`sklearn.tree.DecisionTreeRegressor`). Unlike *DII* and the RBAs, this method is not a filter but an embedded method. The feature selection is determined as a side product during the building of a regressor model. There is no filter algorithm implemented in scikit-learn which can solve a problem as posed here. Two distinct feature importance measures implemented with the approach, the Gini importance and the Permutation importance, lead to feature vectors with a cosine similarity of up to 0.83 with respect to the ground truth. In comparison, the *DII* method with a L_1regularization of 0.0001 (orange in Fig. 1B) finds a weight vector with eight non-zero weights and a cosine similarity of 0.99.

In conclusion, in both examples the *DII* method is able to recover the ground truth weights with good accuracy, and better than the very few other applicable methods, as measured by a larger weight overlap with the ground truth. In the following sections, we apply our feature selection method to cases in which the optimal solution is not known and illustrate how our approach can be used to give an explicit system description by extracting few features from a larger data set.

Identifying the optimal collective variables for describing a free energy landscape of a small peptide

We now illustrate how the *DII* can be used to identify the most informative CVs to describe the free energy landscape of a biomolecule. As opposed to the previous example, in this test the ground truth variables and the input variables are different sets.

We consider a temperature replica-exchange MD simulation (400 ns, 340 K replica analyzed only, $dt = 2$ fs)[50] of the 10-residue peptide CLN025[51], which folds into a β-hairpin. The data set is composed of 1429 frames (subsampled from 41,580 trajectory frames) containing all atom coordinates. The ground truth metric d^B is constructed in the feature space of all the 4278 pairwise distances between the 93 heavy atoms of the peptide, which can be assumed to hold the full conformational information of the system. We consider a feature space A with ten classical CVs that do not depend on knowledge of the folded state of the β-hairpin peptide: Radius of gyration (RGYR), anti-β-sheet content, α-helical content, number of hydrophobic contacts, principal component 1 (PC1), principal component 2 (PC2), principal component residuals, the number of hydrogen bonds in the backbone, in the side chains, and between the backbone and side chains ("Methods").

Since the CV feature space is only 10-dimensional, it is possible to look for the optimal distance d^A by an exhaustive search of all possible 1023 subsets containing one to ten CVs, without using the L_1 regularization to produce sparse solutions. For each subset of CVs, the *DII* is used as a loss to automatically optimize the scaling weights, which are initialized to the inverse standard deviations of the corresponding variables. Even when all feature subsets can be constructed, gradient descent optimization of the *DII* is useful, as the most naive choices of the scaling weights—setting them to the inverse standard deviations of the variables, or all equal to 1—likely define suboptimal distances, since the CVs have different units of measure and importance. The optimization of the feature weights for all 1023 subsets takes about 4.5 h on a CentOS Linux 7 with 24 CPUs Intel Xeon E5-2690 (2.60 GHz) with 15 GB RAM using the function "return_weights_optimize_dii" with 80 epochs (Fig. 2A, green curve).

Figure 2A shows the results of the subset optimizations by computing the *DII* with block cross-validation (see "Methods"). The training and validation *DII*s averaged over all cross-validation splits, show a high degree of consistency, verifying the transferability of the *DII* results between non-overlapping pieces of the trajectory. As shown in the inset graph in Fig. 2A, the *DII* result improves during the gradient descent optimization. The best single CV is anti-β-sheet content, while

https://doi.org/10.1038/s41467-024-55449-7

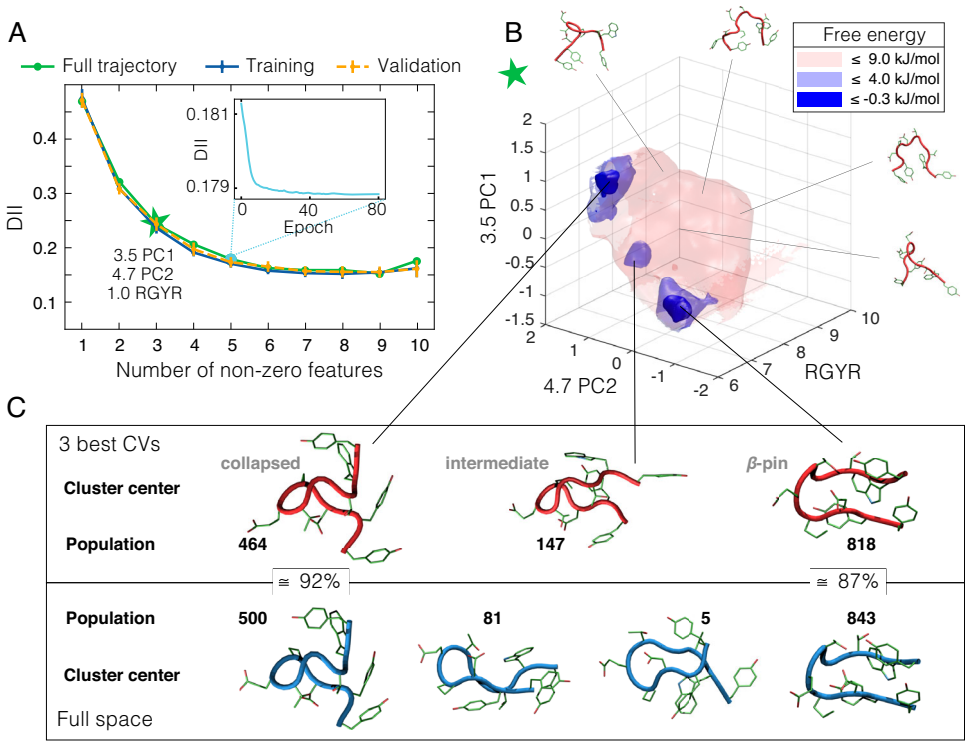

Fig. 2 | *DII* feature selection for describing the free energy landscape and conformations of CLN025. A Green: Optimal Differentiable Information Imbalance (*DII*) results for collective variable (CV) subsets of different sizes with gradient descent optimized weights for 1429 data points evenly sampled from the full trajectory. The green star marks the *DII* result of the optimally scaled 3-plet, which defines the coordinate system for (**B**). Inset: DII gradient descent optimization for the optimal 5-plet. Blue and orange: Average and standard deviations of the *DII* calculated from block cross validation with 4 non-overlapping training data sets and 84 validation sets of 1428 points each. **B** Free energy isosurfaces in the space of the optimal 3-plet of CVs (radius of gyration (RGYR), principal components 1 and 2 (PC1 and PC2), with weights of 1.0, 3.5, and 4.7), corresponding to three different values of the free energy. The renderings around the free energy surfaces show sampled conformations of the peptide at different values of the CVs and free energy. **C** Red and blue renderings are cluster centers obtained from the optimal 3-plet space and from the full space of all pairwise heavy atom distances, respectively. The two main cluster centers of both belong to the dominant peptide conformations: The β-pin and the collapsed denatured state. The collapsed and β-pin clusters identified in the optimal 3-plet space share 92% and 87% of the frames with the corresponding full space clusters. Source data are provided as a Source Data file.

