# On the Interplay of Subset Selection and Informed Graph Neural Networks

Niklas Breustedt[⋆], Paolo Climaco[‡], Jochen Garcke[†‡], Jan Hamaekers[†], Gitta Kutyniok[+#], Dirk A. Lorenz[⋆], Rick Oerder[†], and Chirag Varun Shukla[#].

**Abstract** Machine learning techniques paired with the availability of massive datasets dramatically enhance our ability to explore the chemical compound space by providing fast and accurate predictions of molecular properties. However, learning on large datasets is strongly limited by the availability of computational resources and can be infeasible in some scenarios. Moreover, the instances in the datasets may not yet be labelled and generating the labels can be costly, as in the case of quantum chemistry computations. Thus, there is a need to select small training subsets from large pools of unlabeled data points and to develop reliable ML methods that can effectively learn from small training sets. This work focuses on predicting the molecules' atomization energy in the QM9 dataset. We investigate the advantages of employing domain knowledge-based data sampling methods for an efficient training set selection combined with informed ML techniques. In particular, we show how maximizing molecular diversity in the training set selection process increases the robustness of linear and nonlinear regression techniques such as kernel methods and graph neural networks. We also check the reliability of the predictions made by the graph neural network with a model-agnostic explainer based on the rate-distortion explanation framework.

## 1 Introduction

Modelling the relationship between molecules and their properties is of great interest in several research areas, such as computational drug design [37], material discovery [43] and battery development [3]. The field of computational chemistry offers powerful *ab initio* methods to compute physical and chemical properties of atomic systems.

[†]Fraunhofer SCAI, Sankt Augustin, Germany · [‡]Institut für Numerische Simulation, Universität Bonn, Germany · [+]University of Tromsø, Norway · [#]Ludwig-Maximilians-Universität München, Germany · [⋆]Institut für Analysis und Algebra, Technische Universität Braunschweig, Germany

Unfortunately, these approaches are often limited by their high computational complexity, which restricts their practical applicability to only small sets of molecules. Therefore, machine learning (ML) methods for molecular property prediction have recently gained increased attention in molecular and material science because of their computational efficiency and accuracy on par with established first principle methods [4, 5, 11]. However, to effectively employ ML in real-world problems, there is a need for labelled datasets that can effectively represent the chemical space of interest, i.e., sets of molecules for which the target properties have already been computed using ab initio methods. Thus, on the one hand, accurately choosing which data points to label in the analyzed chemical space is crucial to avoid creating a dataset with redundant information and limiting the required amount of ab initio calculations. On the other hand, it is critical to develop data-efficient ML methods that perform accurate predictions.

Integrating domain knowledge of physical and chemical principles into the dataset selection process and the development of ML techniques is a primary goal of the chemical and material science ML community [31]. Physical and chemical principles, such as spatial invariances, symmetries, algebraic equations and chemical properties, can increase the robustness, reliability and effectiveness of ML methods while reducing the required training data [6, 58].

This work focuses on predicting the atomization energies of molecules in the QM9 dataset [47, 49] and shows how to exploit domain knowledge to select training sets according to specific criteria and how different ML methods may benefit from training on sets selected through such criteria. Specifically, by using Mordred [42], a publicly available library, we generate knowledge-based vector representations of molecules based on their SMILES representation [59] without requiring any ab initio computations. Further, based on such a molecular vector-based representation, we define a training set selection process and can observe that a diversity in the selected subset can increase the reliability of ML methods, indicated by the reduction of the maximum absolute error of the prediction. The maximum absolute error can be interpreted as a measure of robustness, and it is a helpful metric to evaluate ML methods in chemical and material science [65] since the average error alone gives an incomplete impression [19, 55]. Furthermore, this work shows how diversity reduces the gap between the predictive robustness of linear regression-based approaches relying only on the molecular topological information, such as kernel ridge regression (KRR) [32], and non-linear approaches relying on molecular geometric representations obtained through ab initio computations, such as graph neural networks (GNN) [17, 24, 26]. We compare the effectiveness of a diversity-based selection with that of random sampling and of an alternative selection approach based on domain knowledge that focuses on representativeness, i.e., the distribution of chosen properties of the dataset should be present with the same amount in the selected training sets.

Finally, we note that our GNNs are inherently opaque (i.e. the logic flow to the decision-making process of the neural network is obscured). This inherently opaque nature of common deep neural network architectures has led to a rise in demand for trustworthy explanation techniques, which vary in their meaning and validity [48].

Unlike other modalities in computer vision and natural language processing, the non-Euclidean nature of graph-structured data poses a significant challenge to trustworthy and interpretable explanation generation. To this end, there exist a variety of explanation techniques and explanation types [12, 21, 36, 46, 51, 60, 62, 64], the most popular of which are subgraph explanation techniques.

We probe the domain knowledge learned/retained by our GNNs for different sampling strategies through the application of a novel *post-hoc* model-agnostic explanation technique, graph rate-distortion explainer (GRDE). GRDE builds on the existing rate-distortion explanation (RDE) framework [27, 38] to generate *instance-level* subgraph explanations on the input graphs, which highlight the substructures and features in the graph that are most relevant towards the GNNs' predictions.

After describing related work, we give in the following first an overview on three ML models that are designed for the prediction of molecular properties but are based on different underlying working principles. In this way, we hope that our results yield insights for a variety of methods that are used in practical applications. Following that, we discuss two ways of sampling subsets from a larger dataset, one aiming to maximize the diversity of the selected samples and the other seeking to choose a collection of points representative of the set from which we sample. Afterwards, we test the introduced methods, namely the SchNet, KRR and the spatial 3-hop convolution network which is proposed in this work, by performing numerical experiments on the QM9 dataset while putting special emphasis on the effects of the sampling strategies. After a discussion of the numerical results and a comparison between the different ML models, we seek explanations of the model predictions by applying GRDE to one of the employed graph neural networks.

## 2 Related Work

In recent years, there has been growing interest in incorporating domain-specific knowledge into the selection of training data and the development of learning algorithms, which is referred to as informed machine learning [58]. Ideally, the training data selection process should be based purely on the data's features, as labels may be expensive to compute, and should be model-independent so that the selected training data is beneficial for multiple learning models rather than just one. This allows for greater flexibility in model selection and avoids the need for repeating the dataset selection process for each model. Considering these practical aspects, it is clear that a feature-based and model-independent selection process is desirable for efficient and effective machine learning. This section reviews some of the relevant work in this area. Coreset approaches [14] are among the most popular strategies for feature-based and model-independent selection of training datasets. Several of these approaches involve incorporating domain-knowledge into the selection of training data by selecting data points that are representative of the distribution of the target points for which we want to predict the new labels. The simplest and yet one of the most common coreset approaches is uniform sampling, which involves selecting

a random subset of data points from the larger dataset. Uniform sampling is also considered a benchmark for every other selection approach. Unfortunately, uniform sampling does not exploit domain knowledge and can lead to biased results if the dataset is imbalanced or if certain data points are more important than others. To address this issue, importance sampling [7] is an approach that exploits domain knowledge to assign weights to each data point based on its importance or relevance to the problem at hand. The weights are then used for a nonuniform selection of the training set that privileges more important data points. Another class of methods are the grid-based approaches [2], which involve dividing the feature space into a grid and selecting one or more representative points from each grid's cell. This can be useful for problems with a high-dimensional feature space or when there is a need for a more structured selection of data points. Greedy constructions are coreset approaches that iteratively select the most informative data points based on a pre-defined criterion. For instance, well-known greedy selection methods are submodular function maximization algorithms [30]. Greedy approaches can be effective for selecting a small subset of highly informative data points, but they may be computationally expensive for large datasets. Overall, the choice of coreset approach depends on the specific problem and dataset characteristics, as well as computational constraints. See [14] for a more detailed review of coreset approaches. Finally, the field of experimental design [61] offers additional sampling strategies to perform a feature-based selection of the training set that can benefit specific regression model classes, e.g., linear models.

