Advancing ML in ADME(T) prediction: A Focus on Hypothesis Testing and Best Practices

Gintautas Kamuntavičius[1], Orestis Bastas[1], Tanya Paquet[1] and Roy Tal[1]

[1]AI Chemistry, Ro5, 2801 Gateway Drive, Irving, 75063, TX, USA

Abstract

This study, focusing on predicting Absorption, Distribution, Metabolism, Excretion, and Toxicology (ADME(T)) properties, adresses 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. This approach aims to bolster the reliability of ADME(T) predictions, providing more dependable and informative model evaluations.

1 Introduction

Absorption, Distribution, Metabolism, Excretion, and Toxicology (ADME(T)) are a set of processes that are commonly assessed in drug discovery projects as the feasibility of a drug can highly depend on them. A lot of recent and not recent work has gone into building and evaluating ML systems designed to predict molecular properties that are associated to ADME(T).

In particular, there is a variety of models on the TDCommons ADMET leaderboard [1], with a wide variety of models, features, as well as data processing methods ([2], —- make more elaborate examples and citations).

In these works, lot of attention goes into comparing the different ML models and architectures (examples and citations), whereas the selection of compound representations is often not justified or analyzed in a very limited scope. For instance, many approaches use a number of concatenated compound representations that they use all at once to try out various models, sometimes paired with feature selection and feature engineering techniques . While the feature selection justification is lacking, these approaches often yield very good results [3], [4], [5]. In our work, we aim to improve the understanding of the impact of feature concatenation, taking a step further to provide a process that can select dataset-specific, statistically significant compound representation choices.

The benchmark datasets in the ADMET are often criticised with regards to the data cleanliness. Issues range from inconsistent SMILES representations, multiple organic compounds found in a single fragmented SMILES string, to duplicate measurements, and even inconsistent binary labels across duplicate smiles found in both train and test set. In our work we begin by going through a data cleaning procedure, designed to deal with the mentioned problems. This results in removal of a small number of compounds from most datasets[1], as well as some modifications to problematic SMILES representations.

A lot of recent literature focuses on deep-learned compound representations [6], [7], [8], [9]. We investigate how do the DNN compound representations compare to the classical ones in the ADMET prediction domain.

In this paper, we describe three experiments, associated to the following research questions:

1. Which types of algorithms and compound representations are generally suitable for ligand-based machine learning in the ADME(T) domain?
2. Can dataset-specific algorithm and representation choices be made reliably using cross-validation in conjunction with hypothesis testing?

2 Related work

Fang et al. [10] have assessed the efficacy of many popular ML models on their internal assays. While they look at a variety of models, the compounds representations are more limited, exploring only combations of RDKit descriptors and ECFP4 fingerprints. While their experiments were done in a sequential manner and evaluations were carried out on temporal splits, they have also shared a set of ADMET assay results on 3k purchasable compounds. This dataset has been invaluable in our study, allowing to assess the efficacy of public data on internal assay prediction, as well as consider data quality comparisons.

Most recently, Green et al. [11] carried out a study focusing on a wide range of models and features on ADMET and QSAR (Bioactivity) tasks. They

[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.

propose a principled implementation of uncertainty estimation (estimates for both aleatoric and epistemic uncertainty) as well as calibration. The authors only consider regression datasets, allowing for modelling of each dataset as a Gaussian distribution.

3 Methods

3.1 Datasets

Therapeutic Data Commons The Therapeutic Data Commons (TDC) is an initiative that aims to benchmark AI capability across different stages of drug discovery [1]. It provides various resources, including public datasets, via an open Python library []. For this study, we focus on the TDC datasets pertaining to the absorption, distribution, metabolism, excretion, and toxicity (ADME-T) properties of small molecules as laid out in table 1. TDC provides two methods to access the data and its splits, namely `tdc.single_pred` and `tdc.benchmark_group`. The data obtained using either method is mostly consistent for scaffold splits. However, the `ppbr_az` dataset presents an anomaly where the `benchmark_group` method loads a deprecated version, merging data from different species that results in over 600 duplicate compounds with differing values. As such, we used the `single_pred` method to obtain the human-only `ppbr_az` data. For all other datasets, the recommended `benchmark_group` method and scaffold splits were used.

MoleculeNet MoleculeNet is another benchmark for molecular property machine learning []. It offers multiple curated public datasets available through their web page [] or GitHub repository []. The three physico-chemical property datasets where included in this study (Tab. 1). The scaffold split method within the DeepChem library was used to split the datasets.

Biogen Recently, Biogen published in-house property measurements across diverse sets of commercial compounds []. The datasets (Tab. 1) were obtained from their GitHub repository [], and split using the DeepChem library's scaffold split method.

3.2 Data cleaning

Assumptions:

- In vitro/in vivo assays tested in a medium in which the compound/entity is soluble. Lipophilicity also relies on a compounds solution. The effect observed is thus attributed to the parent compound.
- Above not the case for solubility (also for FreeSolv - hydration free energy). Different salts of the same compound have different solubility. Take the stance that we are focusing on inherent compound optimisation, thus interested in the inherent solubility of the parent organic compounds themselves, oppose to salts thereof.
- Thus, we take different data cleaning approaches for in vitro assay properties and solubility.

Definitions:

- To deal with salts, a pre-defined list of 142 salts that includes inorganic and organic components, such as phosphate and citrate, was used. A truncated salt list was also 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. 36 components were excluded. In addition, positive and negative hydrogen ions were added to the salt lists as they were present in the SMILES contained in some datasets, e.g. as `[H+].[Cl-]`.
- Organic compounds were defined as compounds that only consists of the 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}.

Solubility (incl. hydration free energy) datasets: Remove all salts. Check that remaining compounds are organic compounds. Adjust tautomers and canonicalize SMILES in order to have consistent compound and functional group representation.

In vitro/in vivo assays (incl. lipophilicity) datasets: Remove inorganic salts and organometallic compounds. Extract parent organic entities form their salt forms. Adjust tautomers and canonicalize SMILES in order to have consistent compound and functional group representation.

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 (using DataWarrior) to check all is sound.

3.3 Features

RDKit Descriptors RDKit descriptors are a set of molecular descriptors provided by the RDKit cheminformatics toolkit [12]. They comprise 208 features that include physicochemical properties and functional group information.

Table 1: Dataset descriptions.