the best triplet contains RGYR, PC1, and PC2 with weights of 1.0, 3.5, and 4.7. Remarkably, the weight of PC2 is higher than the weight of PC1, confirming that the gradient optimization of the *DII* provides non-trivial results. We estimated the density in the space of the best three scaled variables (Fig. 2B) using point-adaptive k-NN (PAk)[52], implemented in the DADApy package[40]. The free energy derived from this density clearly shows two favorable main states, which are the folded β-hairpin state and a denatured collapsed state[53] with negative values of the free energy in Fig. 2B.

The cluster centers found by Density Peak Clustering in its unsupervised extension[54] are depicted by the renderings denoted "collapsed", "intermediate", and "β-pin" in Fig. 2C, while additional example structures from less favorable free energy regions are shown around Fig. 2B. The clustering was also performed in the full space of all 4,278 heavy atom distances, which holds the full information of the system.

The populations of both, β-pin and collapsed clusters show a remarkable overlap between the clustering structures obtained in the optimal 3-plet case and from the full feature space of 4,278 heavy atom distances. Taking the cluster populations from the full space as ground truth classes, such overlap can be simply measured as the fraction of points (trajectory frames) that belong to the same cluster in both representations, also referred to as cluster purity[55]: The β-hairpin cluster from the 3-plet space has 87% purity, and the collapsed state cluster has 92% purity, considering the full space as reference. Taken together, all clusters have a 89% overall cluster purity towards the full space clusters. This consistency also emerges by visually comparing the red and blue renderings of the two dominant cluster centers (left and right structures in Fig. 2C). As a comparison, running the clustering algorithm using the single best CV, the anti-β-sheet content, brings to an overall cluster purity of 45%, i.e., the trajectory frames clustered into the pin, collapsed, or other clusters using the single best variable, capture 45% of the same frames of the according clusters using the full space for clustering. Hence, no single one-dimensional CV is informative enough to describe CLN025 well, but a combination of only three scaled CVs carries enough information to achieve an accurate description of this system.

Because of the good performance of decision tree regression on the previous example and its ability to handle multi-target (even high-dimensional), continuous ground truth data, we apply this feature selection algorithm also to this use case (Supplementary Information). The best three variables using the Gini importance weights are: 0.29 anti-β-sheet content, 0.25 PC1, 0.1 PC2; using the permutation importance they are: 1.27 PC1, 1.04 anti-β-sheet content, 0.97 PC2. Clustering in these reduced spaces leads to maximum cluster purities compared to the full space clusters of 55% for Gini importance and 63% for the permutation importance and several additional inconsistencies when compared to the full space clustering (Supplementary Fig. S2 and Supplementary text).

We also test the robustness of the method using four uncorrelated trajectory blocks and performing the *DII*-optimization in each of these blocks. The resulting *DII*s, as well as selected features and their weights, show excellent consistency across the blocks (Supplementary Figs. S4 and S5).

https://doi.org/10.1038/s41467-024-55449-7

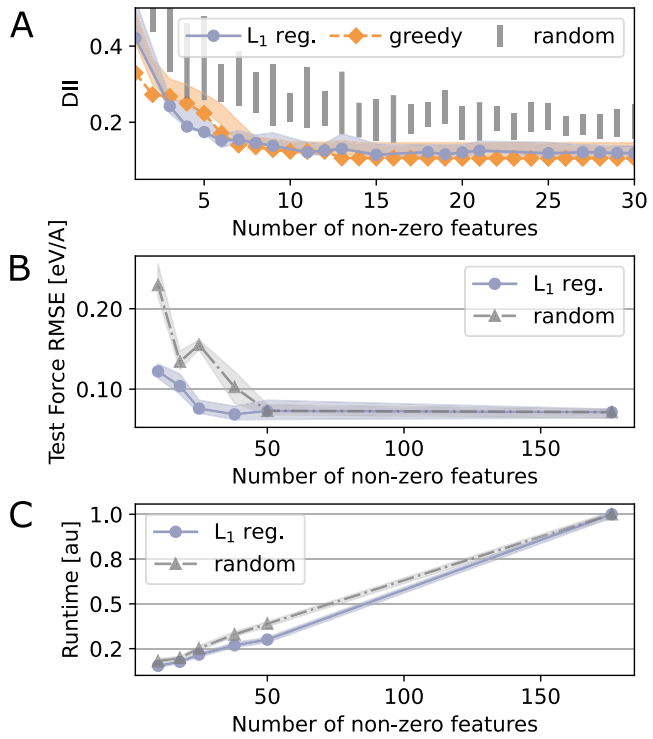

Fig. 3 | *DII* feature selection for efficient training of a Machine Learning Potential (MLP). A Differentiable Information Imbalance (*DII*) selecting the optimal feature subsets from $D_A = 176$ Atom Centered Symmetry Functions (ACSF) descriptors, against a ground truth of $D_B = 546$ Smooth Overlap of Atomic Orbitals (SOAP) descriptors, using a data set of $N \sim 350$ atomic environments. The optimized *DII* per number of non-zero features is shown by blue circles and orange diamonds, using L_1 regularized search and greedy backward selection, respectively. The filled area represents validation data in the form of the minimum and maximum *DII* on 10 batches of ~350 atomic environments other than the ~350 environments used for *DII* feature selection. The *DII* for randomly selecting a certain number of non-zero features is depicted as gray bars between the lowest and highest *DII* found within 10 random selections. **B** Test root-mean-square error (RMSE) with features chosen via L_1 regularized *DII* (blue circles) and at random (gray triangles) by Behler-Parrinello-type MLPs[42] as implemented in n2p2[79,80]. Six MLPs with different train-test splits per number of non-zero features are trained. Markers represent their average RMSE, the filled area shows the range from worst to best performer. **C** Run-time of force and energy prediction on a single structure performed by the same MLPs as in (**B**). The filled area shows the range from worst to best performer, despite being barely visible due to similar run-times across the six MLPs. Source data are provided as a Source Data file.