In this work, incorporating domain knowledge in the learning of algorithms refers to methods which are known as informed graph neural networks. While graph neural networks recently gained increasing attention by the works from Gori et al. [18] and Scarselli et al. [50], the question of how to use domain knowledge to improve the performance of learning methods dates back to the last century (e.g. see [23] or [29]). More recently, physics informed neural networks, which address supervised learning tasks complying with the known laws from physics, are a hot topic in several applications, e.g. to find surface breaking cracks in a metal plate [53] or to solve inverse heat transfer problems [8]. For graph neural networks, based on the message passing principle, i.e. the process of updating so called states or representations attached to each node of a graph using the node's neighbourhood, many different models were proposed (e.g. Graph Attention Networks [57], ChebNet [10], Gated Graph Neural Networks [34]), the most popular being the graph convolutional model by Kipf and Welling [26] which is motivated by an approximation of spectral graph convolutions. Combining incorporating domain knowledge with graph neural networks leads to the very recent informed graph neural networks. In [20] the authors combine theory from thermodynamics with graph neural networks to predict the behaviour of dynamical systems and in [25] combine physical properties of molecules are combined with graph neural networks to predict the cetane number of possible alternative fuels. For more detailed overviews on GNNs or informed neural networks we refer to the book [35] and a recent review [9].

We further build upon the interpretability of graph neural networks in this work by introducing a method akin to perturbation techniques on image data to graph-

structured data. The main goal of interpretability is to invoke transparency in the otherwise opaque prediction process of neural networks, and is further applicable in the detection of bias as well as to explain incorrect classifications in the predictive model. Previous work in interpretability for other modalities such as audio and images [27,38] has shown great success in identifying a neural network's sensitivity to specific subsets of the input data. More specifically, among the variety of local and global interpretability techniques, perturbation [27,28] and gradient-based [54] techniques have been shown to accurately capture a predictive model's sensitivity to some concepts in the input data. These techniques generally seek to optimize a heatmap over the input data such that high-intensity zones are the most relevant to the model's prediction for the given data point. We further discuss this in detail with respect to graphs in section 3.4.

Inspired by the exhaustive work on interpretability for other modalities, several methods [36,46,51,60] have also been proposed for graph-structured data, with perturbation techniques such as GNNExplainer [60] being the baseline for comparison. For a detailed overview of GNN interpretability, we refer to [63]. These techniques, however, have been shown to suffer from unfaithfulness on large graphs since they optimise masks only for small graphs as well as manually threshold their relevance scores. See [1] for a detailed review on the current issues with graph interpretability.

## 3 Methods and Sampling Strategies

This section introduces the approaches we use for predicting the atomization energy, explaining the GNN output and sampling the training data. Subsection 3.1 introduces the benchmark regression model SchNet, a GNN that uses 3-dimensional positional information to predict chemical properties. Next, subsections 3.2 and 3.3 describe KRR and the spatial 3-hop convolution network, respectively. Both these approaches only exploit topological information encoded in the SMILES to perform the energy prediction task. Subsection 3.4 presents the rate-distortion explanation framework for graph data that we use to showcase the domain knowledge learned by the 3-hop convolution network. Finally, subsection 3.5 introduces the approaches we use for the selection of training sets.

### 3.1 SchNet

SchNet is a symmetry-informed neural network model, designed for the prediction of chemical properties by Schütt et al. [52]. In contrast to the methods presented in sections 3.2 and 3.3, it is trained and evaluated on 3-dimensional structural information describing the atomic systems of interest. Usually, the positional information is obtained from computational methods such as density functional theory (DFT).

More formally, for an atomic system with N atoms, SchNet can be used to predict scalar properties as a function $f$ of $3N$ atomic coordinates (nuclear positions) and on $N$ atomic numbers of the corresponding atoms:

$$f : \mathbb{R}^{3N} \times \mathbb{N}^N \to \mathbb{R}. \tag{1}$$

Internally, SchNet operates on a distance-based neighborhood graph, defined by a cutoff radius $r_{\text{cut}}$, in which nodes correspond to the atoms in the atomic system. In this scenario, edges do not necessarily correspond to chemical bonds but merely indicate whether two atoms are closer than the chosen cutoff radius. Hence, the chosen cutoff radius has a direct influence on the graph shown to the model. Similar to other GNNs [17, 26], SchNet operates in a layer-wise fashion by iteratively updating feature representations. At the $l$-th layer each atom, indexed by $i \in \{1, 2, ..., N\}$, is represented by a feature vector $\boldsymbol{x}_i^l \in \mathbb{R}^F$ where $F$ is a hyperparameter. The main layer introduced by Schütt et al. is the continuous-filter convolutional layer: Denoting the atomic positions by $\boldsymbol{r}_i \in \mathbb{R}^3$, this layer updates the atomic features as follows:

$$\boldsymbol{x}_i^{l+1} = \sum_{j \in \mathcal{N}(i)} \boldsymbol{x}_j^l \circ W^l \left( \boldsymbol{r}_i - \boldsymbol{r}_j \right), \tag{2}$$

where $W^l : \mathbb{R}^3 \to \mathbb{R}^F$ is a trainable filter-generating function and $\circ$ denotes element-wise multiplication. In detail, $W^l$ is given as the composition $W^l = \tilde{W}^l \circ \varphi$ of a distance-based radial basis expansion

$$\varphi : \boldsymbol{r}_i - \boldsymbol{r}_j \mapsto \bigoplus_{k=1}^{N_{\text{radial}}} \exp\left( -\gamma \left( \|\boldsymbol{r}_i - \boldsymbol{r}_j\|_2 - \mu_k \right)^2 \right) \tag{3}$$

and a trainable neural network $\tilde{W}^l$ where $0\,\text{A} \leq \mu_k \leq 30\,\text{A}$ are equidistributed centers and $\gamma = 10\,\text{A}$. Here, $\bigoplus$ denotes the direct sum that concatenates the scalar outputs of the radial basis functions to a feature vector in $\mathbb{R}^{N_{\text{radial}}}$ which is then passed into $\tilde{W}^l$. Note that $\varphi$ is invariant with respect to actions of the orthogonal group $O(3)$ which assures that the predictions of SchNet are invariant with respect to translations, rotations and reflections of the input structure as well. Depending on the atomic species, initial embeddings $\boldsymbol{x}_i^0$ are sampled from an $F$-dimensional standard normal distribution and optimized during the training process. In addition, non-linear layers such as dense feed-forward neural networks can be applied to the node features in order to increase the expressiveness of the model.

By summing over the images of a trainable readout function $R : \mathbb{R}^F \to \mathbb{R}$, the final node features in the last layer $L$ are transformed into a prediction of the target property $\hat{y}$:

$$\hat{y} = \sum_{j=1}^{N} R\left( \boldsymbol{x}_j^L \right) \tag{4}$$

Involving only permutation-invariant operations such as the summation over adjacent atoms, the output is invariant with respect to mutual permutations of the atomic positions and atomic species. For more details on the model architecture see [52].

### 3.2 Kernel Ridge Regression

In kernel ridge regression, a vector-based representation of the molecules is mapped into a high-dimensional space using a non-linear map that is implicitly determined by defining a kernel function, which provides a measure of similarity between the molecular representations. The structure-energy relationship is learned in the high-dimensional space. In this work, we use the so-called Gaussian kernel

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) := e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\nu^2}}, \tag{5}$$

where $\|\cdot\|_2$ is the $L_2$-norm and $\nu \in \mathbb{R}$ a kernel hyperparameter to be selected through an optimization process. The kernel ridge regression model is constructed using the selected training set $\{\boldsymbol{x}_i, y(\boldsymbol{x}_i)\}_{i=1}^p$, where $\{\boldsymbol{x}_i\}_{i=1}^p$ are the Mordred [42] based vector representations of the molecules and $\{y(\boldsymbol{x}_i)\}_{i=1}^p$ the associated atomization energies. Once the regression model has been constructed, the predicted energies are given by the scalar values $\tilde{y}(\boldsymbol{x})$ defined as follows

$$\tilde{y}(\boldsymbol{x}) := \sum_{i=1}^{p} \alpha_i k(\boldsymbol{x}, \boldsymbol{x}_i), \tag{6}$$

where the vector $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_p]^T \in \mathbb{R}^p$ is the solution of the following minimization problem

$$\boldsymbol{\alpha} = \underset{\tilde{\boldsymbol{\alpha}}}{\operatorname{argmin}} \sum_{i=1}^{p} (\tilde{y}(\boldsymbol{x}_i) - y(\boldsymbol{x}_i))^2 + \lambda \tilde{\boldsymbol{\alpha}}^T \boldsymbol{K} \tilde{\boldsymbol{\alpha}}. \tag{7}$$

Here, $\boldsymbol{K}$ is the kernel matrix, i.e., $\boldsymbol{K}_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and the parameter $\lambda \in \mathbb{R}$ is the so-called regularization parameter that penalizes larger weights. The analytic solution to the minimization problem in (7) is given by

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

where $\tilde{\boldsymbol{y}} = [\tilde{y}(\boldsymbol{x}_1), \tilde{y}(\boldsymbol{x}_2), \ldots, \tilde{y}(\boldsymbol{x}_p)]^T$. Once the training process has been concluded and the regression parameters $\{\alpha_i\}_{i=1}^p$ have been learned, the energy predictions for molecules not included in the training set can be computed using Equation (6).

