Generalizable, Fast, and Accurate DeepQSPR with `fastprop`

Part 1: Framework and Benchmarks

Jackson W. Burns [1] and William H. Green [1,*]

[1]Massachusetts Institute of Technology, Cambridge, MA
[*]Corresponding: whgreen@mit.edu

January 31, 2024

Abstract

Quantitative Structure-Property/Activity Relationship studies, often referred to interchangeably as QS(P/A)R, seek to establish a mapping between molecular structure and an arbitrary Quantity of Interest (QOI). Since its inception this was done on a QOI-by-QOI basis with new descriptors being devised by researchers to *specifically* map to their QOI. This continued for years and culminated in packages like DRAGON (later E-dragon), PaDEL-descriptor (and padelpy), Mordred, and many others. The sheer number of different packages resulted in the creation of 'meta-packages' which served only to aggregate these other calculators, including tools like molfeat, ChemDes, Parameter Client, and AIMSim. Despite the creation of these aggregation tools, QSPR studies have continued to focus on QOI-by-QOI studies rather than attempting to create a generalizable approach which is capable of modeling across chemical domains.

One contributing factor to this is that QSPR studies have relied almost exclusively on linear methods for regression. Efforts to incorporate Deep Learning (DL) as a regression technique (Deep-QSPR), which would be capable of capturing the non-linear behavior of arbitrary QOIs, have instead focused on using molecular fingerprints as inputs. The combination of bulk molecular-level descriptors with DL has remained largely unexplored, in significant part due to the orthogonal domain expertise required to combine the two. Generalizable QSPR has turned to learned representations primarily via message passing graph neural networks. This approach has proved remarkably effective but is not without drawbacks. Learning a representation requires large datasets to avoid over-fitting or even learn at all, loses interpretability since an embedding's meaning can only be induced retroactively, and needs significant execution time given the complexity of the underlying message passing algorithm. This paper introduces `fastprop`, a software package and general Deep-QSPR framework that combines a cogent set of molecular descriptors with DL to achieve state-of-the-art performance on datasets ranging from tens to tens of thousands of molecules. `fastprop` is designed with Research Software Engineering best practices and is free and open source, hosted at github.com/jacksonburns/fastprop.

Scientific Contribution

`fastprop` is a QSPR framework that achieves state-of-the-art accuracy on datasets of all sizes without sacrificing speed or interpretability. As a software package `fastprop` emphasizes Research Software Engineering best practices, reproducibility, and ease of use for experts across domains.

Keywords

- QSPR
- Learned Representations
- Deep Learning

- Molecular Descriptors

Introduction

Chemists have long sought a method to relate only the connectivity of a molecule to its corresponding molecular properties. The Quantitative Structure-Property Relationship (QSPR) would effectively solve the forward problem of molecular engineering and enable rapid development. Reviews on the topic are numerous and cover an enormous range of scientific subdomains; a comprehensive review of the literature is beyond the scope of this publication, though the work of Muratov and coauthors [1] provides an excellent starting point for further review. An abridged version of the history behind QSPR is presented here to contextualize the approach of `fastprop`.

Historical Approaches

Early in the history of computing, limited computational power meant that significant human expertise was required to guide QSPR models toward effectiveness. This materialized in the form of bespoke molecular descriptors: the Wiener Index in 1947 [2], Atom-Bond Connectivity indices in 1998 [3], and *thousands* of others. To this day descriptors are still being developed - the geometric-harmonic-Zagreb degree based descriptors were proposed by Arockiaraj et al. in 2023 [4]. In each case, domain experts devised an algorithm which mapped a molecular structure to some scalar value. This algorithm would take into account features of the molecule which that expert deduced were relevant to the property at hand. This time consuming technique is of course highly effective but the dispersed nature of this chemical knowledge means that these descriptors are spread out throughout many journals and domains with no single source to compute them all.

The range of regression techniques applied to these descriptors has also been limited. As explained by Muratov et. al [1] QSPR uses linear methods (some of which are now called machine learning) almost exclusively. The over-reliance on this category of approaches may be due to priorities; domain experts seek interpretability in their work, especially given that the inputs are physically meaningful descriptors, and linear methods lend themselves well to this approach. Practice may also have been a limitation, since historically training and deploying neural networks required more computer science expertise than linear methods.

All of this is not to say that DL has *never* been applied to QSPR. Applications of DL to QSPR, i.e. DeepQSPR, were attempted throughout this time period but focused on the use of molecular fingerprints rather than descriptors. This may be at least partially attributed to knowledge overlap between deep learning experts and this sub-class of descriptors. Molecular fingerprints are bit vectors which encode the presence or absence of human-chosen sub-structures in an analogous manner to the "bag of words" featurization strategy common to natural language processing. It is reasonable to assume a DL expert may have bridged this gap to open this subdomain, and its effectiveness proved worthwhile. In the review of DL for QSPR by Ma and coauthors [5] claim that combinations of fingerprint descriptors are more effective than molecular-level descriptors, either matching our outperforming linear methods across a number of ADME-related datasets. This study will later refute that suggestion.

Despite their differences, both classical- and Deep-QSPR shared a lack of generalizability. Beyond the domains of chemistry where many of the descriptors had been originally devised, models were either unsuccessful or more likely simply never evaluated. As interest began to shift toward the prediction of molecular properties which were themselves descriptors (i.e. derived from quantum mechanics simulations) - to which none of the devised molecular descriptors were designed to be correlated - learned representations (LRs) emerged.

Shift to Learned Representations

The exact timing of the transition from descriptors (molecular-level or fingerprints) to LRs is difficult to ascertain. Among the most cited at least is the work of Yang and coworkers in 2019 [6] which laid the groundwork for applying LRs to "Property Prediction" - QSPR by another name. In short, the basic idea is to initialize a molecular graph with only information about its bonds and atoms such as order, aromaticity, atomic number, etc. Then via a Message Passing Neural Network (MPNN) architecture, which is able to

aggregate these atom- and bond-level features into a vector in a manner which can be updated, the 'best' representation of the molecule is found during training. This method proved highly accurate *and* achieved the generalizability apparently lacking in descriptor-based modeling. The corresponding software package Chemprop (later described in [7]) has become a *de facto* standard for property prediction, partially because of the significant development and maintenance effort surrounding the software itself.

Following the initial success of Chemprop numerous representation learning frameworks have been devised, all of which slightly improve performance. The Communicative-MPNN (CMPNN) framework is a modified version of Chemprop with a different message passing scheme to increase the interactions between node and edge features [8]. Uni-Mol incorporates 3D information and relies extensively on transformers [9]. In a "full circle moment" architectures like the Molecular Hypergraph Neural Network (MHNN) have been devised to learn representations for specific subsets of chemistry, in that case optoelectronic properties [10]. Myriad others exist including GSL-MPP (accounts for intra-dataset molecular similarity) [11], SGGRL (trains three representations simultaneously using different input formats) [12], and MOCO (multiple representations and contrastive pretraining) [13].

