Data Efficient Molecular Image Representation Learning using Foundation Models – Supplementary Information

Yonatan Harnik[1], Hadas Shalit Peleg[1], Amit H. Bermano[2], Anat Milo[1]

[1]Department of chemistry, Ben-Gurion University of the Negev, Beer Sheva

[2]School of Computer Science, Tel Aviv University, Tel Aviv

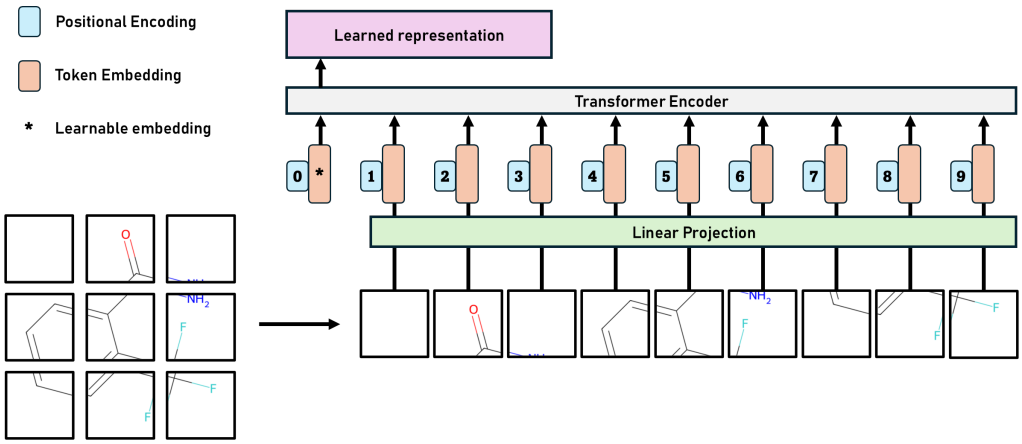
Table of Contents

1. Visual transformers

We introduce MoleCLIP, an image-based molecular representation learning framework designed to produce robust semantic embeddings for molecular inputs. The primary aim of MoleCLIP is to create a powerful molecular image encoder that can be finetuned for various molecular property prediction tasks, enhancing the accuracy and efficiency of these predictions. MoleCLIP relies on OpenAI's CLIP,[1] a foundation model that offers a solid starting point for MoleCLIP's continued pretraining phase on molecular images.

At the core of MoleCLIP's architecture is a vision transformer (ViT). Transformers were originally developed for text processing, and ViT adapts the transformer attention mechanism to images.[2] The ViT workflow involves breaking down an image into a sequence of patches that serve as "tokens", similar to a word in natural language processing. This allows the model to learn relationships between different parts of the image and extract meaningful features from the images. Figure S1 presents the conceptual architecture of ViT. For MoleCLIP, we employ the B/16 variant of ViT, which processes images with dimensions of 224x224 pixels, utilizing a patch size of 16x16 pixels, and includes 12 transformer encoder layers.

Figure S1 – ViT architecture overview, following Dosovitskiy el al.[2]



2. Pretraining

The pretraining of MoleCLIP was performed on molecular image inputs generated by RDKit.[3] ChEMBL-25, comprised of 1,870,421 bioactive drug-like molecules, was used as an unlabeled dataset for pretraining (see (2) in data availability).[4] The molecules were converted from SMILES[5] to images before training using RDKit. To enhance the model's ability to embed molecules effectively, two distinct pretraining tasks were combined during the pretraining phase: supervised structural classification and self-supervised contrastive learning.

a. Structural classification task and loss

Following ImageMol,[6] we incorporated a structural classification pretraining task to enable the encoder to differentiate between various structural groups within the embedding space. This was achieved by performing structural clustering (see below), assigning pseudo-labels to each molecule, and training the model to predict these labels from the embeddings.

i. K-means structural clustering

We employed the K-means algorithm for the clustering task using 166-bit MACCS (Molecular ACCess System) fingerprints[7] extracted using RDKit.[3] Fingerprints are a method for structural molecular encoding by bit representation, specifying the presence or absence of functional groups within a molecule. The bit vector format provides a compact binary representation, making fingerprints useful for cheminformatics tasks such as clustering and molecular similarity evaluation.

Clustering was performed based on the 166-bit representation of the molecules. We used Scikit-learn[8] mini-batch K-means algorithm with a batch size of 20,000. To determine the optimal number of clusters (K), we tested K values ranging from 3 to 3000. The optimal K was identified on the inertia vs. K curve using the Kneedle package for knee-point detection.[9] The knee-point curve for the ChEMBL-25 dataset, shown in Figure S2, indicated

an optimal K value of 300. Consequently, we set clustering labels for our primary model at K = 300 and K = 3000.

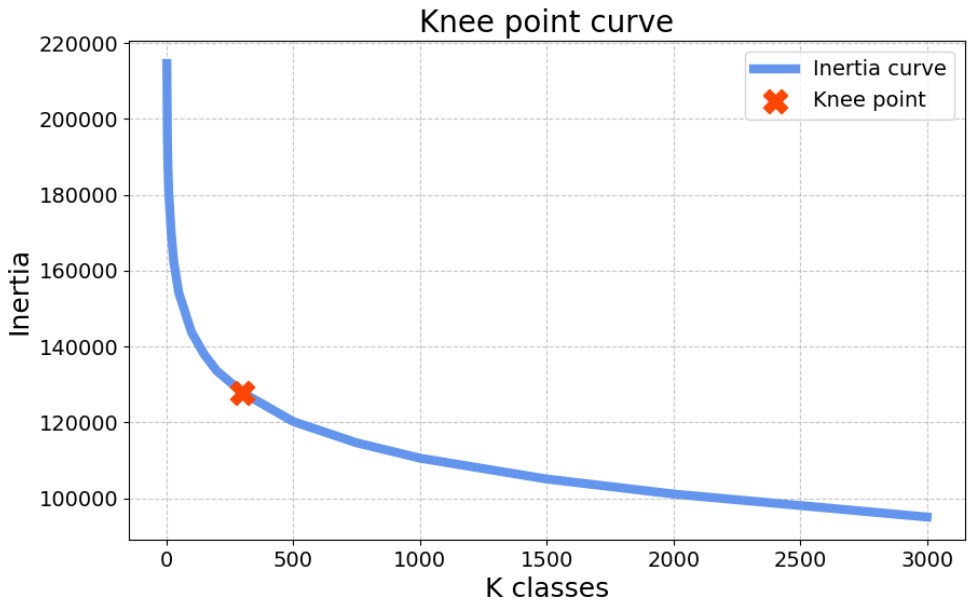


Figure S2 – k-means knee plot for ChEMBL-25

ii. Structural classification training

For the supervised classification task, pseudo-labels were assigned based on the clusters. As depicted in Fig. 1b of the main article, a linear head was added atop the embedding layer to predict the class. The training involved calculating cross-entropy loss between the model's predictions and the assigned pseudo-labels. The structural classification loss for a batch of samples is given by:

$$\mathcal{L}_{SC} = \sum_{n=1}^{N}\sum_{c=1}^{C} y_{n,c}\, log \frac{\exp\,(x_{n.c})}{\sum_{i=1}^{C}\exp\,(x_{n,i})}$$

Where N is the batch size, C is the number of classes, x represents the predicted values, and y is the pseudo label.

b. Contrastive task and loss

In addition to structural classification, a contrastive task was incorporated, inspired by the SimCLR - a simple framework for contrastive learning of visual representations.[10] In this task, unlabeled images were augmented to create pairs of original and augmented images. The model then encoded both sets, training to minimize the embedding distance between similar images while maximizing the distance between different images. We applied this method to molecular images, leveraging CLIP's encoder and workflow.[1] The addition of contrastive learning aims to enhance the model's ability to interpret molecular images and distinguish between molecules based on subtle visual differences.

i. Image augmentations through generation

Contrastive learning was performed between batches of default images and their corresponding augmented versions. Since all images were generated before pretraining, each image was paired with only one augmented version. This differs from the augmentation applied during training (see Section S-c-ii), where random augmentations were introduced in each training iteration. The augmentation process involved randomizing the following RDKit molecular image generation parameters:

- Font type (38 different types)
- Font size (14-24)
- Bond length (10-50 pixels)
- Bond line width (1-5 pixels)
- Multiple bond offset (0.1-0.4)
- Random rotation

We note that the atom labels were kept vertical during the rotation augmentation. This differs from the rotation during training (described in Table S1), where the entire image is rotated as a unit, including the font labels. Figure S3 presents qualitative examples of the various generation-level augmentations.

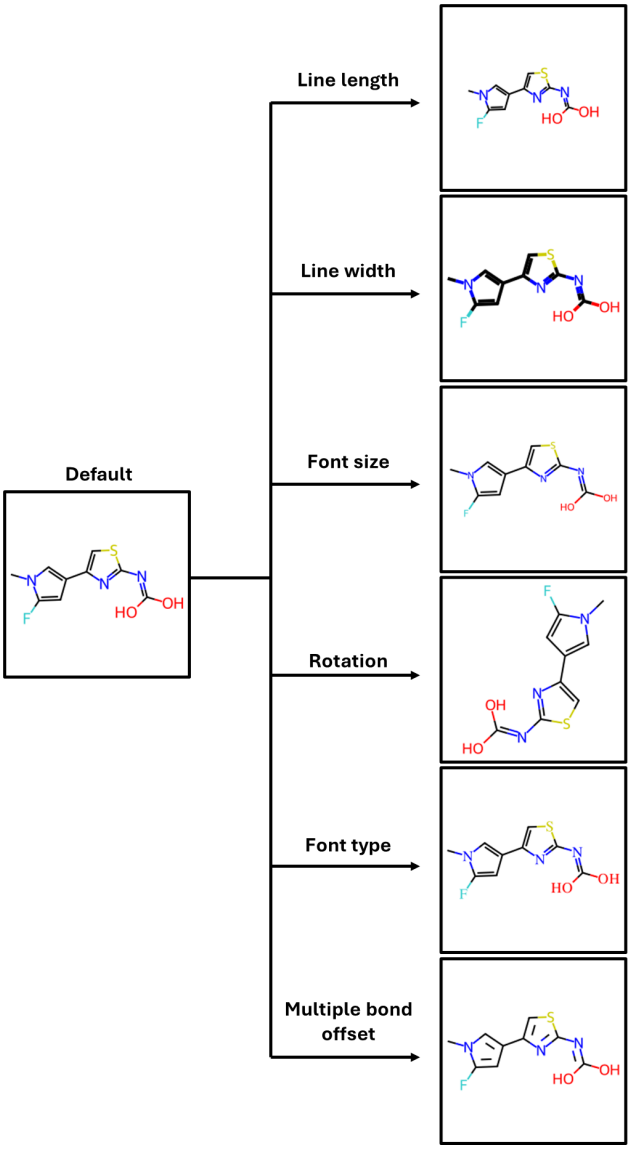


Figure S3 – Generation-level augmentations

ii. Contrastive training

During training, the batches were designed to include molecules from different and similar structural classes. This approach helped the model distinguish between varied and alike structures. Batches of 32 samples from the same class were initially created and then randomly combined to form final training batches of 256 samples.

