Benchmarking ML in ADMET predictions: the practical impact of feature representations in ligand-based models

Gintautas Kamuntavičius*, Tanya Paquet*, Orestis Bastas*, Dainius Šalkauskas, Alvaro Prat, Hisham Abdel Aty, Povilas Norvaišas, and Roy Tal

*Main contributors
AI Chemistry, Ro5, 2801 Gateway Drive, Irving, 75063, TX, USA

Abstract

This study, focusing on predicting Absorption, Distribution, Metabolism, Excretion, and Toxicology (ADMET) properties, addresses the key challenges of ML models trained using ligand-based representations. We propose a structured approach to data feature selection, taking a step beyond the conventional practice of combining different representations without systematic reasoning. Additionally, we enhance model evaluation methods by integrating cross-validation with statistical hypothesis testing, adding a layer of reliability to the model assessments. Our final evaluations include a practical scenario, where models trained on one source of data are evaluated on a different one. This approach aims to bolster the reliability of ADMET predictions, providing more dependable and informative model evaluations.

1 Introduction

The Absorption, Distribution, Metabolism, Excretion, and Toxicology (ADMET) of compounds are commonly estimated throughout drug discovery projects, as the feasibility of a compound to become a viable drug highly depends on it. Through the years, a lot of work has gone into building and evaluating machine learning (ML) systems designed to predict molecular properties that are associated with ADMET. Public curated datasets and benchmarks for ADMET associated properties are becoming increasingly available to the community, creating the opportunity for more widespread exploration of ML algorithms and techniques in this space. The Therapeutics Data Commons (TDC) ADMET leaderboard showcases this [1], highlighting a wide variety of models, features, and processing methods investigated over the past two years.

The studies showcased on the leaderboard often focus on comparing different ML models and architectures, whereas the selection of compound representations is either not justified, or analyzed with limited scope. For instance, many approaches concatenate a number of compound representations at the onset for the assessment of various models. While compound representation and feature selection justification is lacking, these approaches often yield very good results against the TDC benchmarks [2, 3, 4]. In the present study, we aim to improve the understanding of the impact of feature concatenation, taking a step further to provide a process that can inform dataset-specific, statistically significant compound representation choices.

Different deep neural network (DNN) compound representations became prevalent over the past years [5, 6, 7, 8]. We investigate how the DNN compound representations compare to the more classical descriptors and fingerprints in the ADMET ML domain.

In the present study, we conduct experiments to enlighten the following research questions:

- Which types of algorithms and compound representations are generally suitable for ligand-based machine learning in the ADMET domain?
- Can cross-validation hypothesis testing serve as a more robust model comparison than a hold-out test set in the ADMET domain?
- How important are various forms of model optimization in a practical scenario?
- What is the impact on the model performance when available external data of the same property is used in combination with internal data?

Public ADMET datasets are often criticised with regards to data cleanliness. Issues range from inconsistent SMILES representations and multiple organic compounds found in a single fragmented SMILES string, to duplicate measurements with varying values and inconsistent binary labels. Different binary labels for the same SMILES string have also been observed across train and test sets. In order to mitigate these issues in the present study, we begin by applying a set of data cleaning procedures. The data cleaning results in the removal of a number of compounds across

datasets[1].

At a high level, the experiments in this study are carried out sequentially, achieving the following:

1. A best-performing model architecture is identified to both use as a baseline as well as optimized in further experiments;
2. Features are combined iteratively until the best-performing combinations are identified;
3. Hyperparameters of the chosen model architecture are tuned in a dataset-specific manner;
4. Cross-validation hypothesis testing is done in order to assess the statistical significance of the optimization steps;
5. Test set performance is evaluated, assessing the impact of the previous optimization steps, as well as the contrast between the hypothesis test outcomes and test set changes;
6. The optimized models are evaluated in a practical scenario, where models trained on one data source are evaluated on a test set from a different source, for the same property;
7. Finally, the optimized model is trained on a combination of data from two different sources, to mimic the scenario when external data is combined with increasing amounts of internal data.

2 Related work

Fang *et al.* have assessed the performance of many popular ML models on their internal ADME assay data [9]. While they look at a variety of models, the investigation of different compound representations are more limited, exploring only combinations of RDKit descriptors and functional connectivity fingerprints with a radius of 4 (FCFP4). Their experiments were done in a sequential manner and evaluations were carried out on temporal splits. They have also shared their in-house ADME assay results for around 3,000 purchasable compounds. This dataset has been invaluable in our study, allowing us to assess the impact of external data on internal data prediction.

Most recently, Green *et al.* carried out a study on ADMET and quantitative structure-activity relationship (QSAR) tasks, focusing on a wide range of models and features [10]. They propose a principled implementation of uncertainty estimation (estimates for both aleatoric and epistemic uncertainty) as well as calibration, highlighting the superior performance of Gaussian Process (GP) based models in particular. While the group showed GP models to consistently perform the best in bioactivity assays, there was no such clear conclusion for ADMET datasets, for which they have found the optimal model and feature choices to be highly dataset-dependent.

Deng *et al.* have carried out a study [11] that has many similarities with ours; in particular, a number of model were trained with both classical and deep-learned feature representations on ADMET as well as other datasets. The study contains a very thorough analysis and comparison of generally popular ML methods in the area. The authors investigate fixed versus learned (i.e. fine-tuned to the particular dataset) representations, arriving at a conclusion that fixed representations generally outperform learned ones. Moreover, the random forest model architecture was found to be the generally best performing one. In our study we investigate fixed representations, taking a step further to investigate combinations of various representations as well as identify a different best-performing model architecture.

3 Methods

3.1 Data

Datasets Datasets pertaining to the ADMET properties of small molecules were obtained from different public sources as laid out in Table 1.

From TDC [1], the `single_pred` method was used to obtain exclusively human `ppbr_az` data. For all other TDC datasets, the recommended `benchmark_group` method and scaffold splits were used. Kinetic solubility from the National Institute of Health (NIH) as described by Guha *et. al* was obtained from PubChem [12]. The scaffold split method within the DeepChem library was used to split the Biogen and NIH datasets.

Data cleaning Data cleaning was aimed at getting consistent SMILES representations, and to remove noise due to measurement ambiguity. For *in vitro* and *in vivo* assays, we assume that the compound, or salt thereof, is soluble in the medium used, and that the effect observed can be attributed to the parent organic compound in the case of salts.

For solubility, the properties of different salts of the same compound may differ depending on the salt component. All records pertaining to salt complexes were removed from solubility datasets.

The standardisation tool by Atkinson *et al.* were used to clean the compound SMILES strings [13]. We included two modifications to definitions within the tool.

- Boron and silicon were added to the list of organic elements as such that an organic compounds is defined as a compound that only consists of the

[1] Note this technically makes our results incomparable to any of the public leaderboard models, even though the number of removed compounds in the test sets is minor.

following elements: H^{1}, C^{6}, N^{7}, O^{8}, F^{9}, P^{15}, S^{16}, Cl^{17}, Br^{35}, I^{53}, B^{5}, and Si^{14}.
- Positive and negative hydrogen ions were added to the pre-defined salt list as they were present as salt components in some datasets, e.g. as `[H+].[Cl-]`.

In addition, a truncated salt list was created to omit salt components that can in themselves be a parent organic compound with a property measurement e.g. citrate/citric acid. The truncated list was created by excluding components that contain two or more carbons from the tool's pre-defined list. 36 components were excluded as such.

The following steps were the taken to conduct data cleaning across the datasets:

- Remove inorganic salts and organometallic compounds from the datasets.
- Extract organic parent compounds form their salts forms.
- Adjust tautomers to have consistent functional group representation.
- Canonicalize SMILES strings.
- De-duplication. We either keep the first entry if the target values of the duplicates are consistent, or remove the entire group if they are inconsistent. "Consistent" is defined as exactly the same for binary tasks (i.e. the target values of the group are either all 0 or all 1), and within 20% of the inter-quartile range for regression tasks.

Finally, since the datasets are relatively small, visual inspection of the resultant clean datasets were done using DataWarrior [14].

Many of the ADMET endpoints in the datasets are log-transformed. To address highly skewed distributions, we transformed another three of the TDC datasets namely, `clearance_microsome_az`, `half_life_obach` and `vdss_lombardo`. For these, the metrics shown in this work are computed on the log transformed values instead of the original ones listed in Table 1.

3.2 Modeling and evaluation

Models The machine learning algorithms included in the present study range from classical models to more recent neural networks. Included is Support Vector Machines (SVM) [17], tree-based methods comprising Random Forests (RF) [18] and gradient boosting frameworks LightGBM [19] and CatBoost [20], as well as Message Passing Neural Networks (MPNN) as implemented by Chemprop [21] (Ver. 1.6.1).

Features Various descriptors, fingerprints, and embeddings were used on their own or in combination. The following descriptors and fingerprints were implemented using the RDKit cheminformatics toolkit [22]: RDKit descriptors (`rdkit_desc`), Morgan fingerprints with a radius of 2 (`ecfp4`) [23], atom pair fingerprints (`atom_pair`) [24], Avalon Fingerprints (`avalon`) [25], and Extended reduced graph (`erg`) descriptors [26]. Mordred fingerprints (`mordred`) were obtained using the Mordred molecular descriptor calculator [27]. Embeddings used includes Mol2vec (`mol2vec`) [8], Graph Representation frOm self-supervised mEssage passing tRansformer (`grover`) [7], MolFormer (`molformer`) [6], and BARTSMILES (`bartsmiles`) [5]. For MolFormer, we made use of the open-source model version that was trained on 10% of the data. BARTSMILES embeddings were extracted through loading the fairseq Bart model with the pre-trained BARTSMILES checkpoint, encoding the SMILES and extracting their features using fairseq's functions, and averaging them to create a vector for each datapoint. MegaMolBart representations were computed using Nvidia's BioNemo API [28].

Evaluation metrics Following the work from Green *et. al*, we used the normalized root-mean-square error (NRMSE) for regression tasks [10]. It is defined as the RMSE divided by the inter-quartile range (IQR) of the training set. For binary classification tasks, the area under the precision-recall curve (AU-PRC) was used due to the presence of imbalanced datasets.

Statistics The Friedman χ^2 test [29] was used to detect differences across multiple pairwise model and feature comparisons. The test compares ranks rather than actual values, making it appropriate for non-normally distributed data.

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{1}$$

In Equation 1, in the context of our study, N is the number of pairwise comparisons (e.g. all 130 model and dataset pairs that use `rdkit_desc` features), k is the number of methods (e.g. the 10 different feature sets), and R_j is the total sum of the ranks of the method in question (e.g the `rdkit_desc` features).

