Supplementary Information

Two-dimensional interaction parameter histograms as a simple and versatile nanoporous material representation for machine learning prediction of adsorption properties

Tatiane Gercina de Vilas,[1,#] Fernando Fajardo-Rojas,[2,4,#] Omar Mansurov,[1,3] Ruby Devaisher,[1]
Eric Toberer,[2,4] Diego A. Gómez-Gualdrón[1,4] *

[1] Department of Chemical and Biological Engineering, Colorado School of Mines, 1601 Illinois St., Golden CO 80401, USA
[2]Department of Physics, Colorado School of Mines, 1500 Illinois St., Golden CO 80401, USA
[3]Computer Science Department, Colorado School of Mines, 1500 Illinois St., Golden CO 80401, USA
[4] Materials Science Program, Colorado School of Mines, 1601 Illinois St., Golden CO 80401, USA

#These authors contributed equally
*Corresponding author: dgomezgualdron@mines.edu

Table of Content:

Computational Infrastructure Availability

All the computational infrastructure, examples and data used can be found in the following repository:
https://github.com/gomezgualdronlab/2D-IPHs_prediction_infrastructure

The computational infrastructure to build the 2D-IPHs representation can be found in the following repository:
https://github.com/JFajardoRojas/2D-IPHs_builder

Section S1. 2D-IPH example

Note that different MOFs usually have different number of grid points. However, the 2D-IPHs for all MOFs have the same number of bins. Briefly, to build each 2D-IPH, we divided *d* into 20 segments between 0 and 20 Å, and each of the interaction parameters into 74 segments with ε spanning 0-350 K, σ 0.0–4.54 Å, and q –3 to 3 *e*. Thus, each histogram has 20 x 74 = 1,480 bins (see example in **Fig. S1**). The value for each bin is the number of grid points that has particular (q,d) or (ε, d), or (σ, d) combination, normalized by the total number of grid points. In other words, each bin value is a probability density. Grid points with values outside the above ranges are counted in the closest segment (e.g., a grid point with d = 30 Å is counted in the d = 20 Å bin)..

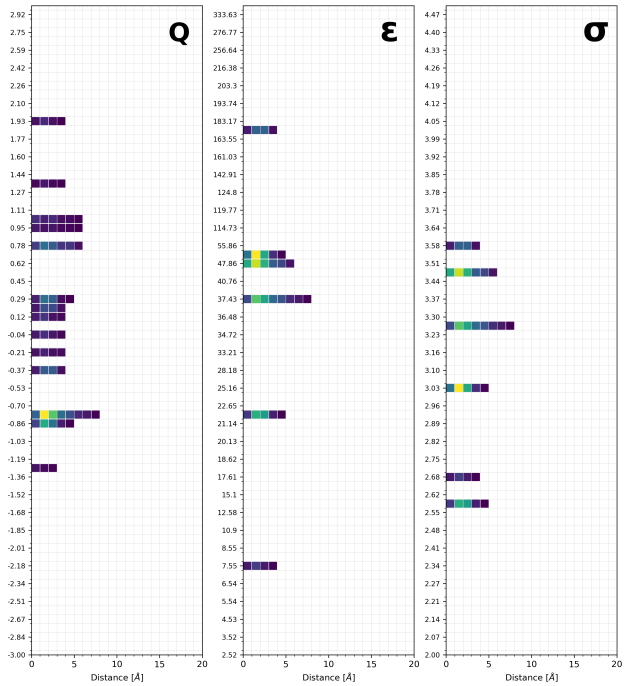

Fig. S1. Representative 2D-IPH for each interaction parameter for one of the MOFs. The value in each bin is a probability density calculated as the number of grid points falling in the distance and interaction parameter range spanned by the bin, normalized by the total number of grid points. White blocks represent bins with no identified interaction parameter-distance pair (probability density equal to zero). Colored blocks represent bins with probability densities higher than zero, with the color scale going from low density (purple) to high density (yellow).

Each histogram is flattened by concatenating their rows, and the flattened histograms are in turn concatenated to produce a feature vector with 1,480 x 3 = 4,440 features. The use of a variance filter is optional and is used as a "knob" to adjust the size of the MOF representation to adapt to variations in training dataset size and computational resources. The rational for the variance filter is the hypothesis that bins whose values are very similar across MOFs are less useful to discriminate MOFs based on their performance, and thus less informative for learning. The variance threshold used to eliminate bins can be considered as a hyperparameter.

Section S2. Adsorption Loading Prediction via General model

Section S2.1. Machine Learning Details

Additional data files for this section can be found at:
https://osf.io/vu3hn/overview
In the folder *General_Model_Data, SFS_Data* and *Scratch_Data*

Machine learning model description. Machine learning models used to predict adsorption loading correspond to fully connected feedforward neural networks (multilayer perceptron, MLPs) implemented using TensorFlow and Keras**.** The models were designed to perform supervised regression of adsorption loading from the assembled MOF–adsorbate–fugacity feature vectors.
Each model consists of:

- An input layer whose dimensionality matches the number of input features,
- A configurable stack of hidden layers,
- A single linear output neuron that predicts the scaled adsorption loading.

All hidden layers employ nonlinear activation functions selected via hyperparameter exploration using a Bayesian approach for the architectural configuration exploration. The output layer uses a linear activation function, consistent with continuous regression targets.

Training procedure. A Bayesian hyperparameter optimization workflow was implemented with KerasTuner (*BayesianOptimization*) on a fixed train/validation/test split. In each trial, a feed-forward multi-layer perceptron (MLP) was trained with Adam for up to 100 epochs (batch size 256), with early stopping on validation loss (MAE) (patience 10, best weights restored) and ***ReducedLROnPlateau*** learning-rate decay. The Bayesian search minimized validation MAE over 20 trials across the space: number of layers (2–5), units per layer (64 / 128 / 256 / 512), dropout (0.0 – 0.4), L2 regularization (1e-6 to 1e-3), learning rate (1e-4 to 3e-3), and activation (ReLU or GELU). After search, the best-performing model was selected, evaluated on the held-out test set (MAE/ root mean squared error RMSE/R^2, including per-adsorbate metrics), and exported together with the fitted preprocessor for fully reproducible inference.

Further details can be found in the repository:
https://github.com/gomezgualdronlab/2D-IPHs_prediction_infrastructure/tree/main/2D-IPHS-Results-General-model

Data preprocessing and scaling. Data processing is implemented as a scikit-learn *ColumnTransformer* fitted only on the **training split** and then reused unchanged for validation/test (to avoid data leakage). The model inputs are assembled from 2D-IPHs (named using the convention "feature_*"), molecular descriptors (chg, bond_length, eps_eff, sig_eff), fugacity, and textural descriptors (LPD, PLD, SA_grav (GSA), VF, PSSD, density (inverse MOF density)). These numerical features are processed with a pipeline of **median imputation** followed by **z-score standardization** (*StandardScaler*).
The fitted preprocessing object is saved (preprocessor.joblib) and consistently applied during training, and inference so all downstream results are in the same transformed feature space.