Limitations

Despite the continuous incremental performance improvements, this area of research has had serious drawbacks. A thru-theme in these frameworks is the increasing complexity of DL techniques and consequent uninterpretability. This also means that actually *using* these methods to do research on real-world dataset requires varying amounts of DL expertise, creating a rift between domain experts and these methods. Perhaps the most significant failure is the inability to achieve good predictions on small [1] datasets. This is a long-standing limitation, with the original Chemprop paper stating that datasets with fewer than 1000 entries see fingerprint-based linear on par with Chemprop [6].

This limitation is especially challenging because it is a *fundamental* drawback of the LR approach. Without the use of advanced DL techniques like pre-training or transfer learning, the model is essentially starting from near-zero information every time a model is created. This inherently requires larger datasets to allow the model to effectively 're-learn' the chemical intuition which was built in to descriptor- and fingerprint-based representations.

Efforts are of course underway to address this limitation, though none are broadly successful. One simple but incredibly computationally expensive approach is to use delta learning, which artificially increases dataset size by generating all possible *pairs* of molecules from the available data (thus squaring the size of the dataset). This was attempted by Nalini et al. [14], who use an unmodified version of Chemprop referred to as 'DeepDelta' to predict *differences* in molecular properties for *pairs* of molecules. They achieve increased performance over standard LR approaches but *lost* the ability to train on large datasets due to simple runtime limitations. Other increasingly complex approaches are discussed in the outstanding review by van Tilborg et al. [15], though such techniques are furthering the consequences of complexity mentioned above.

While iterations on LRs and novel approaches to low-data regimes have been in development, the classical QSPR community has continued their work. A turning point in this domain was the release of `mordred`, a fast and well-developed package capable of calculating more than 1600 molecular descriptors. Critically this package was fully open source and written in Python, allowing it to readily interoperate with the world-class Python DL software ecosystem that greatly benefitted the LR community. Now despite previous evidence that molecular descriptors *cannot* achieve generalizable QSPR in combination with DL, the opposite is shown.

[1] What constitutes a 'small' dataset is decidedly *not* agreed upon by researchers. For the purposes of this study, it will be used to refer to datasets with ~1000 samples or less, which the authors believe better reflects the size of real-world datasets.

Implementation

At its core the `fastprop` 'architecture' is simply the `mordred` molecular descriptor calculator [2] [17] connected to a Feedforward Neural Network (FNN) implemented in PyTorch Lightning [18] (Figure 1). The user simply specifies a set of SMILES [19], a linear textual encoding of molecules, and their corresponding properties. `fastprop` automatically calculates and caches the corresponding molecular descriptors with `mordred`, re-scales both the descriptors and the targets appropriately, and then trains an FNN with to predict the indicated target properties. By default this FNN is two hidden layers with 1800 neurons each connected by ReLU activation functions, though the configuration can be readily changed via the command line interface or configuration file. `fastprop` owes its success to the cogent set of descriptors assembled by the developers of `mordred`, the ease of training FNNs with modern software like PyTorch Lightning, and the careful application of Research Software Engineering best practices that make it as user friendly as the best-maintained alternatives.

Figure 1: `fastprop` logo.

This trivially simple idea has been alluded to in previous published work but neither described in detail nor lauded for its generalizability or accuracy. Comesana and coauthors, based on a review of the biofuels property prediction landscape, claimed that methods (DL or otherwise) using large numbers of molecular descriptors were unsuccessful, instead proposing a feature selection method [20]. As a baseline in a study of photovoltaic property prediction, Wu et al. reported using the `mordred` descriptors in combination with both a Random Forest and an Artificial Neural Network, though in their hands the performance is worse than their bespoke model and no code is available for inspection [21].

Others have also incorporated `mordred` descriptors into their modeling efforts, though none with a simple FNN as described above. Esaki and coauthors started a QSPR study with `mordred` descriptors for a dataset of small molecules, but via an enormously complex modeling pipeline (using only linear methods) removed all but 53 [22]. Yalamanchi and coauthors used DL on `mordred` descriptors as part of a two-headed representation, but their network architecture was sequential hidden layers *decreasing* in size to only 12 features [23] as opposed to the constant 1800 in `fastprop`.

The reason `fastprop` stands out from these studies and contradicts previous reports is for the simple reason that it works. As discussed at length in the Results & Discussion section, this approach matches the performance of leading LR approaches on common benchmark datasets and bespoke QSPR models on small real-world datasets. `fastprop` also overcomes the limitations of LRs discussed above. Because all inputs to the FNN are physically meaningful molecular descriptors, intermediate representations in the FNN are also physically meaningful and can be directly interpreted. The simplicity of the framework enables domain experts to apply it easily and makes model training dramatically faster than LRs. Most importantly this approach is successful on the *smallest* of real-world datasets. By starting from such an informed initialization the FNN can be readily trained on datasets with as few as *forty* training examples (see PAHs).

[2] The original `mordred` package is no longer maintained. `fastprop` uses a fork of `mordred` called `mordredcommunity` that is maintained by community-contributed patches (see github.com/JacksonBurns/mordred-community). Multiple descriptor calculators from the very thorough review by McGibbon et al. [16] could be used instead, though none are as readily interoperable as `mordred`.

Example Usage

`fastprop` is built with ease of use at the forefront of design. To that end, input data is accepted in the immensely popular Comma-Separated Values (CSV) format, editable with all modern spreadsheet editors and completely platform independent. An example specify some properties for benzene is shown below, including its SMILES string:

```
compound,smiles,log_p,retention_index,boiling_point_c,acentric_factor
Benzene,C1=CC=CC=C1,2.04,979,80,0.21
```

`fastprop` itself is accessed via the command line interface, with configuration parameters passed as either command line arguments or in an easily editable configuration file:

```
# pah.yml
# generic args
output_directory: pah
random_seed: 55
problem_type: regression
# featurization
input_file: pah/arockiaraj_pah_data.csv
target_columns: log_p
smiles_column: smiles
descriptors: all
# preprocessing
zero_variance_drop: False
colinear_drop: False
# training
number_repeats: 4
number_epochs: 100
batch_size: 64
patience: 15
train_size: 0.8
val_size: 0.1
test_size: 0.1
sampler: random
```

Training, prediction, and feature importance and then readily accessible via the commands `fastprop train`, `fastprop predict`, or `fastprop shap`, respectively. The `fastprop` GitHub repository contains a Jupyter notebook runnable from the browser via Google colab which allows users to actually execute the above example, which is also discussed at length in the PAHs section, as well as further details about each configurable option.