Feature selection for machine learning potentials

In another use case of the *DII* approach, we demonstrate its capabilities for selecting features for training Behler-Parrinello machine learning potentials (MLPs)[42]. MLPs can learn energy and forces of atomic configurations derived from quantum mechanical calculations. The Behler-Parrinello MLP uses ACSF as inputs for the predictions[56]. The ACSFs are a large set of radial and angular distribution functions, which describe the environment around an atom, and are permutationally, rotationally, and translationally invariant.

The data set used here consists of $N \sim 350$ atomic environments of liquid water molecules, derived from a larger data set that has previously been used to fit a MLP, which can accurately predict various physical properties of water[57]. The input features in this example are 176 ACSF descriptors (see "Methods"). The ACSF descriptor dimensions combinatorially grow with the number of atom types, which makes them computationally costly and makes feature selection attractive[58]. Since the ACSF space is too large for full combinatoric feature selection, we search for sparse solutions using both L_1 regularized *DII* and greedy backward selection ("L_1 reg." and "greedy" in Fig. 3, see "Methods"). We aim to select informative ACSFs before the training to reduce the number of input features and thus reduce the training and prediction time.

While *DII* can be used as an unsupervised feature selector when the feature space is reduced against itself as ground truth, it can also incorporate a separate feature space as ground truth in a supervised fashion. This is especially useful when a comprehensive ground truth exists. In the case of atomic environments, one of the most complete, accurate, and robust descriptions is given by the SOAP descriptors[43,44], based on the expansion of the local density in spherical harmonics. 546 SOAP features ($n_{max} = 6$, $l_{max} = 6$) are defined as the ground truth for feature selection. In this manner, we can put SOAP and ACSF, two comprehensive representations of atomic environments, into relation[59] and show that SOAP is a suitable ground truth to select informative ACSFs as inputs for a MLP. The SOAP space captures the full spacial arrangement of atoms by encoding the local atomic densities and accounting for symmetries[44]. Both SOAP and ACSF descriptor spaces, as well as further local atomic density descriptors, such as the atomic cluster expansion (ACE) representation, have been shown to be compressible without significant loss of information, improving computational efficiency[60,61].

The resulting *DII* for various numbers of ACSFs can be seen in Fig. 3A. With both greedy and L_1 regularized selection, we find that the optimized *DII* asymptotically approaches an optimal value with growing number of non-zero features. However, even relatively small feature spaces with ~10–30 non-zero features have low *DII* values, making effective feature selection possible. We validate the selected features and their weights on validation sets of atomic environments of equal size as the training set. The resulting *DII*s are slightly higher but mostly comparable to the training *DII*s, showcasing the robustness and transferability of the results. As a sanity check for our selection, we also show that randomly selected feature sets have a significantly higher *DII* than optimized sets, meaning they are less informative about the ground truth space (Fig. 3A gray).

To show that the features selected by *DII* are indeed physically relevant, we report in Fig. 3B the root-mean-square error of atomic forces for Behler-Parrinello MLPs using ACSF subsets of different sizes ($n_{ACSF} \in \{10, 18, 25, 38, 50, 176\}$). We find that MLPs with features selected by L_1 regularized *DII* optimization outperform random input features for all tested numbers of input features n_{ACSF}. The difference in prediction accuracy is most pronounced at small n_{ACSF}, where it is least likely that random selection chooses meaningful features. After $n_{ACSF} \approx 20$ input features, the optimized subsets reach an accuracy of <100 meV, which is on par with the original MLP trained on these data[57]. Compressions of local atomic density representations for machine learning potentials have also previously been shown to require a minimum set size of 10–20 PCA features, since further compression fails to faithfully preserve the geometric relationships between data points and leads to increased prediction errors[62]. With $n_{ACSF} = 50$ input features, the MLP performs roughly equally well to using the full data set, while having less than half the run-time (Fig. 3C). This shows that *DII* can be used to select features for downstream tasks such as energy and force fitting in MLPs, by optimizing for a complex ground truth and finding a space with fewer but optimally weighted features that contain the same information.

Discussion

This work presents the Differentiable Information Imbalance, *DII*, designed to automatically learn the optimal distance metric d^A over a set of input features. The metric reproduces the neighborhoods of the data points as faithfully as possible according to a ground truth distance d^B. Here d^A is defined as the Euclidean distance, and the optimization parameters are weights that scale individual features, such

that the presented *DII* is an automatic and universal feature selection and weighting algorithm.

While many other methods are restricted to single variable outputs as "labels" or "targets", *DII* can handle any dimensionality of input and output. Continuous and discrete data is supported and the method can be used in a supervised and unsupervised manner. The weights are optimized automatically, and by using the values of the *DII* as a quality measure one can compare the information content of several feature sets, and select the sets corresponding to the lowest *DII* for each number of features, such as in Supplementary Fig. S6. It is one of very few filter methods that account for feature dependencies but do not rely on explicit feature subset evaluation[26].

In illustrative examples where the optimal feature weights are known, we showed that the *DII* can reliably find the correctly weighted ground truth features out of high-dimensional input spaces. The behavior of the *DII* as a function of the subset size appears to be anti-correlated with the cosine similarity between ground truth and optimized weights. This implies that the *DII* value can be used for assessing the quality of the selected feature subsets when the actual ground truth weights are unknown. The weighted feature sets as provided by *DII* optimization have a higher cosine similarity to the ground truth than sets derived from two other feature selection classes, RBAs[46] and decision tree regressions.

We further applied the method to analyze a MD simulation of a biomolecular system. Extracting a small subset of informative CVs from a pool of many candidate CVs from a MD trajectory is a general problem with both practical and conceptual benefits, including using such CVs in enhanced sampling techniques and obtaining an interpretable description of the free energy landscape. For the peptide CLN025, the selected CVs are the first two principle components (3.5 PC1, 4.7 PC2) and the radius of gyration (1.0 RGYR). Applying clustering in the space of these three scaled CVs leads to the correct identification of the β-pin state and collapsed denatured state of CLN025, in accordance with the clusters built from a much larger feature space, which includes all heavy atom distances. The reduced space clusters are highly meaningful with a 89% overall cluster purity towards the extended space clusters, while reduced variable spaces built from clustering results of the decision tree regression lead to lower cluster purities. Tests of uncorrelated parts of the MD trajectory show great consistency of the results, highlighting the robustness of the method.

In a second application, our method successfully selects highly informative subsets of input features for training a Behler-Parrinello machine learning potential that achieves optimal performance in terms of the mean absolute error of force and energy. We find that using just 50 informative ACSF descriptors selected by our approach, instead of 176, significantly reduces the MLP's computational cost, cutting the runtime by one third while maintaining nearly the same accuracy.