After significant results were obtained from the Friedman χ^2 test, we used the Nemenyi post-hoc test [30] to compare multiple groups.

$$Q_{ij} = \frac{|R_i - R_j|}{\sqrt{\frac{k(k+1)}{6n}}} \tag{2}$$

The Nemenyi test were chosen due to its robustness to the multiple comparisons problem compared to other methods, such as the Wilcoxon rank sum test [31]. This is because the number of comparisons k is taken into account when computing the test statistic Q_{ij} (Eq.

Table 1: Dataset descriptions.

Dataset Name (Table)	Property	Units	Size (Distribution)
TDCommons - regression			**(IQR)**
caco2_wang (A16)	Cell effective permeability	log(Papp)	632 (-5.7; -4.6)
lipophilicity (A36)	Octanol/water distribution	logD	4,199 (-0.6; 0.8)
ppbr_az (A31)	Human plasma protein binding	% bound	1,614 (85; 99)
ld50_zhu (A17)	50% lethal dose	$\log(\text{kg}\cdot\text{mol}^{-1})$	7,308 (1.9; 3.0)
vdss_lombardo (A30)	Steady-state volume of distribution	$\text{L}\cdot\text{kg}^{-1}$	1,105 (0.3; 2.8)
half_life_obach (A23)	Terminal phase half life from IV administration	hr	663 (1.8; 11)
clearance_microsome_az (A22)	Human liver microsome intrinsic clearance	$\text{mL}\cdot\text{min}^{-1}g^{-1}$	1,102 (3.0; 43)
TDCommons - binary		**Positive threshold**	**(# positive)**
bioavailability_ma (A33)	Human oral bioavailability	$F \geq 20\%$	638 (491)
hia_hou (A35)	Human intestinal absorption	$FA > 30\%$	578 (500)
pgp_broccatelli (A34)	P-glycoprotein inhibition	$IC_{50} < 15\ \mu$M, or $> 25\%$ inhibition relative to control	1,212 (647)
bbb_martins (A32)	Blood-brain barrier penetration	logBB $\geq$ -1	1,945 (1,490)
cyp2c9_veith (A27)	CYP2C9 inhibition	inhibition below 57 μM	11,927 (3,993)
cyp2d6_veith (A29)	CYP2D6 inhibition	"	12,960 (2,491)
cyp3a4_veith (A28)	CYP3A4 inhibition	"	12,170 (5,037)
cyp2c9_substrate_carbonmangels (A24)	CYP2C9 metabolism	as per literature annotation [15]	666 (141)
cyp2d6_substrate_carbonmangels (A26)	CYP2D6 metabolism	"	663 (190)
cyp3a4_substrate_carbonmangels (A25)	CYP3A4 metabolism	"	667 (354)
herg (A20)	hERG inhibition	$pIC_{50} \geq 4.4$	603 (414)
ames (A19)	Mutagenicity (bacterial reverse mutation assay)	colony growth in at least 1 of 5 strains	7,220 (3,941)
dili (A18)	Drug-induced liver injury (hepatotoxicity)	according to FDA approved labeling [16]	466 (230)
NIH - regression			**(IQR)**
solubility (A21)	Kinetic aqueous solubility	μg·mL	36,238 (4.0; 35)
Biogen - regression			**(IQR)**
hppb	Human plasma protein binding	log(% unbound)	194 (0.2; 1.5)
rlm (A12)	Rat liver microsomes intrinsic clearance	$\log(\text{mL}\cdot\text{min}^{-1}\cdot\text{kg}^{-1})$	3,054 (1.7; 2.8)
solubility (A13)	Kinetic aqueous solubility	log(μg·mL) (pH 6.8)	2,173 (1.2; 1.7)
hlm (A15)	Human liver microsome intrinsic clearance	$\log(\text{mL}\cdot\text{min}^{-1}\cdot\text{kg}^{-1})$	3,087 (0.7; 1.8)
mdr1-mdck (A14)	P-glycoprotein efflux ratio	log(B-A/A-B)	2,642 (-0.2; 0.9)

2), which follows the studentized range distribution [32].

The non-parametric nature of the statistical test allowed us to use information from both the binary classification and regression tasks, as only the relative rankings are considered. Therefore, we combined the rankings of the NRMSE values for regression tasks with the rankings of the AU-PRC values for binary classification tasks to perform the post-hoc tests.

3.3 Experiments

The experiments carried out in this study were sequential, each making use of the findings of the previous experiments. They were structured as follows:

1. **Model selection**. Initially, we make use of the single-fold experiment setup: train on training set, evaluate of validation set, test set not used. In the first stage, we aim to identify a single model out of the five that we will take forward in more detailed feature combination as well as hyperparameter tuning experiments. A model is trained for each dataset, feature, and model combination, and the models are ranked (1 to 5) in comparison to each other. This results in $25 \cdot 11 \cdot 5 = 1375$ trained models. Subsequently, we also compare models trained using each of the features (except `mordred`)[2] in combination with `rdkit_desc`, a popular feature representation that we have also found to work well. The performance of a model trained on a combination of features is important as it can suggest whether the model architecture will continue performing well as we expand the number of combined features. This results in another 9 feature combinations, corresponding to $25 \cdot 9 \cdot 5 = 1125$ trained models. The models are again ranked (1 to 5), their average rankings are investigated and the significance of the best models' performance is assessed using the Nemenyi test.
2. **Feature combination** Once the model is selected, the next step is to understand the impact of using multiple feature representations on the efficacy of ADMET models. In order to assess this, we iteratively add features one-by-one, starting with the best-performing `rdkit_desc`, evaluating the change in performance across the datasets at every step. Features are added until we observe no improvement from the additional features, thus avoiding the inclusion of noise into the training process. In these experiments, we discovered that the impact of feature choice is very different for regression and binary classification datasets. We therefore perform this iterative feature addition separately for regression and binary classification datasets. Once final feature combinations are selected, hyperparameter optimization is done.
3. **Hyperparameter optimization** The hyperparameter optimization is performed in a dataset-specific manner, unlike the previous optimization steps which were evaluated across all the datasets at once. We use a simple 3-fold cross-validation method, using random search with 20 iteration steps. Random search was chosen due to its robustness compared to grid search when redundant hyperparameters might be involved [33]. The hyperparameter grid is defined in Table 2, chosen based on recommendations from the model authors [20].

Table 2: CatBoost hyperparameter grid used in the random search with 20 iterations. The total size of the grid amounts to 960 possible combinations. The default hyperparameter values are bolded.

Hyperparameter	Values
`depth`	4, **6**, 8, 10
`learning_rate`	0.01, **0.03**, 0.05, 0.1, 0.3
`iterations`	500, **1000**, 2000
`l2_leaf_reg`	1, **3**, 5, 7
`bagging_temperature`	0, **1**, 3, 5

4. **Cross-validation hypothesis testing** To understand whether the optimization choices made in previous steps are meaningful, we perform large scale cross-validation hypothesis testing. In order to maximize the statistical power of the Nemenyi test, we use 10 cross-validation folds. Four model configurations are evaluated:
   - Baseline: `catboost` with default parameters and `rdkit_desc` feature representations.
   - `catboost` with default parameters and `rdkit_desc` + `ecfp4` feature representation combination, which is a popular choice in literature
   - `catboost` with default parameters and the optimized features, as described in Section 4.2.
   - `catboost` with optimized hyperparameters as well as optimized features.

   For each of the four model configuration, we train a model on 25 datasets with 10 folds each, resulting in 250 sets of four paired evaluations to draw the statistical test from.
5. **Test set evaluation** Finally, the baseline as well

[2] `mordred` features were not combined with `rdkit_desc` because the `mordred` descriptors already contain `rdkit_desc`, along with a number of other features.

as optimized models are evaluated on the held-out test set and the improvements are listed for each dataset. Importantly, the performances are shown per-dataset, and they are compared parametrically (NRMSE for regression tasks and AU-PRC for binary classification); in the previous analyses we have used relative ranking to identify the superior models, without considering the magnitude of the improvement. Moreover, together with test set results we observe the hypothesis test results for each dataset from the previous 10-fold CV experiment. The hypothesis test outcome is compared to the change in test set performance.

6. **Transferability**. Properties for which two or more datasets were obtained were used to investigate whether, for a given property, data from one laboratory can be used to predict the measurements from a different laboratory. For these transferability investigations across datasets of the same property, the respective dataset values were transformed to align the measurement units, and overlapping compounds were removed, as well as those with values greater or less than a threshold. The three properties, human plasma protein binding (hPPB), human liver microsome intrinsic clearance (HLM), and kinetic solubility (solubility), for which there are both a Biogen dataset as well as a dataset from TDC (AstraZeneca) or the NIH, were used. For each property, two sets of experiments were conducted.

    - Firstly, the optimized models from the previous experiments are evaluated in a practical scenario: `catboost` models were trained on either the TDC or NIH data, and tested on Biogen data. This was done using all four model configurations described in the cross-validation experiment.
    - Secondly, an experiment is conducted to assess the impact of using external (hypothetically represented by TDC or NIH) together with internal (hypothetically represented by Biogen) data. 50% of the Biogen data were isolated as a test set via a scaffold split. The remaining 50% of the Biogen data was incrementally added (either 1% or 5% increments up to 50%) to the additional dataset's data, resulting in training sets containing increasing proportions of Biogen data. Dataset compositions are summarised in Table 3. `catboost` models with the optimal hyperparameters for the property were constructed with these training sets using the combined features. The models were evaluated based on R^2 against the isolated test set. Models were also trained using only the Biogen portions of the training data to asses the change in performance due to the additional data. The experiments were repeated with 5 different 50% scaffold splits of the Biogen datasets.

Table 3: Maximum combined dataset compositions for transferability experiments.

Property	Maximum combined dataset[a]	Fraction Biogen data[b]	Biogen test set size
hPPB	1,704	0.053	92
HLM	1,781	0.590	1059
solubility	37,308	0.029	1076

[a] TDC or NIH data combined with 50% of the Biogen data. [b] Fraction of the maximum combined dataset represented by the Biogen data.

4 Results

4.1 Model selection

Table 4 shows the relative model performances across the single and combined feature experiments. While `catboost` and `svm` have comparable average rankings in the single feature experiment, `catboost` comes out superior once the features are combined with `rdkit_desc`.

This is corroborated by the p-value heatmaps (Fig. 1, 2). We observe that in the single feature experiment, `catboost` is not significantly different from `svm` and `lightgbm`. However, the difference becomes significant when the combination of two feature sets are used.

Based on these findings, in the subsequent experiments we chose to use the `catboost` model, both for investigating feature combinations and hyperparameter optimization.

4.2 Feature combination

First, we evaluate the `catboost` models trained on a single feature representation, confirming `rdkit_desc` as the superior standalone feature representation in ADMET tasks. The average ranks of each feature across the datasets are shown in Table 5. We can see that its superiority is more prominent in regression tasks compared to binary classification.