### 3.3 Spatial 3-Hop Convolution Network

In addition to the two previous approaches, we propose a third approach which builds on a newly developed spatial graph convolution structure. We call this approach spatial 3-hop convolution network. This approach exploits the graph structure, the node features and optionally edge features for regression or classification but does not need 3-dimensional structural information as is the case for SchNet.

A commonly used graph convolutional network by Kipf & Welling [26] is motivated by an approximation of a spectral convolution. Thereby, they consider spectral convolutions as

$$\boldsymbol{w} \star \boldsymbol{x} = \boldsymbol{U}\boldsymbol{w}\boldsymbol{U}^{\top}\boldsymbol{x}, \tag{9}$$

where $\boldsymbol{w} = diag(\theta) \in \mathbb{R}^{n\times n}$ is a filter, $\boldsymbol{x} \in \mathbb{R}^n$ is a graph signal on a graph with $n$ nodes, $\star$ denotes the spectral graph convolution operator and $\boldsymbol{U}$ is the matrix of eigenvectors from the eigendecomposition of the normalized graph Laplacian $\boldsymbol{I}_n - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$. Moreover, $\boldsymbol{A}$ is the adjacency matrix of the underlying graph, $\mathbf{D}$ is the corresponding degree matrix and $\boldsymbol{I}_n$ is the $n\times n$ identity matrix. This convolution is approximated and generalized to matrix-valued graph signals which leads to the update of the graph convolutional network

$$\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{H}^{(l)}\boldsymbol{W}_0 + \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{H}^{(l)}\boldsymbol{W}_1), \tag{10}$$

where $\boldsymbol{H}^{(l)}$ is the matrix of hidden representations of the $l$-th layer, $\boldsymbol{W}_0$ and $\boldsymbol{W}_1$ are learnable parameters and $\sigma$ denotes the elementwise ReLU function. For the spatial 3-hop convolution layer we do not consider spectral graph convolutions but an intuitive spatial convolution using powers of the graphs adjacency matrix to calculate so called path matrices. Within these, for each node the number of paths of a certain length to every other node is stored. By defining a spatial convolution with path matrices and building a layer of the graph neural network using the convolution, we consider the number of paths of a given length from node $v$ to node $u$ as a measure for the impact of node $v$ on node $u$. Thus, nodes with more paths to the considered node will be taken into account more during the update.

For a graph $G$ with $n$ nodes a path is defined as a sequence of nodes $(1, \dots, k)$ with $k < n$ such that for any $i, j \in (1, \dots, k)$ it is $i \neq j$, i.e. no node appears twice.

With that, we define a spatial $k$-hop graph convolution of a graph signal $\boldsymbol{x} \in \mathbb{R}^n$ with a filter $\boldsymbol{w} \in \mathbb{R}^k$ on an undirected graph $G$ with $n$ nodes as

$$\boldsymbol{w} \star_k \boldsymbol{x} := \sum_{i=0}^{k} \boldsymbol{w}_i \boldsymbol{T}^{(i)} x,$$

where $\boldsymbol{T}^{(i)}$ is a path matrix such that $\boldsymbol{T}^{(i)}_{vu}$ is the number of paths of length $i$ from node $v$ to node $u$.

An approach to computing the needed path matrices is a recursion that starts with the adjacency matrix. Since the adjacency matrix equals the path matrix for paths

of length one it is $\boldsymbol{T}^{(1)} = \boldsymbol{A}$. For every node $i$ and $u$ a neighbor of it, the number of paths of length two from node $i$ to node $j$ equals the number of paths of length one from $u$ to $j$ in which $i$ is not a part of. More generally, the number of paths of length $k$ from a node $i$ to a different node $j$ equals the sum of all paths from node $u$ to $j$ of length $k-1$ over all $u \in \mathcal{N}(i)$ in which $i$ does not appear. Using this, it can be shown that $\boldsymbol{T}^{(2)} = \boldsymbol{A}^2 - \boldsymbol{D}$ and $\boldsymbol{T}^{(3)} = \boldsymbol{A}^3 - \boldsymbol{\Sigma} \circ \boldsymbol{A}$, where $\boldsymbol{A}$ and $\boldsymbol{D}$ are as above and $\boldsymbol{\Sigma}$ is an $n \times n$ matrix with $\boldsymbol{\Sigma}_{ij} = \boldsymbol{D}_{ii} + \boldsymbol{D}_{jj}$. This shows that the 3-hop spatial graph convolution is given by

$$\boldsymbol{w} \star_3 \boldsymbol{x} = (\boldsymbol{w}_0 \boldsymbol{I}_n + \boldsymbol{w}_1 \boldsymbol{A} + \boldsymbol{w}_2(\boldsymbol{A}^2 - \boldsymbol{D}) + \boldsymbol{w}_3(\boldsymbol{A}^3 - \boldsymbol{\Sigma} \circ \boldsymbol{A}))\boldsymbol{x}.$$

Note that the $\boldsymbol{w}_k$'s can be seen as weights for the $k$-hop neighborhoods. A generalization of the former discussion to a signal $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ with $c$ node features for each node (analogously to Kipf & Welling [26]) leads to

$$\boldsymbol{H} = \boldsymbol{X}\boldsymbol{W}_0 + \boldsymbol{A}\boldsymbol{X}\boldsymbol{W}_1 + (\boldsymbol{A}^2 - \boldsymbol{D})\boldsymbol{X}\boldsymbol{W}_2 + (\boldsymbol{A}^3 - \boldsymbol{\Sigma} \circ \boldsymbol{A})\boldsymbol{X}\boldsymbol{W}_3,$$

which results in the spatial 3-hop convolution layer, the message passing layer of the spatial 3-hop convolution network,

$$\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{H}^{(l)}\boldsymbol{W}_0 + \boldsymbol{A}\boldsymbol{H}^{(l)}\boldsymbol{W}_1 + (\boldsymbol{A}^2 - \boldsymbol{D})\boldsymbol{H}^{(l)}\boldsymbol{W}_2 + (\boldsymbol{A}^3 - \boldsymbol{\Sigma} \circ \boldsymbol{A})\boldsymbol{H}^{(l)}\boldsymbol{W}_3),$$

where $\sigma$ is, again, the element-wise ReLU function.

### 3.4 Graph Rate-Distortion Explanations

We now present a formulation for the rate-distortion explanation framework [27,38] for graph data. Given a pre-trained GNN model, $\Phi : \mathbb{R}^{n \times c} \longrightarrow \mathbb{R}^m$ and a set of attributed graphs $G = \{G_1, G_2, ..., G_p\}$ such that $G_i = (V_i, E_i, X_i)$ for all $i \in [1, p]$, our task is to explain the model decision over the set $G$, or more locally, $\Phi(G_i)$. This leads us to the two general branches of explanation techniques: global and local explanations. Global explanation techniques focus on explaining the underlying function learned by the model, $\Phi$. This can be done in a multitude of ways, such as testing the model's sensitivity to a concept [40] or reconstructing graphs from the embedding space learned by the model to reveal important motifs [62]. In general, global explanation techniques, while useful, are hard to construct and are unable to detect finer details on local data points. On the other hand, local explanation techniques, which are the more popular alternative, focus on explaining $\Phi$ for local instances, i.e. $\Phi(G_i)$. Similar to global explanations, there exist a variety of approaches, such as perturbation-based methods [36, 51, 60], surrogate methods [21], gradient-based methods [46], and additive methods [12, 64], each with their benefits and limitations. These techniques aim to extract information from $G_i$ that is most relevant to the local prediction $\Phi(G_i)$. More concretely, given a graph $G_i = (A_i, X_i)$, local explanation techniques commonly attempt to extract a subgraph