Results & Discussion

There are a number of established molecular property prediction benchmarks commonly used in LR studies, especially those standardized by MoleculeNet [24]. Principal among them are QM8 [25] and QM9 [26], often regarded as *the* standard benchmark for property prediction. These are important benchmarks and are included here for completeness, though the enormous size and rich coverage of chemical space (inherent in the design of the combinatorial datasets) means that nearly all model architectures are highly accurate.

Real world datasets, particularly those common in QSPR studies, often number in the hundreds. To demonstrate the applicability of `fastprop` to these regimes, many smaller datasets are selected including some from the QSPR literature that are not established benchmarks. These studies relied on more complex and slow modeling techniques (ARA) or the design of a bespoke descriptor (PAHs) and have not yet come to rely on learned representations as a go-to tool. In these data-limited regimes where LRs sometimes struggle, `fastprop` and its intuition-loaded initialization are highly powerful. To emphasize this point further, the benchmarks are presented in order of size, descending, for first regression and then classification tasks.

Consistently `fastprop` is able to match the performance of LRs on large datasets (O(10,000)+ entries) and compete with or exceed their performance on small datasets ($\leq$O(1,000) entries).

All of these `fastprop` benchmarks are reproducible, and complete instructions for installation, data retrieval and preparation, and training are publicly available on the `fastprop` GitHub repository at github.com/jacksonburns/fastprop.

Benchmark Methods

All benchmarks use 80% of the dataset for training (selected randomly, unless otherwise stated), 10% for validation, and holdout the remaining 10% for testing (unless otherwise stated). Sampling is performed using the `astartes` package [27] which implements a variety of sampling algorithms and is highly reproducible.

Results for `fastprop` are reported as the average value of a metric and its standard deviation across a number of repetitions (repeated re-sampling of the dataset). The number of repetitions is chosen to either match referenced literature studies or else increased from two until the performance no longer meaningfully changes. Note that this is *not* the same as cross-validation.

For performance metrics retrieved from literature it is assumed that the authors optimized their respective models to achieve the best possible results; therefore, `fastprop` metrics are reported after model optimization using the `fastprop train ... --optimize` option. When results are generated for this study using Chemprop, the default settings are used except that the number of epochs is increased to allow the model to converge and batch size is increased to match dataset size and speed up training.

When reported, execution time is as given by the unix `time` command using Chemprop version 1.6.1 on Python 3.8 and includes the complete invocation of Chemprop, i.e. `time chemprop_train ...`. The insignificant time spent manually collating Chemprop results (Chemprop does not natively support repetitions) is excluded. `fastprop` is run on version 1.0.0b2 using Python 3.11 and timing values are reported according to its internal time measurement which was verified to be nearly identical to the Unix `time` command. The coarse comparison of the two packages is intended to emphasize the scaling of LRs and Deep-QSPR and that `fastprop` is, generally speaking, much faster. All models trained for this study were run on a Dell Precision series laptop with an NVIDIA Quadro RTX 4000 GPU and Intel Xeon E-2286M CPU.

Performance Metrics

The evaluation metrics used in each of these benchmarks are chosen to match literature precedent, particularly as established by MoleculeNet [24], where available. It is common to use scale-dependent metrics that require readers to understand the relative magnitude of the target variables. The authors prefer more readily interpretable metrics such as (Weighted) Mean Absolute Percentage Error (W/MAPE) and are included where relevant.

All metrics are defined according to their typical formulae which are readily available online and are implemented in common software packages. Those presented here are summarized below, first for regression: - Mean Absolute Error (MAE): Absolute difference between predictions and ground truth averaged across dataset; scale-dependent. - Root Mean Squared Error (RMSE): Absolute differences *squared* and then averaged; scale-dependent. - Mean Absolute Percentage Error (MAPE): MAE except that differences are relative (i.e. divided by the ground truth); scale-independent, range 0 (best) and up. - Weighted Mean Absolute Percentage Error (WMAPE): MAPE except the average is a weighted average, where the weight is the magnitude of the ground truth; scale-independent, range 0 (best) and up. - Coefficient of Determination (R2): Proportion of variance explained; scale-independent, range 0 (worst) to 1 (best). and classification: - Area Under the Receiver Operating Curve (AUROC, AUC, or ROC-AUC): Summary statistic combining all possible classification errors; scale-independent, range 0.5 (worst, random guessing) to 1.0 (perfect classifier). - Accuracy: Fraction of correct classifications, expressed as either a percentage or a number; scale-independent, range 0 (worst) to 1 (perfect classifier).

Regression Datasets

See Table 1 for a summary of all the regression dataset results. Especially noteworthy is the performance on the ESOL and PAH datasets, which dramatically surpass literature best and have only 55 datapoints, respectively. Citations for the datasets themselves are included in the sub-sections of this section.

Table 1: Summary of regression benchmark results.

Benchmark	Samples (k)	Metric	`Literature Best`	`fastprop`	`Chemprop`
QM9	~134	MAE	0.0047[a]	0.0069	0.0081[a]
OCELOTv1	~25	MAE	0.128[b]	0.158	0.140[b]
QM8	~22	MAE	0.016[a]	0.018	0.019[a]
ESOL	~1.1	RMSE	0.55[c]	0.64	0.67[c]
FreeSolv	~0.6	RMSE	0.82[c]	1.33	1.26[c]
Flash	~0.6	RMSE	13.2[e]	13.3	21.2*
YSI	~0.4	MAE	22.3[f]	13.6	28.9*
HOPV15[g]	~0.3	MAE	1.32[g]	1.55	1.60
Fubrain	~0.3	RMSE	0.44[h]/0.83[d]	0.19/0.74	0.22*/0.97[d]
PAH	~0.06	R2	0.96[i]	0.98	0.59*

a [9] b [10] (summary result is geometric mean across tasks) c [8] d [14] (delta-learning instead of direct prediction) e [28] f [29] g [30] (uses a subset of the complete HOPV15 dataset) h [22] i [4] * These results were generated for this study.

QM9

Originally described in Scientific Data [26] and perhaps the most established property prediction benchmark, Quantum Machine 9 (QM9) provides quantum mechanics derived descriptors for all small molecules containing one to nine heavy atoms, totaling ~134k. The data was retrieved from MoleculeNet [24] in a readily usable format. As a point of comparison, performance metrics are retrieved from the paper presenting the UniMol architecture [9] previously mentioned. In that study they trained on only three especially difficult QOIs (homo, lumo, and gap) using scaffold-based splitting (a more challenging alternative to random splitting), reporting mean and standard deviation across 3 repetitions.