The contrastive loss was computed similarly to the method used in CLIP. However, instead of calculating the similarity between image and text inputs, we focused solely on the similarity between original and

augmented image pairs. The contrastive loss function for a batch of samples (adapted from Zhai et al.) is defined as:[11]

$$\mathcal{L}_{CL} = -\frac{1}{2N}\sum_{n=1}^{N}\left(log\frac{\exp\left(\frac{1}{\tau}\mathrm{I}_{\mathrm{n}}^{\mathrm{o}}\cdot\mathrm{I}_{n}^{\mathrm{a}}\right)}{\sum_{i=1}^{N}\exp\left(\frac{1}{\tau}\mathrm{I}_{\mathrm{n}}^{\mathrm{o}}\cdot\mathrm{I}_{i}^{\mathrm{a}}\right)} + log\frac{\exp\left(\frac{1}{\tau}\mathrm{I}_{\mathrm{n}}^{\mathrm{o}}\cdot\mathrm{I}_{n}^{\mathrm{a}}\right)}{\sum_{i=1}^{N}\exp\left(\frac{1}{\tau}\mathrm{I}_{\mathrm{i}}^{\mathrm{o}}\cdot\mathrm{I}_{n}^{\mathrm{a}}\right)}\right)$$

Where N is the batch size, I^{o} is the normalized embedding of an original image, I^{a} is the normalized embedding of an augmented image, and $\frac{1}{\tau}$ is the temperature scaling factor.

Figure S4 presents a NumPy pseudo code adapted from CLIP for the contrastive loss calculation.[1] In MoleCLIP we set $\frac{1}{\tau}$ to a fixed value of $\frac{1}{\tau} = \frac{1}{0.07} \approx 15$, following CLIP's initialization value.

```
# image_encoder - Vision Transformer
# I_o[n, h, w, c] - minibatch of original images
# I_a[n, h, w, c] - minibatch of augmented images
# W [d_i, d_e] - learned projection of image to embed
# t - fixed temperature parameter

# extract feature representations of both image batches
I_o_f = image_encoder(I_o)
I_a_f = image_encoder(I_a)

# joint embedding [n, d_e]
I_o_e = l2_normalize(np.dot(I_o_f, W), axis=1)
I_a_e = l2_normalize(np.dot(I_a_f, W), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_o_e, I_a_e) * np.exp(1/t) # symmetric loss function
labels = np.arange(n)

loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss = (loss_i + loss_t)/2
```

Figure S4 – NumPy pseudo code for MoleCLIP contrastive loss calculation (taken and adapted from CLIP publication)[1]

c. Preprocessing and training

i. Image preprocessing

All molecular images were generated in 224x224 dimensions before training using the default RDKit image generation settings. Generation-level augmented images were generated as detailed in S1-c-i. During image loading, normalization was performed using OpenAI-CLIP coefficients (**mean** = [0.48145466, 0.4578275, 0.40821073], **std** = [0.26862954, 0.26130258, 0.27577711]).

ii. Image augmentations during training

During the training, random augmentations were added to all images, including those already augmented during the generation. Table S1 indicates all the added augmentations, relevant parameters, and probability of occurrence. Figure S5 shows qualitative examples of the applied augmentations. During pretraining, the default set of augmentations was applied.

Table S1 – MoleCLIP augmentations using Albumentation package functions

Augmentation	Default	Intensive
Downscale	scale_min=0.7, scale_max=0.85, p=0.75, interpolation: downscale - cv2.INTER_AREA, upscale - cv2.INTER_NEAREST	-
LongestMaxSize (following by PadIfNeeded)	max_size = random.randint(135, 224) p=0.75, padding border_mode=1	max_size = random.randint(112, 224) p=0.75, padding border_mode=1
PixelDropout (Salt&Pepper)	dropout_prob=0.01, p=0.3 (performed two times – for salt and for pepper)	dropout_prob=0.05, p=0.5 (performed two times – for salt and for pepper)
Blur	blur_limit = 3, p=0.3	blur_limit=3, p=0.5
GaussNoise	p = 0.3	p = 0.5
ToGray	p = 0.25	p = 0. 5
random rotation	-	limit=360, border_mode=1

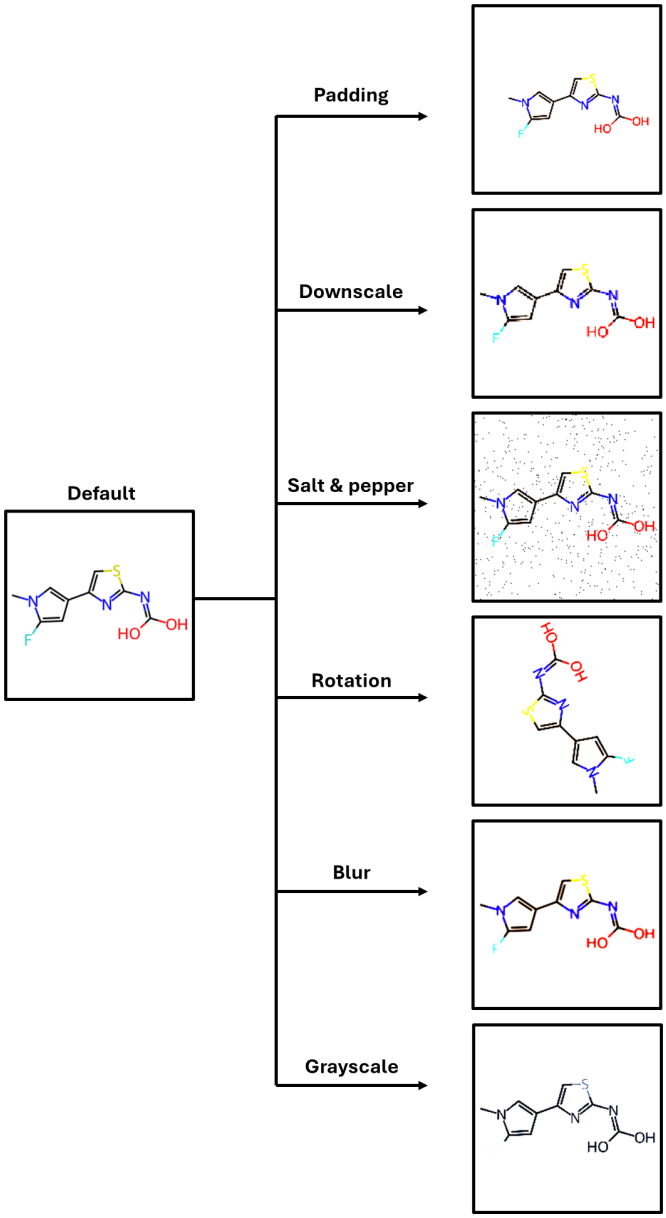


Figure S5 – Augmentations during training

iii. Training parameters

The encoder was pretrained using four Nvidia-T4 GPUs (64 GB RAM) over four epochs on the ChEMBL-25 dataset. The training was performed using the Adam optimizer, with a learning rate $5*10^{-6}$ for the encoder (100 times lower than the CLIP learning rate to prevent catastrophic forgetting)[12] and of 0.01 for the structural classification linear head. We applied a weight decay of 0.1 and a batch size of 256. See (3) in data availability for the pretrained weights of the trained model.

3. Finetuning

a. Datasets

We performed finetuning on four MoleculeNet benchmarks as well as on for catalysis datasets. We selected three molecular classes commonly used in catalysis to curate the catalysis datasets. The primary criterion for dataset selection was that the molecule must be the sole input variable, thus excluding experimental datasets where multiple conditions or substrates change simultaneously. Details on all finetuning datasets, including their size, metrics, and predicted properties, are provided in Table S2.