$\hat{G}_i = (\hat{A}_i, \hat{X}_i) \subseteq G_i$ that is most relevant to the model for its prediction $\Phi(G_i)$. The rate-distortion framework for explaining graphs is a local, post-hoc, model-agnostic explanation technique that comes under the umbrella of perturbation-based graph explainers. Given the pre-trained model $\Phi$ and graph $G_i$, GRDE optimizes a binary deletion mask $S = (S_A, S_X)$ over $G_i$ to obtain a subgraph $\hat{G}_i$ such that $\Phi(\hat{G}_i)$ approximates $\Phi(G_i)$. Mask $S$ thus retains only the edges and features that are most relevant to the model's prediction on $G_i$. Given $A_i \in \mathbb{R}^{n\times n}$ and $X_i \in \mathbb{R}^{n\times f}$, where $n$ is the number of nodes and $f$ is the number of node features, our goal is to optimize masks $S_A \in [0,1]^{n\times n}$ and $S_X \in [0,1]^{n\times f}$. Let $\mathcal{V}_S = (\mathcal{V}_{S_A}, \mathcal{V}_{S_X})$ be probability distributions that can either be chosen manually or learned from the graph dataset. Then the obfuscation on $G_i$, i.e. the subgraph $\hat{G}_i$, can be defined as

$$\hat{G}_i = (\hat{A}_i, \hat{X}_i) = (A_i \odot S_A + (1 - S_A) \odot v_{S_A}, X_i \odot S_X + (1 - S_X) \odot v_{S_X}), \quad (11)$$

where $v_{S_A} \in \mathcal{V}_{S_A}$, $v_{S_X} \in \mathcal{V}_{S_X}$, and $\odot$ denotes element-wise multiplication. Intuitively, this implies that the masks $S$ keep some of the elements in $G_i$ while the elements that are not selected by $S$ are replaced with values from the probability distribution $\mathcal{V}_S$ as 'noise'. In general, the choice of $\mathcal{V}_S$ should be such that the resulting subgraph $\hat{G}_i$ remains within the data manifold, provided that the data manifold is known. Depending on the information in $G_i$, we can use a variety of probability distributions for $(\mathcal{V}_{S_A}, \mathcal{V}_{S_X})$. For example, in the case of a binary adjacency matrix, $\mathcal{V}_{S_A}$ can be the Gumbel-Softmax distribution, whereas for real-valued adjacency matrices and node feature matrices, $\mathcal{V}_S$ can be Gaussian distributions. We can also learn the probability distributions $\mathcal{V}_S$ from the data manifold itself, as previous attempts have shown success with inpainting GANs [27] for this strategy on other data modalities.

Furthermore, we define the expected distortion on $G_i$ with respect to the masks $S$ and perturbation distributions $\mathcal{V}_S$ as

$$\mathcal{D}(G_i, S, \mathcal{V}_S, \Phi) = \mathop{\mathbb{E}}_{v_{S_A}\in\mathcal{V}_{S_A}, v_{S_X}\in\mathcal{V}_{S_X}} \Big[ d(\Phi(G_i), \Phi(\hat{G}_i)) \Big], \quad (12)$$

where $d : \mathbb{R}^m \times \mathbb{R}^m \longrightarrow \mathbb{R}_+$ is the measure of distortion between the two model outputs. Commonly, we can set $d$ as the $\mathcal{L}^2$ distance or the KL-divergence between the two model outputs. Thus, we can define the rate-distortion explanation on $G_i$ as the optimal subgraph $\hat{G}_i$ that solves the minimization problem

$$\min_{S=(S_A,S_X)} \mathcal{D}(G_i, S, \mathcal{V}_S, \Phi) \quad \text{s.t } ||S_A||_0 \leq j, ||S_X||_0 \leq k, \quad (13)$$

where $j, k$ are the desired levels of sparsity for $S_A, S_X$ respectively.

Note that solving equation (13) is $\mathcal{NP}$-hard [38]. Thus, we use an $l_1$ relaxation on equation (13) to get the relaxed optimization problem given by

$$\min_{S=(S_A,S_X)} \mathcal{D}(G_i, S, \mathcal{V}_S, \Phi) + \lambda_A||S_A||_1 + \lambda_X||S_X||_1, \quad (14)$$

where $\lambda_A, \lambda_X > 0$ are hyperparameters to control the sparsity level of the masks. We can further relax the binary masks $S$ by sampling them from the concrete distribution [39] or Gumbel Softmax distribution [22]. This allows us to solve the optimization problem in equation (14) with differentiable techniques such as stochastic gradient descent.

## 3.5 Sampling Strategies

We now introduce two approaches for sampling a set of points from a large dataset. The first method focuses on maximising the diversity of the selected set, while the second aims to select a set that is representative of the whole dataset.

### Diversity

In short, diverse subsets are iteratively selected from $\Omega \subset \mathbb{R}^d$ using the *farthest point sampling* (FPS) algorithm [13], where the resulting subset is a sub-optimal minimizer of the *fill distance*. We denote this approach by FPS.

To maximize diversity of the selection we consider the concept of fill distance. Given a dataset $\Omega \subset \mathbb{R}^d$ consisting of a finite amount of unique points, and $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_p\} \subset \Omega$ a subset of cardinality $p = |X| \in \mathbb{N}$ we define the fill distance of $X$ in $\Omega$ as

$$h_{X,\Omega} := \max_{\boldsymbol{x} \in \Omega} \min_{\boldsymbol{x}_j \in X} \|\boldsymbol{x} - \boldsymbol{x}_j\|_2. \tag{15}$$

Put differently, we have that any point $\boldsymbol{x} \in \Omega$ has a point $\boldsymbol{x}_j \in X$ not farther away than $h_{X,\Omega}$. Notice that, if $X, \bar{X} \subset \Omega$ with $p = |X| = |\bar{X}|$ and $h_{X,\Omega} < h_{\bar{X},\Omega}$ then $X$ consists of data points that are more widely distributed in $\Omega$, thus more diverse, than those in $\bar{X}$.

Fixing the number of points $p \in \mathbb{N}$ we want to select from $\Omega$, we aim to find $X \subset \Omega$ such that

$$X = \operatorname*{argmin}_{\bar{X} \subset \Omega, |\bar{X}| = p} h_{\bar{X},\Omega}. \tag{16}$$

The naive approach to solve the minimization problem in (16) would first require computing the fill distance for all possible sets $X \subset \Omega$ with $|X| = p$ and then choosing one of those sets where the minimum of the fill distance is attained. Unfortunately, such an approach is very time consuming and computationally intractable. Therefore, as an alternative approach we use the FPS algorithm [13]. FPS is a greedy selection method, which means that the points are progressively selected starting from an initial a-priori chosen point, i.e., given a set of selected points $X^s = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_s\} \subset \Omega$ with cardinality $|X^s| = s < p$, the next chosen point is

$$\boldsymbol{x}_{s+1} = \arg\max_{\boldsymbol{x} \in \Omega} \min_{\boldsymbol{x}_j \in X^s} \|\boldsymbol{x} - \boldsymbol{x}_j\|. \tag{17}$$

$\boldsymbol{x}_{s+1}$ is the point which is farthest away from the points in $X^s$ and it is the point where the fill distance $h_{X^s,\Omega}$ is attained. In other words, the next selected sample is the center of the largest empty ball in the dataset.

### Representativeness

We say that data points are representatively selected for the entire dataset, when the distribution of properties in the selected subset are as close as possible to the corresponding distribution in the whole dataset. To this aim, we divide $\Omega$ into clusters and select data points from them so that the distribution of the clusters in the subset resembles that of the whole dataset. For example, if we divide $\Omega$ into two clusters, each containing 50% of the data points, we aim to select a subset consisting of data points which are also equidistributed in the two clusters. The clustering can be performed by clustering algorithms or be based on properties and criteria stemming from domain knowledge, i.e., in the sense of [58] the training data is selected based on scientific knowledge. Furthermore, data points within each cluster are selected using the farthest point sampling, which ensures that in the various clusters a set of diverse data points is chosen. We call this approach cluster-based farthest point sampling (C-FPS).