Reproducibility and workflow automation. All model training, hyperparameter exploration, and complementary files generation were implemented in Python 3.11.5. Hyperparameter combinations were generated systematically, and model configurations were identified using a consistent naming convention encoding activation function, optimizer, learning rate, metric, number of nodes, and number of layers.

This fully scripted approach enables reproduction of individual model configurations and traceability between trained models, preprocessing scalers, and validation performance.

Here we present the combination of parameters that showed with the best performance and therefore were selected for the *general adsorption model*.

Table S1. Optimal parameters for the general adsorption model

Parameter	Value selected
Input dimensions (features)	55
Hidden layers (count)	2
Hidden layers (widths)	128, 256

Hidden activation	GELU
Output layer	Dense (1), linear activation
Loss	MSE
Optimizer	Adam
Learning rate	1.0×10^{-4}
Total parameters (weights + biases)	40,449

Section S2.2. Training and testing details of the General Adsorption Model

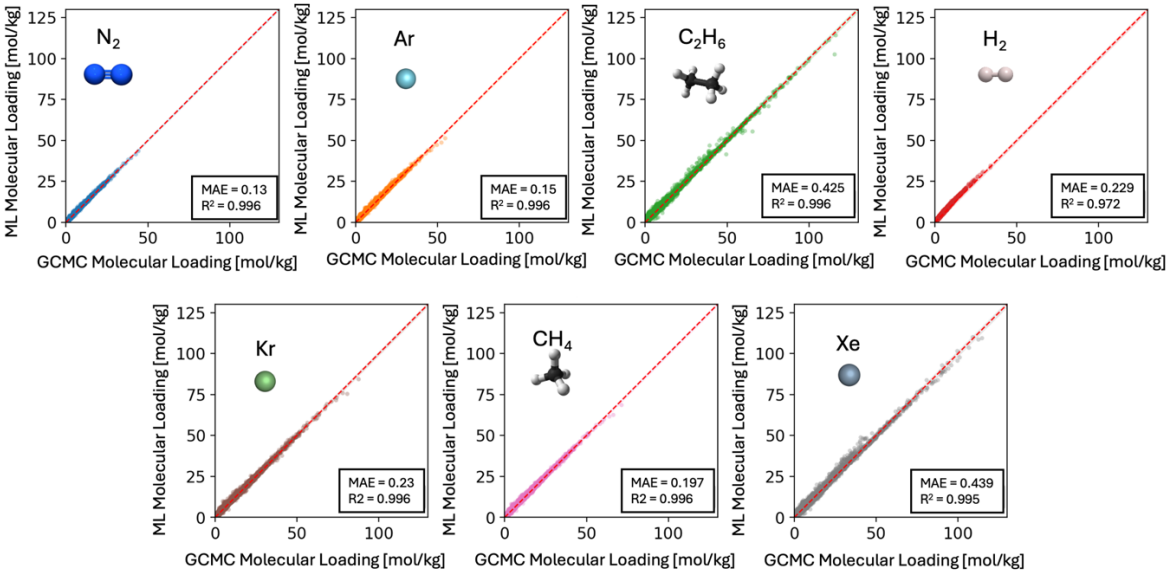

Fig. S2. Molecule adsorption loading prediction on N_2, Ar, C_2H_6, H_2, Kr, CH_4, and Xe. Each plot shows a 95% confidence interval for the prediction. R^2, and mean absolute error (MAE) in each plot.

Dense sampling at low fugacity was employed to anchor adsorption behavior in the low-loading regime, while higher fugacity points ensure continuity with the full pressure grids. Separate anchor sets were defined for training, validation, and testing datasets (**Table S2**).

Table S2. Fugacities used in GCMC simulations, and used in training, validation and testing of the general adsorption model.

Fugacity (bar)		
Validation	**Training**	**Test**
0.02	0.01	0.02
0.04	0.025	0.04
0.06	0.05	0.06

0.08	0.075	0.08
0.2	0.1	0.2
0.4	0.25	0.4
0.6	0.5	0.6
0.8	0.75	0.8
2.5	1	1
30	5	2.5
60	10	5
80	50	10
90	75	25
-	10	50
-	-	100

Section S2.3. Summary of the Data Used in All the Studies

General model data summary (Data from Anderson et al.[1])

Training	4,997,268	MOF + adsorbate + fugacity (200 alchemical adsorbates)
Validation	517,120	MOF + adsorbate + fugacity (200 alchemical adsorbates)
Testing	122,584	MOF + adsorbate + fugacity (7 real molecules* + Helium)

* 7 molecules considered (N_2, Ar, C_2H_6, H_2, Kr, CH_4, Xe)

SFS and Scratch model data summary (GCMC data collected in this study)

Training	12,353	(12,550)	MOF + fugacity + CO_2 (C_3H_8)
Validation	875	(1,198)	MOF + fugacity + CO_2 (C_3H_8)
Testing	6,945	(6,817)	MOF + fugacity + CO_2 (C_3H_8)

Henry's coefficient data summary (Widom Insertion data collected in this study)

Training	25,568	MOF for each molecule considered*
Validation	6,392	MOF for each molecule considered*
Testing	7,990	MOF for each molecule considered*

* 4 molecules considered (H_2O, NH_3, CO_2, N_2)

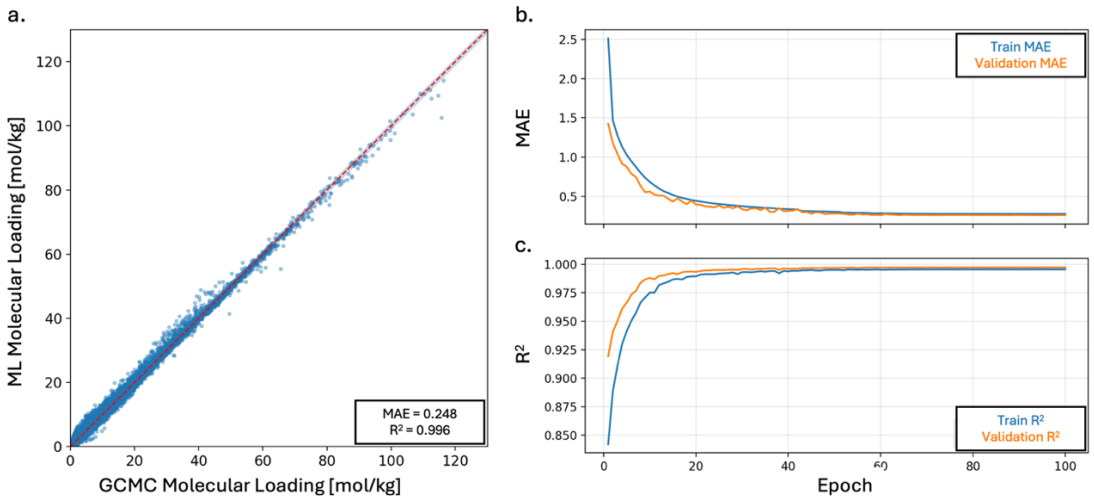

Fig. S3. General model results. a) Parity plot of the GCMC data versus ML predicted data in the test set. b) Learning curves of the Mean Absolute Error (MAE), and c) R^2 as a function of the evaluated epochs.

