Descriptor-Enhanced Machine Learning for Predicting Absorption, Distribution, Metabolism, Excretion, and Toxicity (ADMET) Properties in Drug Discovery

Vaishali Wangikar[1*], Laxmi Kale[1], Ritu Dudhmal[1], Aparna Kulkarni[1], Monali Bansode[1], Prajakta Jadhav[1], Suyoga Bansode[1], Swati Patil[1]

[1]Department of Computer Science and Engineering (Data Science), MIT Academy of Engineering, Alandi (D), Pune, Maharashtra, India

[1]Email: Vaishali.wangikar@gmail.com

[1]Email: laxmi.kale@mitaoe.ac.in

[1]Email: ritu.dudhmal@mitaoe.ac.in

[1]Email: aparna.kulkarni@mitaoe.ac.in

[1]Email: monali.bansode@mitaoe.ac.in

[1]Email: prajakta.jadhav@mitaoe.ac.in

[1]Email: suyoga.bansode@mitaoe.ac.in

[1]Email: swati.patil@mitaoe.ac.in

***Corresponding Author:** Vaishali Wangikar, Department of Computer Science and Engineering (Data Science), MIT Academy of Engineering, Alandi (D), Pune, Maharashtra, India | Email: Vaishali.wangikar@gmail.com

Abstract

Accurate prediction of absorption, distribution, metabolism, excretion, and toxicity (ADMET) properties is essential for reducing late-stage failures in drug discovery and guiding drug-delivery design. Although increasingly complex deep learning models have been proposed for ADMET prediction, their advantages over well-tuned machine learning methods remain unclear. This study presents a rigorous comparison of six predictive models on four representative ADMET endpoints from the PharmaBench dataset, comprising 42,209 compounds selected from a curated collection of 52,482 entries derived from 14,401 bioassays. The evaluated endpoints include lipophilicity (LogD), aqueous solubility, mutagenicity (AMES), and blood-brain barrier (BBB) penetration, covering both regression and classification tasks. The benchmark includes Random Forest, XGBoost, Support Vector Machine (SVM), a Feed-Forward Neural Network (FFNN), a CNN-GNN-RDKit descriptor fusion model, and a descriptor-augmented XGBoost that combines molecular fingerprints with sixteen RDKit physicochemical descriptors. All models were optimized using dataset-specific hyperparameter tuning, evaluated on scaffold-based train/test splits, repeated across five random seeds, and compared using paired *t*-tests.

The results show that the CNN-GNN-RDKit fusion model outperformed the fingerprint-only XGBoost on both regression tasks. However, this advantage disappeared when the same RDKit descriptors were incorporated into XGBoost. The descriptor-augmented XGBoost achieved the best or joint-best performance across all four endpoints, with $R^2 = 0.684 \pm 0.005$ for LogD, $R^2 = 0.761 \pm 0.003$ for aqueous solubility, AUROC $= 0.881 \pm 0.003$ for AMES, and AUROC $= 0.925 \pm 0.002$ for BBB prediction. It significantly outperformed the fusion model on every endpoint (paired *t*-test, all $p < 0.002$), indicating that the performance gains were primarily driven by informative physicochemical descriptors rather than the additional convolutional and graph-learning components. A further methodological finding showed that an untuned single-run evaluation incorrectly ranked the Feed-Forward Neural Network as the best regression model, whereas systematic hyperparameter tuning reversed this ranking.

These findings suggest that, for ADMET prediction, a well-tuned gradient-boosted tree enriched with molecular descriptors can outperform more complex deep learning architectures while requiring substantially lower computational cost. More broadly, reliable model comparison requires fair hyperparameter optimization, repeated evaluation, and statistical significance testing before claims of deep learning superiority can be established.

Keywords: *ADMET prediction; molecular descriptors; feature-matched baseline; gradient boosting; graph neural network; ablation; hyperparameter tuning; PharmaBench; drug delivery*

How to cite this article: Wangikar V, Kale L, Dudhmal R, Kulkarni A, Bansode M, Jadhav P, Bansode S, Patil S. Descriptor-enhanced machine learning for predicting absorption, distribution, metabolism, excretion, and toxicity (ADMET) properties in drug discovery. Int J Drug Deliv Technol. 2026;16(74s): 1158-1166. DOI: 10.25258/ijddt.16.74s.136

1. Introduction

Many drug candidates that work well against their target still fail later in development. Poor absorption, distribution, metabolism, excretion, or toxicity behaviour is a common cause. Machine learning now supports this stage of screening as a matter of routine, and AI-assisted candidates have entered industry pipelines rather than staying in the laboratory (Arnold, 2023). In drug delivery, ADMET properties act as design inputs. Lipophilicity and solubility bear on the choice of carrier, whether a lipid nanoparticle, a polymeric micelle, or an aqueous solution. Blood-brain barrier permeability and mutagenic potential help decide whether a delivery route is worth pursuing. Prediction is therefore useful early, while the delivery system is still being chosen.