Table S2 – Finetuning datasets properties

	Dataset	Number of samples	Targets per sample	Regression/ Classification (metric)	Predicted properties
MoleculeNet	BACE	1513	1	Classification (ROC-AUC)	Binding to β-secretase 1
	BBBP	2039	1	Classification (ROC-AUC)	Blood-brain barrier penetration
	FreeSolv	642	1	Regression (RMSE/MAE)	Hydration free energy
	Esol	1128	1	Regression (RMSE/MAE)	Water solubility
Catalysis	DHBDs	6994	1	Regression (MAE)	DFT calculated HOMO/LUMO gap
	NHCs	95	4	Regression (MAE)	DFT calculated NBO charges
	Phosphines – yield	90	5	Regression (MAE)	Suzuki reaction yields
	Phosphines – selectivity	37	1	Regression (MAE)	Suzuki reaction selectivities

i. MoleculeNet benchmarks

MoleCLIP was evaluated on four MoleculeNet benchmarks, including biophysical and physical chemistry tasks.[13] The biophysical benchmarks included BACE and BBBP, both binary classification tasks. BACE includes

molecular inhibitors labeled according to their binding ability to human β-secretase 1, and BBBP consists of molecules labeled for their ability to penetrate the blood-brain barrier (BBB). The physical chemistry tasks included ESOL and FreeSolv, both regression tasks. ESOL is a compilation of solubility in water for common organic small molecules, whereas FreeSolv is a collection of free hydration energies for small molecules in water. These four datasets were selected for their simplicity, as they are relatively small and contain only one label per molecule.

ii. DHBDs

The evaluation of MoleCLIP on DHBDs was performed based on the OSCAR dataset of 6994 molecules generated through combinatorial enumeration.[14] The target value for prediction was the HOMO/LUMO gap (in eV), derived by subtracting the HOMO energy from the LUMO energy, both computed using density functional theory (DFT). The dataset is available on GitHub (see (4) in data availability).

iii. NHCs

For the NHCs, a dataset of 95 NHC pre-catalysts was collected from the literature. The dataset was curated systematically to include known NHC motifs according to Flanigan et al.,[15] and therefore contains eight structurally distinguished subsets. Figure S6 illustrates these NHC motifs and details the number of samples representing each subset. Despite our efforts to create a balanced dataset, the oxazolidine-based subset contains the fewest molecules due to its paucity in the literature. In contrast, the pyrrolidine-based motif is the most prevalent in organocatalytic studies, resulting in the largest subset in our dataset.

For this dataset, we predicted the DFT-computed natural population analysis (NPA) charges for four atoms: two corresponding to the C-H bonds of the NHC pre-catalysts and two corresponding to the C-C bond of the Breslow intermediate with benzaldehyde (as shown in Figure S7). Geometry optimizations of the pre-catalysts and their respective Breslow

reactive intermediates were performed using Gaussian 16 software.[16] The functional used for DFT calculation is M06-2X, previously benchmarked for thermodynamic and kinetic accuracy of main group elements and non-covalent interactions.[17,18] A triple zeta potential basis-set (def2-TZVP) was chosen based on Zhao and Truhlar's evaluation of the M06-2X functional for organic molecules, indicating that a triple zeta quality is generally more quantitative.[19,20] Charges were calculated using the NBO 3.1 extension.[21] The dataset is available on GitHub (see (5) in data availability).

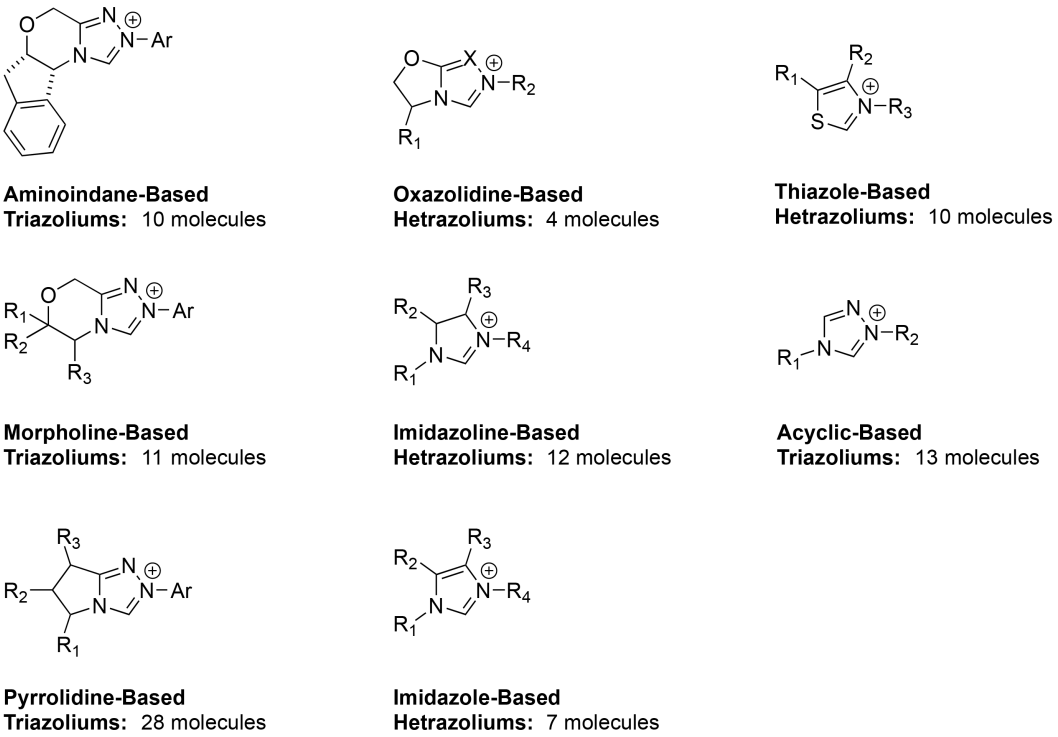


Figure S6 – NHC motifs and their representation in the dataset

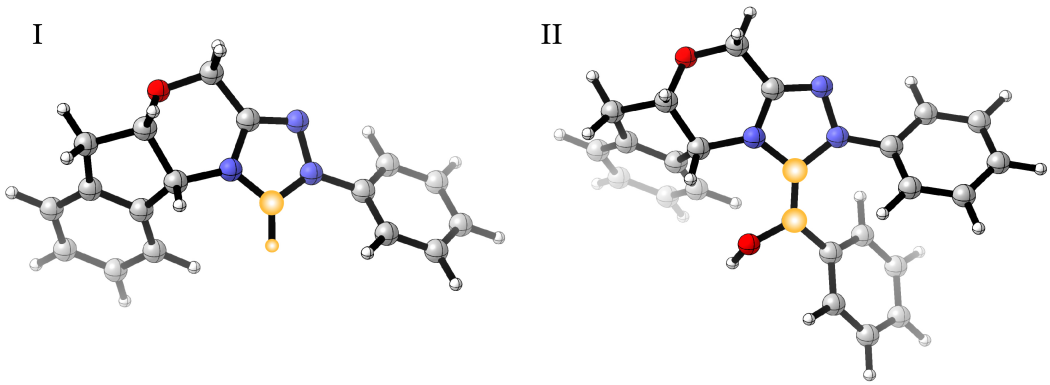


Figure S7 – Examples of the precatalyst (I) and Breslow intermediate with benzaldehyde (II). The four atoms for which DFT-computed NPA charges were predicted are highlighted in yellow. [22] The figure was produced using CYLview20.[22]

iv. Phosphines – yield

The dataset from Newman-Stonebraker et al.,[23] includes 90 phosphines for which high-throughput experimentation was performed. The dataset includes reaction yields from Ni-catalyzed Suzuki coupling reactions using five substrate combinations. We predicted the yields of each combination vs. the 90 phosphines varied across each set. The example reaction is presented in Figure S8-A. The dataset is available on GitHub (see (6) in data availability).

v. Phosphines – selectivity

The data of Niemeyer et al.,[24] selected for the selectivity dataset, includes 37 phosphines. The predicted value in this dataset is ΔΔG‡, which represents the energy difference in kcal/mol between competing pathways to provide each of the two possible enantiomeric products. The ΔΔG‡ values are mathematically derived from the experimentally measured selectivity. The values were collected from a single reaction, illustrated in Figure S8-B. The dataset is available on GitHub (see (7) in data availability).

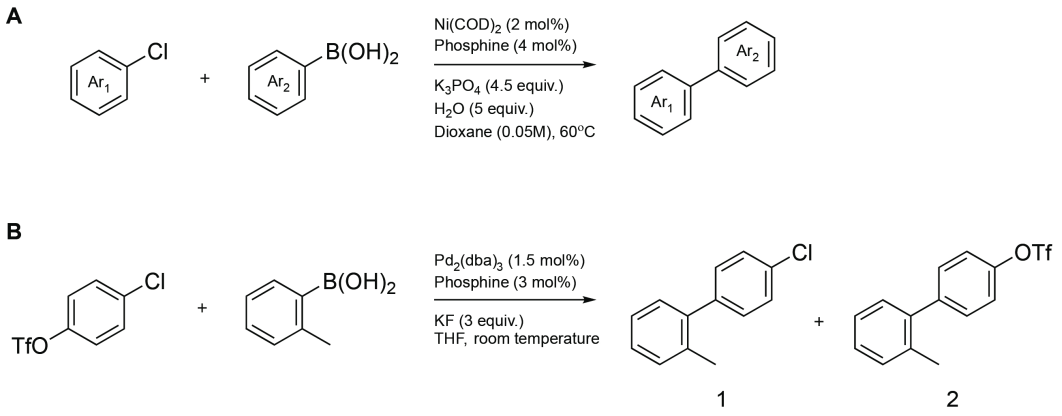


Figure S8 – **A.** Suzuki reaction conducted by Newman-Stonebraker et al.[23] The reaction was performed using 5 combinations of aryls, varying in their functional groups. **B.** Suzuki reaction conducted by Niemeyer et al.[23]

b. Training process

i. Hyperparameter optimization

We performed hyperparameter optimization for MoleCLIP, GEM,[25] and ImageMol.[6] During this process, the best model selection was determined based on the test set results, where the validation loss was minimized. For the finetuning of MoleCLIP, a 3-layer, 512-dimensional multilayer perceptron (MLP) was added on top of the encoder. The encoder and the MLP head were trained concurrently, though with different learning rates. The specific parameters used for hyperparameter scanning of MoleCLIP are detailed in Table S3. Throughout all training sessions of MoleCLIP, the Adam optimizer was employed with the following settings: β_1=0.9, β_2=0.98, ε=1*10^{-6}, weight decay=1*10^{-5}.