Section S3. Adsorption Loading Prediction via Single-Feature Stacking (SFS) Model

Section S3.1. Machine Learning Details

Scratch model: A direct multi-layer perceptron MLP regressor trained end-to-end on the feature space composed by 2D-IPHs (44 features, pruned as described in the manuscript), fugacity, textural descriptors (LPD, PLD, GSA, VF, PSSD, density) on the training/validation/test split to predict directly molecular loading isotherms for CO_2 and C_3H_8. Note that there is no description of the molecular entity since each model is train specifically for a molecule.

SFS model: This approach consists in a two-stage setup. First, an existing expert/base model predicts loading for each sample (This model is the described *General model*). Then, that prediction is appended as a new feature to the downstream feature set (same as *General model*), while some molecule-specific descriptors are dropped (chg, bond_length, eps_eff, sig_eff). A second MLP then is trained on this stacked feature space to predict the molecular loading of an specific molecule, either CO_2 or C_3H_8.

Here, we present the model parameters obtained for Scratch and SFS models to predict CO2 and C3H8 after hyperparameter optimization (**Table S3**).

Table S3. Parameters obtained for CO_2 and C_3H_8 models training at 10% data availability (Scratch and SFS) after hyperparameter exploration.

Parameter	Scratch-CO_2	Scratch-Propane	SFS-CO_2	SFS-Propane
Input dimension (features)	51	51	52	52
Hidden layers (count)	3	3	3	3
Hidden layer widths	256, 128, 64	256, 128, 64	256, 128, 64	256, 128, 64
Hidden activation	ReLU	ReLU	ReLU	ReLU
Output layer	Dense(1), linear	Dense(1), linear	Dense(1), linear	Dense(1), linear
Loss	MSE	MSE	MSE	MSE
Optimizer	Adam	Adam	Adam	Adam
Learning rate	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-3}	1.0×10^{-3}
Total parameters	54,529	54,529	54,785	54,785

Section S3.2. Training details of the Single-Feature Stacking (SFS) Data

Table S4. Fugacities used in GCMC simulations of CO_2 and C_3H_8. Separate pressure sets were employed for training, validation, and testing datasets, with grids defined conditionally on the parent MOF void fraction (VF).

Void fraction regime	Fugacity (bar)		
	Validation	Training	Test
VF < 0.8	0.02	0.01	0.04

	0.08	0.05	0.06
	0.3	0.1	0.6
	1.5	0.5	1
	7	2	15
	50	10	50
	-	80	75
VF $\geq$ 0.8	0.02	0.01	0.04
	0.2	0.1	0.06
	2	2	0.6
	20	10	1
	50	40	15
	90	70	50
	-	100	75

Fugacity grids were selected to span low- to high-loading regimes while minimizing overlap between training, validation, and testing datasets.

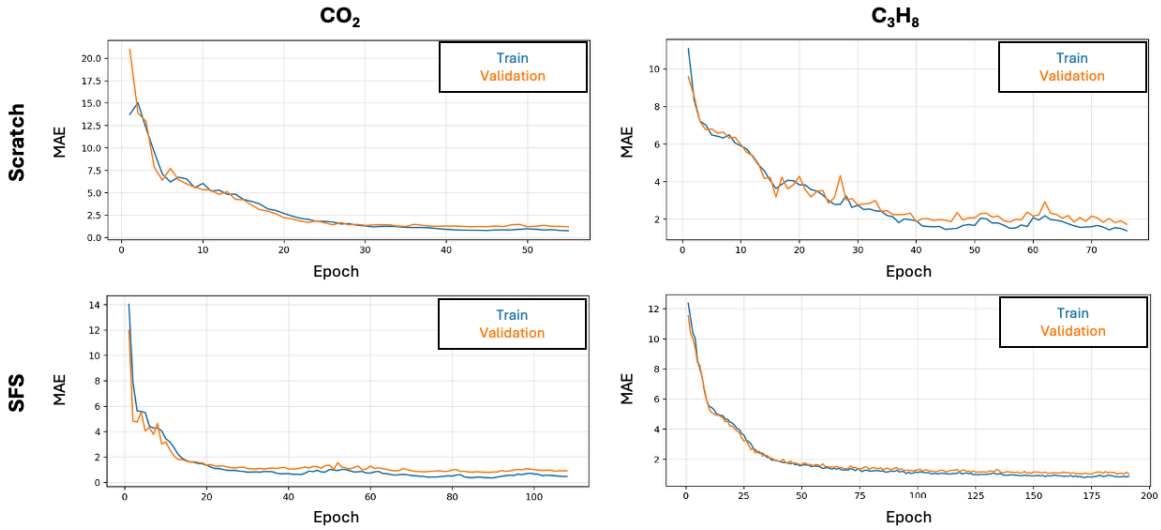

Fig. S4. Learning curves for CO_2 and C_3H_8 prediction via SFS and Scratch models using mean absolute error to follow the model progress through epochs used. These learning curves are the result of training these strategies on 10% of the available data. Training curve in blue and validation curve in orange.

Section S4. Henry's Constant Prediction at full data availability

Section S4.1. Machine Learning Details

Additional data files for this section can be found at:
https://osf.io/vu3hn/overview
In the folder *Henry_Coeff_training_extra_documentation*

Specific dependencies used to construct K_H prediction models. Machine learning models and accompanying data analysis were implemented in Python 3.12.2, for which relevant dependencies can be found in **Table S5.**

Table S5. Computational dependencies used in the training validation and testing of the ML models to predict K_H using the 2D-IPH.

Dependency	Purpose
joblib1.5.0	Feature/target scaler saving
matplotlib3.10.3	Plotting
numpy2.1.3	Data processing and optimized data structure use
pandas2.2.3	Data processing and handling
scikit-learn1.6.1	Machine learning metric calculation, data processing, model performance evaluation
scipy1.15.3	Model performance evaluation and plotting
tensorflow2.19.0	Machine learning model training and saving.