The *DII* is not necessarily a simple monotonic function of the number of non-zero features post-optimization (cardinality). In some cases, the selection of additional features can introduce noise or redundancies that can negatively impact the description of the ground-truth space. Furthermore, if the optimal non-zero features for a two-dimensional description are, say, X^3 and X^{61}, the optimal features for a three dimensional description could be completely different, say X^5, X^9, and X^{44}. The *DII* is hence also not necessarily a submodular function of the number of features.

To extract small subsets of features from high dimensional input data, we implemented two different sparsity inducing heuristics: L_1 regularization and greedy backward optimization. Greedy algorithms have previously been shown to be a fast and effective alternative to convex L_1 regularization in sparse coding[63], and work even if the problem is only approximately submodular[64]. When a feature space is very large, greedy backward optimization will lead to long calculations and L_1 regularization becomes more suitable. Both heuristics are able to find relevant results in the examples presented here.

Like RBAs[26], also *DII* has a computational complexity of $\mathcal{O}(N^2 \cdot D)$, where N is the number of points and D is the number of features. However, by applying a simple subsampling trick (see "Methods"), the computational complexity reduces up to $\mathcal{O}(N \cdot D)$ with a degradation of the accuracy which is barely detectable (Supplementary Fig. S3).

The requirement of a ground truth reference space could pose a difficulty to some applications. In MD simulations, all heavy atom distances are a good, translationally invariant alternative to the set of all atomic positions, if one wants to completely encode the conformation of a molecule. In other cases, if no independent ground truth is known or a lower-dimensional subspace is desired, the full space could be used as ground truth. This approach could be employed, for example, for large gene sequencing data with thousands of features and just hundreds of data points. In this fashion, the method acts as an unsupervised feature selection filter. An open question in this case is the relative weighting of the ground truth features.

Furthermore, even though the method can be applied to any data set, it is most suitable for continuous features. A limitation is given by ground truth metrics with many nominal or binary features, which can lead to a degenerate ground truth rank matrix, making the optimization more difficult.

The Differentiable Information Imbalance introduced in this work could have relevant implications in a wide range of distance-based methods, such as k-NN classification, clustering, and information retrieval. The approach could also be used to identify how much information original features carry compared to otherwise not-interpretable transformations such as UMAP[4] or highly non-linear neural network representations, by optimizing the original features towards such representations. Defining a new feature by combining several input features through a (possibly nonlinear) function might bring to even more compressed and informative representations, although this could reduce interpretability. *DII* has also potential applications beyond feature selection with automatic weighting. Specifically, constructing a distance space $d^A(\boldsymbol{w})$ with a more expressive functional form, compared to the one used in this work, opens up to applications in fields such as dimensionality reduction[65,66] and metric learning[67].

The Differentiable Information Imbalance has been implemented in the Python library DADApy[40] and is well-documented[41], including a tutorial for ease of use. This accessibility allows for a wide audience to explore further use cases and limitations effectively.

Methods

Adaptive softmax scaling factor λ

Qualitatively, the scaling factor λ in the softmax coefficient $c_{ij}(\lambda, \boldsymbol{w})$ defines the size of the neighborhoods in the input space $d^A(\boldsymbol{w})$ used for the rank estimation. Since λ is the same for every data point, regardless of whether the point is an outlier or within a dense cloud, this factor mainly decides how many neighbors are included in dense regions of the data manifold. Importantly, choosing λ too small makes the optimization less efficient, as in the limit $\lambda \to 0$ the derivative of the *DII* (see Eq. (6)) can be shown to vanish for almost all values of the parameters $\boldsymbol{w}$.

To automatically set λ, we take the average of two distance variables, $\hat{d}^A_{\min}$ and $\hat{d}^A_{\text{avg}}$, which heuristically define the "small distance" scale in space d^A. Both of these numbers are based on $\hat{d}^A_i$, here denoting the difference between 2nd and 1st nearest neighbor distances for each data point i, $\hat{d}^A_i = d^A_{ik} - d^A_{ij}$, where $r^A_{ij} = 1$ and $r^A_{ik} = 2$:

$$\hat{d}^A_{\min} := \min_i \hat{d}^A_i, \tag{7a}$$

$$\hat{d}^A_{\text{avg}} := \frac{1}{N}\sum_i \hat{d}^A_i \,. \tag{7b}$$

Setting λ to the average of $\hat{d}^A_{\min}$ and $\hat{d}^A_{\text{avg}}$ at each step of the *DII* optimization has proven to enhance both the speed and stability of convergence. Indeed, using differences between nearest neighbor points to determine λ is more robust than using nearest neighbor distances directly, as in high dimensions first-, second-, and higher-order neighbor distances tend to be very similar on a relative scale[68,69].

Invariance property of the *DII*

In the limit $\lambda \to 0$, the *DII* defined in Eq. (2) is invariant under any global scaling of the distances in space A, $d^A_{ij} \mapsto |c|\, d^A_{ij}$ with $c \in \mathbb{R}$. Similarly, in the small λ regime, $DII(d^A(\boldsymbol{w}) \to d^B)$ is invariant under any uniform scaling of the weight vector, $\boldsymbol{w} \mapsto c\,\boldsymbol{w}$, if $d^A(\boldsymbol{w})$ is built as the usual Euclidean distance in the scaled feature space. This property can be easily verified by observing that the softmax coefficients c_{ij} can be replaced by $\delta_{1,r^A_{ij}}$ when $\lambda \to 0$, and the ranks r^A_{ij} are invariant under a global scaling of the distances d^A_{ij}. The same invariance holds even for $\lambda > 0$ if λ is chosen adaptively ("Methods"), as in the adaptive scheme a global scaling of the distances d^A_{ij} implies a scaling of λ by the same factor, which leaves the c_{ij} coefficients untouched.

Optimization of the *DII*

The optimization of the *DII* is implemented in `FeatureWeighting.return_weights_optimize_dii` in DADApy by gradient descent utilizing the analytic derivative of the *DII*. The default value of the initial feature weights is the inverse standard deviation of each feature. Pseudocodes of the *DII* optimization algorithms are provided in the Supplementary Information (section "Pseudocodes").

Learning rate decay. We employ two different schemes of learning rate decay, (1) cosine learning rate decay and (2) exponential learning rate decay. When both schemes are evaluated, we select the solution with lower *DII* among those found with the two schemes. In the first scheme, the learning rate is updated according to $\eta^k = 0.5\eta^0 \cdot (1 + \cos(\frac{\pi k}{n_{\text{epochs}}}))$, where k denotes the training epoch, η^0 the initial learning rate, and n_{epochs} the total number of epochs in the training. The exponential decay follows $\eta^k = \eta^0 \cdot 2^{\frac{-k}{10}}$. This schedule cuts the learning rate by half every 10 epochs. While the cosine decay leads to optimal results in the absence of L_1 regularization, or for weak regularization, the exponentially decaying learning rate is especially suited for high L_1 regularization. In both schemes, "GD clipping" is used, as described hereafter in the section on L_1 regularization.