`fastprop` achieves 0.0069 ± 0.0002 mean absolute error, whereas Chemprop achieves 0.00814 ± 0.00001 and the UniMol framework manages 0.00467 ± 0.00004. This places the `fastprop` framework ahead of previous learned representation approaches but still trailing UniMol. This is not completely unexpected since UniMol encodes 3D information from the dataset whereas Chemprop and `fastprop` use only 2D. Future work could evaluate the use of 3D-based descriptors to improve `fastprop` performance in the same manner that UniMol has with LRs.

OCELOTv1

The Organic Crystals in Electronic and Light-Oriented Technologies (OCELOTv1) dataset, originally described by Bhat et al. [31], maps 15 quantum mechanics descriptors and optoelectronic properties to ~25k chromophoric small molecules. The literature best model is the Molecular Hypergraph Neural Network (MHNN) [10] which specializes in learned representations for optoelectronic properties and also includes Chemprop as a baseline for comparison. They used a 70/10/20 random split with three repetitions and final performance reported is the average across those three repetitions.

As done in the reference study, the MAE for each task is shown in Table 2. Meanings for each abbreviation are the same as in the original database publication [31]. The geometric mean across all tasks, which accounts for the different scales of the target values better than the arithmetic mean, is also included as a summary statistic. Note also that the relative percentage performance difference between fastprop and chemprop (`fast/chem`) and fastprop and MHNN (`fast/MHNN`) are also included.

Table 2: Per-task OCELOT dataset results. MHNN and Chemprop results are retrieved from the literature [10].

Target	fastprop	Chemprop	fast/chem	MHNN	fast/mhnn
HOMO	0.328	0.330	-0.7%	0.306	7.1%
LUMO	0.285	0.289	-1.5%	0.258	10.4%
H-L	0.555	0.548	1.4%	0.519	7.0%
VIE	0.210	0.191	9.9%	0.178	18.0%
AIE	0.201	0.173	16.2%	0.162	24.0%
CR1	0.057	0.055	3.3%	0.053	7.2%
CR2	0.057	0.053	7.2%	0.052	9.3%
HR	0.108	0.133	-18.5%	0.099	9.5%
VEA	0.194	0.157	23.5%	0.138	40.5%
AEA	0.188	0.154	21.8%	0.124	51.3%
AR1	0.055	0.051	8.2%	0.050	10.4%
AR2	0.049	0.052	-5.1%	0.046	7.3%
ER	0.100	0.098	1.9%	0.092	8.5%
S0S1	0.284	0.249	14.3%	0.241	18.0%
S0T1	0.222	0.150	48.3%	0.145	53.4%
G-Mean	0.151	0.140	7.7%	0.128	17.9%

`fastprop` 'trades places with Chemprop, outperforming on four of the metrics (LUMO, HR, AR2) and under-performing on others. Overall the geometric mean of MAE across all the tasks is ~8% higher, though this result may not be statistically significant. Both `fastprop` and Chemprop are outperformed by the bespoke MHNN model, which is not itself evaluated on any other common property prediction benchmarks.

Although `fastprop` is not able to reach state-of-the-art accuracy on this dataset this result is still promising. None of the descriptors implemented in `mordred` were designed to specifically correlate to these QM-derived targets, yet the FNN is able to learn a representation which is nearly as informative as Chemprop. The fact that a bespoke modeling approach is the most performant is not surprising and instead demonstrates the continued importance of expert input on certain domains. Were some of this expertise to be oriented toward the descriptor generation software `mordredcommunity`, new descriptors could be added to address this apparent shortcoming.

QM8

Quantum Machine 8 (QM8) is the predecessor to QM9 first described in 2015 [25]. It follows the same generation procedure as QM9 but includes only up to eight heavy atoms for a total of approximately 22k molecules. Again, this study used the dataset as prepared by MoleculeNet [24] and compares to the UniMol [9] set of benchmarks as a reference point, wherein they used the same data splitting procedure described previously but regressed all 12 targets in QM8.

UniMol achieved an average MAE across all tasks of 0.00156 ± 0.0001, `fastprop` approaches that performance with 0.0178 ± 0.003, and Chemprop trails both frameworks with 0.0190 ± 0.0001. Much like with QM9 `fastprop` outperforms LR frameworks until 3D information is encoded with UniMol. As previously stated this is achieved despite the targets being predicted not being directly intended for correlation with the `mordred` descriptors.

Of note is that even though `fastprop` is approaching the leading performance on this benchmark, other performance metrics cast doubt on the model performance. The weighted mean absolute percentage error (wMAPE) on a per-task basis is shown in Table 3.

Table 3: Per-task QM8 dataset results.

Metric	wMAPE
E1-CC2	3.6%
E2-CC2	3.3%
f1-CC2	83.0%
f2-CC2	86.7%
E1-PBE0	3.6%
E2-PBE0	3.3%
f1-PBE0	80.6%
F2-PBE0	86.4%
E1-CAM	3.4%
E2-CAM	3.0%
f1-CAM	77.3%
f2-CAM	81.2%
Average	43.0%

At each level of theory (CC2, PBE0, and CAM) `fastprop` is reaching the limit of chemical accuracy on excitation energies (E1 and E2) but is significantly less accurate on oscillator strengths (f1 and f2). This can at least partially be attributed to dataset itself. Manual analysis reveals that nearly 90% of the molecules in the dataset fall within only 10% of the total range of f1 values, which is highly imbalanced. Additionally that 90% of molecules actual f1 values are all near-zero or zero, which are intentionally less represented in the wMAPE metric. Future literature studies should take this observation into account and perhaps move away from this splitting approach toward one which accounts for this imbalance.

ESOL

First described in 2004 [32] and has since become a critically important benchmark for QSPR/molecular property prediction studies. The dataset includes molecular structure for approximately 1.1k simple organic molecules are their corresponding experimentally measured free energy of solvation. This property is a classic target of QSPR studies and is especially well suited for `fastprop`.

The CMPNN [8] model, a derivative of Chemprop, is used for comparison. The performance values are *not* those from the original paper but instead the *corrected* results shown on the CMPNN GitHub page [33]. These numbers are the average and standard deviation across 5 repetitions, each including 5-fold cross validation (60/20/20 random split), for a total of 25 models. `fastprop` performance is reported using the same split sizes across 8 repetitions *without* cross validation; increasing the number of repetitions further did not meaningfully change model performance.

`fastprop` achieves an RMSE of 0.643 ± 0.048 trailing the CMPNN at 0.547 ± 0.011 but matching Chemprop at 0.665 ± 0.052. The same pattern from previous benchmarks is repeated - `fastprop` matches the performance of generic learned representation approaches but is outperformed by bespoke modeling techniques. In this case the CMPNN has been designed to perform better on these benchmark datasets specifically and at an increased cost in both execution time and complexity.