A practical problem runs through the applied literature on ADMET modelling. Comparisons are often reported on one train/test split, with models left near default settings, and without any measure of run-to-run spread. Under those conditions a reported ranking may reflect a real difference between methods, or it may reflect one lucky split, one seed, or uneven tuning effort. This matters because the question a delivery scientist faces is exactly the one such a comparison can answer wrongly: which model to reach for when screening compounds.

This study is built around that problem. Six models are compared on four ADMET endpoints from PharmaBench (Niu et al., 2024). Three are established baselines: Random Forest, XGBoost, and a Support Vector Machine. One is a plain feed-forward neural network. One is a CNN-GNN-RDKit descriptor fusion model that reads the molecular graph directly rather than a fixed fingerprint alone. The last is a deliberately simple ablation: a descriptor-augmented XGBoost that adds sixteen RDKit physicochemical descriptors to the fingerprint. This ablation is the analytical key. The fusion model draws on the fingerprint, the graph, and descriptors at once, so a win over a fingerprint-only tree could come from the deep components or from the descriptors alone. Giving the same descriptors to XGBoost separates the two explanations. Three rules are applied to every model: a per-dataset hyperparameter search on a held-out validation split, retraining across five seeds, and a significance test on any claimed difference.

The main finding is that the performance advantage of the fusion model disappears once XGBoost is enhanced with the same molecular descriptors. The fusion model did beat a fingerprint-only XGBoost on both regression endpoints, which on its own would credit its convolutional and graph parts. Once XGBoost received the same sixteen descriptors, it overtook the fusion model on all four endpoints, at a small fraction of the training cost. The deep architecture bought nothing that a short list of descriptors did not supply more cheaply to a tree. A second, separate result concerns untuned comparisons. An initial single run made a plain neural network look best on regression, yet fair tuning reversed that order. Both results carry the same lesson: an apparent advantage for a complex model must be checked against a fairly tuned, feature-matched baseline before it is believed.

2. Related Work

Benchmark resources for ADMET modelling have grown in recent years. Therapeutics Data Commons gathered 66 datasets across 22 tasks, including an ADMET group of 22 endpoints that range from 475 to about 13,000 molecules each (Huang et al., 2021). MoleculeNet took a broader route and pooled 17 datasets across physical chemistry, biophysics, and physiology, with a combined count above 700,000 molecules, though individual tasks vary widely in size (Wu et al., 2018). PharmaBench was built more recently to address the small size and narrow sourcing of many earlier sets. A large language model-based system mined experimental conditions from 14,401 bioassays and merged them into 52,482 curated entries across eleven properties (Niu et al., 2024).

The PharmaBench authors benchmarked several architectures on their data. The gap between deep learning and classical methods widened on large regression datasets and narrowed, or reversed, on classification datasets such as AMES and BBB (Niu et al., 2024). Independent work agrees. A study that varied feature representations across several ADMET datasets found fixed representations generally beat learned ones, with Random Forest the most consistent performer (Kamuntavičius et al., 2025). Dedicated deep learning platforms report their strongest gains on continuous physicochemical endpoints. ADMET-AI, built on the message-passing network of Yang et al. (2019), is one example (Swanson et al., 2024). Other web platforms, including ADMETlab 3.0 and admetSAR3.0, focus on wider endpoint coverage and decision support (Fu et al., 2024; Gu et al., 2024). OptADMET adds a substructure-modification step to guide lead optimisation (Yi et al., 2024).

Another research direction focuses on adapting deep learning models to the unique challenges of ADMET prediction, including limited data and complex molecular structures. DeepDelta learns the difference in a property between two molecules rather than a single value, and reports gains over ChemProp and Random Forest on ten benchmarks posed this way (Fralish et al., 2023). A multi-task framework improved classification by sharing information across related endpoints (Zhang et al., 2025). Hybrid models that combine fingerprints with graph representations report further gains on property prediction (Liu et al.,

2025). Reviews of the field make a steady point: no single approach wins across all ADMET endpoints, and the model should match the data characteristics of each property (Cheng et al., 2013; Brown et al., 2020; Kumar et al., 2021).

While considerable attention has been given to developing new ADMET prediction models, less emphasis has been placed on ensuring fair model comparison. Gradient-boosted tree models are strong baselines for molecular descriptor data, but their performance can be underestimated when they are evaluated with limited hyperparameter tuning, a single train/test split, or fewer input features than competing deep learning models. This study addresses this issue by comparing a multi-branch CNN-GNN-RDKit fusion model with a feature-matched XGBoost model that uses the same molecular descriptors. This design isolates the contribution of the deep learning architecture from that of the input features. To ensure a fair comparison, all models were evaluated under the same hyperparameter tuning strategy, multi-seed experiments, and statistical significance testing.