Data Processing. After matching the histograms with corresponding K_H values for CO_2, H_2O, N_2, and NH_3, the resulting feature and target data was randomly split into training, validation, and test sets for unbiased model training, with split amounts described in **Table S6**. There is no overlap between each set of data, and the K_H parameters are transformed into $\log_{10}(K_H)$ values to ensure that the target data follows a pseudo-normal distribution for model prediction.

Table S6. Distribution of the available K_H subset in training, validation and testing. Count of the actual number of structures in each partition is included.

Dataset	% Of Total Data	Count of Structures
Training	64%	25,568
Validation	16%	6,392
Test	20%	7,990

Prior to model training, 2D-IPHs data (4,440 histogram features) and target data (4 K_H targets) were scaled using the *scikit-learn StandardScaler* method. Scalers used in the training set to avoid leaking validation and test data into downstream model training efforts.

Scratch Model Training. For all model training and hyperparameter exploration, a fixed training/validation/test split was used, with test data held out from model development.
Hyperparameters were optimized via *GridSearchCV* (*scikit-learn*) over the discrete parameter grids in **Table S7**. Models were trained for 100 epochs, at which point performance gains plateaued. Selection was based on test-set MSE, with R^2 used as tiebreaker when

needed. The search proceeded coarse-to-fin, with optimizer type fixed early and activation functions selected after stabilizing architectural parameters.

Because MLP and CNN architectures share the same base structure, their hyperparameters searches were conducted jointly, where convolution-specific parameters collapse to zero when the MLP outperforms the CNN. Input histograms were reshaped to 2D to enable convolutional operations when needed.

As shown in **Table S7**, the best performing MLP architecture consists of five layers (one input, three hidden, and one output). Dropout of 30% was applied after each hidden layer. The hidden layers use Sigmoid, Sigmoid, and ReLU activation functions, respectively.

Table S7. Parameters used during the hyperparameter exploration of MLPs and CNNs to predict K_H. The table shows the range or specific values used, along the selected parameters that describe an MLP as the model with the best performance.

Hyperparameter	**Tested Value Range**	**Tested Values**	**Final Value**
No. Convolutional Layers (C_L)	0 - 1	-	0
No. Convolution Filters (C_F)	16 - 256	-	-
Kernel Size (S_K)	-	(3, 3), (5, 5), (7, 7), (9, 9)	-
Stride Size (S_S)	-	(3, 3), (5, 5), (7, 7), (9, 9)	-
No. Dense Layers (N_D)	0 - 5	-	3
No. Dense Neurons (N_{DN})	8 - 256	-	128 - 128 - 128
Dropout (D)	0 - 0.5	-	0.3
Batch Size (B)	2 - 512	-	32
Learning Rate (L)	0.0001 - 0.0200	-	0.0014
Optimizer (O)	-	Adam, RMSprop, Momentum	Adam
Activation (A)	-	Sigmoid (S), ReLu (R)	S - S - R

To further test the data splitting distributions, we divided the data in the training/validation/test using three different distributions but following the same percentages. Models in each data splitting were trained and compared in performance. A comparison between prediction accuracies for these new data splits and the scratch model performance are included in **Table S8**.

Table S8. Performance comparison of the models created to predict each molecule K_H from scratch using 4 different splitting distributions (Scracth—the one reported, and 1 to 3 data splits). Mean squared error (MSE) and R^2 used as performance metrics.

Split	**Metric**	**CO_2**	**H_2O**	**N_2**	**NH_3**
Scratch	MSE	0.251	0.184	0.097	0.177
	R^2	0.756	0.819	0.902	0.823
1	MSE	0.277	0.189	0.094	0.187
	R^2	0.739	0.814	0.908	0.819
2	MSE	0.267	0.186	0.109	0.186

	R^2	0.750	0.824	0.891	0.827
3	MSE R^2	0.274 0.742	0.184 0.820	0.097 0.903	0.172 0.829

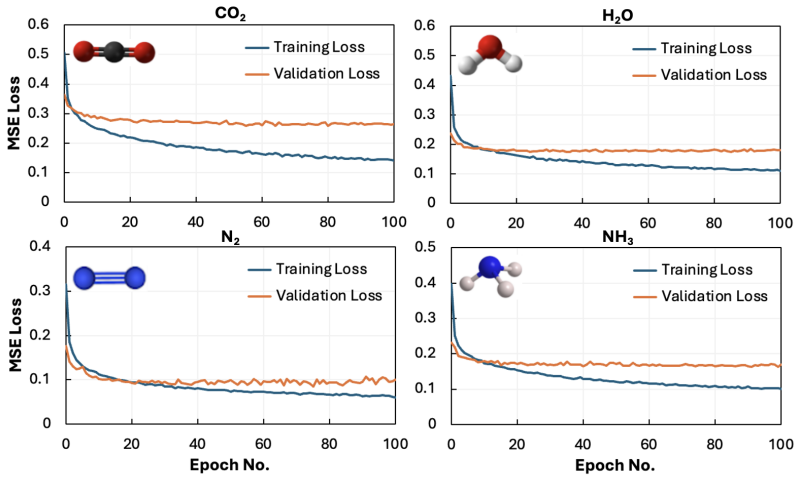

Figure S5. Learning rate curves for the training of the scratch model to predict K_H in CO_2, H_2O, N_2, and NH_3.

Section S5. Henry's Constant Prediction via Transfer Learning

Section S5.1. Machine Learning Details

Transfer Model Training. Frozen representations were obtained by passing input features through a pretrained scratch model and extracting the activations of the final frozen layer (a 128-dimensional vector corresponding to 128 neurons). These activations were then used as input features to train a new MLP composed of the remaining layers with the new target task. Accordingly, each transfer learning configuration consists of a pretrained "*base*" model providing frozen feature representations and a newly trained "*transfer*" model that learns the target-specific mapping. The intermediate features extracted from the base model were used without additional scaling. Because the scratch model consists of three hidden layers, transfer learning was evaluated by freezing layer 1, layers 1-2, and layers 1-3, followed by retraining the remaining unfrozen layers, including the output layer. A summary of transfer learning performance across all configurations is provided in **Table S9**. This comparison corresponds to the training using 100% of the data available.

Table S9. Performance comparison of the models created to predict each molecule K_H using transfer learning strategy. Mean squared error (MSE) and R^2 used as performance metrics. Results present the performance when 1, 1 and 2, and 1 to 3 hidden layers are frozen. For each considered target molecule (H_2O, N_2, NH_3, and CO_2).