Dataset Name (Table)	Property	Size	Units	Property value distribution
TDCommons - regression				IQR
caco2_wang (A10)	Permeability	906	log(Papp)	
ppbr_az (A24)	Human plasma protein binding	1,797	% bound	
ld50_zhu (A11)	50% lethal dose	7,385	$\log(\text{kg}\cdot\text{mol}^{-1})$	
vdss_lombardo (A23)	Steady-state volume of distribution	1,130	$\text{L}\cdot\text{kg}^{-1}$	
half_life_obach (A16)	—	667	hr	
clearance_microsome _az (A15)	Human liver microsome intrinsic clearance	1,102	$\text{mL}\cdot\text{min}^{-1}g^{-1}$	
TDCommons - binary				positive class size
bioavailability_ma (A26)	—	640	%	
hia_hou (A28)	Human intestinal absorption	578	%	
pgp_broccatelli (A27)	P-glycoprotein inhibition	1,212	%	
bbb_martins (A25)	Blood-brain barrier penetration	1,975	%	
cyp2c9_veith (A20)	CYP2C9 inhibition	12,092	%	
cyp2d6_veith (A22)	CYP2D6 inhibition	13,130	%	
cyp3a4_veith (A21)	CYP3A4 inhibition	12,328	%	
cyp2c9_substrate _carbonmangels (A17)	CYP2C9 metabolism	666	%	
cyp2d6_substrate _carbonmangels (A19)	CYP2D6 metabolism	664	%	
cyp3a4_substrate _carbonmangels (A18)	CYP3A4 metabolism	667	%	
hERG (A14)	hERG inhibition	648	%	
AMES (A13)	Carcinogenicity	7,255	%	
DILI (A12)	Drug-induced liver injury	475	%	
MoleculeNet				IQR
lipophilicity (A30)	Octanol/water distribution	4,200	logD	
FreeSolv (A29)	Hydration free energy	642	$\text{kcal}\cdot\text{mol}^{-1}$	
ESOL (A31)	Equilibrium (thermodynamic) aqueous solubility	1,128	$\log(\text{mol}\cdot\text{L}^{-1})$	
Biogen				IQR
rPPB (A**??**)	Rat plasma protein binding	168	log(% unbound)	
hPPB (A**??**)	Human plasma protein binding	194	log(% unbound)	
RLM (A6)	Rat liver microsomes intrinsic clearance	3,054	$\log(\text{mL}\cdot\text{min}^{-1}\cdot\text{kg}^{-1})$	
Solubility (A7)	Kinetic aqueous solubility	2,173	$\log(\mu\text{g}\cdot\text{mL})$ (pH 6.8)	
HLM (A9)	Human liver microsome intrinsic clearance	618	$\log(\text{mL}\cdot\text{min}^{-1}\cdot\text{kg}^{-1})$	
MDR1-MDCK (A8)	P-glycoprotein efflux ratio	2,642	log(B-A/A-B)	

Morgan Fingerprints Morgan fingerprints, also known as Extended Connectivity Fingerprint (ECFP) or circular fingerprints, encode molecular structure information by considering the substructures within a certain radius around each atom in a molecule. The resulting binary fingerprints represent the presence or absence of specific substructures. The algorithm takes into account connectivity (element, number of heavy neighbors, number of hydrogens, charge, isotope, atom in ring) and chemical (Donor, Acceptor, Aromatic, Halogen, Basic, Acidic) features [13]. Morgan fingerprints with a radius of 2 (ECFP4) were implemented using the RDKit cheminformatics toolkit [12].

Atom Pair Fingerprints The fingerprinting approach followed by Atom Pair fingerprints captures pairs of atoms and their interatomic distances in a molecule. The resulting binary fingerprints encode the presence or absence of specific atom pairs, providing a compact representation of molecular structure. The algorithm takes into account each element, number of heavy neighbors, number of pi electrons for each atom, and the topological distance between each atom pair [14]. Atom Pair fingerprints were implemented using the RDKit cheminformatics toolkit [12].

Avalon Fingerprints Avalon fingerprints are also based on atom pairs, but incorporate additional features such as atom environments and pharmacophore-like patterns. [15]. LINK

ErG descriptors Extended reduced graph (ErG) descriptors represent a 2D pharmacophore description method. This representation captures essential molecular features related to pharmacophores, emphasizing two-dimensional aspects of chemical structures, as well as the size and shape of the molecule. The ErG method is tailored to facilitate scaffold hopping [16].

Mol2vec Inspired by word embeddings in natural language processing, Mol2vec uses deep learning techniques to convert molecular structures into fixed-length numerical vectors. These embeddings encode molecular information in such a way that similar compounds have vectors that are closer in the embedding space [9].

MolFormer MolFormer is a transformer-based large chemical language model that generates compressed representations of molecules from SMILES strings. It was trained on around 100 million compounds from the PubChem and ZINC datasets. MoLFormer encodes molecular structures into sequences of tokens similar to natural language processing models [7]. Note we used the open-sourced model that is trained on 10% of the data rather than the full model.

BARTSMILES BARTSMILES is an extension of bidirectional and Auto-Regressive Transformers (BART) applied to chemical structures. The transformer-based model generates chemical feature representations from SMILES strings, focusing on bidirectional contextual embeddings. It was trained on 1.7 billion compounds from the ZINC20 dataset [6] The embeddings for each SMILES molecule were extracted, 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.

GROVER Graph Representation frOm self-supervised mEssage passing tRansformer (GROVER) is a large-scale graph neural network (GNN) that generates chemical feature representations from SMILES strings. It captures the structural and spatial relationships between atoms and bonds and encodes this information into numerical vectors. GROVER was trained on 11 million compounds from the ZINC15 and ChEMBL datasets [8].

3.4 Models

Random Forest Random Forest (RF) is an ensemble machine learning algorithm that operates by constructing a multitude of decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees. The final prediction is then determined by aggregating the predictions of all the trees, providing a more accurate and stable result compared to individual trees [17].

Support Vector Machine Support Vector Machine (SVM) works by finding the hyperplane that maximally separates data points of different classes while maintaining the largest margin. This margin represents the distance between the hyperplane and the nearest data points, ensuring robust generalization to unseen data [18].

LightGBM LightGBM is a gradient boosting framework that employs tree-based learning for efficient and scalable machine learning. It utilizes a histogram-based approach for tree building, which significantly reduces computation time by bucketing continuous feature values [19].

CatBoost CatBoost is a gradient boosting algorithm and is able to automatically handle categorical variables without the need for manual encoding. CatBoost employs a combination of ordered and categorical boosting, enabling it to naturally process categorical features during training. It performs target encoding for categorical variables, where the categorical values are replaced with the average of the target

variable for each category. This approach significantly simplifies the preprocessing steps, making it advantageous for datasets with a mix of categorical and numerical features. The algorithm also incorporates a symmetric tree structure and utilizes a robust ordered boosting scheme to enhance overall predictive performance [20].

MPNN Message Passing Neural Networks (MPNNs), as implemented by Chemprop, are tailored for chemical property prediction. Developed for molecular graph-structured data, MPNNs operate by passing and updating information between atoms, through a series of message passing steps. Messages are passed from the atom to it's neighbors through their connecting bonds multiple times, ensuring information from one atom can reach several steps further than it's immediate neighbors [21].

3.5 Hyperparameter optimization

With such a large scale of data, hyperparameter search can quickly become very costly, especially in conjunction with cross-fold validation. In order to keep the computational resources reasonable, we use a two-step process starting with a single-fold evaluation in order to identify a small, but dataset-specific hyperparameter space to explore.