## 4 Numerical Experiments

### 4.1 QM9 Dataset

In this work, we analyze the publicly available QM9 dataset [47, 49] containing a diverse set of organic molecules. Precisely, the QM9 consists of 133 885 organic molecules in equilibrium with up to 9 heavy atoms of four different types: C, O, N and F. The dataset provides the SMILES [59] representation of the relaxed molecules, their geometric configurations and 19 physical and chemical properties. To guarantee a consistent dataset, we remove all 3054 molecules that failed the consistency test proposed by [47]. Moreover, we remove the 612 compounds for which the RDKit package [33] can not interpret the SMILES. After this preprocessing procedure, we obtain at a smaller version of the QM9 dataset consisting of 130219 molecules.

#### 4.1.1 Knowledge Based Molecular Representation

The domain knowledge based molecular representation we employ is based on Mordred [42], a publicly available library that exploits the molecules' topological information encoded in the SMILES strings to provide 1826 physical and chemical features. Such molecular features are defined as the "final result of a logical and

mathematical procedure, which transforms chemical information encoded within a symbolic representation of a molecule into a useful number or the result of some standardized experiment" [56] and encode scientific knowledge reflecting algebraic equations, logic rules, or invariances [58]. Using the Mordred library, we represent each molecule in the QM9 dataset with a high-dimensional vector where each vector's entry is associated with a distinct feature.

To work with a more compact representation, after generating the Mordred vectors, we use the CUR [41] approach to select a subset of relevant features. The CUR algorithm takes as input the Mordred vector representation of each of the molecules in the analyzed dataset and ranks the significance of the features by associating them with an importance score. We select the first 59 top-ranked features and normalize their values in the range (0,1) using the "MinMaxScaler" function provided by the scikit-learn python library [44]. Moreover, to ensure the uniqueness of the representation, we consider an additional set of features representing the atom type distribution within each molecule. Specifically, for each data point, we add five features, each expressing the amount of atoms of a particular type within the molecule, in percentage. The possible atom types are H, C, O, N and F. In conclusion, the Mordred based representation we employ to sample the QM9 dataset consists of 64-dimensional vectors.

### 4.1.2 Diverse and Representative Sets of Molecules

The knowledge related to the molecules in the QM9 enables us to employ the data sampling strategies introduced in subsection 3.5 to create diverse and representative sets.

Diverse sets are constructed using the FPS algorithm on the Mordred-based vector representations of the molecules in the QM9. The Mordred vectors allow the representation molecules as points in $\mathbb{R}^d$, $d \in \mathbb{N}$, and the definition of a distance between the molecules, the Euclidean distance. Thus, we represent the QM9 as a finite set $\Omega \subset \mathbb{R}^d$ and use the FPS to sample from $\Omega$ a sub-optimal minimizer of the fill distance.

Representative sets are constructed using the procedure introduced in subsection 3.5 consisting of segmenting the QM9 in clusters and then sampling from each cluster so that the distribution of the chosen molecules resembles that of the whole dataset. The segmentation procedure is based on the molecules' topological information and considers their size, atom types and bond types, which following [58], reflects scientific knowledge in the selection of training data so that selected molecular properties are invariant per cluster. Specifically, we define the clusters through a process consisting of three main steps. In the first step, we split the molecules according to their sizes. As a result of this first step, we divided the QM9 dataset into 26 sets. After that, we separate each cluster obtained in the first step into subclusters defined by the different heavy atom types within the molecules. Overall, molecules in the QM9 consist of 4 heavy atom types. Thus, each molecule could consist of 15 different combinations of such atom types, e.g., a molecule can contain up to

four distinct heavy atoms, and for each amount of distinct heavy atom types, various combinations are possible. After this second step, each of the initial 26 clusters is divided into 15 subclusters. The third and final step is further splitting the data points in each subcluster into different sets according to the various bond types present in each molecule. We consider four different bond types: single, double, triple and aromatic bonds. Thus, each of the subclusters is further divided into 15 distinct sets. As a result of this clustering procedure, we divide the QM9 in 5850 different clusters that account for molecular size, atom types and bond types. Molecules within the clusters are selected using the farthest point sampling, which ensures that in the various clusters, a set of diverse molecules is chosen.

### 4.1.3 Sampling the QM9 Dataset

For the experiments, we select training sets of different sizes and according to different strategies from the entire preprocessed QM9 dataset. After that, we test each trained model's predictive accuracy on all the molecules that have not been selected to train it. We construct training sets consisting of 100, 250, 500, 1000 and 5000 samples. Such sets are created following three different selection criteria: random sampling (RDM), as a benchmark, and the two selection strategies introduced in section 3.5, namely, diversity sampling (FPS) and representative sampling (C-FPS). For each sampling strategy and training set size, we run the training set selection process independently five times. For RDM, at each run the points are independently and uniformly selected, while in the case of FPS and C-FPS the initial point to initialize the FPS algorithm is independently selected at random at each run. Thus, for each selection strategy and training set size, each of the analyzed models is trained and tested five times, independently. The test results that follow are averaged over the five runs.

We want to point out that sampling the training data non-randomly will lead to a shift between the training and test distribution, as showcased in Fig. 1, where we compare FPS with a random selection. It is not obvious how such a bias effects the different models. Note that for Fig. 1 we performed the selection twice, with different initialization for FPS and different seeds for the random selection, respectively. We find that changing the initialization for FPS does not lead to a significant change in the distribution, for different seeds in the random selection we make the same observation.

### 4.1.4 Measuring the Error

We evaluate the performances of the employed machine learning methods using three different metrics. Specifically, we consider the mean absolute error (MAE), the root mean squared error (RMSE) and the worst-case error. The mean absolute error (MAE) computes the arithmetic average of the absolute errors between the predicted values $\{\tilde{y}_i\}_{i=1}^N$ and the ground truths $\{y_i\}_{i=1}^N$, that is,

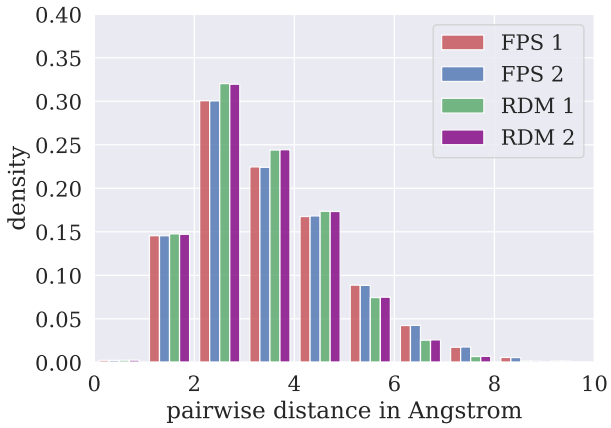



Fig. 1: The distributions of pairwise interatomic distances within a molecule for 5000 molecules sampled with either FPS or randomly (two different splits each) differ.

$$\text{MAE} := \sum_{i=1}^{N} |y_i - \tilde{y}_i|, \tag{18}$$

where $N \in \mathbb{N}$ is the number of data points in the test set used to evaluate the models. The root mean squared error (RMSE) computes the root of the mean squared error, which is the arithmetic average of the squared errors. It is a measure of how spread out the errors are and it is represented by the formula

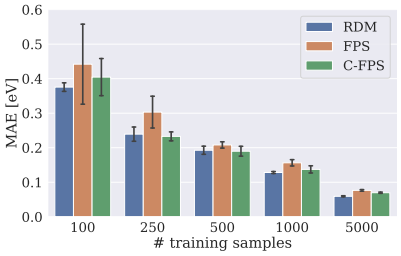
$$\text{RMSE} := \sqrt{\sum_{i=1}^{N} (y_i - \tilde{y}_i)^2}. \tag{19}$$

The worst-case error calculates the maximum absolute error between the predicted values and the ground truths. It is an indicator of the robustness of a model's predictions, and it is defined as

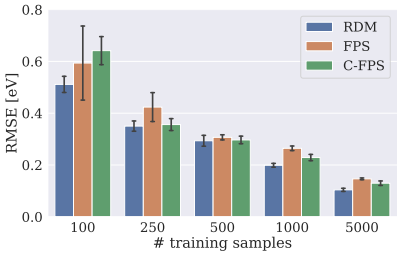
$$\text{worst-case error} := \max_{1 \le i \le N} |y_i - \tilde{y}_i|. \tag{20}$$