L_1 regularization. This method is implemented in DADApy in `FeatureWeighting.return_weights_optimize_dii` when a L_1 penalty different from 0 is chosen, and several different L_1 values are screened in `FeatureWeighting.return_lasso_optimization_dii_search`. Optimizing the *DII* with respect to the feature weights while simultaneously introducing sparsity, i.e., limiting the number of features used, can be considered a convex optimization problem of the form:

$$\min_{\boldsymbol{w}\in\mathbb{R}^D}\left(f(\boldsymbol{w}) + p\,\Omega(\boldsymbol{w})\right), \tag{8}$$

where $f : \mathbb{R}^D \to \mathbb{R}$ is a differentiable function such as $DII\left(d^A(\boldsymbol{w}) \to d^B\right)$, at least locally convex, and $\Omega : \mathbb{R}^D \to \mathbb{R}$ is a sparsity-inducing, non-smooth, and non-Euclidean norm with penalization strength p[70]. We use the L_1 norm, $\Omega(\boldsymbol{w}) = \sum_{\alpha=1}^D |w^\alpha|$ (also called lasso regularization):

$$\min_{\boldsymbol{w}\in\mathbb{R}^D}\left(DII + p\,\Omega(\boldsymbol{w})\right) = \min_{\boldsymbol{w}\in\mathbb{R}^D}\left(\frac{2}{N^2}\sum_{\substack{i,j=1\\(j\neq i)}}^{N} c_{ij}(\lambda, d^A(\boldsymbol{w}))\, r^B_{ij} + p\sum_{\alpha=1}^{D}|w^\alpha|\right) \tag{9}$$

The L_1 norm has the shortcoming that in $N \ll D$ setting, with very few samples but many dimensions, a maximum of N variables can be selected. The L_1 regularization tends to select just one variable from a group of correlated variables and ignore the others[71], which helps building optimal groups of maximally uncorrelated features (see Supplementary Information in ref. 33).

Naive gradient descent with L_1 regularization usually does not produce sparse solutions, as a weight becomes zero only when it falls directly onto zero during the optimization[72]. This is very unlikely with most learning rate regimes. Instead, we employ the two-step weight updating approach also known as "GD clipping"[72]:

$$\begin{aligned} w^\alpha_{t+\frac{1}{2}} &= w^\alpha_t - \frac{\partial DII\left(d^A(\boldsymbol{w}) \to d^B\right)}{\partial w^\alpha} \\ &\textbf{if } w^\alpha_{t+\frac{1}{2}} > 0 \textbf{ then } w^\alpha_{t+1} = \max\left(0, w^\alpha_{t+\frac{1}{2}} - \eta\, p\right) \\ &\textbf{if } w^\alpha_{t+\frac{1}{2}} < 0 \textbf{ then } w^\alpha_{t+1} = \left|\min\left(0, w^\alpha_{t+\frac{1}{2}} + \eta\, p\right)\right| \end{aligned} \tag{10}$$

Here, p denotes the L_1 penalty strength, and t is the epoch index. First, the update is performed only with the GD term, which may result in a change of sign for the weight. Subsequently, the L_1 term is applied, shrinking the weight magnitude. If this shrinkage would change the weight's sign, the weight is instead set to zero. Since the *DII* is sign invariant, all weights are kept positive during the optimization.

Backward greedy optimization. This approach is implemented in DADApy in `FeatureWeighting.return_backward_greedy_dii_elimination`. It starts with a standard optimization run using all the D_A features of the input space. From the solution of the first optimization, the feature corresponding to the smallest weight is discarded (set to zero), and a new optimization with $D_A - 1$ features is carried out. This procedure is iterated until the single most informative feature is left. The greedy backward approach is an alternative to the L_1 regularization and is applicable to moderately large data sets with $D_A \lesssim 100$ features and $N \lesssim 500$ data points, since the computational complexity scales linearly with the number of features.

A linear scaling estimator of the *DII*

The *DII* scales quadratically with the number of points N, with a computational complexity of $\mathcal{O}(N^2 \cdot D)$, where D is the number of features.

The computational time can be dramatically decreased by subsampling the rows of the matrices r_{ij}, d_{ij} and c_{ij} appearing in Eq. (2), reducing them to a rectangular shape $N_{\text{rows}} \times N$ (with $N_{\text{rows}} < N$) (see Supplementary Information "Tests of scalability and robustness"). This subsampling is performed only once at the beginning of the training, so that the rectangular shape of such matrices is kept fixed during all the *DII* optimization. If the *DII* is written as the average of N conditional ranks,

$$DII\left(d^A(\boldsymbol{w}) \to d^B\right) = \frac{2}{N}\frac{1}{N}\sum_{i=1}^{N}\left(\sum_{\substack{j=1\\(j\neq i)}}^{N} c_{ij}(\lambda, d^A(\boldsymbol{w}))\, r^B_{ij}\right) = \frac{2}{N}\langle r^B | r^A \approx 1\rangle\,, \tag{11}$$

the subsampling is equivalent to replacing $1/N\sum_{i=1}^{N}$ with $1/N_{\text{rows}}\sum_{i=1}^{N_{\text{rows}}}$. This means computing the average of N_{rows} conditional ranks instead of N. Different schemes to set N_{rows} result in different scaling laws of the algorithm with respect to N. Setting N_{rows} to a fraction of N (green curve in Supplementary Fig. S3A, $N_{\text{rows}} = N/2$) brings to a quadratic scaling with a smaller prefactor, while sampling a fixed number of points N_{rows} independently of N (red curve, $N_{\text{rows}} = 100$) brings to a linear scaling $\mathcal{O}(N \cdot D)$. In the latter case we observe a striking reduction of the runtime, while the accuracy of the recovered weights is almost perfectly preserved (Supplementary Fig. S3B).