Table S3 – Hyperparameters of MoleCLIP Finetuning

Dataset	MoleculeNet full benchmarks	DHBDs	NHCs, Phosphines
Batch size	64	64	4
Encoder learning rate	5*10^{-6}	5*10^{-6}	5*10^{-6}
MLP head learning rates	1*10^{-3}, 4*10^{-4}, 1*10^{-4}, 4*10^{-5}, 1*10^{-5}	4*10^{-4}, 1*10^{-4}, 4*10^{-5}	1*10^{-3}, 4*10^{-4}, 1*10^{-4}, 4*10^{-5}, 1*10^{-5}
Image augmentation level	None, Default, Intensive	Intensive	None, Default , Intensive
Epochs during hyperparameter optimization	60	60	100
Epochs during Finetuning	60	180	300

For the finetuning of ImageMol, we employed the authors' encoder architecture and initial weights. We maintained batch sizes and number of epochs consistent with those used for MoleCLIP. Following ImageMol methodology, a simple linear head was appended to the encoder and

trained concurrently. The stochastic gradient descent (SGD) optimizer was employed with a momentum of 0.9 and a weight decay of $1*10^{-5}$. Hyperparameter tuning focused only on the learning rate, testing values of 0.0005, 0.005, 0.05, and 0.5 (following the ImageMol author's recommendations of values between 0.0005 and 0.5).

For GEM finetuning, we also used the pretrained weights provided by the original authors, maintaining the exact batch sizes and the number of epochs as for MoleCLIP. A 128-dimensional, 3-layer MLP was added atop the encoder in line with the GEM methodology. Optimization was carried out using the Adam optimizer. We performed a grid search on dropout rates of 0, 0.2, and 0.5, along with encoder-head learning rate pairs of (0.001, 0.001), (0.004, 0.004), and (0.0001, 0.001), following the recommendations of the GEM authors.

We also ran reference evaluations with ECFP representations extracted using RDKit (radius = 2). To model these fixed representations, we employed the same MLP head used for MoleCLIP (3-layer, 512-dimensional). Learning rate tuning was conducted as outlined in Table S3.

ii. Image preprocessing and augmentation

Image preprocessing was consistently applied to the molecular image inputs, following the procedure used during pretraining (refer to S1-d-i). As part of hyperparameter optimization, we evaluated different augmentation strategies for each model. These included training with no augmentation, the default augmentations used for pretraining, and intensive augmentation. The intensive procedure included the same augmentations as the default but with increased probability and adjusted parameters to enhance their impact. Additionally, random rotation augmentation was added to the intensive augmentations. Table S1 details the various augmentations and their corresponding parameters.

For ImageMol, we applied the augmentations recommended by the authors: horizontal flip, grayscale conversion, and random rotation, each

with a 0.5 probability. Notably, for the random rotation in ImageMol, we adhered to the authors' approach of using black fill around the rotated image, whereas for MoleCLIP, we used a white fill.

iii. Evaluation

The finetuning datasets were split to train, validation, and test sets using varied splitting methods and ratios. Evaluations are reported only for the optimal hyperparameters based on the test results at the point where the validation loss was minimal. We used three times more epochs for the final run to ensure the model had reached a stable state. However, for MoleculeNet benchmarks, we maintained a consistent 60 epochs for both hyperparameter optimization and finetuning to enable comparison to the state-of-the-art (SOTA) results in the literature. We used the receiver operating characteristic area under the curve (ROC-AUC) as the metric for classification tasks, whereas the root means squared error (RMSE) and mean absolute error (MAE) were employed for regression tasks.

c. Results

For a summary of all finetuning results, see (8) in data availability

i. MoleculeNet benchmarks

We employed the standard splitting procedures and metrics to benchmark MoleCLIP against several SOTA models on the MoleculeNet datasets. Following a scaffold splitting approach, each dataset was split using an 8:1:1 ratio for training, validation, and testing. This method groups molecules by their Murcko scaffolds, assigning the most frequent scaffolds to the training set and the least frequent to the test set. Scaffold splitting is a more rigorous evaluation method than random splitting, as it challenges the model to generalize from common molecular structures to rare ones.

Table S4 presents MoleCLIP's performance on the four MoleculeNet benchmarks alongside the results of several SOTA models as reported in their respective publications. The optimal results for MoleCLIP, achieved

through hyperparameter tuning, are listed with the corresponding parameters. The results show that MoleCLIP performs comparably to the SOTA models despite being pretrained on significantly fewer data.

Table S4 – This table provides the numerical values illustrated in Fig. 2 of the main manuscript. Performance of MoleCLIP on MoleculeNet benchmarks compared to SOTA models. The table shows performances and optimal hyperparameters for MoleCLIP, with standard deviation errors from three independent runs. SOTA results are from original publications. Pretraining data volume is given for all models as the product of epochs and dataset size. Despite lower pretraining data volume, MoleCLIP achieves comparable performance to SOTA models.

			Classification (ROC-AUC)		**Regression (RMSE)**	
Model	**Model Input**	**Pretraining Data Volume**	BACE (1513 samples)	BBBP (2039 samples)	FreeSolv (642 samples)	Esol (1218 samples)
Baseline	ECFP (radius = 2)	-	0.827±0.006	0.60±0.02	4.12±0.19	1.58±0.03
ChemBERTa-2[26]	String	77M	0.799	0.742	-	0.86
MolCLR[27]	Graph	500M	0.890±0.003	0.736±0.005	2.20±0.20	1.11±0.01
Uni-Mol[28]	Graph	209M	0.857±0.002	0.729±0.006	1.48±0.05	0.79±0.03
GEM[29]	Graph	400M	0.856±0.003	0.724±0.003	1.86±0.09	0.83±0.03
ImageMol[6]	Image	120M	0.839±0.003	0.739±0.003	2.02±0.07	0.97±0.07
MoleCLIP	Image	7.6M	0.829±0.005	0.747±0.009	2.00±0.14	0.97±0.01
MoleCLIP hyperparameters (augmentation level, head learning rate)			Intensive, $4*10^{-4}$	Default, $1*10^{-4}$	Default, $4*10^{-4}$	Intensive, $1*10^{-5}$

ii. Catalysis datasets

We finetuned MoleCLIP, GEM, and ImageMol on four catalysis datasets. The DHBDs, NHCs, and Phosphines-yield datasets were randomly split into training, validation, and test sets using an 8:1:1 ratio. A 6:2:2 split ratio was applied for the Phosphines-selectivity dataset to ensure sufficiently large validation and test sets (seven examples in each). Scaffold splitting, commonly used in the MoleculeNet datasets, was not applied here since it is less meaningful in datasets that contain minimal scaffold variability.

For the NHCs and phosphines datasets, we performed splits using five random seeds (1, 2, 3, 4, 5) and averaged the results across all repetitions. During hyperparameter optimization, each seed was run three times (15 in total hyperparameters set), and models were trained for 100 epochs. We

reran the model with optimal hyperparameters with ten repetitions per seed (50 in total) for 300 epochs. For the DHBDs dataset, given its larger size, we used three random seeds (1, 2, 3). In this case, hyperparameter optimization involved three repetitions per run (nine in total), with models trained for 60 epochs. The final model was rerun with three repetitions per seed for 180 epochs. This procedure was consistently applied to MoleCLIP, ImageMol, and GEM. For error calculations, variance was computed separately for each seed. Pooled variance values and standard deviations were then calculated. Error intervals were derived using standard errors with 50 repeats (nine for the DHBDs) and 95% confidence.

Table S5 – Performance of MoleCLIP on catalysis benchmarks compared to ImageMol and GEM. Optimal hyperparameters are provided for all models. MAE was used as the evaluation metric across all datasets, with error intervals calculated at 95% confidence. Overall, MoleCLIP outperforms ImageMol on all datasets and surpasses GEM on three out of four datasets.

Dataset		**DHBDs (eV)**	**NHCs (Charge)**	**Phosphines - yield (%)**	**Phosphines - selectivity (kcal/mol)**
Number of samples		6994	95	90	37
Baseline (ECFP)	Learning rate	0.00004	0.001	0.0004	0.0004
	Performance (MAE)	0.275±0.003	0.0158±0.0005	12.2±0.2	1.20±0.02
MoleCLIP	Head learning rate	$4*10^{-4}$	$1*10^{-3}$	$1*10^{-4}$	$4*10^{-4}$
	Augmentations level	Intensive	None	None	Intensive
	Performance (MAE)	0.197±0.002	0.0138±0.001	9.5±0.3	0.95±0.04
ImageMol	Learning rate	0.005	0.005	0.005	0.0005
	Performance (MAE)	0.199±0.001	0.0218±0.003	11.3±0.5	1.23±0.06
GEM	Learning rates (Encoder, head)	$4*10^{-3}$, $4*10^{-3}$	$4*10^{-3}$, $4*10^{-3}$	$1*10^{-3}$, $1*10^{-3}$	$1*10^{-4}$, $1*10^{-3}$
	Dropout	0.1	0.1	0.1	0.2
	Performance (MAE)	0.181±0.003	0.0145±0.0007	9.6±0.4	1.03±0.04

Table S5 shows MoleCLIP's performance on the catalysis benchmarks compared to GEM and ImageMol. The reported optimal results include the respective hyperparameters obtained through optimization. The results indicate that MoleCLIP outperforms ImageMol across all datasets and surpasses GEM on three out of four datasets. We note that MoleCLIP's advantage over GEM on the phosphine-yield dataset is not statistically significant.

d. Drug-target affinity

Drug-target affinity (DTA) prediction is a fundamental task in computational drug discovery, aiming to estimate the binding strength between a drug molecule and its biological target.[30] Representation learning can potentially enhance DTA prediction by providing more informative representations of both drug molecules and target proteins. This is usually done by concatenating the learned representation of the drug and the protein sequence, resulting in a fused representation which is used for affinity prediction.

To provide a preliminary evaluation of MoleCLIP in the context of DTA prediction, we used two common benchmark datasets: KIBA and Davis. Since we did not pretrain a dedicated protein encoder, we employed a 1D convolutional neural network (1D-CNN) to encode protein sequences, training it from scratch. This encoder transforms string representations of proteins into 512-dimensional embeddings, using 32 convolutional filters and an embedding dimension of 128. The resulting protein embeddings were concatenated with the 512-dimensional outputs from the pretrained MoleCLIP molecular encoder, forming a combined 1024-dimensional representation. This vector was then passed through a three-layer MLP with 512 hidden units to produce the final prediction.