We chose to investigate feature combinations over more detailed model hyperparameter choices. The nature of model hyperparameters varies very highly depending on the model, and the hyperparameter spaces are generally quite large; it has also been shown to yield limited (although non-zero) improvements [10]. On the other hand, creating a compound representation by combining multiple features is a common practice in the field, but it rarely comes with rigorous justification. We believe this work can help illuminate the efficacy of feature combinations as well as provide a starting-off point for specific datasets regarding the best features to work with.

The strategy we chose for selecting the dataset-specific feature combinations to explore is detailed in Sec 4.1.5.

3.6 Evaluation

Metrics For regression datasets, we use the normalized RMSE metric, following previous work [11]. It is defined as the RMSE divided by the inter-quartile range of the training set, yielding a more intuitive number for a layman with no familiarity with the underlying data distribution. For binary classification datasets, we use the AU-PRC rather than ROC-AUC due to presence of a few imbalanced datasets. While in this study we have stuck to just these two metrics, it is important to note that as long as non-parametric evaluations are done, using different metrics for different datasets is not a problem in theory.

Friedman χ^2 test The Friedman χ^2 test [22] is a non-parametric statistical test used to detect differences in treatments across multiple test attempts or blocks. It's particularly useful when the data violates the assumptions of normality and homogeneity of variances required for repeated measures ANOVA [23]. The test compares the ranks of the treatments rather than their actual values, making it appropriate for ordinal or 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 140 model and dataset pairs that use `rdkit_desc` features in the single-fold experiment 4.1), k is the number of methods (10 total features in the `rdkit_desc` case), R_j is the total sum of the ranks of the method in question.

Nemenyi post-hoc test The Nemenyi post-hoc test [24] is a non-parametric statistical technique used to compare multiple groups after significant results are obtained from the Friedman χ^2 test. The test calculates the pairwise differences between the average ranks of each group or treatment. The significance of these pairwise rank differences is assessed by referencing the calculated test statistics against the studentized range distribution, which results in p-values for each pair of groups. These p-values represent the probability of observing such rank differences if the null hypothesis (that there is no difference between the groups) is true.

We have chosen the Nemenyi test due to its robustness to the multiple comparisons problem compared to other methods, such as the Wilcoxon rank sum test [25]. This is because the number of comparisons k is taken into account when computing the test statistic Q_{ij} (Eq. 2), which follows the studentized range distribution [26].

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

3.7 Log-transformation

Some datasets have very non-standard distributions that make them very hard to train ML models on. A typical approach to mitigate this issue is the log transformation. We follow [11], [3] and others by log-transforming four of the TDCommons ADME(T) datasets, in particular , `clearance_microsome_az`,

half_life_obach and vdss_lombardo. The metrics shown in this work are computed on the log transformed values instead of the original ones.

4 Experiments

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

1. **Single-fold experiment**. A single hold-out validation set is split using the available (non-test) data, using scaffold splits implemented in DeepChem [27], for every dataset. All combinations of the 5 models, 10 features and 26 datasets are trained.These results allow for very thorough pair-wise comparison of model and feature performance in the context of ADMET prediction tasks: every feature is compared 26 x 5 = 130 times, and every model is compared 26 x 10 = 260 times. This number of pairwise comparisons builds confidence in the results of corresponding pairwise hypothesis tests. The results of the single-fold experiment are used to decide on a general ADMET model and feature combination to use as a general ADMET prediction baseline, as well as define a set of hyperparameters (we investigate feature combinations in particular) to explore in the subsequent cross-validation experiment.

2. **Cross-validation experiment**. Once a baseline model choice as well as dataset-specific features of interest are established, a feature combination experiment via 10-fold cross-validation is carried out. The methodology for combining features is outlined in Sec. 4.1.5. This results in 6 pairwise measurements comparing every feature combination to the baseline. For each dataset we perform the Friedman χ^2 test, and if the null hypothesis is rejected, we follow through with the post-hoc Nemenyi test in order to identify which feature(s) outperform the general ADMET baseline model. We refer to the resulting dataset-specific set of features that outperforms a baseline as the "optimized features".

3. **Test set evaluation**. The general ADMET baseline model, as well as the dataset-specific models that have significantly outperformed the baseline, are evaluated on the test set, unseen up until this experiment. If the feature combinations were not found to statistically significantly improve upon the general ADMET baseline model, only the baseline is evaluated on the test set and no comparison is made.

4.1 Single fold experiment

Performing the Friedman χ^2 test on the model and feature comparisons (260, 130 comparisons respectively for each pair) yields p-values on the order of 10^{-126}, 10^{-168} respectively, which allows us to proceed with the post-hoc tests.

Pairwise post-hoc tests for combined binary and regression tasks The non-parametric nature of the tests allows us to use the information from both the binary and regression tasks in the tests, as only the relative rankings are important. Therefore, we combine the rankings of the normalised RMSE values for regression tasks with the rankings of the AU-PRC values for binary classification tasks in order to perform the post-hoc tests.

Feature comparison Figure 1 together with Table 2 show that rdkit_desc are the generally best ranking features across the ADMET datasets, statistically significantly outperforming even the runner-up grover and mol2vec features under Nemenyi test with $p < 0.05$. The mordred descriptors are assessed by the Nemenyi test to be not significantly different than rdkit desc, but they do have a higher average rank even though the mordred descriptors contain rdkit desc features. Moreover, rdkit_desc performance is much more significant in regression tasks (average rank of 2.37) compared to binary classification (average rank of 4.29), even though they still have the lowest rank among both domains. As a result, if we were to post-hoc test the features independently for regression and binary classification tasks, rdkit_desc would not lead to statistically significant improvements. Analogous p-value plots can be found in Sec. A7.1.

Table 2: Average feature rankings in the single-fold experiment. Statistically

Feature	Regression	Binary	Combined
molformer	5.79	5.31	5.54
bartsmiles	6.92	5.85	6.68
grover	3.92	5.20	4.49
mol2vec	4.32	4.75	4.49
atom pair	6.12	6.05	6.06
ecfp4	7.35	5.34	6.39
rdkit desc	**2.37**	4.29	**3.24**
erg	6.72	7.72	7.16
avalon	7.25	5.52	6.42
mordred	4.24	4.97	4.55

Model comparison Figure 2 together with Table 3 show that SVM, together with CatBoost, stand in

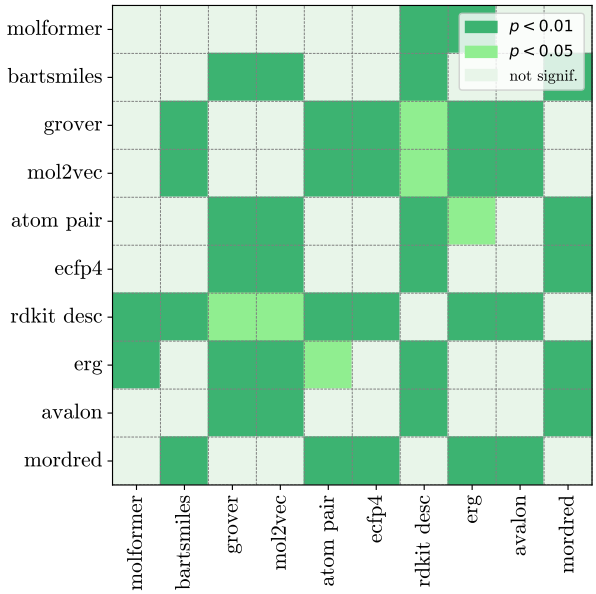

Figure 1: Pair-wise feature performance comparisons across all the dataset and model pairs. Low p-value indicates that there is a significant difference between the performances of a pair of features.