Base Target	Transfer Target	Metric	Freeze 1	Freeze 1-2	Freeze 1-3
CO_2	H_2O	MAE R^2	1.292 0.790	1.525 0.682	1.905 0.528
	N_2	MAE R^2	0.124 0.804	0.187 0.588	0.219 0.429
	NH_3	MAE R^2	0.991 0.779	1.097 0.697	1.392 0.562

H_2O	CO_2	MAE R^2	0.414 0.680	0.447 0.610	0.569 0.467
	N_2	MAE R^2	0.135 0.774	0.188 0.593	0.245 0.303
	NH_3	MAE R^2	0.950 0.795	0.962 0.781	1.075 0.715
N_2	CO_2	MAE R^2	0.415 0.661	0.527 0.475	0.628 0.249
	H_2O	MAE R^2	1.318 0.762	2.055 0.448	2.393 0.154
	NH_3	MAE R^2	1.013 0.746	1.606 0.404	1.813 0.223
NH_3	CO_2	MAE R^2	0.417 0.687	0.422 0.637	0.536 0.510
	H_2O	MAE R^2	1.356 0.787	1.192 0.798	1.507 0.682
	N_2	MAE R^2	0.124 0.799	0.182 0.604	0.242 0.307

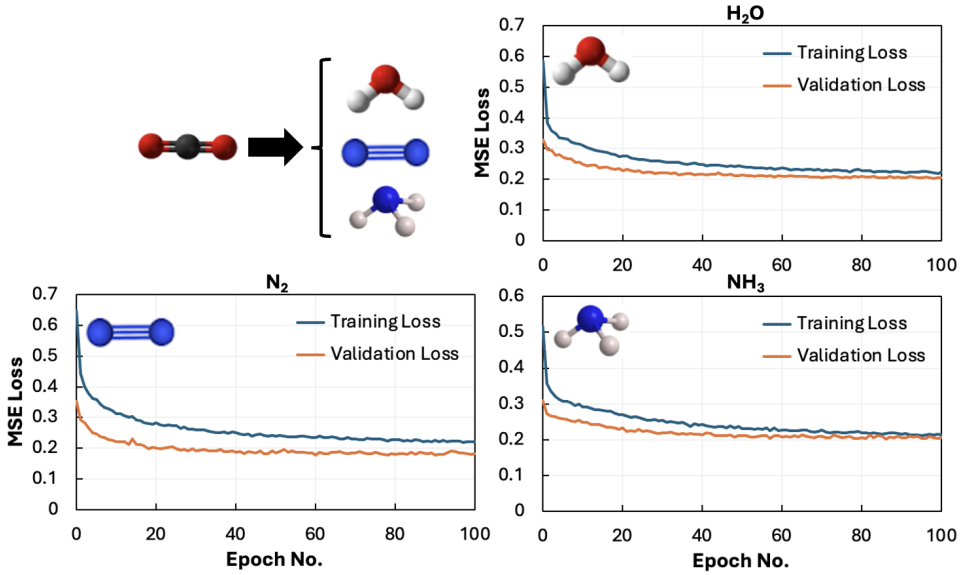

Figure S6. Learning curves for transfer learning with layer 1 frozen and source task CO_2 to predict H_2O, N_2, and NH_3.

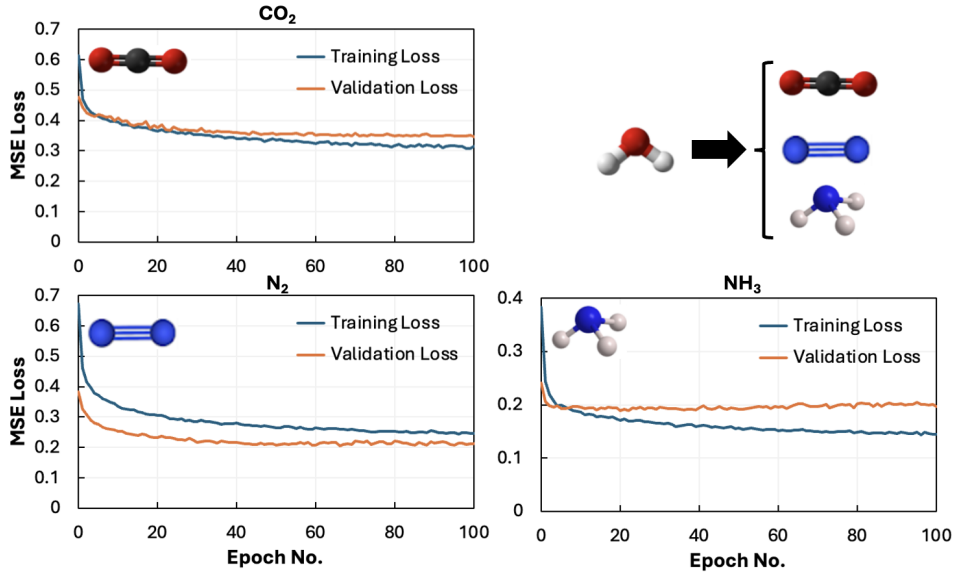

Figure S7. Learning curves for transfer learning with layer 1 frozen and source task H_2O to predict CO_2, N_2, and NH_3.

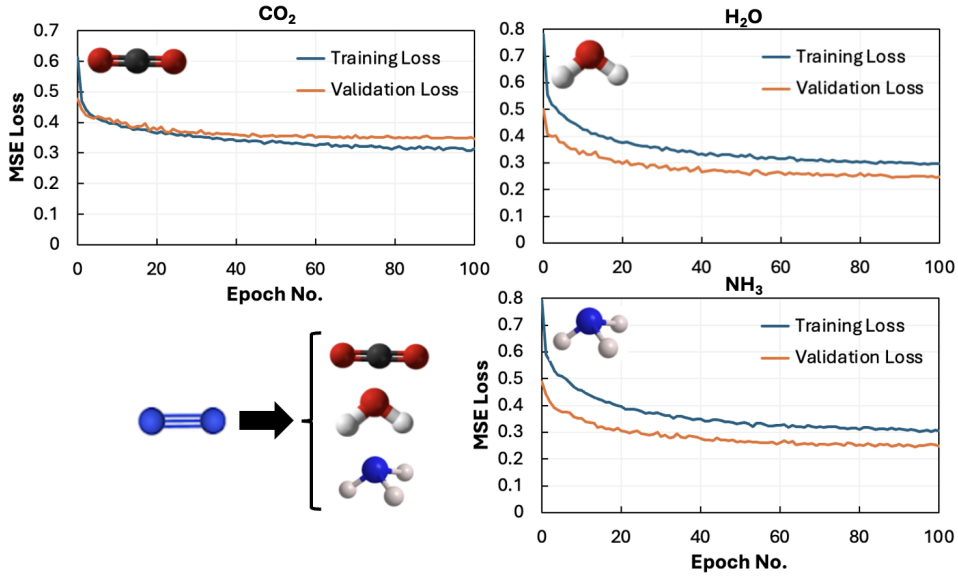

Figure S8. Learning curves for transfer learning with layer 1 frozen and source task N_2 to predict H_2O, CO_2, and NH_3.