The process of iteratively adding features is shown in Table 6 for regression datasets and 7 for binary classification datasets. We find that the `rdkit_desc` + `erg` + `ecfp4` + `avalon` combination performs best on average for regression datasets, and `rdkit_desc` + `erg`

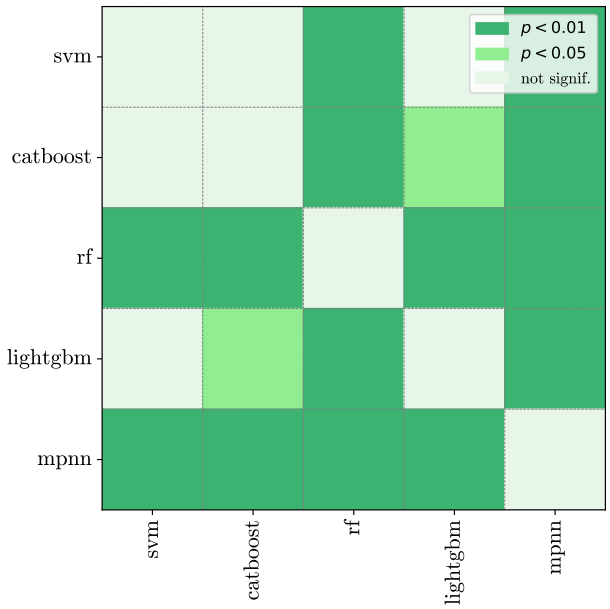

Figure 1: Model comparison, p-value heatmap according to the Nemenyi test of all 11 features trained individually. Each model is trained 275 times, for every dataset and feature combination. Average ranks of each model are shown in Table 4.

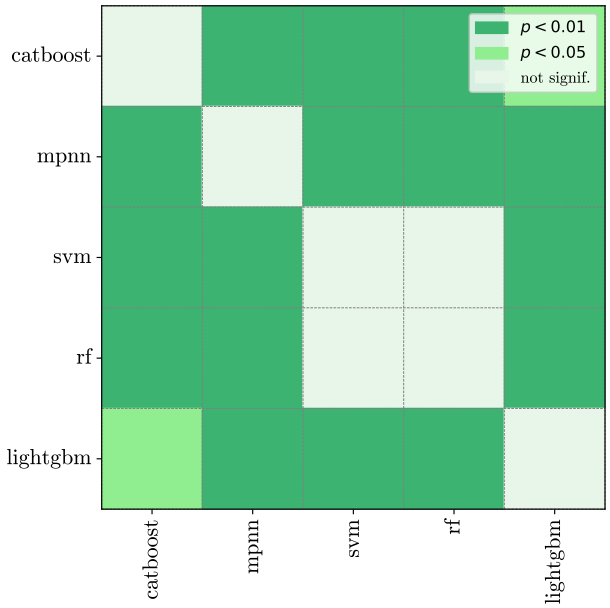

Figure 2: Model comparison, p-value heatmap according to the Nemenyi test of 9 features (previous 11 minus `rdkit_desc` and `mordred`) trained in combination with `rdkit_desc`. Each model is trained 225 times, for every dataset and feature combination. Average ranks of each model are shown in Table 4.

Table 4: Average model rankings in the single-fold experiment. In the single-feature experiment, all 11 features are all trained, resulting in 1325 total models trained across all datasets. For the two features experiment, `rdkit_desc` features are concatenated with the other features (except `mordred`), resulting in 1125 total models. The rankings are relative model performances based on the validation set. Corresponding p-value heatmaps can be seen in Figures 1, 2.

Model	Average rank, single feature	Average rank, two features
`svm`	**2.52**	3.01
`catboost`	**2.36**	**2.07**
`rf`	3.29	3.13
`lightgbm`	2.79	2.52
`mpnn`	4.03	4.28

+ `avalon` for binary classification datasets. Adding more features to these combinations decreases the model performance across the datasets.

Table 5: Feature performance comparison in the single-fold evaluation experiment, using a `catboost` model with default hyperparameters. A model using a particular feature representation is trained 25 times for every dataset, and relative average rankings are shown in the table. Separate rankings are also shown for regression and binary classification datasets, of which there are 12 and 13, respectively.

Feature	Regression	Binary	Overall
`molformer`	6.91	6.00	6.38
`bartsmiles`	8.91	6.85	7.83
`grover`	5.91	6.15	6.08
`mol2vec`	5.82	6.23	6.08
`atom_pair`	7.09	6.62	6.79
`ecfp4`	7.91	6.38	7.04
`rdkit_desc`	**1.91**	**4.31**	**3.21**
`erg`	7.45	9.08	8.33
`avalon`	7.18	5.69	6.42
`mordred`	2.45	**4.23**	3.38
`megamolbart`	4.45	**4.46**	4.46

Table 6: Performance (average rank) of iteratively added feature representations in **regression** datasets. Each set of base features, denoted in the top row, is combined with an extra feature and their performances are ranked compared to each other within same column. For example, in the first step, the `rdkit_desc` + `erg` has the highest average ranking compared to other two feature combinations as well as the baseline `rdkit_desc`; therefore we set it as the base two-feature combination in the subsequent column where a third feature is added in the same manner. This process continues until adding an extra feature does not show improvement.

	`rdkit_desc`	`rdkit_desc` + `erg`	`rdkit_desc` + `erg` + `ecfp4`	`rdkit_desc` + `erg` + `ecfp4` + `avalon`
No added feature	5.0	3.18	3.18	**1.55**
+ `erg`	**2.5**	-	-	-
+ `ecfp4`	3.08	**2.45**	-	-
+ `avalon`	3.5	2.55	**2.27**	-
+ `atom_pair`	4.42	4.0	2.55	2.73
+ `bartsmiles`	9.25	8.0	7.18	6.0
+ `grover`	7.33	7.18	6.27	5.55
+ `megamolbart`	5.83	5.45	4.64	4.27
+ `mol2vec`	5.67	4.82	3.64	2.55
+ `molformer`	8.42	7.36	6.27	5.36

Table 7: Performance (average rank) of iteratively added feature representations in **binary classification** datasets. Each set of base features, denoted in the top row, is combined with an extra feature and their performances are ranked compared to each other within same column. For example, in the first step, the `rdkit_desc` + `avalon` has the highest average ranking compared to other two feature combinations as well as the baseline `rdkit_desc`; therefore we set it as the base two-feature combination in the subsequent column where a third feature is added in the same manner. This process continues until adding an extra feature does not show improvement.

	`rdkit_desc`	`rdkit_desc` + `avalon`	`rdkit_desc` + `avalon` + `erg`
No extra feature	6.46	4.92	**3.46**
+ `avalon`	**4.08**	-	-
+ `erg`	5.38	**3.77**	-
+ `atom_pair`	5.92	4.69	4.08
+ `ecfp4`	4.62	4.38	4.46
+ `megamolbart`	4.38	4.46	4.69
+ `bartsmiles`	6.62	6.08	5.77
+ `grover`	6.38	6.38	5.62
+ `mol2vec`	5.54	5.15	3.62
+ `molformer`	5.62	5.15	4.31

Table 8: Performance of four different CatBoost model configurations, with either default or optimized hyperparameters as well as various feature representation combinations. Each configuration is trained across 25 datasets with 10 folds each, and the configurations' average ranking is reported. The significance of model improvements is visualized in terms of the p-values in Figure 3.

Features	Model	Average rank
`rdkit_desc`	`catboost` default	3.09
`rdkit` + `ecfp4`	`catboost` default	2.84
optimized features	`catboost` default	2.20
optimized features	`catboost` optimized params	**1.84**

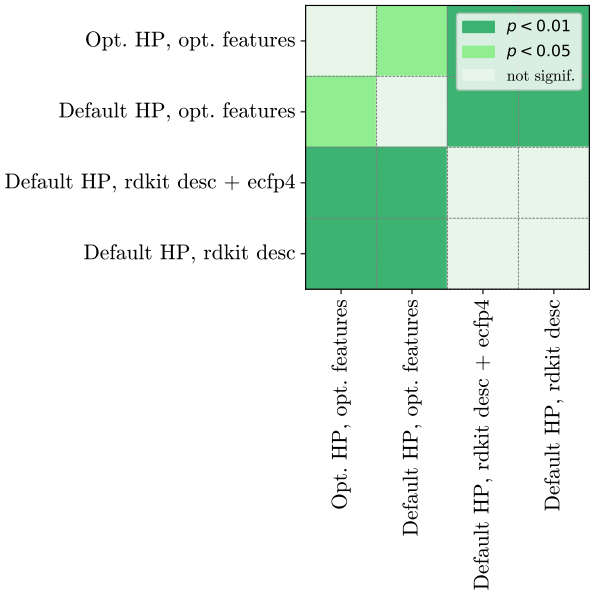

Figure 3: CatBoost model configuration comparison, p-value heatmap according to the Nemenyi test comparing four differently optimized models. Each model configuration is trained 250 times, for each of the 10 folds across 25 datasets. Average rankings of each model configuration are shown in Table 8.

4.3 Cross-validation hypothesis testing

The four chosen model configurations are now trained in a 10-fold cross-validation setting, where the large number of paired performance measurements will allow for a thorough assessment of impact of various optimization steps. We can see the average rankings of each model configuration in Table 8. To create this table, 25 (number of datasets) * 10 (number of CV folds) = 250 sets of four model configurations were compared, resulting in high statistical power of the experiment; the results of the corresponding Nemenyi hypothesis tests are shown in Figure 3.

However, we can also perform these hypothesis tests for every dataset, using only 10 paired measurements for every model configuration. The results of these dataset-specific hypothesis tests (specifically, whether the fully optimized `catboost` model gives significantly different predictions to the baseline) are shown together with the results on the test set in Table 9.

4.4 Test set evaluation

Finally, the four model configurations are evaluated on the test set, after training the models on the entire available training/validation data. The results are shown in Table 9. To visualize the impact of the entire optimization process, we visualize the test set NRMSE and AUPRC performances in Figures 4 and 5 specifically for the baseline and fully optimized (optimized features + hyperparameter optimization) models. The same figures with Pearson R, ROCAUC values are available in Figures A9 and A10, respectively.

For 19 out of 25 datasets, the optimizations have led to an improvement over the baseline model. This number jumps up to 21 out of 25 if either the non-optimized or the optimized `catboost` model with optimized features is considered, as opposed to the models using standard `rdkit_desc` or `rdkit_desc` + `ecfp4` feature combinations. For 12 datasets, hyperparameter optimization leads to a decrease in test set performance compared to the non-optimized model with optimized features.