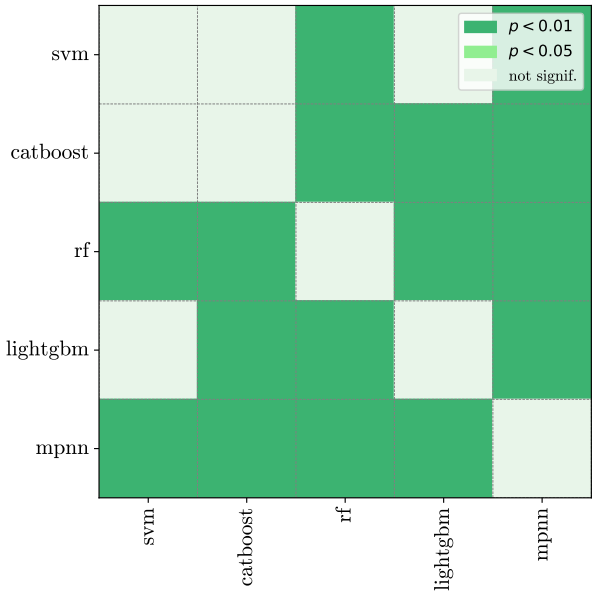

Figure 2: Pair-wise model performance comparisons across all the dataset and feature pairs. Low p-value indicates that there is a significant difference between the performances of a pair of models.

a separate class compared to the other three models, each having an average rank of 2.46, 2.28 respectively compared to the runner-up `LightGBM` at 2.83. The performances of the three models are not significantly different from each other, while being significantly different from the other two with $p < 0.01$.

The outstanding performance of `CatBoost` is aligned with the fact that the top models in most categories come from [3] who employ a CatBoost model combined with a GNN. However, the performance of `SVM` comes as a surprise, as it is often portrayed as a model parallel to the random forest which did not perform nearly as well.

Table 3: Average model rankings in the single-fold experiment. Top two values within each set are shown in bold.

Model	Regression	Binary	Combined
`svm`	**2.37**	**2.61**	**2.46**
`catboost`	**2.37**	**2.21**	**2.28**
`rf`	3.35	3.26	3.29
`lightgbm`	**2.61**	**3.11**	**2.83**
`mpnn`	4.30	3.82	4.13

Baseline model selection We aim to establish a baseline model to compare dataset-specific models to in the subsequent experiment. Based on the observed results in the single-fold experiment, we choose `rdkit_desc` features together with the `CatBoost` model. The feature choice comes from the decisive statistical test, whereas the choice of `CatBoost` versus `SVM` and `LightGBM` is purely based on the average ranking; as the three models have statistically comparable general ADMET performances, we choose a single model dictated by the average rank as the focus of our study is the features rather than the training algorithm.

Dataset-specific feature combinations The feature space explored in the subsequent cross-validation experiment (Sec 4.2) is defined based on the results of feature comparison analysis (Sec 4.1.2) as well as limitations imposed by computational cost[2].

We train `CatBoost` models using concatenated combinations, chosen in the following manner:

- The first feature is chosen as the best performing feature on the single-fold experiment
- Subsequent features are concatenated together one by one, in the order of best performance on the single-fold experiment
- An exception is made for concatenations of `rdkit desc` and `mordred` features. Because `mordred` includes `rdkit desc`, we only choose one of these features in the overall combination, based on which one performed better on the single-fold

[2] If we were to try out all combinations available out of the 10 features, we would have $2^{10} - 1 = 1023$ models to train for every fold.

experiment

- Iteratively larger concatenations of features are tried out until the total number of concatenated features reaches 6.

4.2 Cross validation experiment

For all datasets, Friedman's χ^2 tests had values of less than 0.01, so post-hoc Nemenyi tests were performed for them all. However, only in 6 datasets there were feature combinations that outperformed the baseline with $p < 0.05$. In such cases, an optimal feature combination was chosen based on the feature combination that had the best average rank out of all the combinations that beat the baseline. These results are shown in Table 4.

The general impact of feature combination can be seen in Figure 3. We observe 3, 4 feature combinations to have the best ranking on average across the datasets. Moreover, just using the representation that worked the best on the validation set generally performs worse than the baseline `rdkit desc` features. This can be seen in more detail in Fig A8.

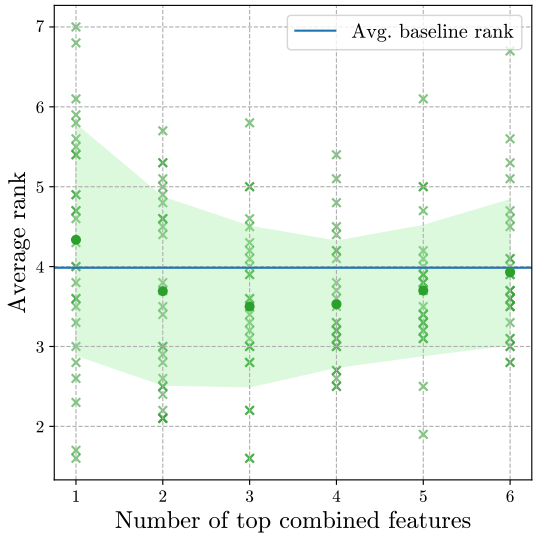

Figure 3: Average rankings across the 10 cross-validation folds are shown for every number of top-k features combined.

4.3 Test set evaluation

The baseline as well as optimized models described in Table 4 were evaluated on the held-out test set. The results are shown in Table 5. We observe a marginal improvement in every one of the six datasets (borderline for lipophilicity).

5 Discussion

The experiments in this study touch upon various aspects of ML application in this domain. The single fold experiment has shown that using the `rdkit desc` representations for ADMET tasks is a safe choice for molecular property prediction tasks in the 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 in the regression tasks compared to binary classification, having average ranks of 2.37 and 4.29 respectively across the N pairs of dataset / model combinations for both task types.

`CatBoost`, `LightGBM` and `SVM` were superior models compared to `mpnn` as well as `random forest`. The poor performance of the `mpnn` is unexpected considering great results found in literature in recent applications. It 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 functions as instructed. Deep learning models generally require some supervision besides the early stopping criteria to ensure that the training has converged.