## 4.2 SchNet

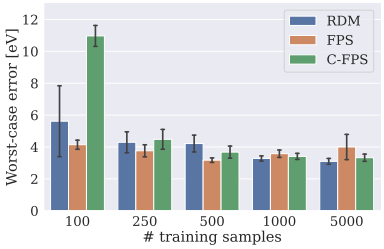
In order to get experimental insights on FPS also for a different class of informed predictive models, we train the publicly available implementation of SchNet from Pytorch Geometric [15] on the defined subsets. In this work, we choose a cutoff radius of 4 A while keeping the other hyperparameters to be the default ones suggested by the Pytorch Geometric implementation (version 2.0.4). Besides the test set that is




(a) MAE




(b) RMSE




(c) Worst-case error

Fig. 2: Results for SchNet.

used for final evaluation, we use random 20 % from the training set for evaluation during training and refer to it as validation set. We minimize the $L2$-loss function with respect to the model parameters with the Adam optimizer using mini-batches of 32 molecules per iteration and a learning rate of $7 \cdot 10^{-4}$. The learning rate is decayed by a factor of 0.8 if the validation error has not improved for 50 epochs. After each epoch a checkpoint is saved if the model has achieved a smaller validation loss than the current best model. The training process is stopped after the model has not improved for 200 epochs (early stopping). The best model is then used for assessing the model performance on the test set.

The first thing we observe is that SchNet does not seem to profit from FPS-based sampling strategies when examining the MAE and RMSE (Fig. 2a and 2b) alone. Random sampling consistently leads to approximately equal or smaller measurements for the MAE and RMSE for all investigated training set sizes. However, for 100, 250 and 500 training samples, the worst case error is reduced by at least 0.5 eV when employing FPS for the training set selection (Fig. 2b). Considering the comparatively small error bars, we expect FPS to be a reliable technique to reduce the worst case error for small (i.e. $\leq$ 500 data points) training sets of QM9. For larger training sets however, this effect vanishes and FPS leads to worse results in the sense of larger worst-case errors. This is possibly due to the fact that FPS is based on Mordred features which yield a rather global description of a molecule. In this sense, FPS selects samples that are maximally far away with respect to those global features. On the

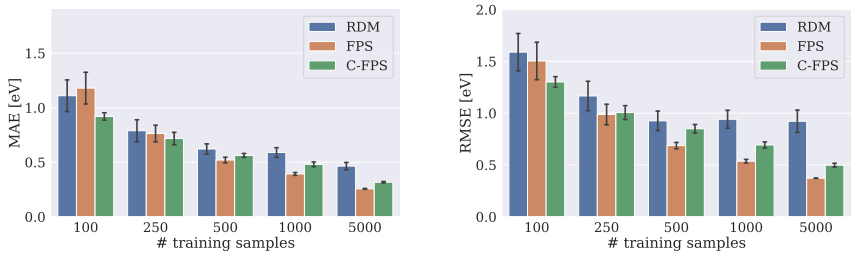
contrary, GNNs strongly exploit local information and we believe this discrepancy to be a possible explanation for the merely small effect induced by FPS. However, in absolute numbers, SchNet yields the lowest error metrics of all tested methods. This meets our expectations since it is the only method incorporating geometric information. In fact, the nuclear positions were obtained through DFT calculations and hence the coordinates already encode highly relevant information for predicting the atomization energy. One could argue that SchNet's input is already part of the solution to the problem and view the use of features derived by ab initio methods as some form of information leakage [16], thus making the learning problem easier.

### 4.3 Kernel Ridge Regression

The kernel and regression hyperparameters were optimized in a grid search for each of the randomly selected training sets of 1000 points. Specifically, we varied the kernel parameter '$\nu$' in the set $\{10^{-1}, 10^0, 10, 10^2, \ldots, 10^7\}$ and the regularization parameter '$\lambda$' in the set $\{10^{-12}, 10^{-10}, 10^{-8}, 10^{-6}, \ldots, 10^0\}$. We selected $\nu = 10^5$ and $\lambda = 10^{-12}$, which is the parameter combination that provides the best performance in terms of the MAE on a randomly chosen test set consisting of 10000 points not considered during training.

Of all predictive models that we investigated, Gaussian kernel regression appears to benefit the most from FPS-based sampling strategies in comparison to random sampling. In particular, FPS and C-FPS improve the obtained RMSE on the test set for all training set sizes as seen in Fig. 3b. However, it is noteworthy that random sampling leads to an increasing RMSE when going from 500 to 1000 or even 5000 training samples. For a possible explanation, we consider the MAE (Fig. 3a) and the worst case error (Fig. 3c). Even though we observe a decreasing MAE with an increasing training set size the worst case error becomes larger with more training samples as well, leading to a stagnating RMSE as it gives a higher weight to outliers than the MAE. FPS appears to alleviate this problem as becomes apparent when considering the comparatively small worst case errors.

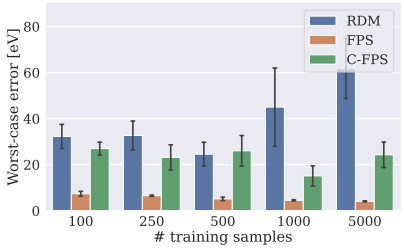
At this point, we can compare the worst case errors of the SMILES-based Kernel ridge regression (KRR) with the worst case error of SchNet. From Fig. 3c it is apparent that FPS significantly reduces the worst-case error of KRR by one order of magnitude compared to random sampling. In order to contextualize this effect better we consider Fig. 4 that shows the worst-case errors of SchNet and KRR side by side for different numbers of training samples. We find KRR to approach the values of SchNet with an increasing number of training samples. In particular, we observe the errors to have the same order of magnitude. This is noteworthy as the KRR only exploits topological information while SchNet requires the atom coordinates obtained from DFT as input.




(a) MAE




(b) RMSE




(c) Worst-case error

Fig. 3: Results for Gaussian Kernel Regression.

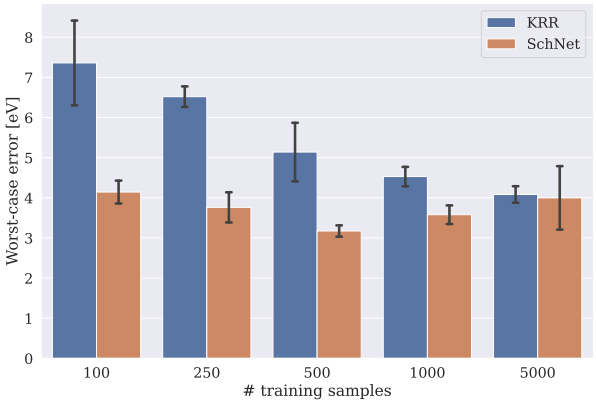
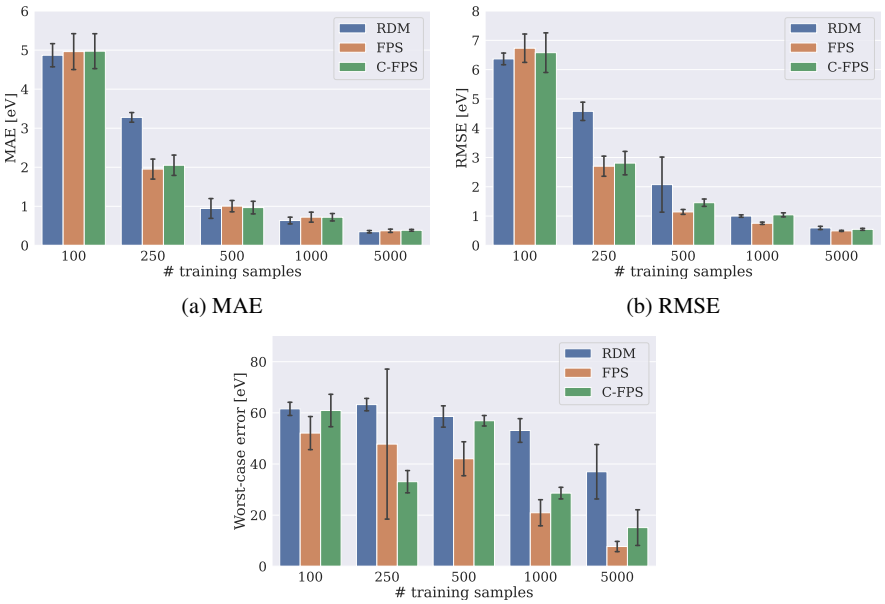



Fig. 4: The worst-case error of KRR can be reduced by FPS such that the order of magnitude is comparable to SchNet.