3. Methodology

3.1 Dataset

Four sub-datasets from PharmaBench were used (Niu et al., 2024). The full benchmark holds eleven sub-datasets and 52,482 entries, built from 14,401 bioassays. The four here, listed in Table 1, cover both task types and represent properties that matter for delivery-system design: LogD and aqueous solubility as continuous endpoints, and AMES and BBB penetration as binary endpoints. Together they hold 42,209 compounds. The remaining seven sub-datasets cover CYP2C9, CYP2D6, and CYP3A4 inhibition, human, mouse, and rat microsomal clearance, and plasma protein binding. They are left for later work.

Sub-dataset	Task type	Total entries	Training set	Test set
LogD (lipophilicity)	Regression	13,068	10,455	2,613
Aqueous solubility	Regression	11,701	9,361	2,340
AMES (mutagenicity)	Classification	9,139	7,312	1,827
BBB penetration	Classification	8,301	6,641	1,660

Table 1. The four PharmaBench sub-datasets used here, with training and test sizes from the scaffold split supplied with the benchmark. Together they hold 42,209 of the 52,482 entries in the full set.

3.2 Molecular Featurisation

Each SMILES string, taken as standardised in PharmaBench, was parsed with RDKit (Landrum et al., 2024) and encoded as a Morgan fingerprint of radius 2 and 1,024 bits (Rogers & Hahn, 2010). This fingerprint is the only input to the four fingerprint-based models: Random Forest, XGBoost, the SVM, and the plain neural network. Those four therefore see an identical feature matrix, so any difference among them reflects the algorithm and its settings rather than the features. The fusion model is a deliberate exception. It also reads the molecular graph and a set of RDKit descriptors, as described in Section 3.5, so that the value of those richer inputs can be measured against the fingerprint-only baselines. Fewer than 1% of entries could not be parsed and were dropped before splitting.

3.3 Models and Hyperparameter Search

Six models were compared: Random Forest (Breiman, 2001), XGBoost (Chen & Guestrin, 2016), a Support Vector Machine with a radial basis function kernel, a feed-forward neural network, a CNN-GNN-RDKit fusion model (Section 3.5), and a descriptor-augmented XGBoost that adds sixteen RDKit descriptors to the fingerprint. None was left at default settings. For each model and each dataset, a small grid was searched on a held-out validation split, taken as 10% of the training set and never from the test set. The configuration with the best validation score, R^2 for regression or AUROC for classification, was carried forward. The grids appear in Table 2. They are compact by design, since the aim is a fair and equal tuning budget for every model rather than each model's global best; an exhaustive search is noted as a limitation in Section 6.

Model	Hyperparameter grid searched (per dataset, on validation split)
Random Forest	n_estimators {200, 300}; max_depth {15, 20, 25}; max_features {sqrt, 0.1}; max_samples 0.7
XGBoost	n_estimators {300, 400}; max_depth {4, 6, 8}; learning_rate {0.03, 0.05, 0.1}; subsample 0.8; colsample_bytree 0.8
SVM	C {1, 10, 100}; gamma {scale, auto}; RBF kernel; trained on a random subsample of up to 4,000 compounds
FFNN	hidden layers {[256,64], [512,128,32]}; learning_rate {5×10^{-4}, 1×10^{-3}}; dropout {0.2, 0.3}; weight_decay {1×10^{-5}, 1×10^{-4}}; Adam; batch 128; up to 80 epochs with early stopping
Fusion	GNN hidden {64, 128}; CNN channels {16, 32}; fusion hidden {64, 128}; dropout {0.2, 0.3}; learning_rate {5×10^{-4}, 1×10^{-3}}; Adam; batch 128; up to 60 epochs with early stopping
XGBoost+desc	same grid as XGBoost, applied to the fingerprint joined with 16 standardised RDKit descriptors

Table 2. Hyperparameter grids searched for each model. The best configuration per dataset, chosen on the validation split, was used for the final multi-seed evaluation.

Random Forest and XGBoost were trained on the raw binary fingerprints, which tree methods do not need scaled. The SVM was trained on standardised fingerprints, with mean and variance fitted on the training set only. Because kernel methods scale poorly, the SVM used a random subsample of up to 4,000 training compounds, a common step at this size. The neural network, described in Section 3.4, was also trained on standardised fingerprints.

3.4 Feed-Forward Neural Network Implementation and Validation

The feed-forward network took a 1,024-unit input, then two or three hidden layers whose width and depth were tuned (Table 2), each with a ReLU activation and dropout (Srivastava et al., 2014), and a single output unit. For regression that unit gave a continuous value trained against mean squared error. For classification it gave a logit trained against binary cross-entropy with an internal sigmoid. One architecture thus served both task types, with only the final loss changed. Training used the Adam optimiser (Kingma & Ba, 2015) with early stopping on the validation split.

To rule out a hidden implementation error, the network was built twice. The first version used PyTorch (Paszke et al., 2019) and its automatic differentiation. The second was written in NumPy, with the forward pass and the backpropagation pass coded by hand, following the classical formulation of Rumelhart, Hinton, and Williams (1986), and with a hand-coded Adam step. This is not a second model. The two share one architecture and should behave alike apart from floating-point precision and random seeding. Across all four datasets the two agreed to within their seed-to-seed spread. That agreement indicates the gradients were coded correctly. All network results in Section 4 come from the PyTorch version; the two are compared in Section 4.4.