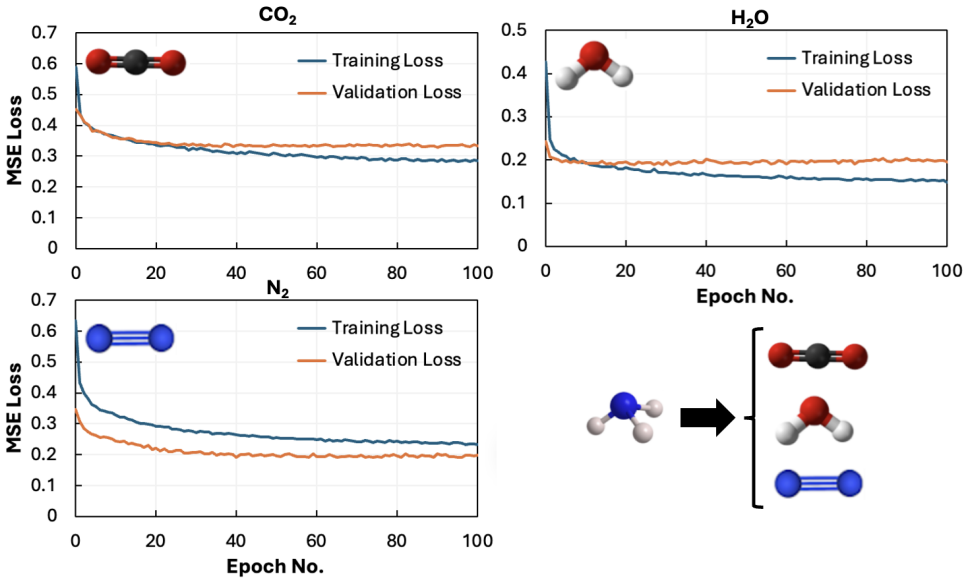

Figure S9. Learning curves for transfer learning with layer 1 frozen and source task NH_3 to predict N_2, H_2O, and CO_2.

Data Scarcity. Different levels of data availability were used to test the effect of data-scarcity in the transfer learning strategy. The data-scarce scenarios studied were 50%, 30%, 10%, 5%, 3%, 1% of the available data per molecule. The test dataset was kept fixed and not used in any training process to prevent data leakage. The data-scarce datasets were subsequently processed through the same machine learning pipelines used in the full-data experiments. When transfer learning was evaluated under data-scarce conditions, the reduced datasets were used exclusively to train the transfer learning models, while the source model corresponded to the scratch model trained on the full dataset.

A full summary of all models trained with data scarcity can be found in **Tables S10** to **Table S17.**

Table S10. Performance summary of the CO_2 as source task to transfer for the considered molecules. (-) represent scratch model. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Base Target	Transfer Target	% Data	MAE	R^2	Top 800	Top 400	Top 200
CO_2	-	50%	0.330	0.740	556	230	98
		10%	0.357	0.661	529	211	92
		5%	0.413	0.604	494	197	78
		3%	0.421	0.539	494	199	68
		1%	0.559	0.259	375	138	42
	H_2O	50%	1.270	0.787	637	281	120
		10%	1.377	0.752	629	278	117
		5%	1.426	0.723	610	268	116
		3%	1.504	0.689	567	266	118
		1%	1.531	0.673	565	258	104
	N_2	50%	0.127	0.795	636	293	139
		10%	0.132	0.771	614	277	129
		5%	0.144	0.729	592	266	124
		3%	0.154	0.694	590	253	126

		1%	0.178	0.605	562	252	114
		50%	0.967	0.784	601	263	107
		10%	0.981	0.747	567	239	100
	NH_3	5%	1.064	0.720	554	246	101
		3%	1.093	0.706	553	248	107
		1%	1.189	0.671	494	240	106

Table S11. Performance summary of the H_2O as source task to transfer for the considered molecules. (-) represent scratch model. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Base Target	Transfer Target	% Data	MAE	R^2	Top 800	Top 400	Top 200
H_2O	-	50%	1.142	0.807	635	287	115
		10%	1.262	0.779	617	279	111
		5%	1.352	0.737	613	277	113
		3%	1.450	0.694	598	260	101
		1%	1.707	0.571	548	252	74
	CO_2	50%	0.417	0.669	520	224	95
		10%	0.410	0.643	505	204	92
		5%	0.432	0.626	504	199	89
		3%	0.453	0.599	487	202	86
		1%	0.490	0.528	436	177	76
	N_2	50%	0.134	0.777	629	283	133
		10%	0.149	0.724	601	277	117
		5%	0.155	0.693	589	266	114
		3%	0.167	0.659	564	268	121
		1%	0.203	0.490	535	248	119
	NH_3	50%	0.974	0.788	595	261	116
		10%	0.907	0.793	595	260	114
		5%	0.917	0.786	591	260	107
		3%	0.921	0.779	592	246	105
		1%	0.916	0.776	593	253	108

Table S12. Performance summary of the N_2 as source task to transfer for the considered molecules. (-) represent scratch model. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Base Target	Transfer Target	% Data	MAE	R^2	Top 800	Top 400	Top 200
N_2	-	50%	0.082	0.895	700	323	156
		10%	0.099	0.853	661	289	139
		5%	0.111	0.811	623	276	124
		3%	0.124	0.791	608	268	129
		1%	0.189	-0.022	569	241	108
	CO_2	50%	0.386	0.659	532	213	93
		10%	0.412	0.613	496	206	87
		5%	0.438	0.587	489	201	77
		3%	0.444	0.550	474	202	76
		1%	0.516	0.466	443	175	62
	H_2O	50%	1.340	0.743	622	284	119

		10%	1.481	0.689	598	279	108
		5%	1.639	0.629	562	270	116
		3%	1.669	0.592	553	248	100
		1%	2.039	0.480	519	255	87
		50%	1.016	0.732	576	249	110
		10%	1.118	0.673	553	228	84
	NH_3	5%	1.260	0.605	507	213	84
		3%	1.311	0.575	497	210	88
		1%	1.434	0.468	451	172	72

Table S13. Performance summary of the NH_3 as source task to transfer for the considered molecules. (-) represent scratch model. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Base Target	Transfer Target	% Data	MAE	R^2	Top 800	Top 400	Top 200
		50%	0.842	0.809	594	267	111
		10%	0.900	0.767	578	243	101
	-	5%	0.974	0.722	537	240	94
		3%	1.022	0.716	561	240	93
		1%	1.214	0.616	517	204	79
		50%	0.396	0.691	527	220	101
		10%	0.402	0.669	504	214	88
	CO_2	5%	0.411	0.649	499	204	88
		3%	0.404	0.639	496	207	92
		1%	0.461	0.583	453	187	83
NH_3		50%	1.253	0.797	652	289	114
		10%	1.178	0.796	647	287	113
	H_2O	5%	1.231	0.789	643	278	106
		3%	1.274	0.785	624	274	110
		1%	1.304	0.771	634	271	109
		50%	0.134	0.772	635	279	135
		10%	0.135	0.758	614	271	118
	N_2	5%	0.148	0.709	587	256	113
		3%	0.158	0.681	586	247	111
		1%	0.205	0.464	576	254	114