FreeSolv

The Free Energy of Solvation (FreeSolv) database is a more curated alternative to ESOL developed and maintained by Mobley et al. in 2014 [34]. It contains approximately 0.6k molecules and their experimental *and* calculated hydration free energy. This benchmark is the smallest 'recognized' benchmark often reported in the molecular property prediction literature.

Again the CMPNN study is used as a reference and the same procedure described in ESOL is followed. `fastprop` achieves an RMSE of 1.33 ± 0.21, once again trailing trailing the CMPNN at 0.82 ± 0.15 but matching Chemprop at 1.26 ± 0.11.

Flash

First assembled and fitted to by Saldana and coauthors [28] the dataset (Flash) includes around 0.6k entries, primarily alkanes and some oxygen-containing compounds, and their literature-reported flash point. The reference study reports the performance on only one repetition, but manually confirms that the distribution of points in the three splits follows the parent dataset. The split itself was a 70/20/10 random split, which is repeated four times for this study.

Using a complex multi-model ensembling method, the reference study achieved an RMSE of 13.2, an MAE of 8.4, and an MAPE of 2.5%. `fastprop` matches this performance, achieving 13.3 ± 2.1 RMSE, 9.4 ± 0.8 MAE, and 2.6% ± 0.1% MAPE. Chemprop, however, struggles to match the accuracy of either method. It manages an RMSE of 21.2 ± 2.2 and an MAE of 13.8 ± 2.1 and does not report MAPE.

Critically, `fastprop` dramatically outperforms both methods in terms of training time. The reference model required significant manual intervention to create a model ensemble, so no single training time can be fairly identified. `fastprop` arrived at the indicated performance without any manual intervention in only 30 seconds, 13 of which were spent calculating descriptors. Chemprop, in addition to not reaching the same level of accuracy, took 5 minutes and 44 seconds to do so - more than ten times the execution time of `fastprop`.

YSI

Assembled by Das and coauthors [29] from a collection of other smaller datasets, this dataset maps ~0.4k molecular structures to a unified-scale Yield Sooting Index (YSI), a molecular property of interest to the combustion community. The reference study performs leave-one-out cross validation to fit a per-fragment contribution model, effectively a training size of >99%, without a holdout set. Though this is not standard practice and can lead to overly optimistic reported performance, the results will be carried forward regardless. The original study did not report overall performance metrics, so they have been re-calculated for this study using the predictions made by the reference model as provided on GitHub [3]. For comparison `fastprop` and Chemprop use a more typical 60/20/20 random split and 8 repetitions. Results are summarized in Table 4.

Table 4: YSI results.

Model	MAE	RMSE	WMAPE
Reference	22.3	50	0.3
`fastprop`	13.6 ± 2.1	54 ± 13	14.5 ± 2.2
Chemprop	28.9 ± 6.5	63 ± 14	~

`fastprop` significantly outperforms both other models when considering MAE, especially impressive in the case of the reference model which was trained on far more data. When considering RMSE, which penalizes large errors more than MAE, all models are similarly performant. Finally the WMAPE shows that the reference model makes much smaller errors on the highest YSI molecules compared to `fastprop`. Taken in combination with the MAE and RMSE values, which are respectively worse and competitive with `fastprop`, the model is likely highly overfit to the training data due to the cross-validation strategy.

Also notable is the difference in training times. Chemprop takes 7 minutes and 2 seconds while `fastprop` completes in only 38 seconds, again a factor of ten faster.

HOPV15 Subset

The HOPV15 Subset is a collection of ~0.3k organic photovoltaic compounds curated by Eibeck and coworkers from the larger Harvard Organic Photovoltaic (HOPV15 [35]) based on criteria described in their paper [30]. This dataset is unique in that the target property Power Conversion Efficiency is both experimentally measurable or can be derived from quantum mechanics simulations, but regardless is not a 'classical' target of QSPR. After applying a variety of established modeling techniques Eibeck et al. achieved a best-case MAE of

[3]Predictions are available at this permalink to the CSV file on GitHub.

1.32 ± 0.10 averaged across 5 randomly selected 60/20/20 data splits by using a simple molecular fingerprint representation and support vector regression.

In the course of this study it was found that changing the random seed when using only 5 repetitions would lead to dramatically different model performance. Thus, for this benchmark the number of repetitions was set *higher* than the reference study at 15. `fastprop` reports an average MAE of 1.55 ± 0.20 and an RMSE of 1.93 ± 0.20, in line with the performance of Chemprop at an MAE of 1.60 ± 0.15 and an RMSE of 1.97 ± 0.16. Execution time is again dramatically different with Chemprop taking 11 minutes and 35 seconds whereas `fastprop` took only 2 minutes and 3 seconds of which 1 minute and 47 seconds was spent calculating descriptors for the abnormally large molecules in HOPV15.

Fubrain

First described by Esaki and coauthors, the Fraction of Unbound Drug in the Brain (Fubrain) dataset is a collection of about 0.3k small molecule drugs and their corresponding experimental experimentally measured unbound fraction in the brain, a critical metric for drug development [22]. This specific target in combination with this dataset size makes this benchmark highly relevant for typical QSPR studies.

The study that first generated this dataset used `mordred` descriptors but as is convention they strictly applied linear modeling methods. Using both cross validation and and external test sets, they had an effective training/validation/testing split of 0.64/0.07/0.28 which will be repeated 4 times here for comparison. All told, their model achieved an RMSE of 0.53 averaged across all testing data.

In only 44 seconds, of which 36 are spent calculating descriptors, `fastprop` far exceeds the reference model with an RMSE of 0.19 ± 0.03 Under the same conditions Chemprop approaches `fastprop`'s performance with an RMSE of 0.22 ± 0.04 but requires 5 minutes and 11 seconds to do so, in this case a 7 times performance improvement for `fastprop` over Chemprop.

Delta-Fubrain Also noteworthy for the Fubrain dataset is that it has been subject to the delta-learning approach to small dataset limitations. DeepDelta [14] performed a 90/0/10 cross-validation study of the Fubrain dataset in which the training and testing molecules were used to generate all possible pairs and then the differences in the property [4] were predicted rather than absolute values. They reported an RMSE of 0.830 ± 0.023, whereas a Chemprop model trained to directly predict property values was only able to reach an accuracy of 0.965 ± 0.019 when evaluated on its capacity to predict property differences.