Extraction of collective variables from the CLN025 MD simulation

All CVs were extracted from the MD simulation using PLUMED 2[73]. The ground truth pairwise heavy atom distances were computed using the "DISTANCE" CV on all pairs of non-hydrogen atoms. The RGYR was obtained with the "GYRATION" CV and the C_α atoms. The number of hydrophobic contacts were calculated using the "COORDINATION" CV ($R_0 = 0.45$) and using the amino acids THR, TRP, and TYR of CLN025, and sidechain carbons not directly bonded with an electronegative atom. The number of hydrogen bonds was also calculated using the "COORDINATION" CV ($R_0 = 0.25$). For backbone H-bonds and sidechain H-bonds only hydrogens and oxygens of the backbone and the sidechain were considered, respectively, while for the sidechain-to-backbone interactions, the cross of these were considered. For the quantification of the alpha-helical content and the anti-parallel beta sheet content, the CVs "ALPHARMSD" and "ANTIBETARMSD" were used with all residues of the peptide. For the principle components PC1, PC2, and the PCA residual, first a pdb file containing the average structure of the trajectory and the two first principle directions was created using the CVs "COLLECT_FRAMES_ATOMS" with all heavy atoms, and "PCA" using the previous output and optimal alignment. Subsequently, each frame of the trajectory was projected onto the two principle components referenced in the pdb file using "PCAVARS".

Block cross validation of CLN025

To account for the equilibration of the system, the first ~15 ns of the trajectory were discarded throughout the analysis (1580 of 41,580 trajectory frames). Block cross validation (Fig. 2A) was carried out by splitting the remaining frames into 4 consecutive blocks. The training blocks were built by subsampling each block to every 7th frame to decorrelate, leaving 1428 points per training block. The optimal tuple and weight results from each training block were used to calculate the *DII* in 21 validation sets built from the remaining three blocks (repeatedly subsampling each block with stride 7, starting from frames 1 to 7), totaling 84 validation sets.

ACSF and SOAP descriptors

The systems for creating ACSF and SOAP descriptors are based on 1593 liquid H_2O structures whose forces and energies were found using DFT via the CP2K[74] package with the revPBE0-D3 functional. We use the DScribe Python package[75,76] to calculate SOAP and ACSF descriptors from the atomic positions. The data points were chosen as follows: The 1593 structures (with 64 H_2O molecules each) yielded 192,000 atomic environments, from which a subset of ~350 was sampled to reduce the computational time of feature selection. The ACSF descriptors were constructed on a grid of hyperparameters (G2: $\eta \in [10^{-3}, 10^{0.5}]$ logspace $n_\eta = 15$, $R_S = 0$, G4: $\eta \in [10^{-3}, 10^{0.5}]$ logspace $n_\eta = 6$, $\zeta \in \{1, 4\}$, $\lambda \in [-1, 1]$ linspace $n_\lambda = 4$, $R_S = 0$), resulting in 176 (+2 cutoff functions) different features for each atomic environment. The 546 SOAP descriptors were selected with $n_{\max} = 6$, $l_{\max} = 6$ and a cutoff radius of 6 Å.

The optimization of ACSF with respect to the ground truth of SOAP is carried out starting from $\gamma_i = 1 \; \forall \; i \in [1, 176]$.

JAX version of DII

In order to benefit from modern machine learning GPU-based calculation speed, a GPU-compatible implementation written with the JAX library[77] is also provided within the same package.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

The data generated by feature selection in this study have been deposited on OSF at the following URL: https://osf.io/swtg5. The processed molecular dynamics and H_2O structure data are also available at OSF. The data files necessary for carrying out all analyses and source data are available at the same OSF URL. Source data are provided with this paper.

Code availability

The Python code to replicate and extend our study is available on GitHub at the following URL: https://github.com/sissa-data-science/DADApy under the Apache License 2.0. The according documentation can be found in ref. 41. The code at the time of publishing can be built under the Apache License 2.0 from: https://doi.org/10.5281/zenodo.14277899[78].

References

1. Guyon, I. & Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res.* **3**, 1157–1182 (2003).
2. Sarder, M. A., Maniruzzaman, M. & Ahammed, B. Feature selection and classification of leukemia cancer using machine learning techniques. *Mach. Learn. Res.* **5**, 18–27 (2020).
3. Wang, Y., Yao, H. & Zhao, S. Auto-encoder based dimensionality reduction. *Neurocomputing* **184**, 232–242 (2016).
4. McInnes, L., Healy, J., Saul, N. & Großberger, L. Umap: uniform manifold approximation and projection. *J. Open Source Softw.* **3**, 861 (2018).
5. Ehiro, T. Feature importance-based interpretation of umap-visualized polymer space. *Mol. Inform.* **42**, 2300061 (2023).
6. van der Maaten, L. & Hinton, G. Visualizing data using t-sne. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008).
7. Bussi, G. & Tribello, G. A. *Analyzing and Biasing Simulations with PLUMED* 529–578 (Springer New York, 2019). https://doi.org/10.1007/978-1-4939-9608-7_21.
8. Yun, K. K., Yoon, S. W. & Won, D. Interpretable stock price forecasting model using genetic algorithm-machine learning regressions and best feature subset selection. *Expert Syst. Appl.* **213**, 118803 (2023).
9. Chen, Y., Zhang, J. & Qin, X. Interpretable instance disease prediction based on causal feature selection and effect analysis. *BMC Med. Inform. Decis. Mak.* **22**, 51 (2022).
10. Remeseiro, B. & Bolon-Canedo, V. A review of feature selection methods in medical applications. *Comput. Biol. Med.* **112**, 103375 (2019).
11. Sozio, E. et al. The role of asymmetric dimethylarginine (ADMA) in COVID-19: association with respiratory failure and predictive role for outcome. *Sci. Rep.* **13**, 9811 (2023).
12. Pathan, M. S., Nag, A., Pathan, M. M. & Dev, S. Analyzing the impact of feature selection on the accuracy of heart disease prediction. *Healthc. Anal.* **2**, 100060 (2022).
13. Chandrashekar, G. & Sahin, F. A survey on feature selection methods. *Comput. Electr. Eng.* **40**, 16–28 (2014).
14. Wu, X. et al. Supervised feature selection with orthogonal regression and feature weighting. *IEEE Trans. Neural Netw. Learn. Syst.* **32**, 1831–1838 (2021).