## 4.4 Spatial 3-Hop Convolution Network

In accordance with the previous sections, we train the method presented in Section 3.3 on subsets of QM9. The network consists of several updates by the spatial 3-hop




(a) MAE

(b) RMSE

(c) Worst-case error

Fig. 5: Results for spatial 3-hop convolution network

convolution layer, followed by an aggregation layer to obtain graph features which are further processed by linear layers. During training we minimize the L1-loss function with respect to the models parameters with the Adam optimizer, use a learning rate of $2 \cdot 10^{-4}$ and a batch size of 32 molecules per iteration. We train each model for 500 epochs and choose the best model with respect to a validation set (20% of the training set) for measuring the model performance on a test set.

Apart from the model trained on 250 samples, we do not observe significant differences among the different sampling strategies when considering only the MAE (Fig. 5a). When training on 250 samples, the FPS-based methods appear to lead to an advantage and reduce the MAE in comparison to random sampling. For larger training sets random sampling seems to catch up and perform on par with FPS-based sampling. Considering the RMSE (Fig. 5b), our observations are somewhat different. In particular, we find FPS and C-FPS to outperform random sampling for most sizes of the training set. In line with the other methods, FPS reduces the worst-case error in comparison to random sampling (Fig. 5c).

We observe comparatively large values for all metrics, especially for small training set sizes. For example, the MAE for 100 training samples obtained with FPS amounts to approximately 5 eV. This is around 4 times larger than what we measure for KRR and more than 10 times larger than the value of SchNet. This was to be expected, since both KRR and SchNet are relatively data efficient. We note that the good performace of SchNet is to be expected as it uses more features than the spatial 3-

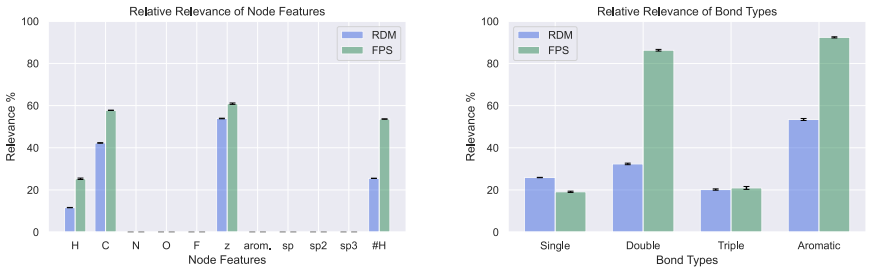
hop convolution network and especially uses the positions of the atoms (a powerful information which allows to compute the atomization energy explicitely). The KRR has the advantage of being a kernel method which has empirically shown to be effective in the realm of small datasets [45]. However, for larger training sets the relative difference between the methods becomes smaller: For 500 training samples, KRR yields only a two times smaller and SchNet only a 5 times smaller MAE. Moreover, the spatial 3-hop convolution network can benefit the most from larger datasets, i.e. we observe a significant improvement in performance whenever the size of the dataset is increased.

## 4.5 Explanation

With GRDE framework from Section 3.4 we now investigate the domain knowledge learned by the spatial 3-hop convolution network from Section 3.3 using the sampling strategies from Section 3.5.

### Setup of the Experiments

For the experiments, we utilize the spatial 3-hop convolution network from Section 3.3 that has been pre-trained using the sampling strategies from Section 3.5. More specifically, we compare explanations on the pre-trained model for the cases of random sampling (RDM) and diversity sampling (FPS) of 5000 samples as the training dataset. We fix the distortion measure $d$ as the $L^2$ distance for a regression task, and randomly initialize masks $S$. Furthermore, given the sparsity of the data, we also set a low value on $\lambda_A, \lambda_X = 20$ (which corresponds to choosing 10-15% of the non-zero elements in the respective masks) and set $(v_{S_A}, v_{S_X})$ to null. Since the QM9 dataset possesses edge features, we optimize $S_A = [S_{A_1}, S_{A_2}, ..., S_{A_h}]$ where $A_i$ is the adjacency matrix with respect to the edge feature $i$ $\forall i \in \{1, 2, ..., h\}$; $h$ being the number of edge features. The results that follow are obtained as an average over 3 independent runs on 100 graphs randomly sampled from the respective test datasets. Since the setup produces positive relevance masks, i.e., the masks only obfuscate features that exist for each node/edge, and do not show the relevance of the lack of a feature for a node/edge), we aggregate and average the node- and edge-wise scores to obtain feature-wise scores. Furthermore, we offset the imbalance in the scores by weighting them with respect to the frequency of their occurrence over the sampled data. Our explanation query is as follows: *For a randomly selected graph unseen by the pre-trained model, which features does the model consider important for its prediction?*




(a) Node feature explanations

(b) Edge feature explanations

Fig. 6: Results for feature-level GRDE.

### Results

From Fig. 6a and Fig. 6b, we see that the model trained using FPS places a stronger importance in both edge and node features than in the case of RDM, especially in the case of atomic numbers (Z), double bonds and aromatic rings. In comparison, single bonds, though the most frequent of the bond types in the sampled data are not considered as important. Furthermore, Fig. 6a shows that the model requires certain nodes to be specific atom types and have a certain type of neighborhood for its predictions, as can be seen from the atomic number (Z) as well as the importance of carbon (C) and the number of hydrogen atoms (#H) surrounding the node in the case of FPS. In the case of RDM, we also have a clear indication that edge features are not as important as the node features, whereas this is significantly more balanced in the case of FPS. Finally, we find that, though there exist non-zero values for some node features in the sampled graphs such as in the case of Nitrogen (N) and Oxygen (O), GRDE does not attribute any importance to them. This implies that the model treats these features as noise and ignores them regardless of the sampling strategy used.

## 5 Conclusion

In this work, we employed three informed ML models to predict the atomization energy of molecules in the QM9 dataset. We used KRR with a kernel obtained from molecular topological features, a geometry-based GNN (SchNet) and a topology-based GNN. We saw that maximizing molecular diversity in the training set selection process improves the accuracy and robustness of those methods. Our main finding is that by training topology-based ML models with sets of diverse molecules, we can significantly reduce their test maximum absolute error, thus increasing their robustness to distribution shifts. For SchNet, this effect was still observable but only for small training sets. Moreover, by maximizing diversity in the training sets,

we could substantially reduce the gap between the maximum absolute errors of a topology-based regression method as KRR and the SchNet, which is a geometry-based GNN. This work proposes only an empirical investigation, in the field of molecular property prediction, on the effects of maximizing molecular diversity in the training set selection. Ongoing research seeks to provide a theoretical foundation for the observed empirical results.

We believe that reducing the worst-case error is of great importance for applications that require a high degree of robustness but have limited budget for data generation. One example would be the application of Machine Learning Interatomic Potentials (MLIPs) for molecular dynamics simulations. In this scenario, the predictions of the model are used to integrate the equations of motion and to compute particle trajectories. Thus, large errors in the predictions could potentially lead to a failure of the simulation and techniques to prevent this are needed. Investigating this scenario in particular, could be a direction of future research.

**Acknowledgements** This work was supported in part by the BMBF-project 05M20 MaGriDo (Mathematics for Machine Learning Methods for Graph-Based Data with Integrated Domain Knowledge) and in part by the Fraunhofer Cluster of Excellence »Cognitive Internet Technologies«.