3.5 CNN-GNN-RDKit Descriptor Fusion Model

The fusion model reads each molecule in three ways and combines them, so it is not limited to a single fixed fingerprint. It has three parallel branches. The first is a one-dimensional convolutional branch. It treats the 1,024-bit fingerprint as a sequence and applies two convolutional layers of kernel size 7 and stride 2, then adaptive max-pooling, so it can pick up local patterns of co-occurring bits. The second is a graph branch. It works on the molecular graph directly, where each atom carries a 35-dimensional feature vector (element, degree, hydrogen count, formal charge, hybridisation, aromaticity, and ring membership) and each bond forms an edge. Two graph convolutional layers (Kipf & Welling, 2017) and global mean pooling produce a fixed-length embedding of the connectivity. The third is a descriptor branch, a small network over sixteen RDKit descriptors such as molecular weight, calculated logP, topological polar surface area, hydrogen-bond donor and acceptor counts, rotatable-bond count, aromatic-ring count, and fraction of sp3 carbons. The three branch outputs are joined and passed through a dense head with one output unit. Training used mean squared error for regression or binary cross-entropy for classification, with the Adam optimiser (Kingma & Ba, 2015) and early stopping. The graph branch used PyTorch Geometric (Fey & Lenssen, 2019). The design assumes that fingerprints, learned graph embeddings, and descriptors carry partly different information, so a model that reads all three should gain where a single fixed representation falls short.

The sixth model tests that assumption head-on. It is an ablation, not a new architecture: an ordinary XGBoost trained on the fingerprint with the same sixteen descriptors added as extra columns, standardised on the training set, and tuned over the same grid as the plain XGBoost. If the fusion model's advantage over a fingerprint-only tree comes from its convolutional or graph branches, this tree should stay behind the fusion model. If the advantage comes from the descriptors, this tree should match or beat it. That single comparison is the analytical centre of the study, and its result appears in Sections 4.1 to 4.3.

3.6 Train/Test Splitting, Multi-Seed Protocol, and Evaluation

All four sub-datasets used the scaffold-based split supplied with PharmaBench. Compounds are grouped by Bemis-Murcko scaffold (Bemis & Murcko, 1996) before splitting, so related structures do not fall on both sides. A scaffold split is a harder test than a random split, because it penalises a model that leans on memorising near-duplicate structures. Scores under a scaffold split should therefore not be compared directly with random-split scores from elsewhere.

After the best configuration for a model and dataset was fixed, it was retrained from scratch five times with different seeds, affecting weight initialisation, bootstrap sampling, dropout, and any subsampling. Each run was scored on the held-out test set. All results in Section 4 are the mean ± standard deviation of those five runs, so the spread across runs is visible rather than hidden. Regression used mean absolute error (MAE), root mean squared error (RMSE), and the coefficient of determination (R^2). Classification used area under the ROC curve (AUROC), area under the precision-recall curve (AUPRC), accuracy, and F1. Any claim that one model beats another rests on a paired t-test across the five seeds (Section 4.3).

3.7 Implementation

All work was done in Python. RDKit handled featurisation. Scikit-learn (Pedregosa et al., 2011) supplied Random Forest, the SVM, and the metrics. XGBoost supplied gradient boosting. PyTorch (Paszke et al., 2019) ran the neural network, and PyTorch Geometric (Fey & Lenssen, 2019) supplied the graph layers of the fusion model. SciPy (Virtanen et al., 2020) ran the paired t-tests. The hand-coded reproducibility check used NumPy only. Code and the exact settings for every model are available from the author on request. Figure 1 sets out the full pipeline.

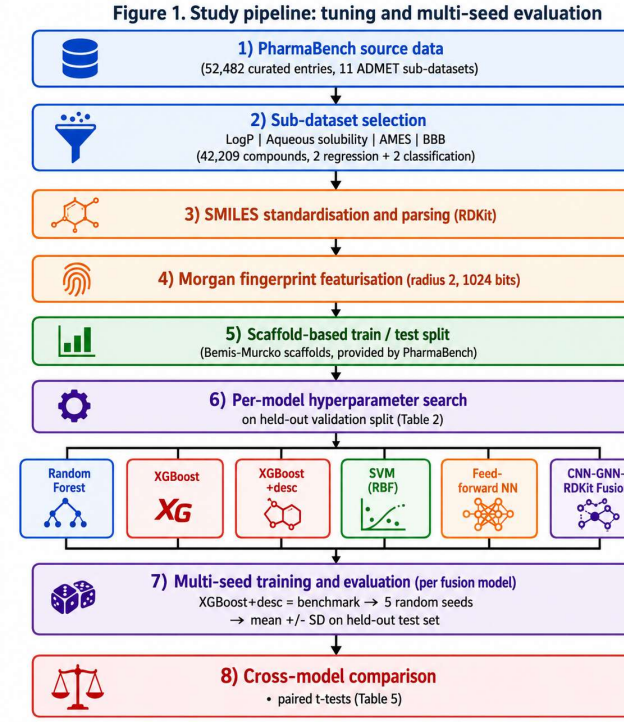

Figure 1. Study pipeline. One featurisation and scaffold-splitting stage feeds a per-model hyperparameter search on a validation split, then five-seed retraining of the best configuration, then evaluation on the held-out test set. The descriptor-augmented XGBoost (XGBoost+desc) adds sixteen RDKit descriptors to the fingerprint as a feature-matched ablation of the fusion model.