Table S14. Relative performance of the scratch model to predict K_H of H_2O versus the use of NH_3 as source task to predict H_2O using transfer learning at different data-scarce scenarios. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Model	Data (%)	n = 800	n = 400	n = 200
Scratch	100	649	291	120
$NH_3 \rightarrow H_2O$		653	288	112
Scratch	50	644	287	119
$NH_3 \rightarrow H_2O$		652	289	114
Scratch	30	636	281	116
$NH_3 \rightarrow H_2O$		653	286	119

Scratch	10	612	276	106
$NH_3 \rightarrow H_2O$		647	287	113
Scratch	5	613	267	109
$NH_3 \rightarrow H_2O$		643	278	106
Scratch	3	576	252	102
$NH_3 \rightarrow H_2O$		624	274	110
Scratch	1	556	244	101
$NH_3 \rightarrow H_2O$		634	271	109

Table S15. Relative performance of the scratch model to predict K_H of NH_3 versus the use of H_2O as source task to predict NH_3 using transfer learning at different data-scarce scenarios. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Model	**Data (%)**	**n = 800**	**n = 400**	**n = 200**
Scratch	100	607	265	116
$H_2O \rightarrow NH_3$		602	268	122
Scratch	50	600	260	120
$H_2O \rightarrow NH_3$		595	261	116
Scratch	30	591	259	111
$H_2O \rightarrow NH_3$		591	266	114
Scratch	10	556	243	110
$H_2O \rightarrow NH_3$		595	260	114
Scratch	5	558	237	101
$H_2O \rightarrow NH_3$		591	260	107
Scratch	3	530	191	78
$H_2O \rightarrow NH_3$		592	246	105
Scratch	1	484	208	74
$H_2O \rightarrow NH_3$		593	253	108

Table S16. Relative performance of the scratch model to predict K_H of CO_2 versus the use of NH_3 as source task to predict CO_2 using transfer learning at different data-scarce scenarios. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Model	**Data (%)**	**n = 800**	**n = 400**	**n = 200**
Scratch	100	569	245	104
$NH_3 \rightarrow CO_2$		529	221	99
Scratch	50	566	243	101

$NH_3 \rightarrow CO_2$		527	220	101
Scratch	30	546	229	96
$NH_3 \rightarrow CO_2$		524	221	94
Scratch	10	511	208	83
$NH_3 \rightarrow CO_2$		504	214	88
Scratch	5	496	202	86
$NH_3 \rightarrow CO_2$		499	204	88
Scratch	3	500	192	75
$NH_3 \rightarrow CO_2$		496	207	92
Scratch	1	433	165	50
$NH_3 \rightarrow CO_2$		453	187	83

Table S17. Relative performance of the scratch model to predict K_H of N_2 versus the use of NH_3 as source task to predict N_2 using transfer learning at different data-scarce scenarios. The table includes statistics of the practical test conducted in the identification of the top-*n* MOFs using simulated K_H as reference.

Model	**Data (%)**	**n = 800**	**n = 400**	**n = 200**
Scratch	100	709	335	158
$NH_3 \rightarrow N_2$		645	281	134
Scratch	50	702	328	159
$NH_3 \rightarrow N_2$		635	279	135
Scratch	30	700	319	155
$NH_3 \rightarrow N_2$		629	269	131
Scratch	10	665	300	132
$NH_3 \rightarrow N_2$		614	271	118
Scratch	5	628	280	128
$NH_3 \rightarrow N_2$		587	256	113
Scratch	3	626	278	125
$NH_3 \rightarrow N_2$		586	247	111
Scratch	1	603	252	85
$NH_3 \rightarrow N_2$		576	254	114

Section S6. Model variation tests

Section S6.1. Performance Comparison of Different Representation Variants in the Prediction of Molecular Loading and Henry's Coefficient

For molecular loading, we assessed the performance of two representation approaches: *i)* 2D-IPH only, and *ii)* 2D-IPH + textural properties. Textural properties include the largest pore diameter (LPD), pore limiting diameter (PLD), gravimetric surface area (SAG), void fraction (VF), and standard deviation of the pore size distribution (PSSD). Both representations performed comparably (MAE ~ 0.3-0.2, R^2 ~ 0.98-0.99) (**Fig. S10**). The 2D-IPH representation alone captures the essential structure-property relationships, with textural properties providing marginal gains in terms of R^2 and MAE.

For Henry's coefficient, we assessed the performance of three representation approaches: *i)* 2D-IPHs only, and *ii)* 2D-IPH + textural properties, and *iii)* textural properties only. The inclusion of textural properties to the 2D-IPH does not reflect on the prediction performance of the model (**Fig. S11**). Moreover, consistent with Henry's coefficient being interaction–dominated rather than geometry–driven, when considering textural properties representation alone the obtained model presents the worst performance (**Fig. S12**).

These observations are consistent with the hypothesis that the detailed energetic information encoded in 2D-IPHs is particularly useful for adsorption properties at low pressure (fugacity) and dilute regime. At high pressure (fugacity), where adsorption loadings start to be more limited by available physical space than by energetics, the more "summarizing" textural properties bring valuable information.

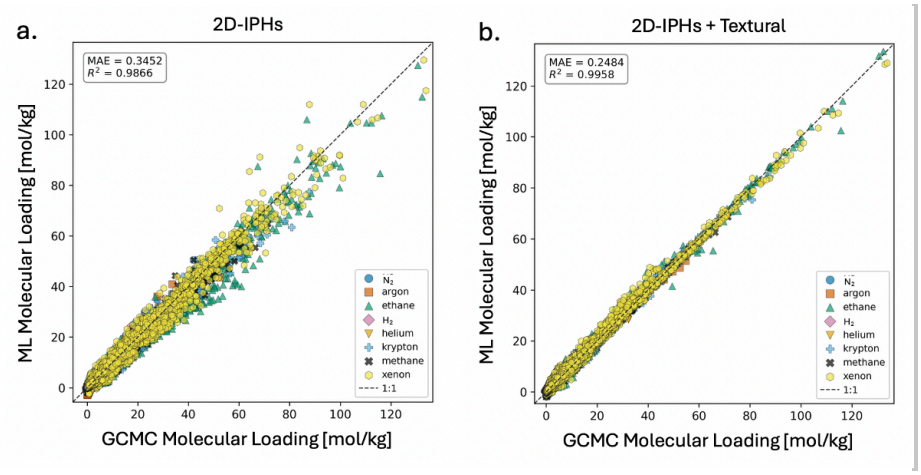

Fig. S10. a) ML predictions of loading by general model using only 2D-IPHs (reduced to only 44 features) as input. b) ML predictions of loading by general using 2D-IPHs along with textural properties as input.