15. Hastie, T. & Tibshirani, R. *Generalized Additive Models* (Wiley Online Library, 1990).
16. Maldonado, S., Weber, R. & Basak, J. Simultaneous feature selection and classification using kernel-penalized support vector machines. *Inf. Sci.* **181**, 115–128 (2011).
17. Maldonado, S. & López, J. Dealing with high-dimensional class-imbalanced datasets: embedded feature selection for SVM classification. *Appl. Soft Comput.* **67**, 94–105 (2018).
18. Liu, Y., Ye, D., Li, W., Wang, H. & Gao, Y. Robust neighborhood embedding for unsupervised feature selection. *Knowl. Based Syst.* **193**, 105462 (2020).
19. Wang, H. & Hong, M. Distance variance score: an efficient feature selection method in text classification. *Math. Probl. Eng.* **2015**, 695720 (2015).
20. He, X., Cai, D. & Niyogi, P. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems*, vol. 18 (eds Weiss, Y., Schölkopf, B. & Platt, J.) (MIT Press, 2005). https://proceedings.neurips.cc/paper_files/paper/2005/file/b5b03f06271f8917685d14cea7c6c50a-Paper.pdf
21. Cai, D., Zhang, C. & He, X. Unsupervised feature selection for multi-cluster data. In *Proc. 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, 333–342 (Association for Computing Machinery, 2010). https://doi.org/10.1145/1835804.1835848.
22. Boutsidis, C., Drineas, P. & Mahoney, M. W. Unsupervised feature selection for the k-means clustering problem. In *Advances in Neural Information Processing Systems*, vol. 22 (eds Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C. & Culotta, A.) (Curran Associates, Inc., 2009). https://proceedings.neurips.cc/paper_files/paper/2009/file/c51ce410c124a10e0db5e4b97fc2af39-Paper.pdf.
23. Kraskov, A., Stögbauer, H. & Grassberger, P. Estimating mutual information. *Phys. Rev. E* **69**, 066138 (2004).
24. Ståhle, L. & Wold, S. Analysis of variance (ANOVA). *Chemom. Intell. Lab. Syst.* **6**, 259–272 (1989).
25. Almuallim, H., Dietterich, T. G. et al. Learning with many irrelevant features. in *AAAI*, vol. 91, 547–552 (Citeseer, 1991).
26. Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S. & Moore, J. H. Relief-based feature selection: introduction and review. *J. Biomed. Inform.* **85**, 189–203 (2018).
27. Kira, K. & Rendell, L. A. A practical approach to feature selection. in *Machine Learning Proceedings 1992* 249–256 (Elsevier, 1992).
28. Hopf, K. & Reifenrath, S. Filter methods for feature selection in supervised machine learning applications—review and benchmark 2111.12140. https://arxiv.org/abs/2111.12140 (2021).
29. Campadelli, P., Casiraghi, E., Ceruti, C. & Rozza, A. Intrinsic dimension estimation: relevant techniques and a benchmark framework. *Math. Probl. Eng.* **2015**, 759567 (2015).
30. Facco, E., d'Errico, M., Rodriguez, A. & Laio, A. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Sci. Rep.* **7**, 12140 (2017).
31. Allegra, M., Facco, E., Denti, F., Laio, A. & Mira, A. Data segmentation based on the local intrinsic dimension. *Sci. Rep.* **10**, 16449 (2020).
32. Zhang, R., Nie, F., Li, X. & Wei, X. Feature selection with multi-view data: a survey. *Inf. Fusion* **50**, 158–167 (2019).
33. Wild, R. et al. Maximally informative feature selection using information imbalance: application to Covid-19 severity prediction. *Sci. Rep.* **14**, 10744 (2024).
34. Nie, F., Li, J. & Li, X. Parameter-free auto-weighted multiple graph learning: a framework for multiview clustering and semi-supervised classification. In *Proc. Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, 1881–1887 (AAAI Press, 2016).
35. Glielmo, A., Zeni, C., Cheng, B., Csányi, G. & Laio, A. Ranking the information content of distance measures. *PNAS Nexus* **1**, pgac039 (2022).
36. Kandy, A. K. A., Rossi, K., Raulin-Foissac, A., Laurens, G. & Lam, J. Comparing transferability in neural network approaches and linear models for machine-learning interaction potentials. *Phys. Rev. B* **107**, 174106 (2023).
37. Donkor, E. D., Laio, A. & Hassanali, A. Do machine-learning atomic descriptors and order parameters tell the same story? the case of liquid water. *J. Chem. Theory Comput.* **19**, 4596–4605 (2023).
38. Darby, J. P. et al. Tensor-reduced atomic density representations. *Phys. Rev. Lett.* **131**, 028001 (2023).
39. Tatto, V. D., Fortunato, G., Bueti, D. & Laio, A. Robust inference of causality in high-dimensional dynamical processes from the information imbalance of distance ranks. *Proc. Natl. Acad. Sci. USA* **121**, e2317256121 (2024).
40. Glielmo, A. et al. DADApy: distance-based analysis of data-manifolds in Python. *Patterns* **3**, 100589 (2022).
41. DADApy-Authors. Distance-based analysis of data-manifolds in Python (DADApy), accessed 28 March 2024; https://dadapy.readthedocs.io/en/latest/index.html (2021).
42. Behler, J. & Parrinello, M. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.* **98 14**, 146401 (2007).
43. Bartók, A. P., Kondor, R. & Csányi, G. On representing chemical environments. *Phys. Rev. B* **87**, 184115 (2013).
44. De, S., Bartók, A. P., Csányi, G. & Ceriotti, M. Comparing molecules and solids across structural and alchemical space. *Phys. Chem. Chem. Phys.* **18**, 13754–13769 (2016).
45. Springer, M. D. & Thompson, W. E. The distribution of products of beta, gamma and Gaussian random variables. *SIAM J. Appl. Math.* **18**, 721–737 (1970).
46. Urbanowicz, R. J., Olson, R. S., Schmitt, P., Meeker, M. & Moore, J. H. Benchmarking relief-based feature selection methods for bioinformatics data mining. *J. Biomed. Inform.* **85**, 168–188 (2018).
47. Spolaôr, N., Cherman, E. A., Monard, M. C. & Lee, H. D. Relieff for multi-label feature selection. In *Proc. 2013 Brazilian Conference on Intelligent Systems* 6–11 (IEEE, 2013).
48. Zhang, J., Liu, K., Yang, X., Ju, H. & Xu, S. Multi-label learning with relief-based label-specific feature selection. *Appl. Intell.* **53**, 18517–18530 (2023).
49. Pedregosa, F. et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
50. Carli, M. & Laio, A. Statistically unbiased free energy estimates from biased simulations. *Mol. Phys.* **119**, e1899323 (2021).
51. Honda, S. et al. Crystal structure of a ten-amino acid protein. *J. Am. Chem. Soc.* **130**, 15327–15331 (2008).
52. Rodriguez, A., D'Errico, M., Facco, E. & Laio, A. Computing the free energy without collective variables. *J. Chem. Theory Comput.* **14**, 1206 – 1215 (2018).
53. McKiernan, K. A., Husic, B. E. & Pande, V. S. Modeling the mechanism of CLN025 beta-hairpin formation. *J. Chem. Phys.* **147**, https://www.scopus.com/inward/record.uri?eid=2-s2.0-85029471768&doi=10.1063%2f1.4993207&partnerID=40&md5=9f4b0c0ca0ef5e562b09ca0650466fbe (2017).
54. d'Errico, M., Facco, E., Laio, A. & Rodriguez, A. Automatic topography of high-dimensional data sets by non-parametric density peak clustering. *Inf. Sci.* **560**, 476 – 492 (2021).
55. Manning, C. D., Raghavan, P. & Schütze, H. Introduction to information retrieval. *Flat Clustering* 349–375 (Cambridge University Press, 2008).
56. Behler, J. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *J. Chem. Phys.* **134 7**, 074106 (2011).
57. Cheng, B., Engel, E. A., Behler, J., Dellago, C. & Ceriotti, M. Ab initio thermodynamics of liquid and solid water. *Proc. Natl. Acad. Sci. USA* **116**, 1110–1115 (2018).