Deep learned feature representations did not outperform `rdkit desc` by themselves; however, combinations of `rdkit desc` with `grover`, `mol2vec` as well as `molformer` features were successfully used to improve models on certain datasets as shown in Table 4. `grover` has been the most impactful deep-learned representation of the three, improving performance on three different datasets both in the cross validation and on the test set. This shows that while human engineered features explain the most in the molecular property prediction tasks, there is still some useful information in the deep learned representations that can be utilized.

Surprisingly, statistical tests for Friedman χ^2 all passed for every single dataset, even though in some cases upon further testing the Nemenyi test has showed that not a single pair of features performs significantly differently.

The datasets for which better features were established are all relatively large datasets. The smallest dataset out of the six is the `lipophilicity` dataset, with the `cyp` datasets containing 10k+ compounds. One explaining factor is the much larger fold sizes compared to the other datasets, which in turn leads to more homogeneous model performances across the folds and therefore lower p-values[3]. In any case, this

[3] We initially experimented with using 5 folds for each dataset instead of 10. This has the benefit that the test sizes are larger, which could stabilize the performances for the smaller datasets; however, having a sample size of 5 instead of 10 was a bigger factor in increasing the p-values, leading to even fewer

Table 4: Cross-validation experiment results. Best average ranking feature combinations are shown for datasets with statistically significant difference based on the Nemenyi test with $p < 0.05$ across the 10 cross-validation folds. Average ranks are shown out of 7 total ranks (baseline and 6 feature combinations).

Dataset	Optimized features	Avg. baseline rank	Avg. optimized features rank
cyp2c9_veith	`ecfp4` + `grover` + `mol2vec` + `molformer` + `rdkit_desc`	4.9	**2.5**
cyp3a4_veith	`avalon` + `ecfp4` + `mol2vec` + + `mordred`	6.2	**2.7**
cyp2d6_veith	`ecfp4` + `rdkit_desc`	5.8	**2.2**
ames	`avalon` + `grover` + `mordred`	5.3	**1.6**
lipophilicity	`erg` + `grover` + `mordred`	6.2	**2.8**
ld50_zhu	`atom_pair` + `avalon` + `mordred`	6.2	**1.6**

Table 5: Average model rankings in the single-fold experiment. Top two values within each set are shown in bold.

Dataset	Metric	Baseline	Optimized
cyp2c9_veith	AUPRC	0.763	**0.782**
cyp3a4_veith	AUPRC	0.854	**0.878**
cyp2d6_veith	AUPRC	0.677	**0.694**
ames	AUPRC	0.890	**0.908**
lipophilicity	NRMSE	0.426	**0.423**
ld50_zhu	NRMSE	0.817	**0.804**

finding suggests that the combination of small dataset size and noise makes it very hard to identify statistically significant model improvements in most ADMET datasets. A possible way to move forward would be to extend feature search to model hyperparameter exploration in order to seek out even better models, in which case statistically significant differences over the baseline might appear. However, the more models are evaluated, the more problematic it is to take into account the multiple comparisons problem: attempts to take it into account (e.g. the Nemenyi test or the Bonferroni correction) make the p-values larger across the board, whereas not taking it into account runs the risk of incorrectly assigning statistical significance to a model that works better by chance. Defining small, but well reasoned hyperparameter search grids appears to be the safest way to search for better models.

Four out of the six datasets for which improved feature combinations were found are binary. This is in line with the analysis in Sec A7.1 which showed that the baseline `rdkit desc` features work great for regression tasks, but in binary classification tasks the best performing features were very dataset-dependent and none of the features stood out as universally high ranking.

Average rank analysis of the feature combinations in Figure 3 shows that combinations of 3-4 features lead to the best tradeoff between representation expressivity and noise. Interestingly, using only the best performing feature on average leads to a slightly worse rank than just using the `rdkit desc` representation. This hints at mutual information shared between the datasets, as well as validates the approach of selecting a general baseline based on the entire domain: by using `rdkit desc` features that worked well across the board instead of dataset-specific best single feature baseline, we avoided overfitting to the validation set used in the single fold experiment.

statistically significant improvements.

6 Conclusions

The research provides a detailed analysis of compound representations and machine learning techniques in ADME(T) tasks. We verified, with the help of thorough pairwise comparisons and hypothesis testing, that RDKit descriptors are highly effective for molecular property predictions, particularly in regression tasks. Although deep-learned features alone did not surpass RDKit descriptors, their combinations with features like GROVER and Mol2Vec showed improved performance in specific datasets. A contribution of this study is the identification of feature combinations that are better suited for six different datasets, enhancing CatBoost model performance. The systematic approach to baseline selection, feature selection and the incorporation of statistical methods in model comparison marks an advancement in the accuracy and reliability of ML applications in this area.

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] Oscar Méndez-Lucio, Christos Nicolaou, and Berton Earnshaw. Mole: a molecular foundation model for drug discovery. *arXiv preprint arXiv:2211.02657*, 2022.

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

[4] Gemma Turon and Miquel Duran-Frigola. ZairaChem: automated ML modelling for chemistry datasets, November 2022.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] Rdkit: Open-source cheminformatics.

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

[14] 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.

[15] 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.

[16] 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.

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

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

[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] 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.

[23] Ronald A. Fisher. Studies in crop variation i. 1921. First application of the analysis of variance to data analysis.

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

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

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

[27] Bharath Ramsundar, Peter Eastman, Patrick Walters, Vijay Pande, Karl Leswing, and Zhenqin Wu. *Deep Learning for the Life Sciences*. O'Reilly Media, 2019. `https://www.amazon.com/Deep-Learning-Life-Sciences-Microscopy/dp/1492039837`.

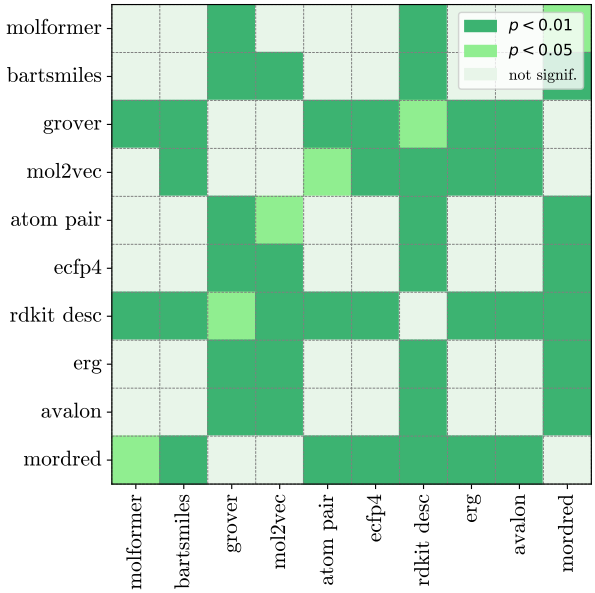

Figure 4: Pairwise post-hoc tests for feature comparison, for **regression** datasets only.