We initialized the molecular encoder with pretrained MoleCLIP weights and trained the model on each dataset for 200 epochs using a learning rate

of 0.0005, following the GraphDTA training protocol.[31] The datasets were split using the same 80/20 train-test split. Mean squared error (MSE) results are reported in Table S6. Whereas our model does not outperform current state-of-the-art methods, it achieves better results compared to WideDTA, which demonstrates the versatility of MoleCLIP towards various tasks.

It is important to note that the model architecture is imbalanced, as it leverages a high capacity pretrained encoder for molecular inputs but a lightweight, randomly initialized protein encoder. This exercise was merely aimed at piquing curiosity toward the use of MoleCLIP in drug affinity tasks. We anticipate that incorporating a more sophisticated, pretrained protein encoder, together with a careful hyperparameter optimization to balance the two modalities, could substantially enhance the overall performance of the model on the DTA task.

Table S6 – Performance of MoleCLIP on DTA prediction task compared to other existing models. MSE was used as the evaluation metric across all datasets.

Model	Davis (30,056 samples)	KIBA (118,254 samples)
WideDTA[32]	0.262	0.179
GraphDTA[31]	0.229	0.139
SAG-DTA[33]	0.209	0.130
HiSIF-DTA[34]	0.191	0.120
MoleCLIP	0.246	0.170

4. Analysis

a. R-group replacement

We curated additional ESOL and FreeSolv datasets versions by modifying the molecular images, specifically replacing certain functional groups with numbered R-groups, as indicated in Table S7. This replacement was executed using a dedicated code substituting specific functional groups with corresponding R-groups. All R-groups were added in black, ignoring the specific atom colors usually assigned by RDKit for each functional group. The code for generation of R-replaced molecular images is available in our GitHub repository (see (1) in data availability).

Table S7 – R-group numbering for functional groups

R-number	Functional group
–R1	$–CH_3$
–R2	–OH
–R3	$–NH_2$
–R4	–SH
–R5	–F
–R6	–Cl
–R7	–Br
–R8	–I

The original and modified datasets were evaluated using MoleCLIP and ImageMol, following similar procedures as outlined in S2-b, including hyperparameter optimization. The evaluation process for all experiments involved training for 60 epochs with three random seeds and three repetitions per seed for hyperparameter optimization, followed by 180 epochs and five repetitions per seed for the final evaluation. MAE was used as the evaluation metric across all tests. Error intervals for the performance differences were calculated at 95% confidence based on the standard errors of the two values. Detailed results in Table S8 indicate that MoleCLIP's relative performance compared to ImageMol significantly

improves in the R-replaced datasets versus the original datasets, demonstrating MoleCLIP's superior robustness to distribution shifts. The results are detailed in (7) in data availability.

Table S8 – Evaluation of MoleCLIP and ImageMol on the original and R-replaced datasets for Esol and FreeSolv. MAE was used as the evaluation metric across all datasets, with error bars representing 95% confidence intervals.

Dataset		Esol Original	Esol R replaced	FreeSolv Original	FreeSolv R replaced
MoleCLIP	Head learning rate	0.001	0.00001	0.0001	0.001
	Augmentations level	Default	Default	Default	Default
	Performance (MAE)	0.484±0.007	0.514±0.010	0.77±0.05	1.00±0.06
ImageMol	Learning rate	0.005	0.005	0.0005	0.005
	Performance (MAE)	0.466±0.014	0.532±0.013	0.78±0.03	1.14±0.05
%Δ Difference **100 * (ImageMol - MoleCLIP) / MoleCLIP**		-3.7±2.8 %	3.5±2.9 %	1.3±6.9 %	14.0±6.7 %

To extend this analysis to catalysis-related datasets, we first selected our largest catalysis dataset, which is the DHBDs dataset (6,994 samples) involving HOMO/LUMO gap prediction. In this case, however, R-group replacement had a minimal impact on the performance of both MoleCLIP and ImageMol (see Table S9). These slight changes were too small for drawing meaningful insights, and suggest that, for this specific task, our R-group modification does not significantly affect prediction accuracy.

Thus, we performed the analysis also on the Phosphines-yield dataset, comprising of only 90 samples. Whereas both models showed reduced performance following the R-replacement, MoleCLIP exhibited a smaller performance drop compared to ImageMol, indicating greater robustness (see Table S9). Although both models exhibited reduced performance upon this replacement, MoleCLIP's performance was still better (11.5±0.3) than the baseline (12.2±0.2, see Table S5), whereas ImageMol's performance dropped significantly below it (14.4±0.7). We note that due to the reliance

on a fixed fingerprint representation, the baseline cannot undergo R-group replacement. Impressively, MoleCLIP's performance following R-group replacement was comparable to ImageMol's original performance on this dataset (11.3±0.5). Thus, despite the limited dataset size, MoleCLIP exhibited a much stronger performance in this data size regime, even with this distribution shift.

Table S9 – Evaluation of MoleCLIP and ImageMol on the original and R-replaced datasets for the DHBDs and Phosphines-yield datasets. MAE was used as the evaluation metric across all datasets, with error bars representing 95% confidence intervals.

Dataset		**DHBDs Original**	**DHBDs R replaced**	**Phosphines-yield Original**	**Phosphines-yield R replaced**
MoleCLIP	Head learning rate	0.00004	0.00004	0.0001	0.0001
	Augmentations level	Intensive	Intensive	None	None
	Performance (MAE)	0.197±0.002	0.199±0.02	9.5±0.3	11.5±0.3
ImageMol	Learning rate	0.005	0.005	0.005	0.005
	Performance (MAE)	0.199±0.001	0.202±0.002	11.3±0.5	14.4±0.7
%Δ Difference **100 * (ImageMol - MoleCLIP) / MoleCLIP**		1.0±1.1 %	1.5±1.4 %	18.9±6.1 %	25.2±6.6 %

b. Domain-focused pretraining

i. Pretraining on buried-volume prediction

Using the phosphines domain as a test case, we conducted a domain-specific pretraining session and evaluated its effectiveness using phosphine-related prediction sets. This selected pretraining task focused on predicting buried volume values, which are known to correlate with yield and selectivity activity cliffs within the phosphines domain.[23] For this purpose, we utilized the Kraken dataset, which includes 1,540 literature-sourced molecules and their corresponding DFT-calculated properties. The selected training targets were the minimum and Boltzmann buried volume values.

The pretraining phase was initialized from the ChEMBL-pretrained weights of the MoleCLIP primary model. This phase was executed similarly to a finetuning process, where the MoleCLIP encoder and an additional 3-layer MLP head (512-dimensional) were trained simultaneously. The focused pretraining lasted for 300 epochs with a constant learning rate of $5*10^{-6}$, weight decay of $1*10^{-5}$, and batch size of 64. We applied the default augmentation procedure (Table S1) from the pretraining of MoleCLIP's primary model.

The resulting model's performance, referred to as MoleCLIP_{bv}, was assessed on phosphines-yield and phosphines-selectivity datasets following the data-splitting procedure outlined in S2-c-ii. Hyperparameter optimization was performed in alignment with the primary MoleCLIP model. We compared the performance differences between MoleCLIP_{bv} and the primary MoleCLIP model. Detailed results and optimal hyperparameters for fine-tuning are presented in Table S10, demonstrating that MoleCLIP_{bv} outperforms the primary MoleCLIP model.

Table S10 – Results and optimal hyperparameters for the domain-focused pretrained model evaluated on the phosphines-yield and phosphines-selectivity datasets. The evaluation metric is MAE, with errors represented as 95% confidence intervals. Performance differences are compared to the primary MoleCLIP model results, as detailed in Table S5. The domain-focused pretrained model significantly improves performance over the primary MoleCLIP model, demonstrating the effectiveness of domain-specific pretraining in enhancing prediction accuracy for phosphine-related properties.

Dataset		**Phosphines - yield (%)**	**Phosphines - selectivity (kcal/mol)**
MoleCLIP$_{bv}$ ChEMBL – 4 epochs Kraken (buried volume) - 100 epochs	Head learning rate	0.00001	0.0004
	Augmentations level	Default	Intensive
	Performance (MAE)	9.1±0.3	0.82±0.03
%Δ Difference **100 * (MoleCLIP – MoleCLIP$_{bv}$) / MoleCLIP**		4.2±3.3 %	13.7±4.5 %

iii. Controls

To evaluate the significance of each component in the MoleCLIP$_{bv}$ pretraining workflow, we conducted a series of control pretraining experiments. These experiments were designed as follows:

- **Control I** – Pretraining on phosphine domain-specific molecules instead of ChEMBL.

  We only utilized MoleCLIP's primary tasks of contrastive learning and structural classification on phosphines. This control model was also initialized with CLIP weights. For this control, we curated a synthetic phosphines dataset with approximately 1.9 million samples through combinatorial enumeration, analogous to the approach used for the Kraken dataset.[35] This process involved 12,610 phosphanides (12,034 from PubChem[36] and 576 from Kraken) used as substituents for generating phosphine ligands. By iterating over all substituents, we generated all possible PA_3 ligands and PAB_2 ligands by randomly selecting 149 additional substituents in each iteration. This process resulted in 1,891,500 molecules (12,610 * 150). For a SMILES version of the dataset, see (3) in data availability. We trained two CLIP-initialized MoleCLIP encoders using the following two parameter sets for structural classification: $K_1 = 300$, $K_2 = 3000$ and $K_1 = 30$, $K_2 = 300$. All other pretraining parameters and procedures followed MoleCLIP's primary model pretraining protocol as specified in S1-e.
- **Control II –** Adding a phosphine-specific pretraining stage to our ChEMBL pretrained model without the buried volume task.