Only for 11 datasets the optimizations were statistically significant according to the Nemenyi test using a 10-fold CV method. In all 11 cases, an improvement on the test set performance was observed as well.

4.5 Transferability

The selected model configurations were subsequently evaluated in a more practical scenario, compared to the manually created test set splits. Multiple public datasets for the same property, but from different sources, were found for hPPB, HLM, solubility. Respectively, 9, 11 and 20 overlapping compounds were found in the dataset pairs for hPPB, HLM and solubility. After adjusting the values to reflect similar units, Pearson correlations of 0.98, 0.92, and 0.89 were obtained for the respective properties of the overlapping compounds measured in different laboratories (Fig. 6). Given the high correlation, we assessed the influence of using data from one laboratory on the

Table 9: Test set evaluation of model configurations at four different degrees of optimization. In the 10-fold CV experiment, each configuration was trained 10 times for each dataset, allowing us to perform Nemenyi hypothesis test to compare the model configurations. In the penultimate column, the outcome of the hypothesis ($p < 0.05$ or not) is shown, specifically comparing the fully optimized (features and hyperparameters) model to the baseline model. In the last column, we also denote whether the fully optimized configuration outperformed the baseline on the held-out test set.

Dataset	`catboost default, rdkit_desc`	`catboost default, rdkit_desc + ecfp4`	`catboost default, optimized features`	`catboost optimized, optimized features`	10-fold CV Nemenyi test $p < 0.05$	Test set improvement
TDCommons - regression	**NRMSE** (↓)					
caco2_wang	0.342	0.336	**0.323**	0.335	✓	✓
lipophilicity	0.437	0.428	0.408	**0.400**	✓	✓
ppbr_az	0.847	0.859	0.830	**0.829**	✗	✓
ld50_zhu	0.817	0.811	0.781	**0.752**	✓	✓
vdss_lombardo	0.414	0.412	0.411	**0.410**	✗	✓
half_life_obach	**0.552**	0.561	0.562	0.557	✗	✗
clearance_microsome_az	**0.429**	0.437	0.437	0.453	✗	✗
TDCommons - binary	**AUPRC** (↑)					
bioavailability_ma	0.890	0.882	**0.909**	0.900	✗	✓
hia_hou	**0.994**	0.993	**0.994**	0.991	✗	✗
pgp_broccatelli	0.935	0.937	**0.939**	0.935	✗	✗
bbb_martins	0.976	0.979	**0.981**	0.978	✗	✓
cyp2c9_veith	0.763	0.773	0.778	**0.789**	✓	✓
cyp2d6_veith	0.677	0.694	**0.712**	0.709	✓	✓
cyp3a4_veith	0.854	**0.877**	0.874	0.873	✓	✓
cyp2c9_substrate_carbonmangels	0.418	0.416	**0.448**	0.345	✗	✗
cyp2d6_substrate_carbonmangels	0.688	0.683	**0.719**	0.707	✗	✓
cyp3a4_substrate_carbonmangels	0.680	0.696	0.709	**0.712**	✗	✓
herg	0.938	0.936	**0.957**	0.944	✗	✓
ames	0.890	0.896	0.896	**0.902**	✗	✓
dili	0.867	0.883	0.887	**0.894**	✗	✓
NIH - regression	**NRMSE** (↓)					
solubility	0.469	0.462	0.457	**0.441**	✓	✓
Biogen - regression	**NRMSE** (↓)					
rlm	0.507	0.489	0.485	**0.474**	✓	✓
solubility	0.905	0.911	**0.877**	0.933	✗	✗
hlm	0.425	0.422	0.424	**0.409**	✓	✓
mdr1-mdck	0.452	0.441	0.431	**0.427**	✓	✓

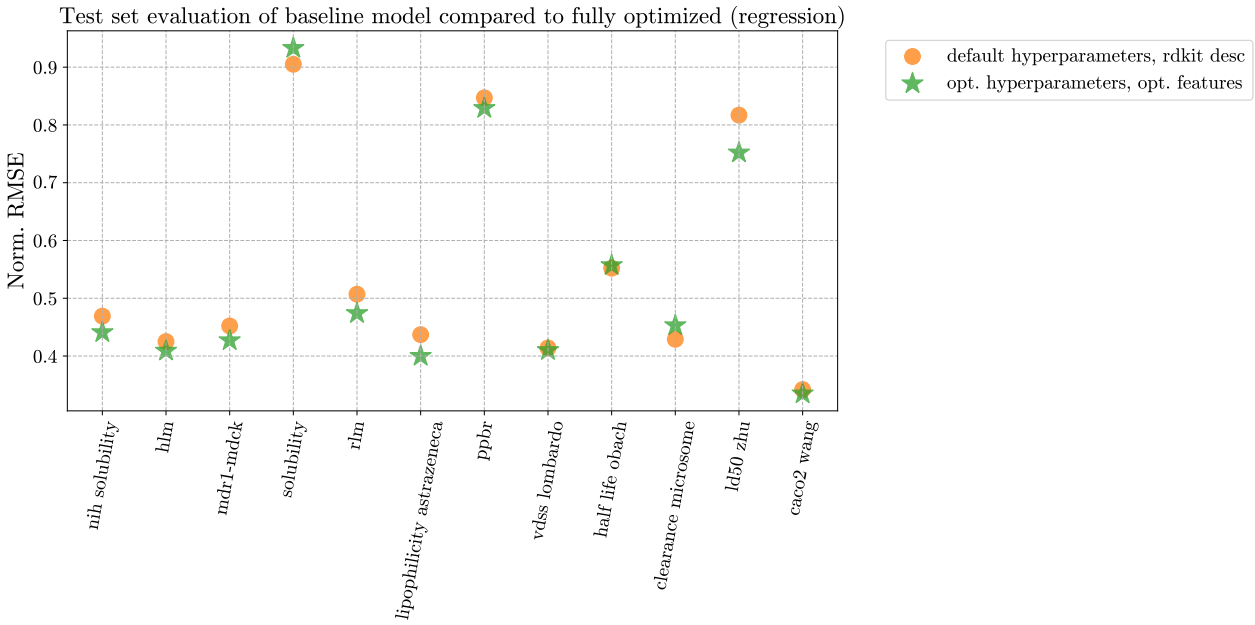

Figure 4: Comparison of baseline (CatBoost default hyperparameters + `rdkit_desc` features) versus fully optimized (CatBoost optimized hyperparameters + optimized features) on the test set of each dataset. Exact values, including all four model configurations, can be seen in Table 9. Pearson R values can be seen in Figure A9.

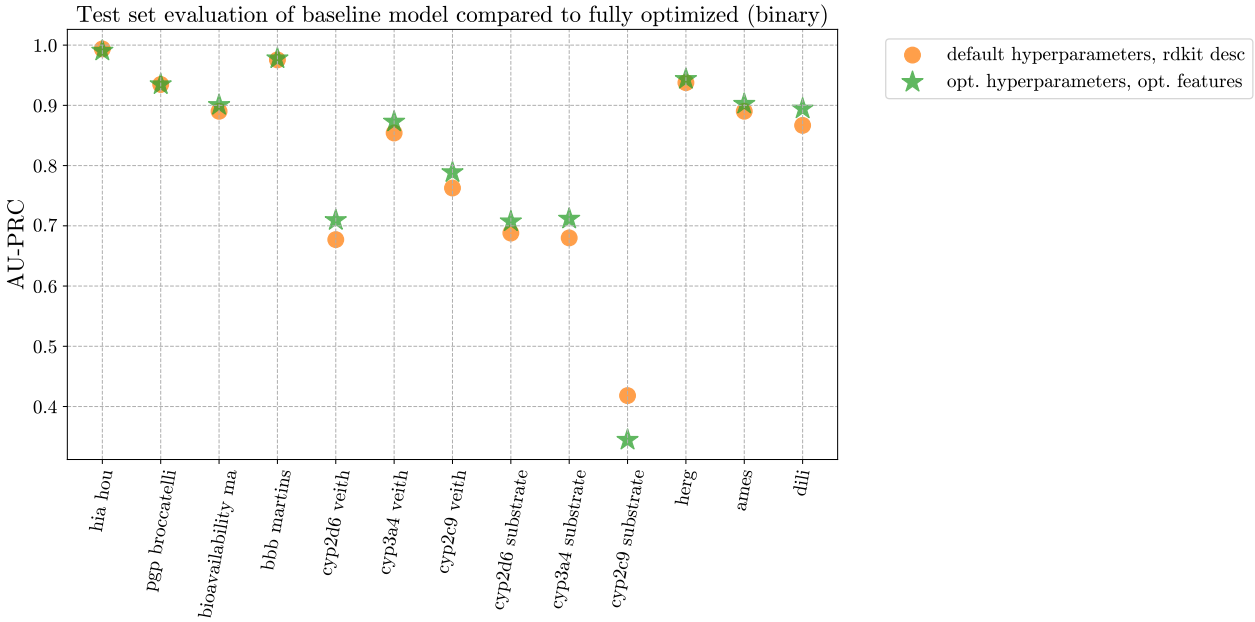

Figure 5: Comparison of baseline (CatBoost default hyperparameters + `rdkit_desc` features) versus fully optimized (CatBoost optimized hyperparameters + optimized features) on the test set of each dataset. Exact values, including all four model configurations, can be seen in Table 9. ROC-AUC values can be seen in Figure A10.

predictive outcome of the properties measured in a different laboratory.

Table 10: Performance of models at various levels of optimization when trained on data from one source (TDC or NIH) and evaluated on data from a different source (Biogen).

Features	Model	hPPB RMSE (R^2)	HLM RMSE (R^2)	Sol. RMSE (R^2)
`rdkit_desc`	`catboost` default	0.345 (0.534)	0.366 (0.053)	0.301 (0.410)
`rdkit` + `ecfp4`	`catboost` default	0.345 (0.534)	0.364 (0.043)	0.291 (0.433)
optimized features[a]	`catboost` default	0.355 (0.523)	0.360 (0.051)	0.277 (0.458)
optimized features[a]	`catboost` optimized params	0.345 (0.531)	0.361 (0.057)	**0.274** (**0.465**)

[a] `rdkit_desc`, `ecfp4`, `erg`, and `avalon` fingerprints

Firstly, the different levels of model optimization, the baseline `catboost` model compared to optimal feature combinations and hyperparameters, had minimal benefit when training on TDC data and testing on Biogen data for hPPB and HLM (Table **??**). In the case of solubility, where models were trained with NIH data and tested on Biogen data, a decrease of RMSE (and increase in model correlation coefficient, R^2) was observed alongside the iteratively optimized features and hyperparameters. For hPPB and solubility, R^2 (percent variance explained) of around 50% (0.5) were achieved (Table **??**). This relates to around 30% percent of standard deviation explained,[3] meaning that the models trained on the TDC and NIH data yields errors that are around 30% smaller on average than those of a constant-only model (e.g. predicting the mean). Looking at the correlation between the predicted and measured values indicates that the hPPB and solubility models tend to over-predict the lower values (Fig. 7). The HLM model offered little advantage over predicting the mean (3% smaller errors on average compared to predicting the mean given the R^2 of 0.06). Since the TDC data for both hPPB and HLM stems from AstraZeneca data, we sought to see whether compound similarity between the training and test sets would provide an explanation for the difference in model performance. However, the Tanimoto similarities (based on `ecfp4`) between test set compounds and that of the most and least similar compounds in the training sets, as well as the average similarity, indicated no significant differences (Fig. A12).