7 Appendix

7.1 Pairwise Nemenyi tests for binary, regression tasks separately

The outcomes of the post-hoc tests are different depending on the task type; one might come to different conclusions regarding model/feature performance for the specific task type compared to viewing and testing the features altogether like it is done in Sec. 4.1. Figures A4, A5, A6, A7 show the task-specific pairwise post-hoc test p-values.

In particular, there is a significant difference between the `rdkit_desc` feature performance between regression and binary datasets. In regression, it is the undoubtedly best performing feature, which is not the case in binary datasets. Even though it has the best average ranking, there is no statistically significant difference between the performance of `rdkit_desc` and most other features.

Moreover, regarding models, in the binary classification tasks CatBoost and SVM stand much more clearly in their own class (as suggested by the average ranking in Table 3).

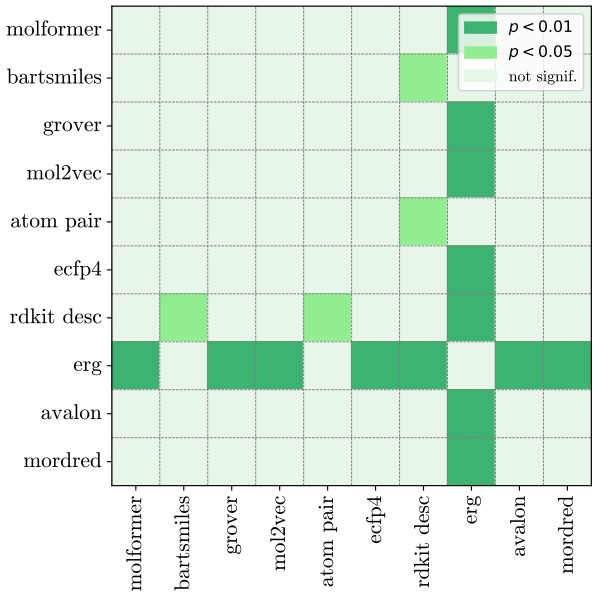

Figure 5: Pairwise post-hoc tests for feature comparison, for **binary classification** datasets only.

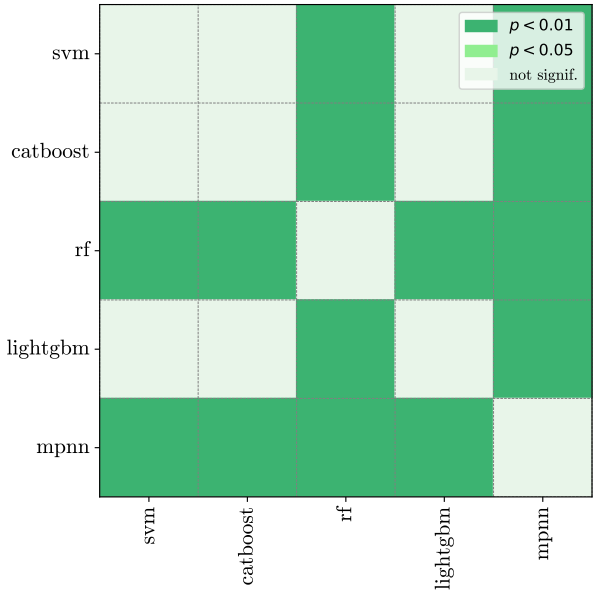

Figure 6: Pairwise post-hoc tests for model comparison, for **regression** datasets only.

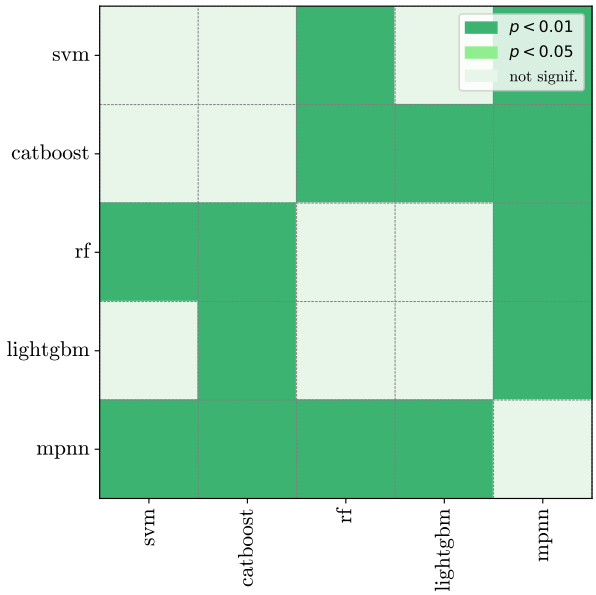

Figure 7: Pairwise post-hoc tests for model comparison, for **binary classification** datasets only.