4. Results

4.1 Regression

Table 3 reports tuned, five-seed regression performance on LogD and aqueous solubility. Two comparisons matter. First, among the fingerprint-only models the fusion model was strongest. It beat a fingerprint-only XGBoost on both endpoints, raising LogD R^2 from 0.587 to 0.638 and solubility R^2 from 0.533 to 0.720. Taken alone, that would credit its convolutional and graph branches. Second, the descriptor-augmented XGBoost was better still. It reached $R^2 = 0.684 \pm 0.005$ on LogD and 0.761 ± 0.003 on solubility, above the fusion model on both. The same descriptors that the fusion model read through a branch were used more effectively by a tree. Standard deviations are small next to the gaps between models, so the order is stable across seeds.

Model	LogD MAE	LogD RMSE	LogD R^2	Sol. MAE	Sol. RMSE	Sol. R^2
Random Forest	0.817±0.001	1.085±0.001	0.413±0.001	1.120±0.003	1.496±0.003	0.390±0.003
XGBoost	0.672±0.003	0.910±0.003	0.587±0.003	0.971±0.014	1.309±0.013	0.533±0.009
SVM	0.768±0.004	1.037±0.008	0.463±0.008	1.084±0.014	1.448±0.015	0.429±0.012
FFNN	0.697±0.004	0.954±0.007	0.546±0.007	0.989±0.010	1.331±0.012	0.518±0.008
Fusion	0.638±0.006	0.852±0.007	0.638±0.006	0.747±0.012	1.014±0.018	0.720±0.010
XGBoost+desc	0.572±0.005	0.796±0.006	0.684±0.005	0.673±0.005	0.936±0.007	0.761±0.003

Table 3. Regression performance on the LogD and aqueous solubility test sets (tuned models, mean ± SD over five seeds, scaffold split). Lower MAE and RMSE are better; higher R^2 is better. The descriptor-augmented XGBoost row is shaded as the best model on both endpoints, above the fusion model despite its simpler form.

Figure 2 shows the same regression results as bars, with error bars for the standard deviation across seeds. On R^2, the descriptor-augmented XGBoost is the tallest bar for both endpoints, at 0.684 on LogD and 0.761 on solubility. The fusion model is next, at 0.638 and 0.720. The three fingerprint-only baselines follow: plain XGBoost (0.587 and 0.533), the neural network (0.546 and 0.518), the SVM (0.463 and 0.429), and Random Forest last (0.413 and 0.390). The gap is widest on solubility, where the descriptor-augmented tree stands well clear of every other model.

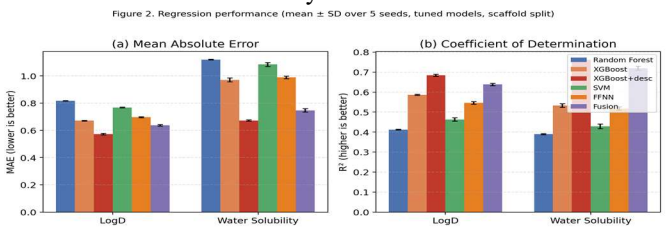

Figure 2. Regression performance on the LogD and aqueous solubility test sets (mean ± SD over five seeds). Panel (a) shows mean absolute error, where

lower is better; panel (b) shows R², where higher is better. The descriptor-augmented XGBoost (XGBoost+desc) leads on both panels for both endpoints, above the fusion model.

4.2 Classification

Table 4 reports tuned, five-seed classification performance on AMES and BBB. The descriptor-augmented tree again came out on top. On AMES it had the best AUROC, 0.881 ± 0.003, above plain XGBoost (0.871) and Random Forest (0.868), with the fusion model well back at 0.844. On BBB it tied Random Forest for the best AUROC, 0.925, above plain XGBoost (0.919) and the fusion model (0.895). Here the fusion model did not even lead the fingerprint-only trees. On these binary endpoints its extra architecture was not just redundant but a slight cost.