We continued to use the optimized features and hyperparameters in further experiments to assess the influence of including Biogen data in the training sets. Omitting, and then incrementally increasing the amount of Biogen data in the training sets of the models either does not affect, in the case of hPPB, or increases R^2 as the Biogen proportion of training data increases (Fig. 8). The increasing effect is most pronounced in the case of HLM, where a the proportionally higher amount of Biogen data is available and included (up to almost 60% of the training set). In all cases, when only using Biogen data for training, the increase in data increases the R^2. For HLM with its higher proportions of Biogen data, it reaches a point where there is no benefit to including the additional TDC data compared to only using the Biogen data. However, the inclusion of additional data generally reduces the R^2 standard deviation when less Biogen data is included, indicating that the additional data affords better generalizability across different scaffold folds.

5 Discussion

The experiments in this study extensively explore ML models and features utilized in the ADMET space, focusing on statistical significance when comparing between them.

The initial model selection experiment via single-fold evaluation has shown that when using a single feature representation, `catboost`, `svm` and `lightgbm` all perform statistically similarly according to the Nemenyi post-hoc test. However, when two feature representations are used (`rdkit_desc` combined with other features), `catboost` comes out as a superior model architecture, yielding significantly better performance with $p < 0.05$ under the Nemenyi test.

`random forest` as well as `mpnn` model architectures performed poorly in our experiments. The poor performance of the `mpnn` is surprising considering it often is shown to yield good results in the literature [21, 34, 35]. This could be explained by the hands-off approach of our study, in which we did not supervise the training process of either model, simply making use of the training and prediction CLI scripts as instructed. Deep learning models often require some supervision besides the early stopping criteria to ensure that the training has converged properly. Moreover, deep learning models typically require more data compared to decision tree based architectures. Therefore, we investigate the differences in model performance between the baseline `mpnn` and `catboost` models trained with `rdkit_desc` features, with the dataset size in mind. This data

[3] The percent by which the standard deviation of the errors is less than the standard deviation of the dependent variable = 1 - square root(1 - R^2)

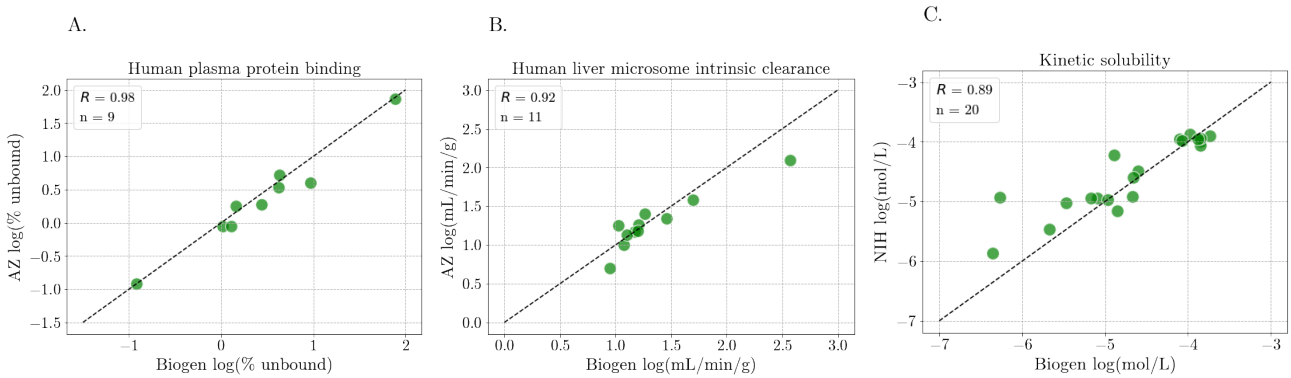

Figure 6: Correlation between the property values for overlapping compounds across the datasets for A) hPPB, B) HLM, and C) solubility.

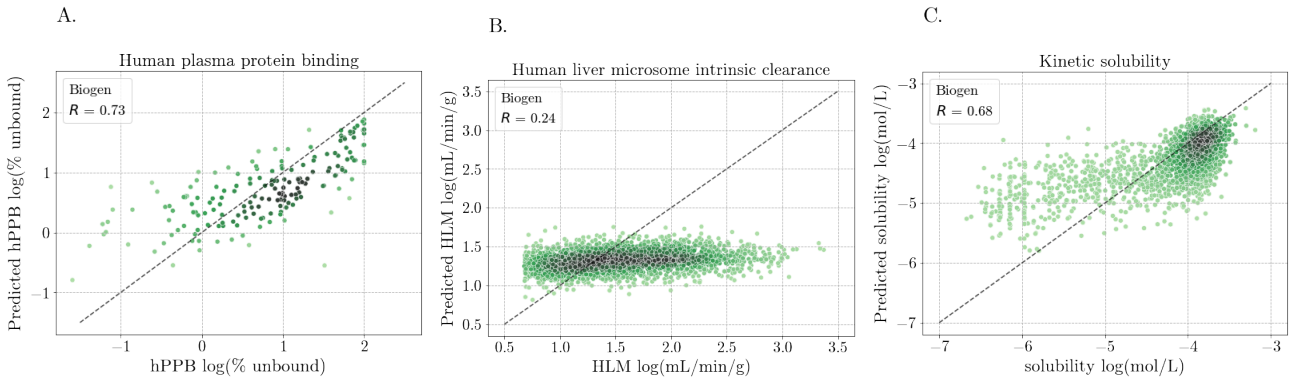

Figure 7: Correlation between the measured and predicted property values for models trained on data from one source and tested data from another across A) hPPB, B) HLM, and C) solubility.

can be seen in Table A11. We observe that as the dataset size increases, the model performances become more and more comparable, with `mpnn` ultimately outperforming baseline `catboost` on the NIH solubility dataset containing ∼36k compounds.

The feature performance investigation has showed that using the `rdkit_desc` representation of compounds is a safe choice for molecular property prediction tasks in the ADMET domain, performing significantly better used as a single feature compared to any other single feature across all models. However, it is a much more reliable representation for regression tasks compared to binary classification, attaining average ranks of 1.91 and 4.31 respectively across pairs of dataset-model combinations for both task types.

Iterative feature addition (Table 6 and 7) has resulted in combinations that yield significantly better performance for both regression as well as binary classification tasks across the datasets. In both cases, using only `rdkit_desc` features, or a common `rdkit_desc` + `ecfp4` feature combination does not bring out the best possible model performance in the datasets. The improvement that comes from using the optimized set of features is more significant than the improvement of subsequent hyperparameter optimization, as seen both in the cross-validation analysis (Table 8) as well as the test set evaluation (Table 9). Moreover, adding extra features results in decreased performance, suggesting that the added noise outweighs the additional representational power.

The optimal sets of features for regression[4] and binary classification[5] tasks contain only standard cheminformatics representations, showing that deep-learning based features (both LLM and graph-based) do not perform well in the ADMET domain. Notably, as a standalone feature, `megamolbart` has achieved an average ranking of 4.46 in the binary classification datasets, compared to the 4.31 of `rdkit_desc`, which is the only time in the study where a deep-learning based feature representation has come close to the standard cheminformatics ones. A possible interpretation for why deep-learning features do not perform well in this domain is the high-noise, low-data issue: the vast information encoded in the high-dimensional compound embeddings of deep-learning models might be too hard to uncover when guided only by the highly noisy and sparse data.

Overall, the model optimization has been impactful

[4] regression: `rdkit_desc` + `erg` + `ecfp4` + `avalon`
[5] binary: `rdkit_desc` + `erg` + `avalon`

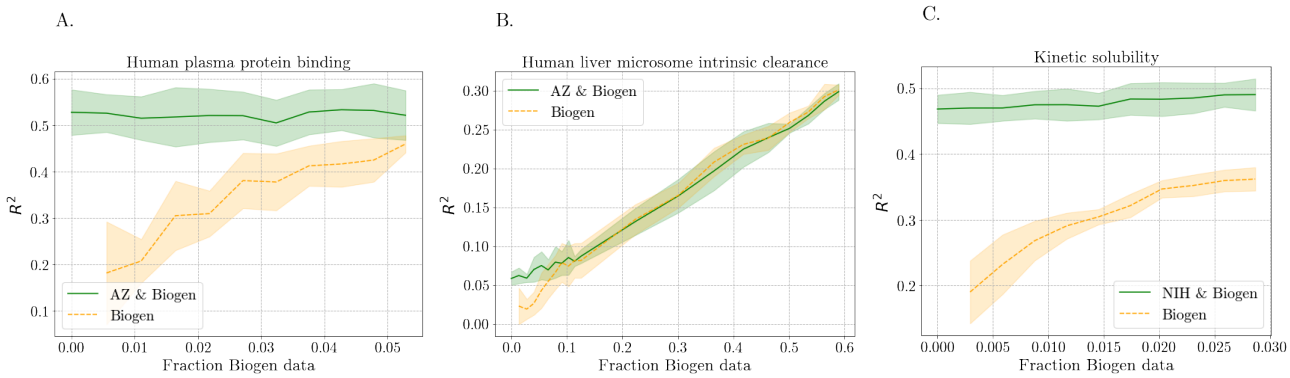

Figure 8: Performance of models trained with increasing amounts of Biogen data by itself, or combined with TDC or NIH data. The standard deviation across 5-folds are indicated by the shading around the means for experiments across A) hPPB, B) HLM, and C) solubility.

in increasing the test set performance. Non-optimized models with either `rdkit_desc` or `rdkit_desc`+`ecfp4` features only performed best on 4 out of 25 test sets, whereas one of the optimized model configurations performed the best on the remaining 21. Interestingly, while the hyperparameter optimization was shown to be statistically significant in the cross-validation experiment, it had mixed results in increasing the performance on the test set, where for 12 datasets it has led to a decrease in performance. However, only 3 of the 12 datasets were regression datasets, suggesting that the optimization is less robust in the binary classification setting.