  We applied a sequential training strategy by further pretraining the ChEMBL-pretrained model using phosphine-specific data and applying MoleCLIP's primary tasks. Since the model faced only realistic molecules during the ChEMBL pretraining, we also wished to include realistic molecules during the continued pretraining phase. Hence, we filtered the synthetic phosphines dataset designed for **Control I** using the SCScore model that evaluates synthesizability (based on the

number of reaction steps required for synthesis).[37] We excluded molecules with SCScore values above 3.5, resulting in a refined dataset containing 151,497 molecules. For a SMILES version of the dataset, see (3) in data availability. Pretraining on this refined was conducted for ten epochs, equivalent to one epoch on approximately 1.5 million molecules. For this experiment, we used the structural classification parameters K_1 = 300 and K_2 = 3000, as used in MoleCLIP's primary model. All pretraining parameters and procedures followed MoleCLIP's primary model pretraining protocol as specified in S1-e.

- **Control III -** Pretraining only on the domain-specific buried volume prediction task.

  This control model was initialized with CLIP weights and was only pretrained on the buried volume task. The pretraining spanned 300 epochs and adhered to the same procedure used for primary domain-focused pretraining (as described in S3-c-i). For this

The performance of these control models on the phosphines-yield and selectivity datasets is summarized in Table S11. The results are detailed in (7) in data availability. The results demonstrate that these control models generally exhibit inferior finetuning performance compared to the MoleCLIP primary model. This highlights the effectiveness of the methodology used for training MoleCLIP$_{bv}$, in integrating domain-specific knowledge within the training and utilizing it for the phosphine-specific task.

Table S11 – Results and optimal hyperparameters of the controls based on the phosphines-yield and phosphines-selectivity datasets. The inferior performance of the control models compared to the MoleCLIP primary model highlights the necessity of all components of MoleCLIP$_{bv}$ training methodology (see Table S10) for effective domain-specific pretraining. Errors represented as 95% confidence intervals.

Dataset		Yield	Selectivity
Control - I Phosphines - 4 epochs K_1 = 30, K_2 = 300	Performance (MAE)	9.9±0.4	1.12±0.05
	Augmentations level	Intensive	Intensive
	Head learning rate	0.00001	0.0004
Control - I Phosphines - 4 epochs K_1 = 300, K_2 = 3000	Performance (MAE)	9.8±0.4	1.08±0.06
	Augmentations level	Intensive	Intensive
	Head learning rate	0.00001	0.001
Control - II ChEMBL – 4 epochs Phosphines filtered - 10 epochs K_1 = 300, K_2 = 3000	Performance (MAE)	9.6±0.3	1.14±0.06
	Augmentations level	Intensive	Intensive
	Head learning rate	0.00001	0.001
Control – III * CLIP initialized Kraken (buried volume) - 100 epochs	Performance (MAE)	13.8±0.4	1.70±0.01
	Augmentations level	None	Intensive
	Head learning rate	0.00001	0.001

* For control III, because of the poor results, we did not perform a complete evaluation based on 50 repeats, and the results are reported for 50 repeats (five seeds, three repeats per seed, 100 epochs)

c. Ablation study

Several ablated models were trained to evaluate the effects of different aspects of the model on finetuning performance. The trainings followed the same procedures as performed for the primary model, including all parameters of pretraining, finetuning, and hyperparameter optimization. The control models were evaluated on MoleculeNet benchmarks under 8:1:1 train/validation/test scaffold splitting. The detailed results of these controls are presented in Table S12.

i. Pretraining from scratch

To evaluate the impact of the CLIP foundation model on our results, we attempted to pretrain the ViT-B/16 encoder from scratch. However, the

model failed to converge despite experimenting with various learning rates and temperature factors. Consequently, we have not included any finetuning results from this control experiment.

ii. K values for Structural classification

As the optimal K value for ChEMBL-25 was 300, we set the clustering labels for our primary model to K_1 = 300 and K_2 = 3000. Thus, we experimented with K_1 = 30 and K_2 = 300 to assess the impact of higher K values on promoting finer structural distinctions within the latent space.

iii. Temperature factor

The temperature factor in contrastive learning is a hyperparameter applied to the loss function that influences the smoothness of the learned latent space, where a higher temperature results in a smoother latent space. In MoleCLIP, we fixed the temperature parameter at $\tau = 0.07$. To evaluate its effect on model performance, we also tested temperatures of $\tau = 0.05$ and $\tau = 0.1$.

iv. Pretraining period

As the number of epochs for the primary model was set to 4, we evaluated the model performance after a shorter training period of one epoch and a longer training period of ten epochs.

v. Pretraining data

To assess the impact of pretraining data on MoleCLIP's performance, we pretrained MoleCLIP using the ImageMol dataset, which comprises 10-million drug-like molecules extracted from PubChem. The model was trained for two epochs, corresponding to the same number of iterations as ten epochs on the ChEMBL dataset.

vi. Augmentations ablation

The effect of augmentation during pretraining was evaluated by removing the during-training augmentation but not removing the generation stage augmentations that were applied during the contrastive task.

vii. Pretraining tasks ablation

To evaluate the effect of each of the two pretraining tasks, we conducted pretraining with each one of the tasks alone.

Table S12 – Finetuning results of control models on the MoleculeNet benchmarks under scaffold splitting. Errors represented as standard deviations based on three independent runs.

	Classification (ROC-AUC)		**Regression (RMSE)**	
Control	BACE (1513 samples)	BBBP (2039 samples)	FreeSolv (642 samples)	Esol (1218 samples)
Baseline (ECFP)	0.827±0.006	0.60±0.02	4.12±0.19	1.58±0.03
Primary MoleCLIP model	0.829±0.005	0.747±0.009	2.00±0.14	0.99±0.01
K values: 30, 300	0.833±0.010	0.719±0.007	1.92±0.09	0.89±0.02
$\boldsymbol{\tau = 0.05}$	0.823±0.004	0.717±0.006	2.14±0.11	0.92±0.03
$\boldsymbol{\tau = 0.1}$	0.808±0.015	0.708±0.021	2.34±0.21	0.96±0.03
1 epoch	0.841±0.011	0.712±0.009	2.09±0.08	1.00±0.05
10 epochs	0.815±0.003	0.726±0.012	2.28±0.16	0.95±0.01
Pubchem-10m (2 epochs)	0.827±0.009	0.706±0.008	2.17±0.19	0.93±0.03
No augmentations during pretraining	0.830±0.005	0.742±0.019	2.17±0.14	0.93±0.02
Only contrastive task	0.815±0.020	0.664±0.020	2.66±0.06	1.15±0.04
Only classification task	0.833±0.017	0.707±0.006	2.41±0.05	1.03±0.03

d. Interpretability

We note that interpretability methods usually rely on qualitative manual assessment of the results. Whereas this approach can be valuable for gaining intuition and incorporating expert knowledge, it also introduces potential bias through the selection of specific examples and subjective interpretation. Given these considerations, interpretability results should be approached with caution and are best viewed as exploratory tools that offer intuition about model behavior, rather than as evidence of what the model has learned.

To gain insight into the model's internal decision-making process we applied two interpretability techniques:

- **t-distributed Stochastic Neighbor Embedding (t-SNE)** - To better understand the representations learned during pretraining, we applied visualization techniques to illustrate different aspects of the model's latent space. This was done by t-SNE, allowing us to project high-dimensional features into a two-dimensional space. By inspecting samples within the embedded space, we aimed to derive qualitative insights about how the model organizes the encoded information and differentiates between data points.
- **Saliency mapping** – This method was employed in order to identify the regions of an input image that most strongly influenced the model's performance, either during the pretraining or finetuning phase. This involved computing the gradient of the output with respect to the input image, thereby providing an assessment of each region's importance. Since our architecture is based on a ViT, which processes images as 16×16 pixel patches (tokens), we computed saliency at the patch level. The resulting token-level importance scores were then interpolated back to the original image resolution (224×224) to generate a heatmap.

i. Role of pretraining tasks

To assess the contribution of each pretraining task to the learned embeddings and downstream performance, we analyzed their effects on the organization of the latent space and the model's predictive ability. To isolate the effect of each of our two tasks (structural classification and contrastive learning) we compared our primary model with models trained in our ablation study: one trained only with contrastive learning and another with only the structural classification task.

To make the visualization of this large space tractable, we constructed a subset of the ChEMBL dataset containing 3,000 molecules, comprising 10 randomly selected representatives from each of the 300 classes used in the structural classification task. The impact of contrastive learning was evaluated using t-SNE to visualize the embedding of each molecule alongside its augmented version. We then measured the distance in the latent space between pairs of differently augmented versions of the same molecule. We performed this analysis with our primary model (trained with both tasks) and with a classification-only model. As shown in Figure S9, the model lacking the contrastive objective failed to embed augmented pairs in close proximity. This observation is supported quantitatively by a significantly higher mean distance between corresponding pairs in the classification-only model.