AMES (mutagenicity):

Model	AUROC	AUPRC	Accuracy	F1
Random Forest	0.868± 0.001	0.886± 0.001	0.789± 0.003	0.800± 0.003
XGBoost	0.871± 0.002	0.888± 0.002	0.791± 0.007	0.806± 0.008
SVM	0.806± 0.005	0.832± 0.008	0.742± 0.003	0.751± 0.003
FFNN	0.829± 0.007	0.844± 0.005	0.748± 0.010	0.763± 0.011
Fusion	0.844± 0.004	0.858± 0.005	0.761± 0.010	0.779± 0.016
XGBoost+desc	0.881± 0.003	0.895± 0.004	0.793± 0.007	0.807± 0.007

BBB penetration:

Model	AUROC	AUPRC	Accuracy	F1
Random Forest	0.925± 0.000	0.960± 0.001	0.845± 0.000	0.894± 0.000
XGBoost	0.919± 0.002	0.957± 0.002	0.851± 0.003	0.892± 0.002
SVM	0.899± 0.002	0.939± 0.001	0.840± 0.006	0.888± 0.003
FFNN	0.903± 0.007	0.946± 0.006	0.837± 0.005	0.880± 0.004
Fusion	0.895± 0.005	0.941± 0.005	0.839± 0.004	0.883± 0.002
XGBoost+desc	0.925± 0.002	0.960± 0.001	0.860± 0.002	0.899± 0.002

Table 4. Classification performance on the AMES and BBB test sets (tuned models, mean ± SD over five seeds, scaffold split). Higher is better on all four metrics. The descriptor-augmented XGBoost row is shaded as the best or joint-best model on both endpoints.

Figure 3 shows the AMES and BBB results as bars for AUROC and F1. The pattern differs from the regression panels. On AUROC the descriptor-augmented XGBoost is the tallest or joint-tallest bar on both endpoints, at 0.881 on AMES and 0.925 on BBB. The fusion model, shown in purple, sits below the tree ensembles rather than above them, at 0.844 on AMES and 0.895 on BBB. The F1 panel shows the same order. On these binary endpoints the deep model does not lead.

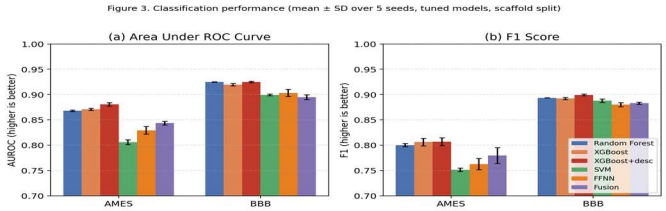

Figure 3. Classification performance on the AMES and BBB test sets (mean ± SD over five seeds). Panel (a) shows AUROC; panel (b) shows F1. Higher is better on both. The descriptor-augmented XGBoost (XGBoost+desc) is the best or joint-best model on both endpoints, while the fusion model trails the fingerprint-only trees.

4.3 Statistical Significance

The key comparison was a paired t-test between the descriptor-augmented XGBoost and the CNN-GNN-RDKit fusion model across five random seeds using the primary evaluation metric for each endpoint. Table 5 gives the result. The descriptor-augmented tree beat the fusion model on all four endpoints: by $\Delta R^2 = 0.046$ on LogD ($p = 0.0002$) and 0.042 on solubility ($p = 0.0015$), and by ΔAUROC = 0.037 on AMES ($p = 0.0001$) and 0.030 on BBB ($p = 0.0004$). Every difference is significant at $p < 0.002$. The two models differ only in form: one is a deep three-branch network reading fingerprint, graph, and descriptors, the other a tree reading fingerprint and the same descriptors. The tree wins throughout. The convolutional and graph branches therefore added nothing beyond handing the descriptors to a tree. The earlier point is thus explained. The fusion model beat a fingerprint-only XGBoost on regression because of the descriptors it read, not because of its deep architecture.

Endpoint	Primary metric	XGBoost+desc − Fusion	t (4 df)	p-value
LogD	R^2	+0.046	13.85	0.0002
Water Solubility	R^2	+0.042	7.75	0.0015
AMES	AUROC	+0.037	14.90	0.0001
BBB	AUROC	+0.030	10.72	0.0004

Table 5. Paired t-tests over five seeds, comparing the descriptor-augmented XGBoost with the fusion model

on each endpoint's primary metric. A positive difference favours the tree. The tree wins on all four endpoints (all $p < 0.002$), so the fusion model's convolutional and graph branches add no value beyond the descriptors.

4.4 Neural Network Reproducibility Check

The PyTorch and NumPy implementations of the Feed-Forward Neural Network produced consistent results across all four datasets, with differences remaining within the expected variation across random seeds. This agreement confirms the correctness of the implementation, and a single set of FFNN results is therefore reported throughout Section 4

.4.5 The Effect of Tuning on the Baselines

A second result concerns the effect of the protocol on the baselines. An initial untuned run, with each model at common default settings and a single training pass, made the plain neural network look like the best regression model among the fingerprint-based models. It beat XGBoost on LogD, 0.573 against 0.531, and on solubility, 0.504 against 0.485. On its own, that would suggest a neural network is the model of choice for continuous endpoints. These untuned figures and their tuned counterparts are set side by side in Table 6.