## References

1. Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, 2023.
2. P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via coresets. In *Combinatorial and Computational Geometry*, volume 52, pages 1–30. Cambridge University Press, 2005.
3. James Barker, Laura-Sophie Berg, Jan Hamaekers, and Astrid Maass. Rapid prescreening of organic compounds for redox flow batteries: A graph convolutional network for predicting reaction enthalpies from SMILES. *Batteries & Supercaps*, 4(9):1482–1490, jun 2021.
4. Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P. Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E. Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13(1), may 2022.
5. Anton Bochkarev, Yury Lysogorskiy, Sarath Menon, Minaam Qamar, Matous Mrovec, and Ralf Drautz. Efficient parametrization of the atomic cluster expansion. *Phys. Rev. Materials*, 6:013804, Jan 2022.
6. Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik J Bekkers, and Max Welling. Geometric and physical quantities improve e(3) equivariant message passing. In *International Conference on Learning Representations*, 2022.
7. Vladimir Braverman, Dan Feldman, and Harry Lang. New frameworks for offline and streaming coreset constructions. *ArXiv*, abs/1612.00889, 2016.
8. Shengze Cai, Zhicheng Wang, Sifan Wang, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 143(6), 2021.
9. Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

10. Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3844–3852, Red Hook, NY, USA, 2016. Curran Associates Inc.
11. Volker L. Deringer, Albert P. Bartók, Noam Bernstein, David M. Wilkins, Michele Ceriotti, and Gábor Csányi. Gaussian process regression for materials and molecules. *Chemical Reviews*, 121(16):10073–10141, 2021. PMID: 34398616.
12. Alexandre Duval and Fragkiskos D Malliaros. Graphsvx: Shapley value explanations for graph neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 302–318. Springer, 2021.
13. Y. Eldar, M. Lindenbaum, M. Porat, and Y.Y. Zeevi. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*, 6(9):1305–1315, sep 1997.
14. Dan Feldman. Core-sets: Updated survey. In *Sampling Techniques for Supervised or Unsupervised Tasks*, pages 23–44. Springer International Publishing, oct 2019.
15. Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.
16. Johannes Gasteiger, Chandan Yeshwanth, and Stephan Günnemann. Directional message passing on molecular graphs via synthetic coordinates. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.
17. Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1263–1272. JMLR.org, 2017.
18. M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks*, volume 2, pages 729–734 vol. 2, 2005.
19. Tim Gould and Stephen G. Dale. Poisoning density functional theory with benchmark sets of difficult systems. *Phys. Chem. Chem. Phys.*, 24:6398–6403, 2022.
20. Quercus Hernández, Alberto Badías, Francisco Chinesta, and Elías Cueto. Thermodynamics-informed graph neural networks. *arXiv preprint arXiv:2203.01874*, 2022.
21. Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 2022.
22. Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-softmax. In *International Conference on Learning Representations*, 2017.
23. Wayne H Joerding and Jack L Meador. Encoding a priori information in feedforward networks. *Neural Networks*, 4(6):847–856, 1991.
24. Peter Bjørn Jørgensen, Karsten Wedel Jacobsen, and Mikkel Nørgaard Schmidt. Neural message passing with edge updates for predicting properties of molecules and materials. In *32nd Conference on Neural Information Processing Systems, NIPS 2018*, 2018.
25. Yeonjoon Kim, Jaeyoung Cho, Nimal Naser, Sabari Kumar, Keunhong Jeong, Robert L McCormick, Peter C St John, and Seonah Kim. Physics-informed graph neural networks for predicting cetane number with systematic data quality analysis. *Proceedings of the Combustion Institute*, 2022.
26. Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017. https://openreview.net/forum?id=SJU4ayYgl.
27. Stefan Kolek, Duc Anh Nguyen, Ron Levie, Joan Bruna, and Gitta Kutyniok. A rate-distortion framework for explaining black-box model decisions. *xxAI - Beyond Explainable AI*, page 91–115, 2022.
28. Stefan Kolek, Robert Windesheim, Hector Andrade Loarca, Gitta Kutyniok, and Ron Levie. Explaining image classifiers with multiscale directional image representation. *arXiv preprint arXiv:2211.12857*, 2022.
29. Mark A Kramer, Michael L Thompson, and Phiroz M Bhagat. Embedding theoretical models in neural networks. In *1992 American Control Conference*, pages 475–479. IEEE, 1992.

30. Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability*, 3:71–104, 2014.
31. H J Kulik, T Hammerschmidt, J Schmidt, S Botti, M A L Marques, M Boley, M Scheffler, M Todorović, P Rinke, C Oses, A Smolyanyuk, S Curtarolo, A Tkatchenko, A P Bartók, S Manzhos, M Ihara, T Carrington, J Behler, O Isayev, M Veit, A Grisafi, J Nigam, M Ceriotti, K T Schütt, J Westermayr, M Gastegger, R J Maurer, B Kalita, K Burke, R Nagai, R Akashi, O Sugino, J Hermann, F Noé, S Pilati, C Draxl, M Kuban, S Rigamonti, M Scheidgen, M Esters, D Hicks, C Toher, P V Balachandran, I Tamblyn, S Whitelam, C Bellinger, and L M Ghiringhelli. Roadmap on machine learning in electronic structure. *Electronic Structure*, 4(2):023004, jun 2022.
32. S. Y. Kung. *Kernel Methods and Machine Learning*. Cambridge University Press, apr 2014.
33. G. Landrum. Rdkit:. *Open-source cheminformatics*, 2012.
34. Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.
35. Zhiyuan Liu and Jie Zhou. Introduction to graph neural networks. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(2):1–127, 2020.
36. Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.
37. Junshui Ma, Robert P. Sheridan, Andy Liaw, George E. Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure–activity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, feb 2015.
38. Jan MacDonald, Stephan Wäldchen, Sascha Hauch, and Gitta Kutyniok. A rate-distortion framework for explaining neural network decisions. *arXiv preprint arXiv:1905.11092*, 2019.
39. Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.
40. Lucie Charlotte Magister, Dmitry Kazhdan, Vikash Singh, and Pietro Liò. GCExplainer: Human-in-the-loop concept-based explanations for graph neural networks. In *3rd ICML Workshop on Human in the Loop Learning*, 2021. arXiv preprint arXiv:2107.11889.
41. Michael W. Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, jan 2009.
42. Hirotomo Moriwaki, Yu-Shi Tian, Norihito Kawashita, and Tatsuya Takagi. Mordred: A molecular descriptor calculator. *Journal of Cheminformatics*, 10(1), feb 2018.
43. Tim Mueller, Aaron Gilad Kusne, and Rampi Ramprasad. Machine learning in materials science. In *Reviews in Computational Chemistry*, pages 186–273. John Wiley & Sons, Inc, may 2016.
44. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.
45. Max Pinheiro, Fuchun Ge, Nicolas Ferré, Pavlo O. Dral, and Mario Barbatti. Choosing the right molecular machine learning potential. *Chem. Sci.*, 12:14396–14413, 2021.
46. Phillip E. Pope, Soheil Kolouri, Mohammad Rostami, Charles E. Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10764–10773, 2019.
47. Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014.
48. Ribana Roscher, Bastian Bohn, Marco F. Duarte, and Jochen Garcke. Explainable Machine Learning for Scientific Insights and Discoveries. *IEEE Access*, 8(1):42200–42216, 2020.
49. 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. PMID: 23088335.

50. Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.
51. Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. Interpreting graph neural networks for {nlp} with differentiable edge masking. In *International Conference on Learning Representations*, 2021. https://openreview.net/forum?id=WznmQa42ZAx.
52. K. T. Schütt, P.-J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko, and K.-R. Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 992–1002, Red Hook, NY, USA, 2017. Curran Associates Inc.
53. Khemraj Shukla, Patricio Clark Di Leoni, James Blackshire, Daniel Sparkman, and George Em Karniadakis. Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks. *Journal of Nondestructive Evaluation*, 39:1–20, 2020.
54. Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.
55. 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. *Nature Communications*, 11(1):4428, sep 2020.
56. R Todeschini and V Consonni. *Molecular Descriptors for Chemoinformatics*. Wiley-VCH, 2009.
57. Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. https://openreview.net/forum?id=rJXMpikCZ.
58. Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Michał Walczak, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. Informed Machine Learning - A Taxonomy and Survey of Integrating Knowledge into Learning Systems. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):614–633, 2023.
59. David Weininger. SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Modeling*, 28(1):31–36, feb 1988.
60. Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.
61. Kai Yu, Jinbo Bi, and Volker Tresp. Active learning via transductive experimental design. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 1081–1088, New York, NY, USA, 2006. Association for Computing Machinery.
62. Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 430–438, 2020.
63. Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
64. Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. In *International Conference on Machine Learning*, pages 12241–12252. PMLR, 2021.
65. Viktor Zaverkin, David Holzmüller, Ingo Steinwart, and Johannes Kästner. Exploring chemical and conformational spaces by batch mode deep active learning. *Digital Discovery*, 2022.