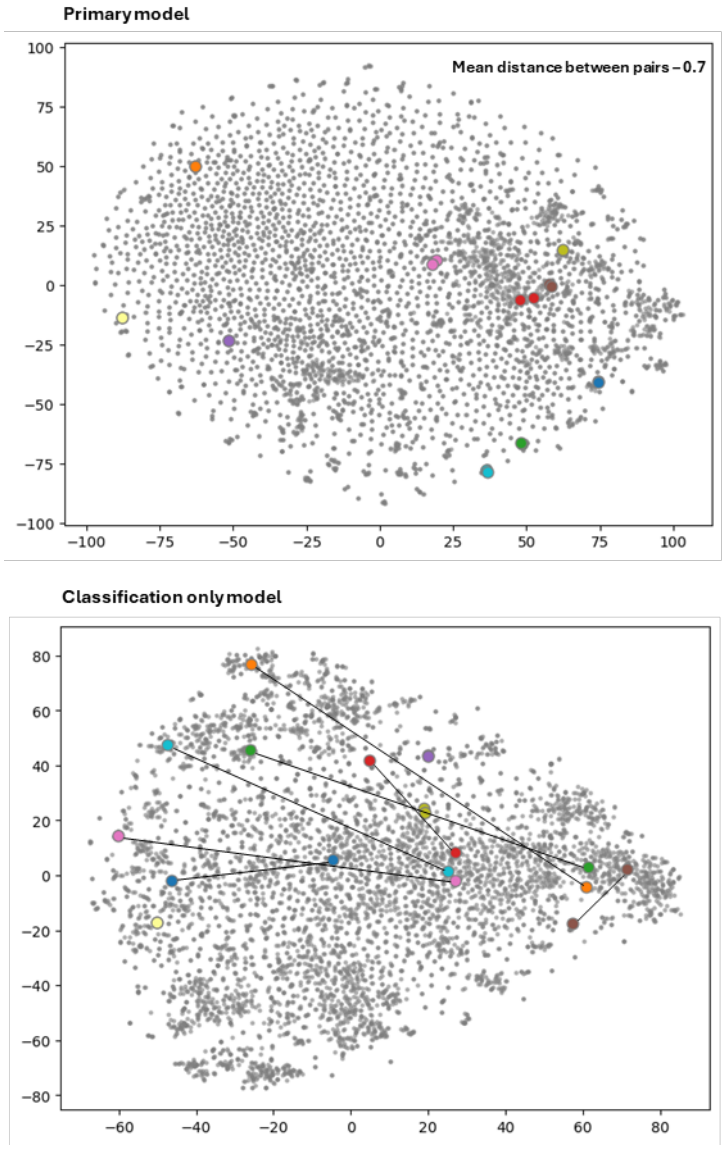


Figure S9 - t-SNE visualization of the latent space for 3,000 molecules from ChEMBL and their augmented versions. In our primary model (top), augmented pairs are embedded in close proximity, while the model trained with the classification task only (bottom) shows a significantly greater distance between augmented pairs.

Next, we evaluated the role of the structural classification task in shaping the latent space organization. We created a new ChEMBL subset, by randomly picking 10 structural classes from those used in the structural classification task, and randomly sampling 300 molecules out of each class. This resulted in a subset of 3000 molecules. We visualized the embeddings of all the molecules using t-SNE, assigning a unique color to

each cluster. We compared the results for our primary model and the contrastive-only model. The contrastive-only model exhibited a less structured latent space and was unable to clearly separate structural classes (see Figure S10). This is further supported by a higher Davies–Bouldin Index, indicating less differentiation between classes compared to our primary model.

Figure S10 - t-SNE visualization of embeddings for 3,000 molecules from ChEMBL, colored by K-means clustering (k=10) based on MACCS fingerprints. The primary model (top) shows better separation between clusters compared to the contrastive-only model (bottom). The contrastive-only model also has higher Davies–Bouldin Index, indicating poorer structural organization.

We aimed to assess how the pretraining tasks influenced performance during finetuning. To do this, we used the NHCs dataset due to its relatively small size, which facilitated manual interpretation, and because it was curated by our lab, allowing us to draw better chemical insights. Each of our three models, the primary model, the contrastive-only model, and the classification-only model, was trained on the NHC charge prediction finetuning task. We then visualized their saliency maps to examine which regions of the input images were most influential in the model's predictions (Figure S11).

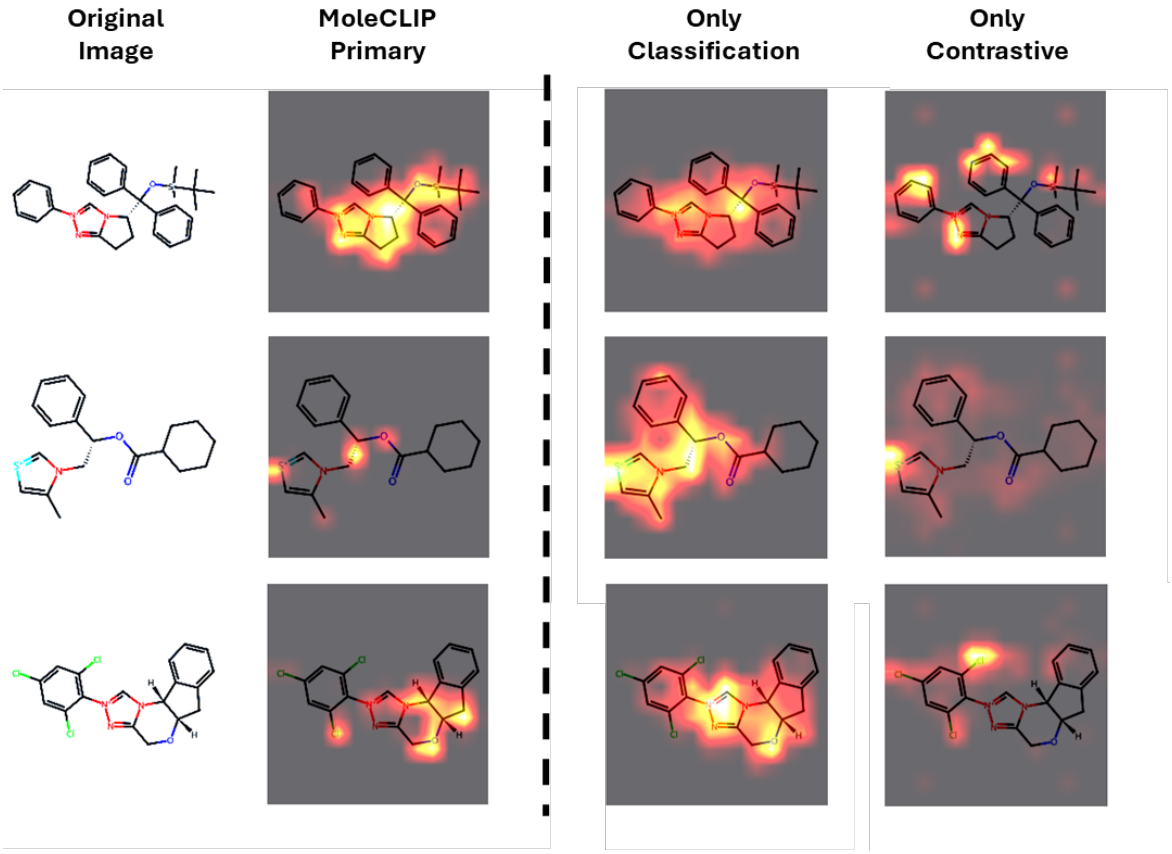


Figure S11 - Saliency map comparison across models on the NHCs finetuning task using three different pretrained models: contrastive-only, classification-only, and the primary model. Brighter regions indicate higher attention by the model.

The results reveal distinct patterns across the models. The contrastive-only model primarily focused on peripheral groups, which aligns with its objective of distinguishing molecules based on fine-grained, localized differences. In contrast, the classification-only model concentrated on the

central scaffold, as this structure typically defines the molecule's broader structural class. Interestingly, the primary model appeared to integrate both strategies: it focuses on both peripheral groups and the central scaffold and, in some cases, highlighted regions not emphasized by either of the other models. These observations suggest that each pretraining task contributes complementary information.

ii. Effect of generation-level augmentation on pretraining

During pretraining, particularly under contrastive learning, the model is exposed to various augmentations of molecular images, including variations of line-width, font size, type, etc. Such augmentations may affect recognizability of chemically meaningful substructures, especially when functional groups become proportionally smaller in larger molecules. To investigate whether the model remains robust to these variations, we examined saliency maps generated using our primary model for the same molecules under different augmentations. This analysis allowed us to evaluate whether the model consistently pays attention to the same chemically relevant regions regardless of image augmentations.

Visualizations such as the positive examples presented in Figure S12 (top) were the most commonly observed, generally supporting the model's robustness to these variations, with the model usually focusing on the same functional groups across augmentations. Nonetheless, we observed a few negative cases where the model failed to refer to similar regions. These failures were more common in very large molecules, where the visual scale of individual groups is reduced, making them harder to detect. We also note that our augmentation procedure includes sampling font sizes from a fixed range. As a result, images of large molecules that originally have smaller font sizes as their default, often only have augmented images with larger font sizes. While this improves the visibility of functional groups, it can obscure bond structures and could lead to misinterpretations.