`fastprop` is able to overcome these limitations. Using the same model from above (re-trained to predict log-transformed values), which has *less* training data than DeepDelta even *before* the augmentation, `fastprop` achieves 0.740 ± 0.087 RMSE after pairing all withheld test molecules. Increasing the amount of training data while retaining some samples for early stopping yields only small improvements, showing that `fastprop` may be approaching the irreducible error of Fubrain. With an 89/1/10 split the RMSE of `fastprop` decreases to 0.7118 ± 0.1381, though with significantly increased variance due to small size of the testing data. Regardless, the execution time and scaling issues of DeepDelta and the inaccuracy of Chemprop are effectively circumvented by `fastprop`.

PAHs

Originally compiled by Arockiaraj et al. [4] the Polycyclic Aromatic Hydrocarbons (PAH) dataset contains water/octanol partition coefficients (logP) for exactly 55 polycyclic aromatic hydrocarbons ranging in size from napthalene to circumcoronene. This size of this benchmark is an ideal case study for the application of `fastprop`. Using expert insight the reference study designed a novel set of molecular descriptors that show a strong correlation to logP, with correlation coefficients ranging from 0.96 to 0.99 among the various new descriptors.

For comparison, `fastprop` and Chemprop are trained using 8 repetitions of a typical 80/10/10 random split - only **44** molecules in the training data. `fastprop` matches the performance of the bespoke descriptors with a

[4] Although the original Fubrain study reported untransformed fractions, the DeepDelta authors confirmed via GitHub that DeepDelta was trained on log base-10 transformed fraction values, which is replicated here.

correlation coefficient of 0.976 ± 0.027. This corresponds to an MAE of 0.160 ± 0.035 and an MAPE of 2.229 ± 0.061%. Chemprop effectively fails on this dataset, achieving a correlation coefficient of only 0.59 ± 0.24, an MAE of 1.04 ± 0.33 (one anti-correlated outlier replicate removed). Despite the large parameter size of the `fastprop` model relative to the training data, it readily outperforms Chemprop in the small-data limit.

For this unique dataset, execution time trends are inverted. `fastprop` takes 2 minutes and 44 seconds, of which 1 minute and 44 seconds were spent calculating descriptors for these unusually large molecules. Chemprop completes in 1 minute and 17 seconds, on par with the training time of `fastprop` without considering descriptor calculation.

Classification Datasets

See Table 5 for a summary of all the classification dataset results. Especially noteworthy is the performance on QuantumScents dataset, which outperforms the best literature result. Citations for the datasets themselves are included in the sub-sections of this section.

Table 5: Summary of classification benchmark results.

Benchmark	Samples (k)	Metric	Literature Best	`fastprop`	Chemprop
HIV	~41	AUROC	0.81[a]	0.81	0.77[a]
QuantumScents	~3.5	AUROC	0.88[b]	0.91	0.85[b]
SIDER	~1.4	AUROC	0.67[c]	0.64	0.65[c]
Pgp	~1.3	AUROC	0.94[e]	0.92	0.89[e]
ARA	~0.8	Accuracy	91[d]	89	82*

a [9] b [36] c [8] d [37] e [38] * These results were generated for this study.

HIV Inhibition

Originally compiled by Riesen and Bunke [39], this dataset includes the reported HIV activity for approximately 41k small molecules. This is an established benchmark in the molecular property prediction community and the exact version used is that which was standardized in MoleculeNet [24]. This dataset is unique in that the labels in the original study include three possible classes (a *multiclass*) regression problem whereas the most common reported metric is instead lumping positive and semi-positive labels into a single class to reduce the task to *binary* classification; both are reported here. UniMol is again used as a point of comparison, and thus an 80/10/10 scaffold-based split with three repetitions is used.

For binary classification `fastprop`'s AUROC of 0.81 ± 0.04 matches the literature best UniMol with and 0.808 ± 0.003 [9]. This corresponds to an accuracy of 96.8+/-1.0% for `fastprop`, which taken in combination with AUROC hints that the model is prone to false positives. Chemprop performs worse than both of these models with a reported AUROC of 0.771 ± 0.005.

When attempting multiclass classification, `fastprop` maintains a similar AUROC of 0.818+/-0.019 AUROC Accuracy suffers a prodigious drop to 42.8 ± 7.6%, now suggesting that the model is prone to false negatives. Other leading performers do not report performance metrics on this variation of the dataset.

QuantumScents

Compiled by Burns and Rogers [36], this dataset contains approximately 3.5k SMILES and 3D structures for a collection of molecules labeled with their scents. Each molecule can have any number of reported scents from a possible 113 different labels, making this benchmark a a Quantitative Structure-Odor Relationship. Due to the highly sparse nature of the scent labels a unique sampling algorithm (Szymanski sampling [40]) was used in the reference study and the exact splits are replicated here for a fair comparison.

In the reference study, Chemprop achieved an AUROC of 0.85 with modest hyperparameter optimization and an improved AUROC of 0.88 by incorporating the atomic descriptors calculated as part of QuantumScents.

`fastprop`, using neither these descriptors nor the 3D structures, outperforms both models with an AUROC of 0.910+/-0.001 with only descriptors calculated from the molecules' SMILES. The GitHub repository contains an example of generating custom descriptors incorporating the 3D information from QuantumScents and passing these to `fastprop`; impact on the performance was negligible.

SIDER

First described by Kuhn et al. in 2015 [41], the Side Effect Resource (SIDER) database has become a standard property prediction benchmark. This challenging dataset maps around 1.4k compounds, including small molecules, metals, and salts, to any combination of 27 side effects - leading performers are only slightly better than random guessing (AUROC 0.5).

Among the best performers in literature is the previously discussed CMPNN [8] with a reported AUROC of 0.666+/-0.007, which narrowly outperforms Chemprop at 0.646+/-0.016. Using the same approach, `fastprop` achieves a decent AUROC of 0.636+/-0.019. Despite many of the entries in this dataset being atypical for `mordred` near-leading performance is still possible, supporting the robustness and generalizability of this framework.

Pgp

First reported in 2011 by Broccatelli and coworkers [42], this dataset has since become a standard benchmark and is included in the Therapeutic Data Commons (TDC) [43] model benchmarking suite. the dataset maps approximately 1.2k small molecule drugs to a binary label indicating if they inhibit P-glycoprotein (Pgp). TDC serves this data through a Python package, but due to installation issues the data was retrieved from the original study instead. The recommended splitting approach is a 70/10/20 scaffold-based split which is done here with 4 replicates.

The model in the original study uses a molecular interaction field but has since been surpassed by other models. According to TDC the current leader [38] on this benchmark has achieved an AUROC of 0.938 $\pm$ 0.002 [5]. AOn the same leaderboard Chemprop [6] achieves 0.886 $\pm$ 0.016 with the inclusion of additional molecular features. `fastprop` yet again approaches the performance of the leading methods and outperforms Chemprop, here with an AUROC of 0.919 $\pm$ 0.013 and an accuracy of 84.5 $\pm$ 0.2%.