7.2 Data cleaning

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

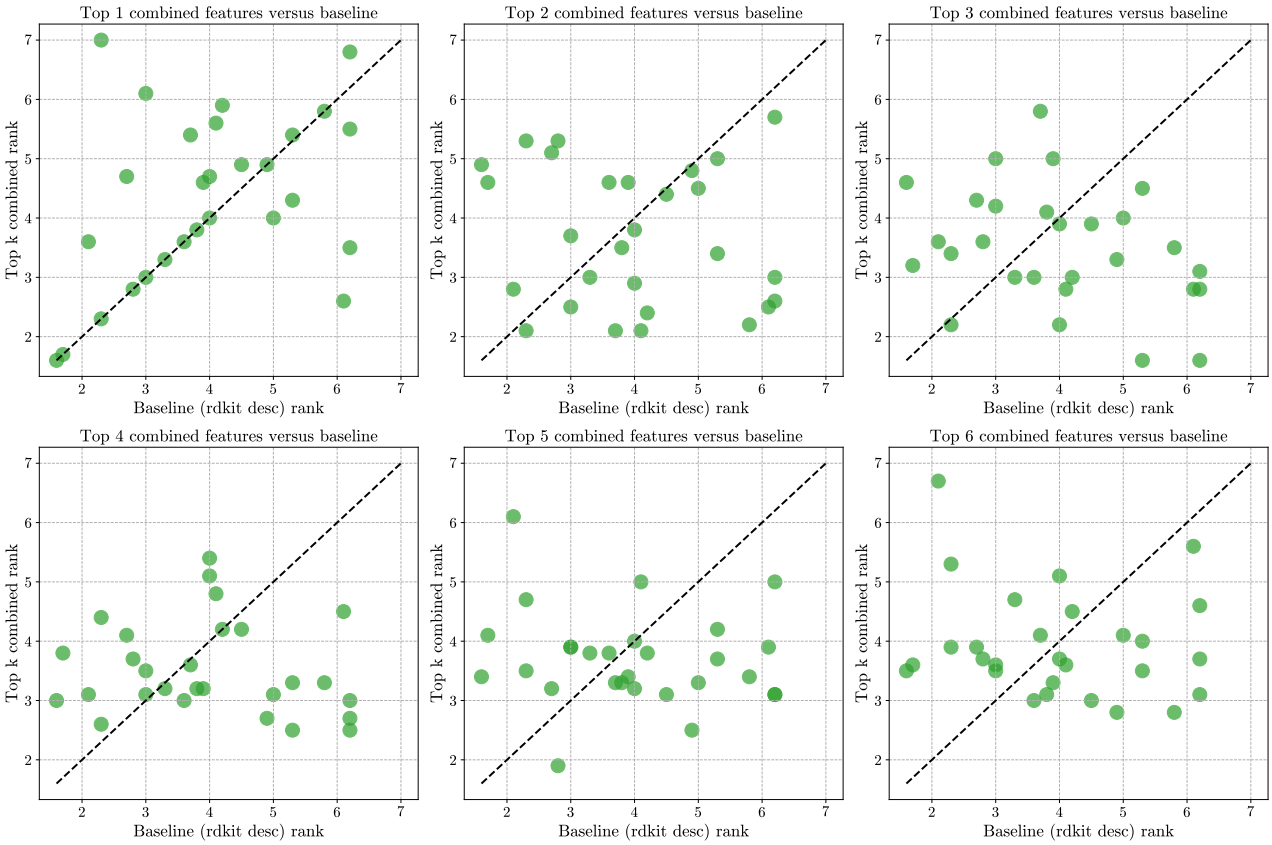

Figure 8: Baseline (rdkit desc) model performance compared to top-k performing feature concatenations based on the single fold experiment. Points along the X=Y axis in the first panel show that for many datasets it was `rdkit desc` itself that was the best performing feature.

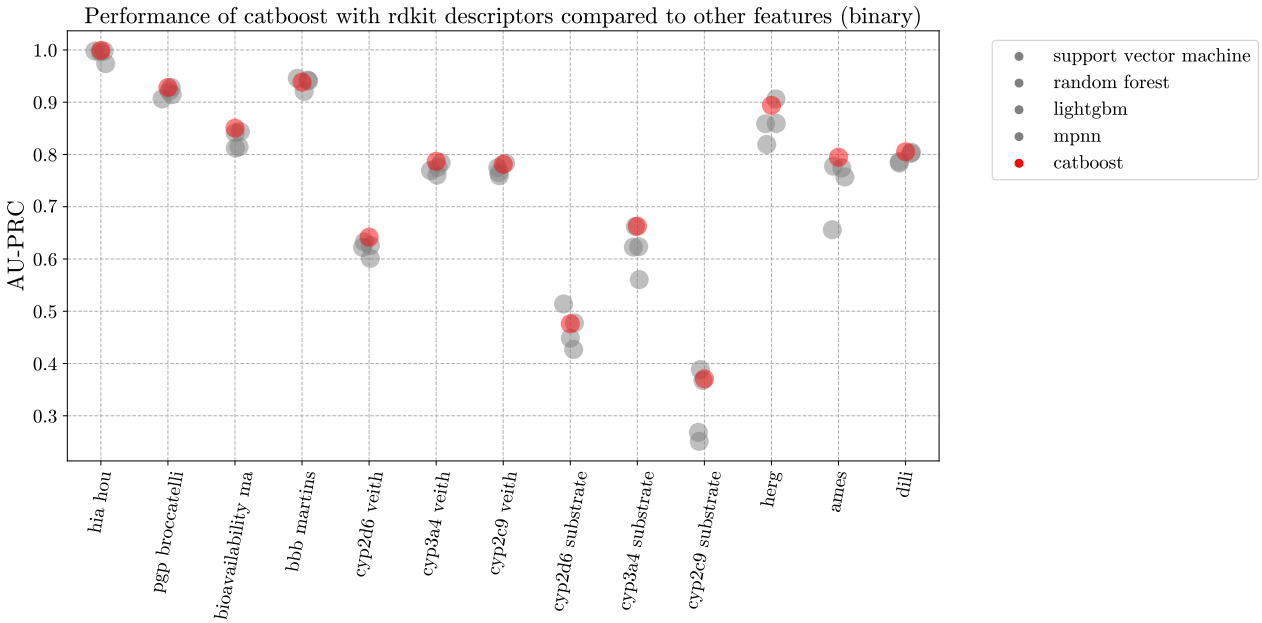

Figure 9: Performance of CatBoost with rdkit descriptors in the single fold experiment, compared to other models that also use rdkit descriptors, on the binary classification datasets.

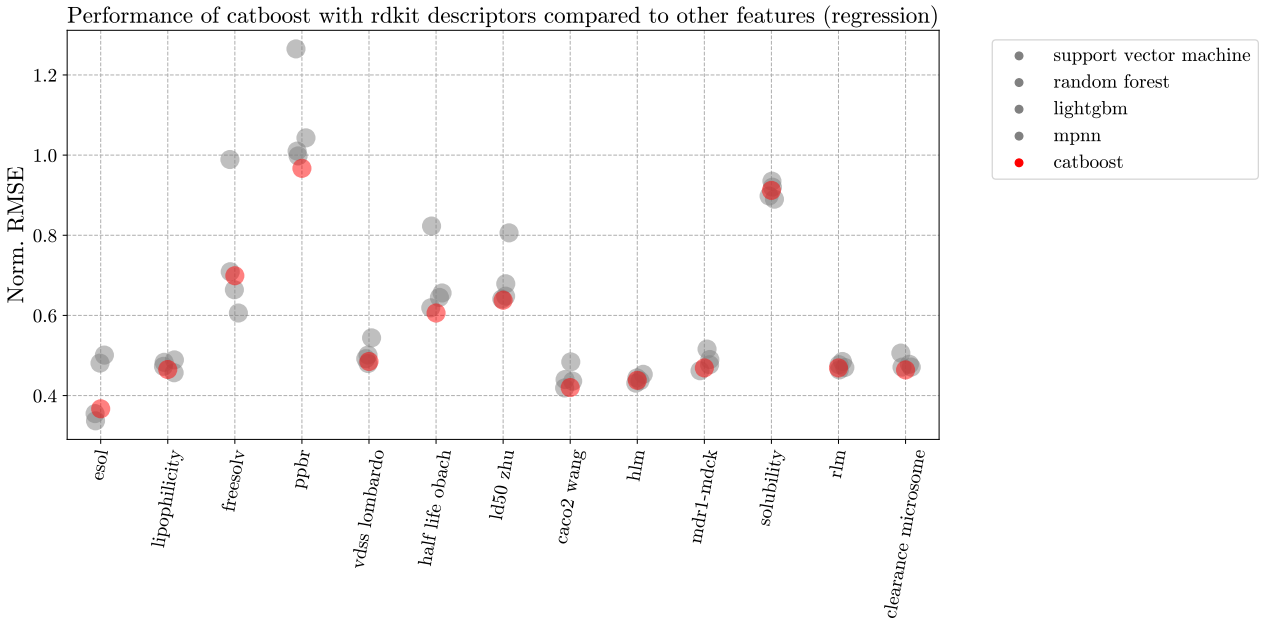

Figure 10: Performance of CatBoost with rdkit descriptors in the single fold experiment, compared to other models that also use rdkit descriptors, on the regression datasets.