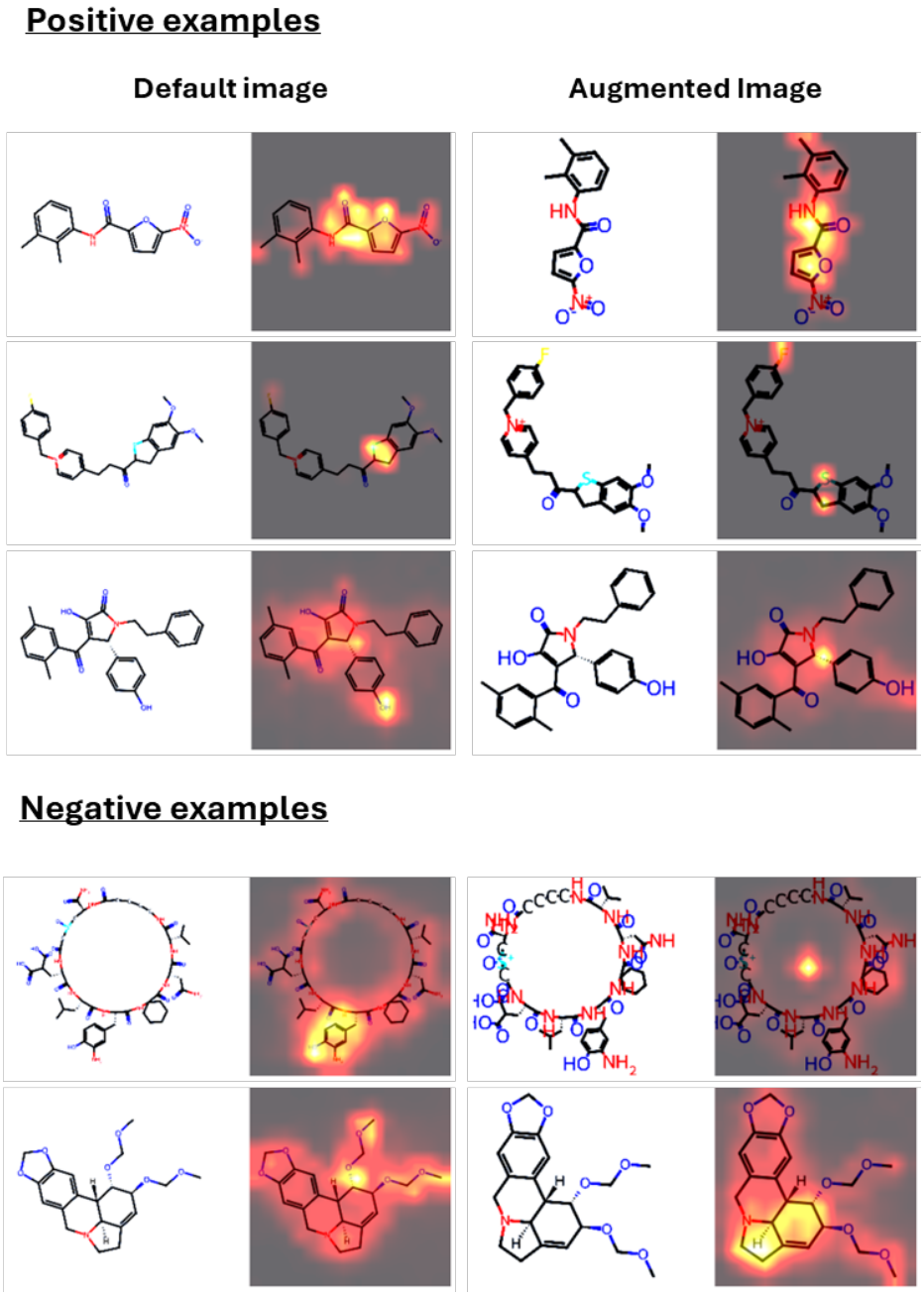


Figure S12 - Saliency visualizations for a single molecule presented under different generation-level augmentations (e.g., bond length, font variation). Brighter regions indicate higher attention by the model. The top section shows positive examples where the model focuses on to the same molecular substructures despite the augmentations. These are representative of most of the saliency maps we evaluated. The bottom section shows negative examples where the model's attention shifts to different regions across augmentations, which were not commonly encountered.

Data availability

The code, datasets, and results related to MoleCLIP are openly available through the following links:

(1) MoleCLIP code and resources – https://github.com/Milo-group/MoleCLIP

(2) ChEMBL-25 dataset – https://ftp.ebi.ac.uk/pub/databases/chembl/ChEMBLdb/releases/chembl_25/

(3) Pretrained weights and curated phosphines datasets – https://zenodo.org/records/13826016

(4) DHBDs dataset – https://github.com/Milo-group/MoleCLIP/tree/main/Datasets/DHBDs

(5) NHCs dataset – https://github.com/Milo-group/MoleCLIP/tree/main/Datasets/NHCs

(6) Phosphines – yield dataset – https://github.com/Milo-group/MoleCLIP/tree/main/Datasets/Phosphines_yield

(7) Phosphines – selectivity dataset – https://github.com/Milo-group/MoleCLIP/tree/main/Datasets/Phosphines_selectivity

(8) Summary of all finetuning results – https://github.com/Milo-group/MoleCLIP/tree/main/Paper_results

References

1. Radford, A. *et al.* Learning Transferable Visual Models From Natural Language Supervision. *Proc Mach Learn Res* **139**, 8748–8763 (2021).

2. Dosovitskiy, A. *et al.* An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ICLR 2021 - 9th International Conference on Learning Representations* (2020).

3. Landrum, G. *et al.* rdkit/rdkit: 2023_09_1 (Q3 2023) Release Beta. (2023) doi:10.5281/zenodo.8413907.

4. *CHEMBL Database Release 25*. (2019) doi:10.6019/CHEMBL.database.25.

5. Weininger, D. SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules. *J Chem Inf Comput Sci* **28**, 31–36 (1988).

6. Zeng, X. *et al.* Accurate prediction of molecular properties and drug targets using a self-supervised image representation learning framework. *Nat Mach Intell* **4**, 1004–1016 (2022).

7. Durant, J. L., Leland, B. A., Henry, D. R. & Nourse, J. G. Reoptimization of MDL keys for use in drug discovery. *J Chem Inf Comput Sci* **42**, 1273–1280 (2002).

8. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *J Mach Learn Re*s **12**, 2825–2830 (2011).

9. Satopää, V., Albrecht, J., Irwin, D. & Raghavan, B. Finding a 'kneedle' in a haystack: Detecting knee points in system behavior. in *Proceedings - International Conference on Distributed Computing Systems* 166–171 (2011). doi:10.1109/ICDCSW.2011.20.

10. Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. in *International Conference on Machine Learning* 1597–1607 (2020). doi:10.5555/3524938.3525087.

11. Zhai, X., Mustafa, B., Kolesnikov, A. & Beyer, L. Sigmoid Loss for Language Image Pre-Training. in *Proceedings of the IEEE International Conference on Computer Vision* 11941–11952 (Institute of Electrical and Electronics Engineers Inc., 2023). doi:10.1109/ICCV51070.2023.01100.

12. Kirkpatrick, J. *et al.* Overcoming catastrophic forgetting in neural networks. *Proc Natl Acad Sci U S A* **114**, 3521–3526 (2017).

13. Wu, Z. *et al.* MoleculeNet: a benchmark for molecular machine learning. *Chem Sci* **9**, 513–530 (2018).

14. Gallarati, S. *et al.* OSCAR: an extensive repository of chemically and functionally diverse organocatalysts. *Chem Sci* **13**, 13782–13794 (2022).

15. Flanigan, D. M., Romanov-Michailidis, F., White, N. A. & Rovis, T. Organocatalytic Reactions Enabled by N-Heterocyclic Carbenes. *Chem Rev* **115**, 9307–9387 (2015).

16. Frisch, M. J. *et al.* Gaussian˜16 Revision C.01. Preprint at (2016).

17. Valero, R., Gomes, J. R. B., Truhlar, D. G. & Illas, F. Good performance of the M06 family of hybrid meta generalized gradient approximation density functionals on a difficult case: CO adsorption on MgO(001). *J Chem Phys* **129**, 124710 (2008).

18. Zhao, Y. & Truhlar, D. G. The M06 suite of density functionals for main group thermochemistry, thermochemical kinetics, noncovalent interactions, excited states, and transition elements: Two new functionals and systematic testing of four M06-class functionals and 12 other functionals. *Theor Chem Acc* **120**, 215–241 (2008).

19. Weigend, F. & Ahlrichs, R. Balanced basis sets of split valence, triple zeta valence and quadruple zeta valence quality for H to Rn: Design and assessment of accuracy. *Phys Chem Chem Phys* **7**, 3297–3305 (2005).

20. Weigend, F. Accurate Coulomb-fitting basis sets for H to Rn. *Phys Chem Chem Phys* **8**, 1057–1065 (2006).

21. Glendening, E. D., Reed, A. E., Carpenter, J. E. & Weinhold, F. NBO Version 3.1. included in Gaussian 16. (2016).

22. Legault, C. Y. CYLview20. http://www.cylview.org (2020).

23. Newman-Stonebraker, S. H. *et al.* Univariate classification of phosphine ligation state and reactivity in cross-coupling catalysis. *Sci (1979)* **374**, 301–308 (2021).

24. Niemeyer, Z. L., Milo, A., Hickey, D. P. & Sigman, M. S. Parameterization of phosphine ligands reveals mechanistic pathways and predicts reaction outcomes. *Nat Chem 2016 8:6* **8**, 610–617 (2016).

25. Fang, X. *et al.* Geometry-enhanced molecular representation learning for property prediction. *Nat Mach Intell* **4**, 127–134 (2022).

26. Ahmad, W., Simon, E., Chithrananda, S., Grand, G. & Ramsundar, B. ChemBERTa-2: Towards Chemical Foundation Models. Preprint at https://doi.org/10.1088/2632-2153/acdb30 (2022).

27. Wang, Y., Wang, J., Cao, Z. & Barati Farimani, A. Molecular contrastive learning of representations via graph neural networks. *Nat Mach Intell 2022 4:3* **4**, 279–287 (2022).

28. Zhou, G. *et al.* Uni-Mol: A Universal 3D Molecular Representation Learning Framework. Preprint at https://doi.org/10.26434/chemrxiv-2022-jjm0j-v2 (2022).

29. Fang, X. *et al.* Geometry-enhanced molecular representation learning for property prediction. *Nat Mach Intell* **4**, 127–134 (2022).

30. Zhang, Y. *et al.* A survey of drug-target interaction and affinity prediction methods via graph neural networks. *Comput Biol Med* **163**, 107136 (2023).

31. Nguyen, T. *et al.* GraphDTA: predicting drug–target binding affinity with graph neural networks. *Bioinform***37**, 1140–1147 (2021).

32. Öztürk, H., Ozkirimli, E. & Özgür, A. WideDTA: prediction of drug-target binding affinity (2019). Preprint at https://doi.org/10.48550/arXiv.1902.04166

33. Zhang, S. *et al.* SAG-DTA: Prediction of Drug–Target Affinity Using Self-Attention Graph Network. *Int J Mol Sci 2021, Vol. 22, Page 8993* **22**, 8993 (2021).

34. Bi, X., Zhang, S., Ma, W., Jiang, H. & Wei, Z. HiSIF-DTA: A Hierarchical Semantic Information Fusion Framework for Drug-Target Affinity Prediction. *IEEE J Biomed Health Inform* (2023) doi:10.1109/JBHI.2023.3334239.

35. Gensch, T. *et al.* A Comprehensive Discovery Platform for Organophosphorus Ligands for Catalysis. *J Am Chem Soc* **144**, 1205–1217 (2022).

36. Kim, S. *et al.* PubChem 2023 update. *Nucleic Acids Res* **51**, D1373–D1380 (2023).

37. Coley, C. W., Rogers, L., Green, W. H. & Jensen, K. F. SCScore: Synthetic Complexity Learned from a Reaction Corpus. *J Chem Inf Model* **58**, 252–261 (2018).