In every set of comparisons, the Friedman χ^2 test was easily passed with $p < 0.01$; however, the post-hoc Nemenyi test did not always identify significant differences under $p < 0.05$. In particular, passing the 10-fold cross-validation hypothesis test is harder than improving the test set performance; often even when a substantial improvement in the test set is observed, the hypothesis test did not pass with $p < 0.05$. However, for the 11 datasets where the hypothesis test was passed, a test set improvement was observed too. These results suggest that hypothesis testing can be utilized as a more robust alternative to single hold-out set evaluation, allowing to identify model improvements that are much more likely to generalize. A case for when such a robustness can be useful is seen in the results of the transferability experiment.

Out of the three datasets used in the transferability experiment, model optimization yielded a noticeable improvement only in the `solubility` dataset, as seen in Table 10. This aligns with our test set observations (Table 9): for `clearance_microsome_az`, neither the hypothesis test was passed nor a test set improvement was observed, whereas the opposite was true for the NIH solubility dataset. The remaining dataset presents an argument for why the robustness of hypothesis might be preferred to a test set evaluation: for the `ppbr_az` dataset, we observe an improvement in the test set. We might therefore incorrectly expect an improvement in the transferability experiment. However, the hypothesis test was rejected; hence if we were to be guided by the hypothesis test, the lack of improvement is no surprise.

Provided that the assay conditions and endpoints are similar, using data from a different laboratory in predictive models could be beneficial when no data from the laboratory in question, represented by Biogen data in this study, is available (Fig. 7). The performance of the models seem to be property dependent, with similarity to compounds in the training set not providing any indication of whether the model would perform well.

As more Biogen data is included in model training, the benefit of additional TDC or NIH training data decreases (Fig. 8). In part, it appears to depend on the amount of additional data compared to Biogen data. For hPPB and HLM, the use of the additional data starts to become insignificant when around 5% Biogen data is available (Fig. 8). In the case of solubility, a similar trend is apparent. However, as much more NIH than Biogen data is available for solubility, the point at which the additional NIH data becomes insignificant was not reached and can thus not be concluded.

6 Conclusions

This study provides a detailed analysis of compound representations and machine learning techniques in ADMET tasks. The systematic approach taken across baseline model and feature selection, and the incorporation of statistical methods in comparisons, advances the accuracy and reliability of ML applications in the ADMET molecular property space. Furthermore, the systematic approach can reveal the degree of benefit afforded by the combined use of data from different

sources.

References

[1] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *arXiv preprint arXiv:2102.09548*, 2021.

[2] Jim Notwell. Maplight tdc: Source code for maplight's therapeutics data commons (tdc) admet benchmark group submission. `https://github.com/maplightrx/MapLight-TDC`, 2023.

[3] Gemma Turon and Miquel Duran-Frigola. Zairachem: automated ml modelling for chemistry datasets, 11 2022.

[4] Oloren-AI. Oce: The first infinitely composable library for reproducibly implementing sota molecular property prediction/qsar techniques. `https://github.com/Oloren-AI/olorenchemengine`, 2023. MIT License.

[5] Gayane Chilingaryan, Hovhannes Tamoyan, Ani Tevosyan, Nelly Babayan, Lusine Khondkaryan, Karen Hambardzumyan, Zaven Navoyan, Hrant Khachatrian, and Armen Aghajanyan. Bartsmiles: Generative masked language models for molecular representations, 2022.

[6] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence*, 4(12):1256–1264, 2022.

[7] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 33:12559–12571, 2020.

[8] Sabrina Jaeger, Simone Fulle, and Samo Turk. Mol2vec: unsupervised machine learning approach with chemical intuition. *Journal of chemical information and modeling*, 58(1):27–35, 2018.

[9] Cheng Fang, Ye Wang, Richard Grater, Sudarshan Kapadnis, Cheryl Black, Patrick Trapa, and Simone Sciabola. Prospective validation of machine learning algorithms for absorption, distribution, metabolism, and excretion prediction: An industrial perspective. *Journal of Chemical Information and Modeling*, 2023.

[10] Jacob Green, Cecilia Cabrera Diaz, Maximilian AH Jakobs, Andrea Dimitracopoulos, Mark van der Wilk, and Ryan D Greenhalgh. Current methods for drug property prediction in the real world. *arXiv preprint arXiv:2309.17161*, 2023.

[11] Jianyuan Deng, Zhibo Yang, Hehe Wang, Iwao Ojima, Dimitris Samaras, and Fusheng Wang. A systematic study of key elements underlying molecular property prediction. *Nature Communications*, 14(1):6395, 2023.

[12] Rajarshi Guha, Thomas S. Dexheimer, Aimee N. Kestranek, Ajit Jadhav, Andrew M. Chervenak, Michael G. Ford, Anton Simeonov, Gregory P. Roth, and Craig J. Thomas. Exploratory analysis of kinetic solubility measurements of a small molecule library. *Bioorganic Medicinal Chemistry*, 19(13):4127–4134, 2011.

[13] Francis Atkinson. Standardiser: A tool for standardising molecules for use in qsar modelling. `https://github.com/flatkinson/standardiser`, 2014.

[14] Thomas Sander, Joel Freyss, Modest von Korff, and Christian Rufener. Datawarrior: An open-source program for chemistry aware data visualization and analysis. *Journal of Chemical Information and Modeling*, 55(2):460–473, 2015.

[15] Slobodan Rendic. Summary of information on human cyp enzymes: human p450 metabolism data. *Drug Metabolism Reviews*, 34(1-2):83–448, 2002.

[16] Minjun Chen, Vikrant Vijay, Qiang Shi, Zhichao Liu, Hong Fang, and Weida Tong. Fda-approved drug labeling for the study of drug-induced liver injury. *Drug Discovery Today*, 16(15):697–703, 2011.

[17] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[18] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[19] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154, 2017.

[20] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[21] Esther Heid, Kevin P Greenman, Yunsie Chung, Shih-Cheng Li, David E Graff, Florence H Vermeire, Haoyang Wu, William H Green, and Charles J McGill. Chemprop: A machine learning package for chemical property prediction. 2023.

[22] Rdkit: Open-source cheminformatics. `https://www.rdkit.org`.

[23] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

[24] Raymond E. Carhart, Dennis H. Smith, and R. Venkataraghavan. Atom pairs as molecular features in structure-activity studies: definition and applications. *J. Chem. Inf. Comput. Sci.*, 25:64–73, 1985.

[25] Peter Gedeck, Bernhard Rohde, and Christian Bartels. Qsar- how good is it in practice? comparison of descriptor sets on an unbiased cross section of corporate data sets. *Journal of chemical information and modeling*, 46(5):1924–1936, 2006.

[26] Nikolaus Stiefl, Ian A Watson, Knut Baumann, and Andrea Zaliani. Erg: 2d pharmacophore descriptions for scaffold hopping. *Journal of chemical information and modeling*, 46(1):208–220, 2006.

[27] Hirotomo Moriwaki, Yu-Shi Tian, Norihito Kawashita, and Tatsuya Takagi. Mordred: a molecular descriptor calculator. *Journal of Cheminformatics*, 10(4):1758–2946, 2018.

[28] NVIDIA Corporation. Bionemo: A library for biological neural network models. `https://github.com/NVIDIA/BioNeMo`, 2023.

[29] Milton Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[30] P.B. Nemenyi. *Distribution-free Multiple Comparisons*. PhD thesis, Princeton University, 1963.

[31] Frank Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in Statistics: Methodology and Distribution*, pages 196–202. Springer, 1992.

[32] Student. Errors of routine analysis. *Biometrika*, 19(1/2):151–164, 1927.

[33] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(10):281–305, 2012.

[34] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. Chemberta: large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020.

[35] Xu Han, Ming Jia, Yachao Chang, Yaopeng Li, and Shaohua Wu. Directed message passing neural network (d-mpnn) with graph edge attention (gea) for property prediction of biofuel-relevant species. *Energy and AI*, 10:100201, 2022.

7 Appendix

7.1 Test set evaluation over different metrics

In the main body of paper, only NRMSE and AUPRC values are used for the datasets. Here we include the test set evaluations on the regression and binary classification tasks using other two popular metrics, Pearson R and ROC-AUC, respectively.

7.2 MPNN versus CatBoost comparison

A common notion is that deep learning models require more data to be trained compared to decision tree based architectures. Therefore, we investigate the differences in model performance between the baseline `mpnn` and `catboost` models trained with `rdkit_desc` features, with the dataset size in mind. This data can be seen in Table A11. We observe that as the dataset size increases, the model performances become more and more comparable.

7.3 Additional transferability information

The figures below shows the Tanimoto similarity distributions of test set compounds rfom one source to that for training set compounds from a different source as discusses in the transferability experiment. Further, we include the predicted vs. measured correlations for training sets consisting of 50% of the Biogen data, and also the combination of the 50% Biogen data and the TDC (hPPB and HLM) or NIH (solubility) data respectively.

7.4 Data cleaning

The tables below show all the dataset-specific information behind the data cleaning that was carried out.

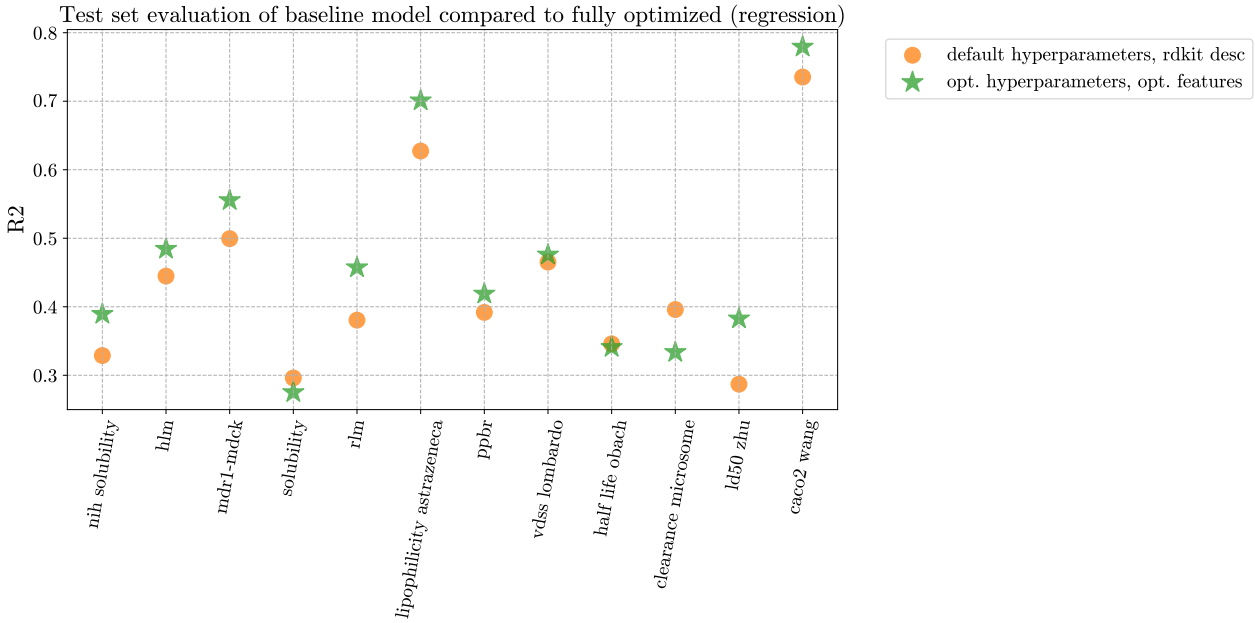

Figure 9: Comparison of baseline (CatBoost default hyperparameters + `rdkit_desc` features) versus fully optimized (CatBoost optimized hyperparameters + optimized features) on the test set of each dataset, evaluated via R2.