The tuned, five-seed results tell a different story. Once XGBoost received a fair search, its regression R^2 rose past the neural network on both endpoints, and the earlier order reversed. The network stayed respectable, but its apparent lead was an artefact of unequal tuning rather than a real gain. The same caution applies to the fusion model, whose apparent regression lead over a fingerprint-only tree fell away once the tree received matched features (Section 4.3). Figure 4 places the primary metric for all six models across all four endpoints in one view, and the descriptor-augmented XGBoost is the tallest or joint-tallest bar in every group.

Endpoint	XGBoost untuned	XGBoost tuned	FFNN untuned	FFNN tuned
LogD (R^2)	0.531	0.587	0.573	0.546
Solubility (R^2)	0.485	0.533	0.504	0.518

Table 6. Regression R^2 for XGBoost and the plain neural network before and after tuning. Untuned figures come from a single default-setting run; tuned figures are the five-seed means from Table 3. Untuned, the network leads on both endpoints; after a fair search, XGBoost overtakes it, which shows how an untuned comparison can invert a ranking.

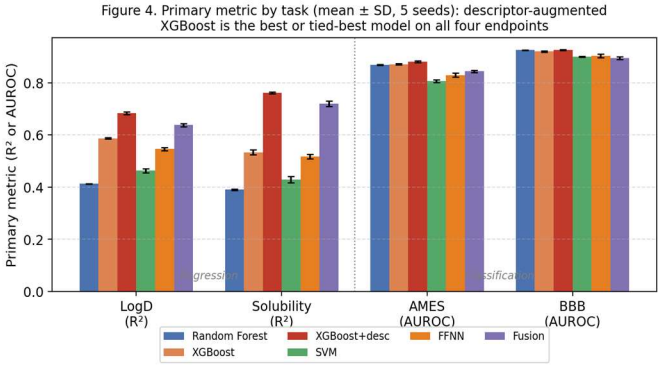

Figure 4. Primary metric across all four endpoints (R^2 for regression, AUROC for classification; mean ± SD over five seeds). The descriptor-augmented XGBoost (XGBoost+desc) is the best or joint-best model on every endpoint, above the fusion model throughout.

5. Discussion

The results provide a clear conclusion. The CNN-GNN-RDKit fusion model outperformed the fingerprint-only XGBoost on the regression tasks, suggesting an advantage for the deep architecture. However, this advantage disappeared when the same sixteen RDKit descriptors were added to XGBoost. The descriptor-augmented XGBoost achieved the best performance across all four endpoints (Table 5), showing that the improvement was driven primarily by the descriptors rather than the convolutional or graph-learning components. On the classification tasks, the fusion model did not outperform even the fingerprint-only tree models. These findings are consistent with previous studies demonstrating the strong performance of gradient-boosted trees on molecular descriptor data (Kamuntavičius et al., 2025; Niu et al., 2024).

The largest improvements were observed for lipophilicity and solubility, where descriptors such as LogP, topological polar surface area, and hydrogen-bond counts provide highly informative molecular features. Gradient-boosted trees effectively utilize these descriptors, explaining their superior performance. These findings suggest that, for these datasets, informative molecular descriptors contribute more to prediction accuracy than increased model complexity.

The descriptor-augmented XGBoost offered the best balance of accuracy, efficiency, and simplicity. Its results show that performance gains can be driven more by informative molecular descriptors than by increased model complexity, emphasizing the need for feature-matched baselines when evaluating deep learning models.

6. Limitations

The proposed benchmarking framework provides a strong foundation for future ADMET prediction research. The current evaluation used a fixed

molecular representation, a single fusion architecture, four representative PharmaBench datasets, and a balanced hyperparameter search to ensure fair model comparison.

Future work will extend this framework by incorporating richer molecular representations, additional ADMET endpoints, larger hyperparameter search spaces, and more repeated experiments to further strengthen the robustness and generalizability of the findings. Evaluating the SVM on larger training sets as computational resources improve will also provide a more comprehensive comparison.

7. Conclusion

Six models were compared on four PharmaBench ADMET datasets using the same evaluation protocol, including hyperparameter tuning, five independent runs, and statistical significance testing. The CNN-GNN-RDKit fusion model outperformed the fingerprint-only XGBoost on the regression tasks. However, when the same sixteen RDKit descriptors were added to XGBoost, it achieved the best performance across all four endpoints ($p < 0.002$) with much lower computational cost. This shows that the performance improvement came mainly from the molecular descriptors rather than the deep learning architecture. The study also highlights that fair hyperparameter tuning can change model rankings, emphasizing the need for rigorous benchmarking. Overall, a well-tuned XGBoost model with molecular descriptors provides an accurate, efficient, and practical solution for ADMET prediction. Future work will extend the evaluation to the remaining PharmaBench datasets and additional molecular representation methods.

Data Availability

The PharmaBench dataset is openly available at https://github.com/mindrank-ai/PharmaBench and https://figshare.com/articles/dataset/PharmaBench_Enhancing_ADMET_benchmarks_with_large_language_models/25559469, under the terms set out in Niu et al. (2024).