ARA

The final benchmark is one which closely mimics typical QSPR studies. Compiled by Schaduangrat et al. in 2023 [37], this dataset maps ~0.8k small molecules to a binary label indicating if the molecule is an Androgen Receptor Antagonist (ARA). The reference study introduced DeepAR, a highly complex modeling approach, which achieved an accuracy of 0.911 and an AUROC of 0.945.

For this study an 80/10/10 random splitting is repeated four times on the dataset since no analogous split to the reference study can be determined. Chemprop takes 16 minutes and 55 seconds to run on this dataset and achieves only 0.824+/-0.020 accuracy and 0.898+/-0.022 AUROC. `fastprop` takes only 2 minutes and 4 seconds (1 minute and 47 seconds for descriptor calculation) and is competitive with the reference study in performance, achieving a 89.1+/-4.0% accuracy and 0.951+/-0.018 AUROC.

Limitations and Future Work

Execution Time

Although `fastprop` is consistently around an order of magnitude faster to train than learned representations when using a GPU, execution time is a minor concern when considering the enormous labor invested in dataset generation. For day-to-day work it is convenient but the correctness of `fastprop`, especially on small datasets, is more important. Note that due to the large size of the FNN in `fastprop` it will typically be

[5] See the TDC Pgp leaderboard.

slower than Chemprop when training on a CPU since Chemprop uses a much smaller FNN and associated components.

Regardless, there is an clear performance improvement to be had by reducing the number of descriptors to a subset of only the most important. Future work will address this possibility to decrease time requirements for both training by reducing network size and inference by decreasing the number of descriptors to be calculated for new molecules. This has *not* been done in this study for two reasons: (1) to emphasize the capacity of the DL framework to effectively perform feature selection on its own via the training process, de-emphasizing unimportant descriptors; (2) as discussed above, training time is small compared ot dataset generation time.

Coverage of Descriptors

`fastprop` is fundamentally limited by the types of chemicals which can be uniquely described by the `mordred` package. Domain-specific additions which are not just derived from the descriptors already implemented will be required to expand its application to new domains.

For example, in its current state `mordred` does not include any connectivity based-descriptors that reflect the presence or absence of stereocenters. While some of the 3D descriptors it implements could implicitly reflect sterochemistry, more explicit descriptors like the Stereo Signature Molecular Descriptor [44] may prove helpful in the future if re-implemented in `mordred`.

Interpretability

Though not discussed here for the sake of length, `fastprop` already contains the functionality to perform feature importance studies on trained models. By using SHAP values [45] to assign a scalar 'importance' to each of the input features, users can determine which of the `mordred` descriptors has the largest impact on model predictions. Future studies will demonstrate this in greater detail.

Availability

- Project name: fastprop
- Project home page: github.com/jacksonburns/fastprop
- Operating system(s): Platform independent
- Programming language: Python
- Other requirements: pyyaml, lightning, mordredcommunity, astartes
- License: MIT

Declarations

Availability of data and materials

`fastprop` is Free and Open Source Software; anyone may view, modify, and execute it according to the terms of the MIT license. See github.com/jacksonburns/fastprop for more information.

All data used in the Benchmarks shown above is publicly available under a permissive license. See the benchmarks directory at the `fastprop` GitHub page for instructions on retrieving each dataset and preparing it for use with `fastprop`, where applicable.

Competing interests

None.

Funding

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Department of Energy Computational Science Graduate Fellowship under Award Number DE-SC0023112.

Authors' contributions

Initial ideation of `fastprop` was a joint effort of Burns and Green. Implementation, benchmarking, and writing were done by Burns.

Acknowledgements

The authors acknowledge Haoyang Wu, Hao-Wei Pang, and Xiaorui Dong for their insightful conversations when initially forming the central ideas of `fastprop`.

Disclaimer

Cited Works

1. Muratov EN, Bajorath J, Sheridan RP, et al (2020) QSAR without borders. Chem Soc Rev 49:3525–3564. https://doi.org/10.1039/D0CS00098A
2. Wiener H (1947) Structural determination of paraffin boiling points. Journal of the American Chemical Society 69:17–20. https://doi.org/10.1021/ja01193a005
3. Estrada E, Torres L, Rodriguez L, Gutman I (1998) An atom-bond connectivity index: Modelling the enthalpy of formation of alkanes
4. Arockiaraj M, Paul D, Clement J, et al (2023) Novel molecular hybrid geometric-harmonic-zagreb degree based descriptors and their efficacy in QSPR studies of polycyclic aromatic hydrocarbons. SAR and QSAR in Environmental Research 34:569–589. https://doi.org/10.1080/1062936x.2023.2239149
5. Ma J, Sheridan RP, Liaw A, et al (2015) Deep neural nets as a method for quantitative structure-activity relationships. Journal of Chemical Information and Modeling 55:263–274. https://doi.org/10.1021/ci500747n
6. Yang K, Swanson K, Jin W, et al (2019) Analyzing learned molecular representations for property prediction. Journal of Chemical Information and Modeling 59:3370–3388. https://doi.org/10.1021/acs.jcim.9b00237
7. Heid E, Greenman KP, Chung Y, et al (2024) Chemprop: A machine learning package for chemical property prediction. Journal of Chemical Information and Modeling 64:9–17. https://doi.org/10.1021/acs.jcim.3c01250
8. Song Y, Zheng S, Niu Z, et al (2021) Communicative representation learning on attributed molecular graphs. In: Proceedings of the twenty-ninth international joint conference on artificial intelligence. Yokohama, Yokohama, Japan
9. Zhou G, Gao Z, Ding Q, et al (2023) Uni-mol: A universal 3D molecular representation learning framework. In: The eleventh international conference on learning representations

10. Chen J, Schwaller P (2023) Molecular hypergraph neural networks

11. Zhao B, Xu W, Guan J, Zhou S (2023) Molecular property prediction based on graph structure learning. arXiv preprint arXiv:231216855

12. Wang Z, Jiang T, Wang J, Xuan Q (2024) Multi-modal representation learning for molecular property prediction: Sequence, graph, geometry

13. Zhu Y, Chen D, Du Y, et al (2024) Molecular contrastive pretraining with collaborative featurizations. Journal of Chemical Information and Modeling 64:1112–1122. https://doi.org/10.1021/acs.jcim.3c01468

14. Schaduangrat N, Anuwongcharoen N, Charoenkwan P, Shoombuatong W (2023) DeepAR: A novel deep learning-based hybrid framework for the interpretable prediction of androgen receptor antagonists. Journal of Cheminformatics 15:50. https://doi.org/10.1186/s13321-023-00721-z