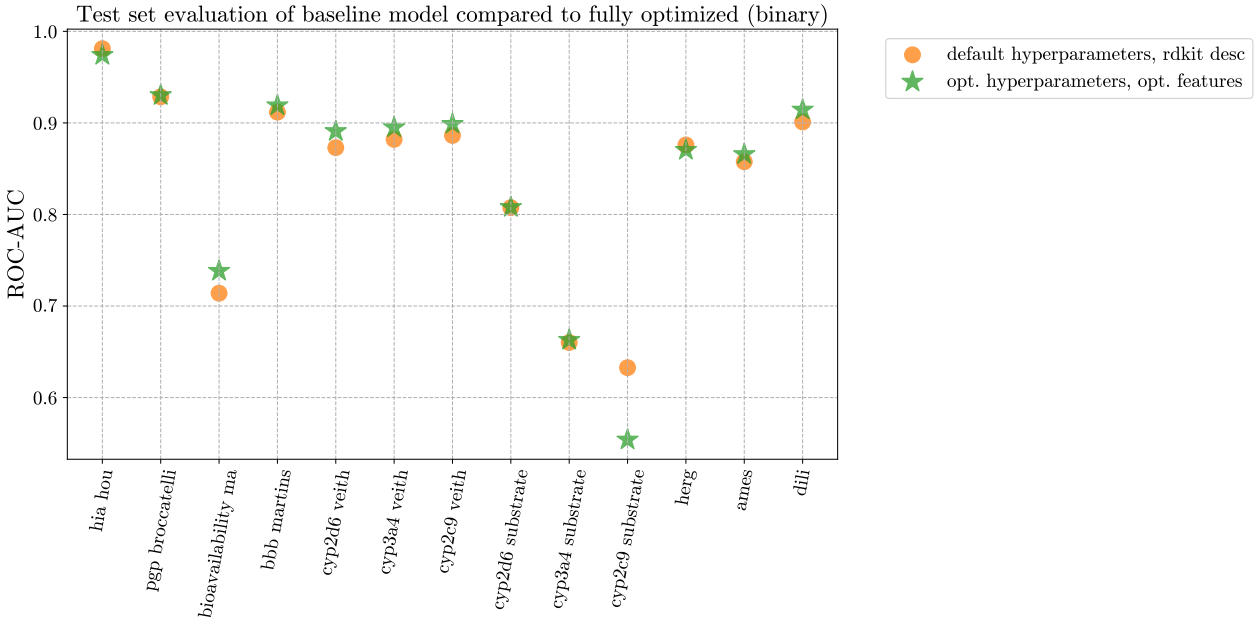

Figure 10: Comparison of baseline (CatBoost default hyperparameters + `rdkit_desc` features) versus fully optimized (CatBoost optimized hyperparameters + optimized features) on the test set of each dataset, evaluated via ROC-AUC.

Table 11: `mpnn` and `catboost` performance comparison on the regression datasets, based on the dataset sizes. The results are based on the single-fold hold-out set experiment (validation, not test).

Dataset Name	Dataset Size	`mpnn` R^2	`catboost` R^2
half_life_obach	662	0.16	**0.25**
caco2_wang	753	0.52	**0.61**
clearance_microsome_az	1102	0.26	**0.32**
vdss_lombardo	1105	0.35	**0.45**
ppbr_az	1614	0.37	**0.44**
solubility	2173	**0.28**	**0.28**
mdr1-mdck	2642	0.43	**0.50**
rlm	3054	**0.47**	**0.49**
hlm	3087	**0.39**	**0.40**
ld50_zhu	7323	0.14	**0.25**
nih_solubility	36238	**0.32**	0.27

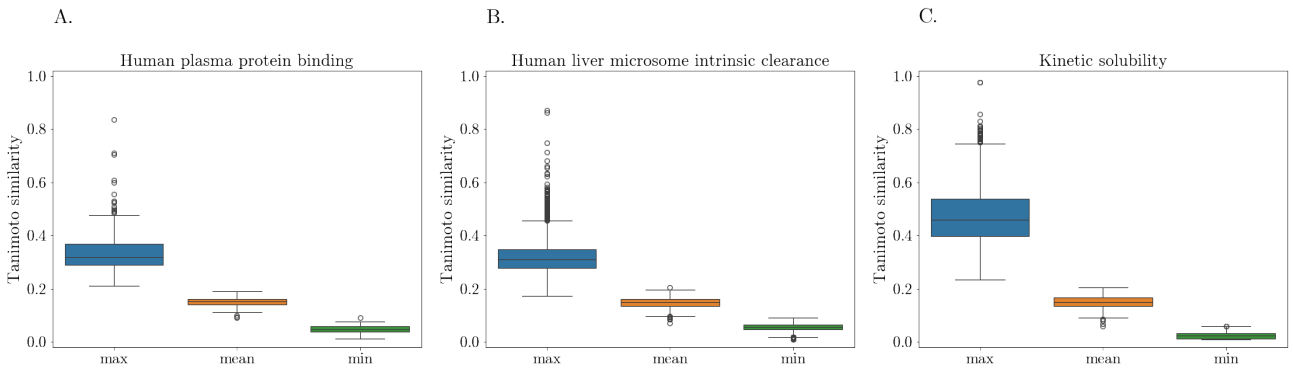

Figure 11: Distribution of Tanimoto similarity between compounds in the test set and those in the training set. For each compound in the test set, only the most similar (max) compound in the training set, the average similarity to all training set compounds (mean), and the least similar compound (min). Training data from one source and test data from another spans the properties A) hPPB, B) HLM, and C) solubility.

Table 12: Data cleaning breakdown for rlm

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	3054	2991	63	0
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	30			
canonicalized	21			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	6			
25 Charge-seperate sulphoxides	5			
04 4-pyrimidone -> 2-pyrimidone (any)	1			

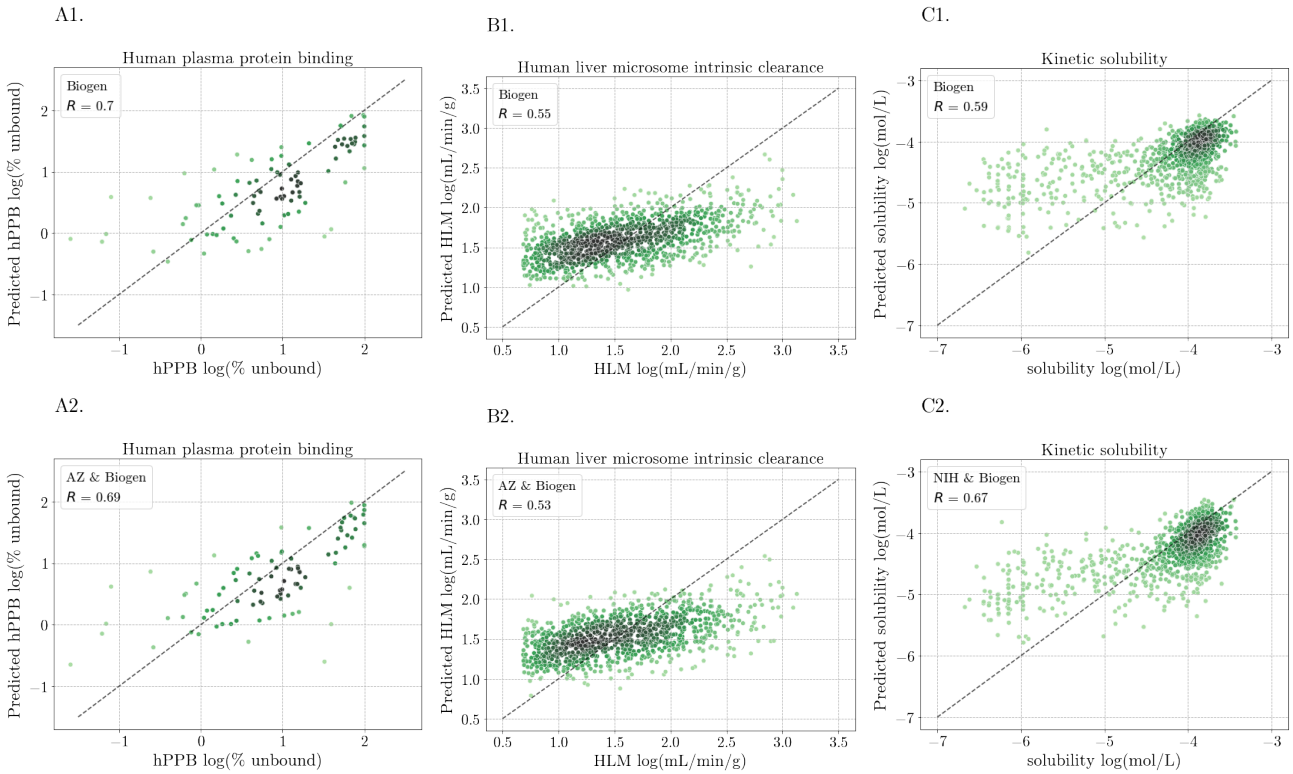

Figure 12: Correlation between predicted and measured values from models trained with 50% Biogen data (1st panels), and the 50% Biogen data combined with the TDC or NIH data (2nd panels) across A) hPPB, B) HLM, and C) solubility.