References

1. Arnold, C. (2023). Inside the nascent industry of AI-designed drugs. Nature Medicine, 29, 1292-1295.

2. Bemis, G. W., & Murcko, M. A. (1996). The properties of known drugs. 1. Molecular frameworks. Journal of Medicinal Chemistry, 39(15), 2887-2893.

3. Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.

4. Brown, N., Ertl, P., Lewis, R., Luksch, T., Reker, D., & Schneider, N. (2020). Artificial intelligence in chemistry and drug design. Journal of Computer-Aided Molecular Design, 34, 709-715.

5. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).

6. Cheng, F., Li, W., Liu, G., & Tang, Y. (2013). In silico ADMET prediction: Recent advances, current challenges and future trends. Current Topics in Medicinal Chemistry, 13(11), 1273-1289.

7. Fey, M., & Lenssen, J. E. (2019). Fast graph representation learning with PyTorch Geometric. In ICLR Workshop on Representation Learning on Graphs and Manifolds.

8. Fralish, Z., Chen, A., Skaluba, P., & Reker, D. (2023). DeepDelta: Predicting ADMET improvements of molecular derivatives with deep learning. Journal of Cheminformatics, 15(1), 101.

9. Fu, L., Shi, S., Yi, J., Wang, N., He, Y., Wu, Z., Peng, J., Deng, Y., Wang, W., Wu, C., Lyu, A., Zeng, X., Zhao, W., Hou, T., & Cao, D. (2024). ADMETlab 3.0: An updated comprehensive online ADMET prediction platform. Nucleic Acids Research, 52(W1), W422-W431.

10. Gu, Y., Yu, Z., Wang, Y., et al. (2024). admetSAR3.0: A comprehensive platform for exploration, prediction and optimization of chemical ADMET properties. Nucleic Acids Research, 52(W1), W432-W438.

11. Huang, K., Fu, T., Gao, W., Zhao, Y., Roohani, Y., Leskovec, J., Coley, C. W., Xiao, C., Sun, J., & Zitnik, M. (2021). Therapeutics Data Commons: Machine learning datasets and tasks for drug discovery and development. arXiv preprint arXiv:2102.09548.

12. Kamuntavičius, G., Paquet, T., Bastas, O., Šalkauskas, D., Prat, A., Abdel Aty, H., Pabrinkis, A., Norvaišas, P., & Tal, R. (2025). Benchmarking ML in ADMET predictions: The practical impact of feature representations in ligand-based models. Journal of Cheminformatics, 17, 108.

13. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR).

14. Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In Proceedings of the 5th International Conference on Learning Representations (ICLR).

15. Kumar, A., Kini, S. G., & Rathi, E. (2021). A recent appraisal of artificial intelligence and in silico ADMET prediction in the early stages of drug discovery. Mini-Reviews in Medicinal Chemistry, 21(18), 2788-2800.

16. Landrum, G., et al. (2024). RDKit: Open-source cheminformatics software (Version 2024.03). https://www.rdkit.org

17. Liu, S., Chen, M., Yao, X., & Liu, H. (2025). Fingerprint-enhanced hierarchical molecular graph neural networks for property prediction. Journal of Pharmaceutical Analysis, 15, 101242.

18. Niu, Z., Xiao, X., Wu, W., Cai, Q., Jiang, Y., Jin, W., Wang, M., Yang, G., Kong, L., Jin, X., Yang, G., & Chen, H. (2024). PharmaBench: Enhancing ADMET benchmarks with large language models. Scientific Data, 11, 985.
19. Paszke, A., Gross, S., Massa, F., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems, 32.
20. Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
21. Rogers, D., & Hahn, M. (2010). Extended-connectivity fingerprints. Journal of Chemical Information and Modeling, 50(5), 742-754.
22. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533-536.
23. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929-1958.
24. Swanson, K., Walther, P., Leitz, J., Mukherjee, S., Wu, J. C., Shivnaraine, R. V., & Zou, J. (2024). ADMET-AI: A machine learning ADMET platform for evaluation of large-scale chemical libraries. Bioinformatics, 40(7), btae416.
25. Virtanen, P., Gommers, R., Oliphant, T. E., et al. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. Nature Methods, 17, 261-272.
26. Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., & Pande, V. (2018). MoleculeNet: A benchmark for molecular machine learning. Chemical Science, 9(2), 513-530.
27. Yang, K., Swanson, K., Jin, W., et al. (2019). Analyzing learned molecular representations for property prediction. Journal of Chemical Information and Modeling, 59(8), 3370-3388.
28. Yi, J., Shi, S., Fu, L., et al. (2024). OptADMET: A web-based tool for substructure modifications to improve ADMET properties of lead compounds. Nature Protocols, 19, 1105-1121.
29. Zhang, H., Ma, F., Su, J., Yang, X., Wang, L., Ye, W.-C., & Liu, L. (2025). Quantum-enhanced multi-task learning with learnable weighting for pharmacokinetic and toxicity prediction. arXiv preprint arXiv:2509.04601.