15. Tilborg D van, Brinkmann H, Criscuolo E, et al (2024) Deep learning for low-data drug discovery: Hurdles and opportunities. ChemRxiv. https://doi.org/10.26434/chemrxiv-2024-w0wvl

16. McGibbon M, Shave S, Dong J, et al (2023) From intuition to AI: Evolution of small molecule representations in drug discovery. Briefings in Bioinformatics 25: https://doi.org/10.1093/bib/bbad422

17. Moriwaki H, Tian Y-S, Kawashita N, Takagi T (2018) Mordred: A molecular descriptor calculator. Journal of Cheminformatics 10: https://doi.org/10.1186/s13321-018-0258-y

18. Falcon W, The PyTorch Lightning team (2019) PyTorch Lightning

19. Weininger D (1988) SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. Journal of Chemical Information and Computer Sciences 28:31–36. https://doi.org/10.1021/ci00057a005

20. Comesana AE, Huntington TT, Scown CD, et al (2022) A systematic method for selecting molecular descriptors as features when training models for predicting physiochemical properties. Fuel 321:123836. https://doi.org/https://doi.org/10.1016/j.fuel.2022.123836

21. Wu J, Wang S, Zhou L, et al (2020) Deep-learning architecture in QSPR modeling for the prediction of energy conversion efficiency of solar cells. Industrial & Engineering Chemistry Research 59:18991–19000. https://doi.org/10.1021/acs.iecr.0c03880

22. Esaki T, Ohashi R, Watanabe R, et al (2019) Computational model to predict the fraction of unbound drug in the brain. Journal of Chemical Information and Modeling 59:3251–3261. https://doi.org/10.1021/acs.jcim.9b00180

23. Yalamanchi KK, Kommalapati S, Pal P, et al (2023) Uncertainty quantification of a deep learning fuel property prediction model. Applications in Energy and Combustion Science 16:100211. https://doi.org/https://doi.org/10.1016/j.jaecs.2023.100211

24. Wu Z, Ramsundar B, Feinberg EN, et al (2018) MoleculeNet: A benchmark for molecular machine learning

25. Ramakrishnan R, Hartmann M, Tapavicza E, Lilienfeld OA von (2015) Electronic spectra from TDDFT and machine learning in chemical space. The Journal of Chemical Physics 143: https://doi.org/10.1063/1.4928757

26. Ramakrishnan R, Dral PO, Rupp M, Lilienfeld OA von (2014) Quantum chemistry structures and properties of 134 kilo molecules. Scientific Data 1: https://doi.org/10.1038/sdata.2014.22

27. Burns JW, Spiekermann KA, Bhattacharjee H, et al (2023) Machine learning validation via rational dataset sampling with astartes. Journal of Open Source Software 8:5996. https://doi.org/10.21105/joss.05996

28. Saldana DA, Starck L, Mougin P, et al (2011) Flash point and cetane number predictions for fuel compounds using quantitative structure property relationship (QSPR) methods. Energy & Fuels 25:3900–3908. https://doi.org/10.1021/ef200795j

29. Das DD, St. John PC, McEnally CS, et al (2018) Measuring and predicting sooting tendencies of oxygenates, alkanes, alkenes, cycloalkanes, and aromatics on a unified scale. Combustion and Flame 190:349–364. https://doi.org/10.1016/j.combustflame.2017.12.005

30. Eibeck A, Nurkowski D, Menon A, et al (2021) Predicting power conversion efficiency of organic photovoltaics: Models and data analysis. ACS Omega 6:23764–23775. https://doi.org/10.1021/acsomega.1c02156

31. Bhat V, Sornberger P, Pokuri BSS, et al (2023) Electronic, redox, and optical property prediction of organic pi-conjugated molecules through a hierarchy of machine learning approaches. Chemical Science 14:203–213. https://doi.org/10.1039/d2sc04676h

32. Delaney JS (2004) ESOL: Estimating aqueous solubility directly from molecular structure. Journal of Chemical Information and Computer Sciences 44:1000–1005. https://doi.org/10.1021/ci034243x

33. Song Y, Zheng S, Niu Z, et al (2020) CMPNN README

34. Mobley DL, Guthrie JP (2014) FreeSolv: A database of experimental and calculated hydration free energies, with input files. Journal of Computer-Aided Molecular Design 28:711–720. https://doi.org/10.1007/s10822-014-9747-x

35. Lopez SA, Pyzer-Knapp EO, Simm GN, et al (2016) The harvard organic photovoltaic dataset. Scientific Data 3: https://doi.org/10.1038/sdata.2016.86

36. Burns JW, Rogers DM (2023) QuantumScents: Quantum-mechanical properties for 3.5k olfactory molecules. Journal of Chemical Information and Modeling 63:7330–7337. https://doi.org/10.1021/acs.jcim.3c01338

37. Schaduangrat N, Anuwongcharoen N, Charoenkwan P, Shoombuatong W (2023) DeepAR: A novel deep learning-based hybrid framework for the interpretable prediction of androgen receptor antagonists. Journal of Cheminformatics 15: https://doi.org/10.1186/s13321-023-00721-z

38. Notwell JH, Wood MW (2023) ADMET property prediction through combinations of molecular fingerprints

39. Riesen K, Bunke H (2008) IAM graph database repository for graph based pattern recognition and machine learning. In: Vitoria Lobo N da, Kasparis T, Roli F, et al (eds) Structural, syntactic, and statistical pattern recognition. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 287–297

40. Szymański P, Kajdanowicz T (2017) A network perspective on stratification of multi-label data. In: Luís Torgo PB, Moniz N (eds) Proceedings of the first international workshop on learning with imbalanced domains: Theory and applications. PMLR, pp 22–35

41. Kuhn M, Letunic I, Jensen LJ, Bork P (2015) The SIDER database of drugs and side effects. Nucleic Acids Research 44:D1075–D1079. https://doi.org/10.1093/nar/gkv1075

42. Broccatelli F, Carosati E, Neri A, et al (2011) A novel approach for predicting p-glycoprotein (ABCB1) inhibition using molecular interaction fields. Journal of Medicinal Chemistry 54:1740–1751. https://doi.org/10.1021/jm101421d

43. Huang K, Fu T, Gao W, et al (2021) Therapeutics data commons: Machine learning datasets and tasks for therapeutics. CoRR abs/2102.09548: https://doi.org/10.48550/arXiv.2102.09548

44. Carbonell P, Carlsson L, Faulon J-L (2013) Stereo signature molecular descriptor. Journal of Chemical Information and Modeling 53:887–897. https://doi.org/10.1021/ci300584r

45. Lundberg S, Lee S-I (2017) A unified approach to interpreting model predictions