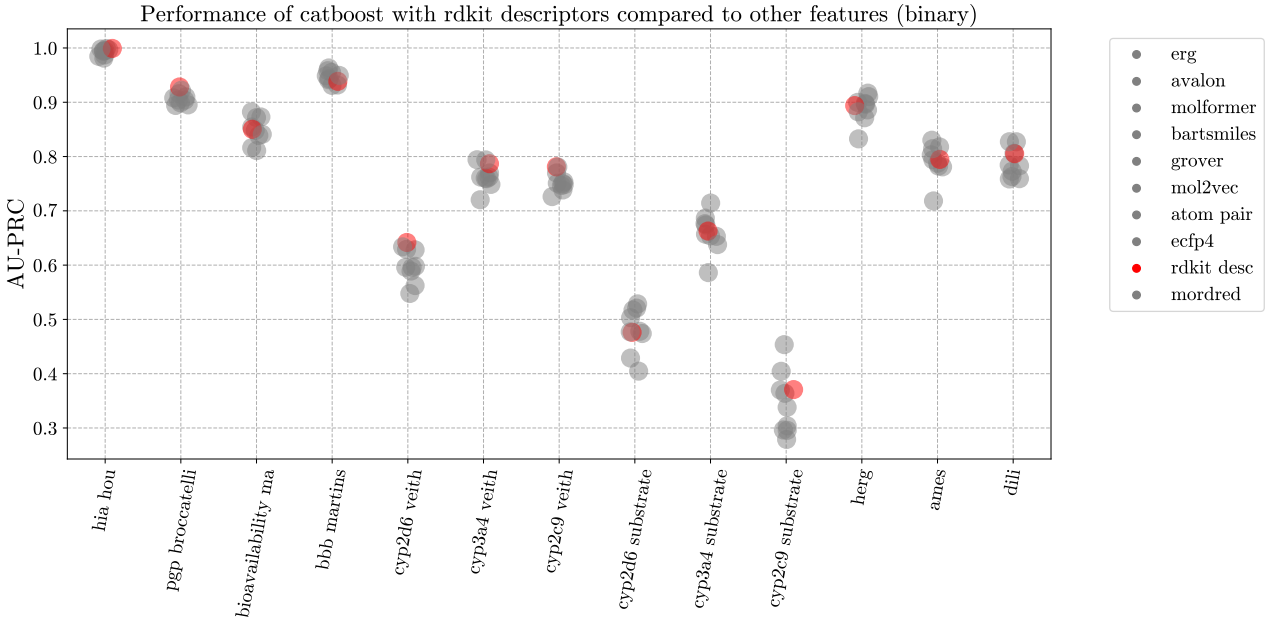

Figure 11: Performance of CatBoost with rdkit descriptors in the single fold experiment, compared to CatBoost trained on other features, on the binary classification datasets.

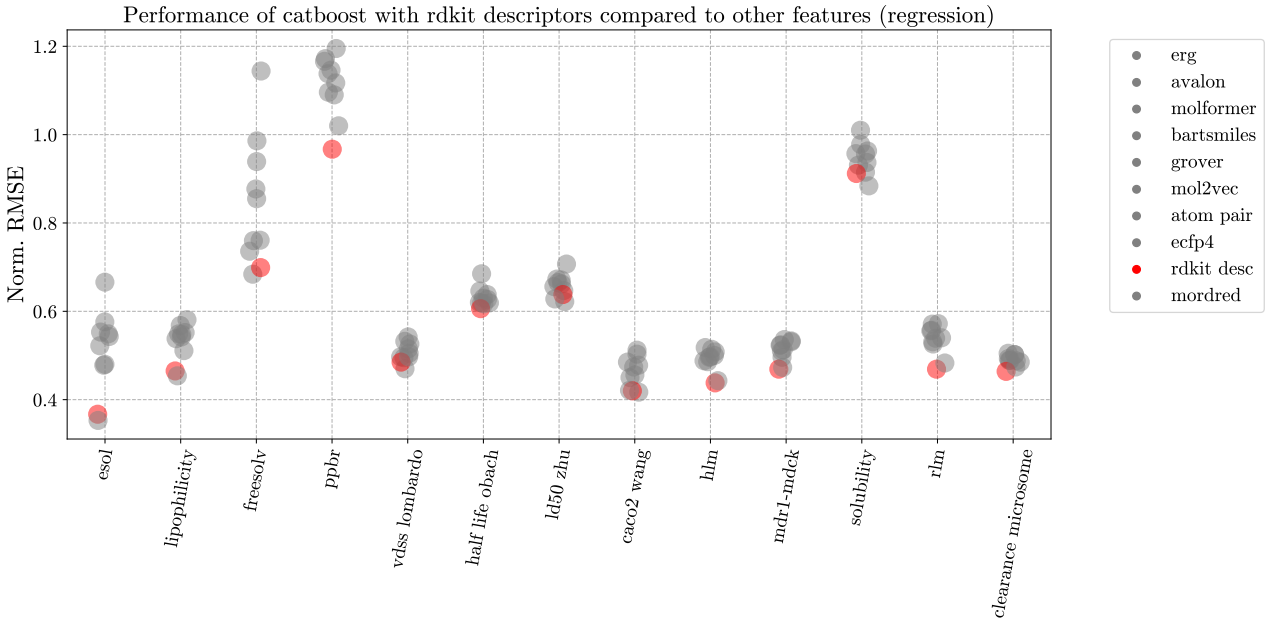

Figure 12: Performance of CatBoost with rdkit descriptors in the single fold experiment, compared to CatBoost trained on other features, on the regression datasets.

Table 6: 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			

Table 7: 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 8: 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 9: 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 10: 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 11: 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 12: 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 13: 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;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 14: 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 15: 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 16: 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 17: 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 18: 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 19: 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 20: 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 21: 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 22: 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 23: 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 24: 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 25: 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 26: 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 27: 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 28: 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 29: Data cleaning breakdown for freesolv

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	642	636	3	3
Removal reason	**Count**			
no_non_salt_or_inorganic	3			
Transformation description	**Count**			
25 Charge-seperate sulphoxides	2			
canonicalized	1			

Table 30: 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			

Table 31: Data cleaning breakdown for esol

Metric	**Total**	**Unchanged**	**Transformed**	**Removed**
SMILES count	1128	1102	14	12
Removal reason	**Count**			
duplicate	10			
inconsistent_duplicate	2			
Transformation description	**Count**			
02 2-hydroxy pyridine -> 2-pyridone	5			
25 Charge-seperate sulphoxides	4			
14 Enol -> Ketone 2	2			
13 Enol -> Ketone 1	2			
03 4-hydroxy pyridine -> 4-pyridone (within-ring)	1			