58. Keith, J. A. et al. Combining machine learning and computational chemistry for predictive insights into chemical systems. *Chem. Rev.* **121**, 9816–9872 (2021).
59. Musil, F. et al. Physics-inspired structural representations for molecules and materials. *Chem. Rev.* **121**, 9759–9815 (2021).
60. Darby, J. P., Kermode, J. R. & Csányi, G. Compressing local atomic neighbourhood descriptors. *npj Comput. Mater.* **8**, 166 (2022).
61. Zeni, C., Rossi, K., Glielmo, A. & de Gironcoli, S. Compact atomic descriptors enable accurate predictions via linear models. *J. Chem. Phys.* **154**. https://doi.org/10.1063/5.0052961 (2021).
62. Zeni, C., Anelli, A., Glielmo, A. & Rossi, K. Exploring the robust extrapolation of high-dimensional machine learning potentials. *Phys. Rev. B* **105**, 165141 (2022).
63. Ren, H., Pan, H., Olsen, S. & Moeslund, T. *Greedy vs. L1 Convex Optimization in Sparse Coding: Comparative Study in Abnormal Event Detection*, vol. 37 (MIT Press, 2015). International Conference on Machine Learning 2015; Conference date: 01-06-2015.
64. Halabi, M. E. & Jegelka, S. Optimal approximation for unconstrained non-submodular minimization. In *Proceedings of the 37th International Conference on Machine Learning* 3961–3972, vol. 119 (eds Daumé III, H. & Singh, A.) (PMLR, 2020). http://proceedings.mlr.press/v119/halabi20a/halabi20a.pdf.
65. Van der Maaten, L. J. P., Postma, E. O. & Van Den Herik, H. J. Dimensionality reduction: a comparative review. *J. Mach. Learn. Res.* **10**, 1–41 (2009).
66. Glielmo, A. et al. Unsupervised learning methods for molecular simulation data. *Chem. Rev.* **121**, 9722–9758 (2021).
67. Bellet, A., Habrard, A. & Sebban, M. Metric learning. *Synth. Lectures Artif. Intell. Mach. Learn.* **9**, 1–151 (2015).
68. Beyer, K., Goldstein, J., Ramakrishnan, R. & Shaft, U. When is "nearest neighbor" meaningful? In *Database Theory—ICDT'99* (eds Beeri, C. & Buneman, P.) 217–235 (Springer Berlin Heidelberg, 1999).
69. Hinneburg, A., Aggarwal, C. & Keim, D. What is the nearest neighbor in high dimensional spaces? In *Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt, 2000* 506–515, vol. 1 (Morgan Kaufmann, 2000). http://nbn-resolving.de/urn:nbn:de:bsz:352-opus-70224.
70. Bach, F., Jenatton, R., Mairal, J. & Obozinski, G. Optimization with sparsity-inducing penalties. *Found. Trends Mach. Learn.* **4**, 1–106 (2011).
71. Zou, H. & Hastie, T. Regularization and variable selection via the elastic net. *J. R. Stat. Soc.: Ser. B* **67**, 301–320 (2005).
72. Tsuruoka, Y., Tsujii, J. & Ananiadou, S. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Annual Meeting of the Association for Computational Linguistics*. https://api.semanticscholar.org/CorpusID:18431463 (2009).
73. Tribello, G. A., Bonomi, M., Branduardi, D., Camilloni, C. & Bussi, G. Plumed 2: New feathers for an old bird. *Comput. Phys. Commun.* **185**, 604–613 (2014).
74. Lippert, G., Hutter, J. & Parrinello, M. The Gaussian and augmented-plane-wave density functional method for ab initio molecular dynamics simulations. *Theor. Chem. Acc.* **103**, 124–140 (1999).
75. Himanen, L. et al. DScribe: library of descriptors for machine learning in materials science. *Comput. Phys. Commun.* **247**, 106949 (2020).
76. Laakso, J. et al. Updates to the dscribe library: new descriptors and derivatives. *J. Chem. Phys.* **158**, 234802 (2023).
77. Bradbury, J. et al. JAX: composable transformations of Python +NumPy programs. http://github.com/google/jax (2018).
78. Wodaczek, F. & Wild, R. Felixwodaczek/dii-molecular-systems: v1.0.1. https://doi.org/10.5281/zenodo.14277899 (2024).
79. Singraber, A., Behler, J. & Dellago, C. Library-based lammps implementation of high-dimensional neural network potentials. *J. Chem. Theory Comput.* **15**, 1827–1840 (2019).
80. Singraber, A., Morawietz, T., Behler, J. & Dellago, C. Parallel multistream training of high-dimensional neural network potentials. *J. Chem. Theory Comput.* **15**, 3075–3092 (2019).

Acknowledgements

The authors thank Dr. Matteo Carli for providing the CLN025 replica exchange MD trajectory and Matteo Allione for the fruitful discussions connected with the idea of the linear scaling estimator. This work was partially funded by NextGenerationEU through the Italian National Centre for HPC, Big Data, and Quantum Computing (Grant No. CN00000013 received by A.L.). A.L. also acknowledges financial support by the region Friuli Venezia Giulia (project F53C22001770002 received by A.L.).

Author contributions

Concept: A.L., R.W., V.D.T., F.W. and B.C. Model and algorithm development: R.W., V.D.T., F.W., B.C., and A.L. Code implementation: R.W., F.W. and V.D.T. Figures: R.W., F.W., and V.D.T. Writing: R.W., V.D.T., F.W., B.C., and A.L.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41467-024-55449-7.

Correspondence and requests for materials should be addressed to Alessandro Laio.

Peer review information *Nature Communications* thanks the anonymous, reviewer(s) for their contribution to the peer review of this work. A peer review file is available.