Table 13: Data cleaning breakdown for solubility

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	2173	2140	33	0
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	18			
canonicalized	7			
25 Charge-seperate sulphoxides	4			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	4			

Table 14: Data cleaning breakdown for mdr1-mdck

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	2642	2587	55	0
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	26			
canonicalized	17			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	7			
25 Charge-seperate sulphoxides	4			
04 4-pyrimidone -> 2-pyrimidone (any)	1			

Table 15: Data cleaning breakdown for hlm

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	3087	3023	64	0
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	30			
canonicalized	21			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	7			
25 Charge-seperate sulphoxides	5			
04 4-pyrimidone -> 2-pyrimidone (any)	1			

Table 16: Data cleaning breakdown for caco2_wang

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	910	587	45	278
Removal reason	**Count**			
inconsistent_duplicate	256			
duplicate	21			
multi_component	1			
Transformation description	**Count**			
canonicalized	30			
02 2-hydroxy pyridine -> 2-pyridone	4			
25 Charge-seperate sulphoxides	4			
13 Enol -> Ketone 1	3			
04 4-pyrimidone -> 2-pyrimidone (any)	3			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	1			

Table 17: Data cleaning breakdown for ld50_zhu

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	7385	7221	87	77
Removal reason	**Count**			
inconsistent_duplicate	66			
duplicate	8			
no_non_salt_or_inorganic	3			
Transformation description	**Count**			
25 Charge-seperate sulphoxides	48			
13 Enol -> Ketone 1	22			
04 4-pyrimidone -> 2-pyrimidone (any)	8			
canonicalized	3			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	3			
10 Fix heterocyclic tautomer 2	2			
02 2-hydroxy pyridine -> 2-pyridone	1			

Table 18: Data cleaning breakdown for dili

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	475	442	24	9
Removal reason	**Count**			
multi_component	6			
multi_organic_salt	2			
no_non_salt_or_inorganic	1			
Transformation description	**Count**			
canonicalized	11			
13 Enol -> Ketone 1	7			
25 Charge-seperate sulphoxides	4			
04 4-pyrimidone -> 2-pyrimidone (any)	2			

Table 19: Data cleaning breakdown for ames

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	7278	7069	151	58
Removal reason	**Count**			
inconsistent_duplicate	38			
duplicate	16			
no_non_salt_or_inorganic	4			
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	42			
canonicalized	35			
13 Enol -> Ketone 1	24			
10 Fix heterocyclic tautomer 2	14			
25 Charge-seperate sulphoxides	12			
11 Fix heterocyclic tautomer 3	11			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	9			
11 Fix heterocyclic tautomer 3 and 02 2-hydroxy pyridine -> 2-pyridone	1			
07 hydropyridin-4-imine -> 4-amino-pyridine	1			
21 Fix 1,3 conjugated cation (non-aromatic)	1			
14 Enol -> Ketone 2	1			

Table 20: Data cleaning breakdown for herg

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	655	333	270	52
Removal reason	**Count**			
duplicate	29			
inconsistent_duplicate	16			
multi_component	6			
no_non_salt_or_inorganic	1			
Transformation description	**Count**			
canonicalized	260			
25 Charge-seperate sulphoxides	5			
04 4-pyrimidone -> 2-pyrimidone (any)	2			
13 Enol -> Ketone 1	2			
02 2-hydroxy pyridine -> 2-pyridone	1			

Table 21: Data cleaning breakdown for nih_solubility

Metric	Total	Unchanged	Transformed	Removed
Dataset count	36240	35775	463	2
Transformation description	**Count**			
04 4-pyrimidone -> 2-pyrimidone (any)	317			
14 Enol -> Ketone 2	73			
13 Enol -> Ketone 1	46			
25 Charge-separate sulphoxides	16			
canonicalized	7			
11 Fix heterocyclic tautomer 3	3			
04 4-pyrimidone -> 2-pyrimidone (any);14 Enol -> Ketone 2	1			

Table 22: Data cleaning breakdown for clearance_microsome_az

Metric	Total	Unchanged	Transformed	Removed
SMILES count	1102	1065	37	0
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	27			
canonicalized	7			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	3			

Table 23: Data cleaning breakdown for half_life_obach

Metric	Total	Unchanged	Transformed	Removed
SMILES count	667	595	68	4
Removal reason	**Count**			
inconsistent_duplicate	2			
multi_component	1			
duplicate	1			
Transformation description	**Count**			
canonicalized	48			
13 Enol -> Ketone 1	12			
02 2-hydroxy pyridine -> 2-pyridone	5			
04 4-pyrimidone -> 2-pyrimidone (any)	2			
06 hydropyridin-2-imine -> 2-amino-pyridine (N-subst.)	1			

Table 24: Data cleaning breakdown for cyp2c9_substrate_carbonmangels

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	669	613	53	3
Removal reason	**Count**			
duplicate	3			
Transformation description	**Count**			
canonicalized	40			
13 Enol -> Ketone 1	8			
04 4-pyrimidone -> 2-pyrimidone (any)	2			
25 Charge-seperate sulphoxides	1			
01 hydroxy imine -> carboxamide	1			
02 2-hydroxy pyridine -> 2-pyridone	1			

Table 25: Data cleaning breakdown for cyp3a4_substrate_carbonmangels

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	670	614	53	3
Removal reason	**Count**			
duplicate	3			
Transformation description	**Count**			
canonicalized	40			
13 Enol -> Ketone 1	8			
04 4-pyrimidone -> 2-pyrimidone (any)	2			
25 Charge-seperate sulphoxides	1			
02 2-hydroxy pyridine -> 2-pyridone	1			
01 hydroxy imine -> carboxamide	1			

Table 26: Data cleaning breakdown for cyp2d6_substrate_carbonmangels

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	667	612	51	4
Removal reason	**Count**			
duplicate	2			
inconsistent_duplicate	2			
Transformation description	**Count**			
canonicalized	38			
13 Enol -> Ketone 1	8			
04 4-pyrimidone -> 2-pyrimidone (any)	2			
25 Charge-seperate sulphoxides	1			
02 2-hydroxy pyridine -> 2-pyridone	1			
01 hydroxy imine -> carboxamide	1			

Table 27: Data cleaning breakdown for cyp2c9_veith

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	12092	10958	969	165
Removal reason	**Count**			
multi_component	76			
no_non_salt_or_inorganic	43			
duplicate	23			
inconsistent_duplicate	19			
manual_inspection	2			
sanity_check	1			
multi_organic_salt	1			
Transformation description	**Count**			
canonicalized	833			
13 Enol -> Ketone 1	97			
25 Charge-seperate sulphoxides	28			
14 Enol -> Ketone 2	5			
07 hydropyridin-4-imine -> 4-amino-pyridine	2			
02 2-hydroxy pyridine -> 2-pyridone	2			
13 Enol -> Ketone 1;14 Enol -> Ketone 2	1			
01 hydroxy imine -> carboxamide	1			

Table 28: Data cleaning breakdown for cyp3a4_veith

Metric	Total	Unchanged	Transformed	Removed
SMILES count	12328	11246	924	158
Removal reason	**Count**			
multi_component	78			
no_non_salt_or_inorganic	42			
inconsistent_duplicate	22			
duplicate	14			
manual_inspection	1			
multi_organic_salt	1			
Transformation description	**Count**			
canonicalized	785			
13 Enol -> Ketone 1	96			
25 Charge-seperate sulphoxides	31			
14 Enol -> Ketone 2	6			
07 hydropyridin-4-imine -> 4-amino-pyridine	3			
01 hydroxy imine -> carboxamide	1			
02 2-hydroxy pyridine -> 2-pyridone	1			
13 Enol -> Ketone 1;14 Enol -> Ketone 2	1			

Table 29: Data cleaning breakdown for cyp2d6_veith

Metric	Total	Unchanged	Transformed	Removed
SMILES count	13130	11962	998	170
Removal reason	**Count**			
multi_component	80			
no_non_salt_or_inorganic	40			
inconsistent_duplicate	30			
duplicate	17			
manual_inspection	1			
sanity_check	1			
multi_organic_salt	1			
Transformation description	**Count**			
canonicalized	830			
13 Enol -> Ketone 1	118			
25 Charge-seperate sulphoxides	39			
14 Enol -> Ketone 2	6			
07 hydropyridin-4-imine -> 4-amino-pyridine	3			
13 Enol -> Ketone 1;14 Enol -> Ketone 2	1			
01 hydroxy imine -> carboxamide	1			

Table 30: Data cleaning breakdown for vdss_lombardo

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	1130	394	711	25
Removal reason	**Count**			
duplicate	15			
inconsistent_duplicate	10			
Transformation description	**Count**			
canonicalized	673			
13 Enol -> Ketone 1	16			
25 Charge-seperate sulphoxides	10			
04 4-pyrimidone -> 2-pyrimidone (any)	7			
06 hydropyridin-2-imine -> 2-amino-pyridine (N-subst.)	2			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	2			
10 Fix heterocyclic tautomer 2	1			

Table 31: Data cleaning breakdown for ppbr_az

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	1614	1572	42	0
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	19			
canonicalized	11			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	10			
01 hydroxy imine -> carboxamide	2			

Table 32: Data cleaning breakdown for bbb_martins

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	2030	1743	202	85
Removal reason	**Count**			
duplicate	44			
inconsistent_duplicate	36			
multi_component	5			
Transformation description	**Count**			
canonicalized	159			
13 Enol -> Ketone 1	28			
04 4-pyrimidone -> 2-pyrimidone (any)	9			
25 Charge-seperate sulphoxides	6			

Table 33: Data cleaning breakdown for bioavailability_ma

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	640	580	58	2
Removal reason	**Count**			
no_non_salt_or_inorganic	1			
multi_component	1			
Transformation description	**Count**			
canonicalized	36			
13 Enol -> Ketone 1	10			
25 Charge-seperate sulphoxides	6			
04 4-pyrimidone -> 2-pyrimidone (any)	3			
02 2-hydroxy pyridine -> 2-pyridone	2			
06 hydropyridin-2-imine -> 2-amino-pyridine (N-subst.)	1			

Table 34: Data cleaning breakdown for pgp_broccatelli

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	1218	1157	55	6
Removal reason	**Count**			
duplicate	6			
Transformation description	**Count**			
canonicalized	44			
13 Enol -> Ketone 1	5			
04 4-pyrimidone -> 2-pyrimidone (any)	3			
25 Charge-seperate sulphoxides	2			
01 hydroxy imine -> carboxamide	1			

Table 35: Data cleaning breakdown for hia_hou

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	578	529	49	0
Transformation description	**Count**			
canonicalized	29			
25 Charge-seperate sulphoxides	8			
13 Enol -> Ketone 1	7			
01 hydroxy imine -> carboxamide	4			
02 2-hydroxy pyridine -> 2-pyridone	1			

Table 36: Data cleaning breakdown for lipophilicity

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	4200	4094	105	1
Removal reason	**Count**			
no_non_salt_or_inorganic	1			
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	63			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	37			
01 hydroxy imine -> carboxamide	2			
25 Charge-seperate sulphoxides	1			
canonicalized	1			
04 4-pyrimidone -> 2-pyrimidone (any)	1			