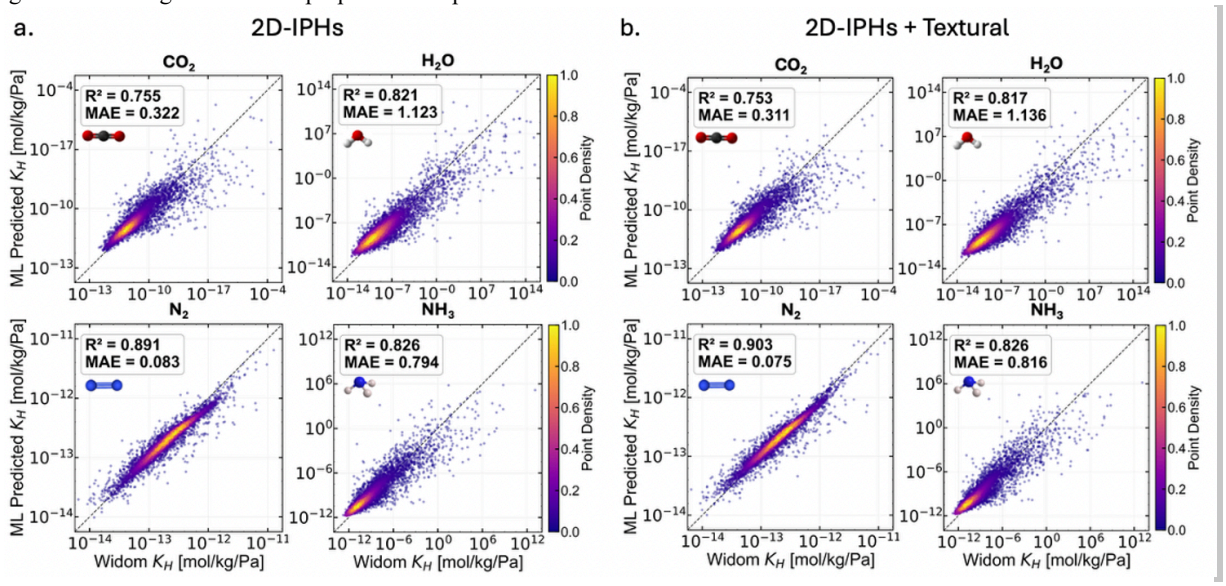

Fig. S11. a) ML predictions of K_H using only 2D-IPHs (all 4,440 features) as input. b) ML predictions of K_H using 2D-IPHs along with textural properties as input.

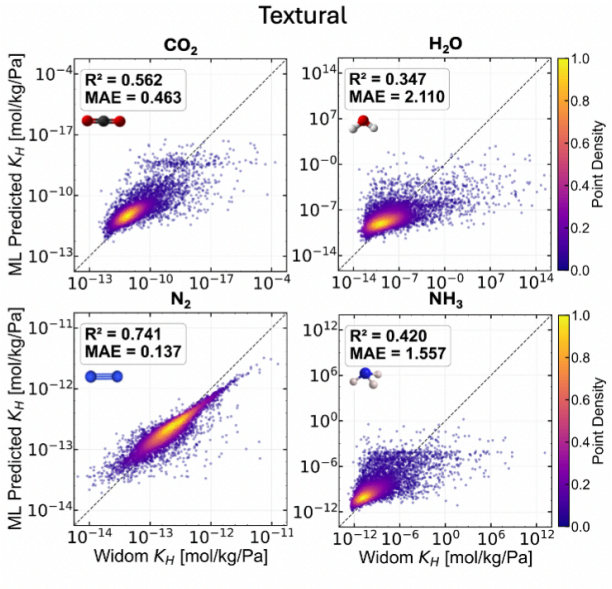

Fig. S12. ML predictions of K_H using only textural properties as input.

Section S6.2. Grid Resolution

To assess sensitivity to grid and bin resolution, we computed 2D-IPHs for ten randomly selected MOFs across all combinations of grid resolutions (0.5, 1.0, and 2.0 Å) and bin counts (20 and 40), using our default configuration (1.0 Å, 20 bins) as reference. Rather than retraining ML models for each configuration, we use pairwise feature similarity with respect to the reference as a direct proxy for expected model behavior. Representations deviating substantially from the reference will yield degraded predictive performance, as the encoded structural information diverges from that on which the models were optimized.

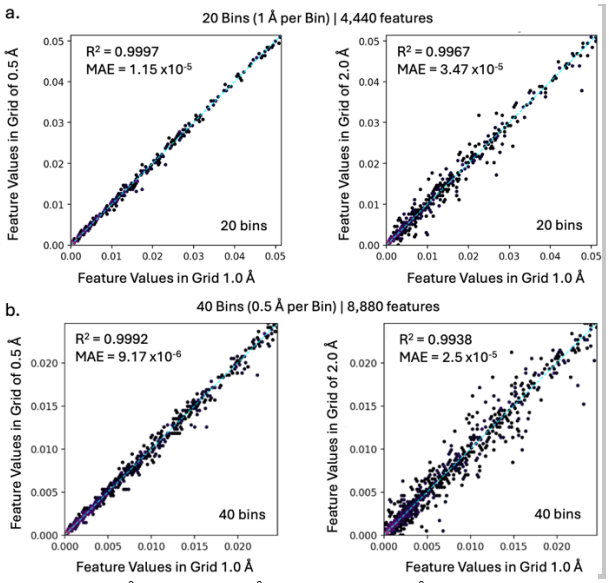

Fig. S13. Similarity plot between grid resolution of 1.0 Å versus 0.5 Å (left), and 2.0 Å (right) for a) 20 bins and b) 40 bins resolution used to obtain the 2D-IPHs.

As shown in **Fig. S13**, the 0.5 Å / 20-bin variant most closely reproduces the reference features, suggesting comparable ML performance at moderately increased computational cost. The 2.0 Å grid produces the largest deviations, likely due to increase proximity of coarse grid points to less chemically relevant atoms (due to the current implementation of the 2D-IPHs code) within local adsorption sites. Doubling the bin count to 40 expands the feature vector to 8,880 dimensions, introducing substantial memory overhead and computational cost making it an unfavorable trade-off. Overall, these results confirm that the default 1.0 Å / 20-bin configuration strikes the best balance between representational quality and computational efficiency.

References

1. Anderson, R., Biong, A. & Gómez-Gualdrón, D. A. Adsorption Isotherm Predictions for Multiple Molecules in MOFs Using the Same Deep Learning Model. *J. Chem. Theory Comput.* **16**, 1271–1283 (2020).