X.C01 Machine Learning for Molecular Engineering Spring 2024 Problem Set #6

David David
MIT ChemE
davidgoh@mit.edu

Abstract

We benchmark regression models predicting partition coefficients from solvent and solute molecular representations. Chemprop D-MPNN outperforms fingerprint-based XGBoost and FNN. Featurization using RDKit descriptors and hyperparameter tuning using Optuna improves model accuracy. We applied an ensemble of tuned MPNNs on 20 folds and submit the mean predictions for Kaggle.

1. Data Preprocessing

The data consists of solvent and solute SMILES, the partition coefficient ($\log K$), and the solvation free energy (ΔG). The regression models incorporate both local and global features of the molecule. Local representations are generated using molecular graphs or through fingerprints. We first discuss the global features, which are generated using RDKit descriptors of properties that are relevant to solubility (Landrum et al., 2024).

The RDKit generated descriptors are as follows: the Labute accessible surface area (`LabuteASA`) describes the van der Waals surface of a solute accessibly by the solvent; the topological polar surface area (`TPSA`) describes the polarity of a molecule; the octanol-water partition coefficient (`MolLogP`) describes the lipophilicity of a molecule; the molar refractivity (`MolMR`) describes the polarizability of a molecule; the proportion of aromatic atoms (`AromProp`), which affects its solubility and lipophilicity; the number of hydrogen bond acceptors (`NumHAcceptors`) and donors (`NumHDonors`), which affects the strength of solvent-solute interactions; the number of rotatable bonds (`NumRotatableBonds`), which affects the packing of the solute (Labute, 2000; Ertl et al., 2000; Wildman & Crippen, 1999; Ritchie et al., 2011). As a first approximation, we assume similar features are descriptive of the properties of the solvent in the two-component mixture.

To explore the relevance of these descriptors, we visualize the structure of the dataset using the TSNE (t-Stochastic Neighbor Embedding) of the concatenated solvent and solute Morgan fingerprints, labelled with normalized values of $\log K$ and the descriptors (van der Maaten & Hinton, 2008). The target partition coefficient $\log K$ is largest at the center (Fig. 3A). The solute descriptors `LabuteASA`, `TPSA`, `MolLogP`, `MolMR`, and `NumRotatableBonds` show patterns that visually correlate with $\log K$, indicating that they are viable features (Fig. 3B). Trends in other descriptors are unclear. Using simple linear regression, all features have $p < 10^{-5}$ except for solvent `NumHDonors`, solvent `MolLogP`, and solute `NumHAcceptors`. We first train and tune our models using all descriptors before masking the features on the best performing model to evaluate their importance. Descriptors are normalized.

2. Feature Representation

Local representations of the molecule's structure are generated through message passing on the directed bonds of the molecule's graph for the Directed Message Passing Neural Network (D-MPNN) (Yang et al., 2019) and through circular fingerprints (e.g., Morgan) for the XGBoost (eXtreme Gradient Boosting) model, FNN (Feedforward Neural Network), and Random Forest model (Rogers & Hahn, 2010).

Fingerprints are a standard way of representing structural information of small molecules in medicinal chemistry. They are inexpensive to calculate and encode various functional groups with minimal collision. However, they can be limited in describing global features of the molecules (Capecchi et al., 2020), which may be relevant for physicochemical properties like solubility. In contrast, message passing on the molecular graph allows a model to learn a representation that incorporates both local and global information. With sufficient data and tuning, the learned representation can outperform fingerprints. Hence, we benchmark models using both representations in the context of our dataset.

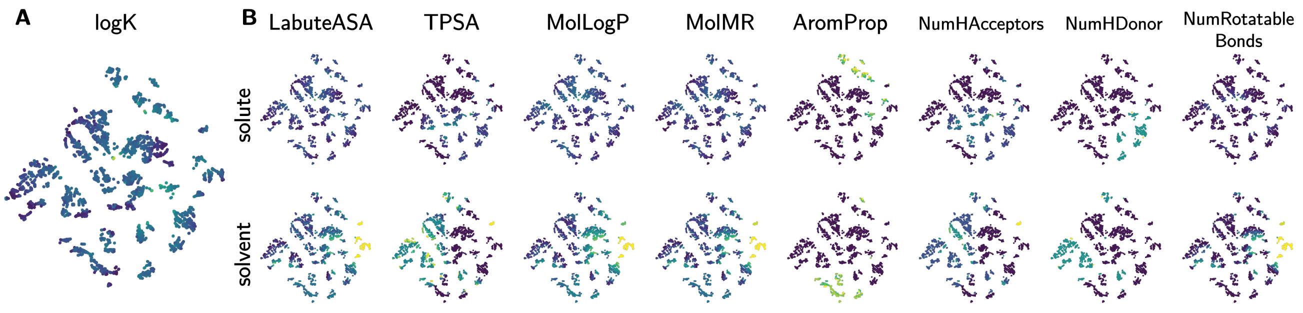

Figure 1. A) TSNE of Morgan fingerprints with solvent-solute partition coefficient labels B) the same TSNE labelled with Labute accessible surface area, topological polar surface area, Wildman-Crippen lipophilicity (octanol-water partition coefficient) and molar refractivity, the proportion of aromatic atoms, the number of hydrogen bond acceptors and donors, and the number of rotatable bonds.

Table 1. 5-fold cross validation R2 scores for different fingerprint methods on the tuned XGBoost model

Trial	Fingerprint	R2 Score
0	Morgan	0.9795
1	RDKit	0.9774
2	Avalon	0.9796
3	TopologicalTorsion	0.9780

We implement the D-MPNN using Chemprop (Heid et al., 2024). We use a separate message passing layer to encode the solvent and solute molecules into a 300-element vector each, before concatenating them with the global descriptors into a 616-element vector. The message passing layer encodes the graphs using learned weights, which are shared across all molecules. This representation vector is the input into a FNN, which we describe in Sec. 3. Incorporating solvent descriptors requires a small modification to Chemprop's `MulticomponentTrainingBatch`, which we implement through a custom class.

We use fingerprints (2048 bits) concatenated with global descriptors to train the XGBoost model and FNN. Using our tuned model (Sec. 3), we tested the models' accuracy using RDKit, Morgan, Topological Torsion, and Avalon fingerprints using 5-fold cross validation (Table 1) (O'Boyle & Sayle, 2016). Avalon and Morgan fingerprints have similar performances that outperform the rest. For simplicity, we use the more common Morgan fingerprints.

3. Model Architecture

We train four different models: 1) a D-MPNN with dropout, and acting on Morgan fingerprints, 2) an XG-

Table 2. Model hyperparameters optimized using Optuna

Model	Hyperparameters
D-MPNN	`hidden_dim, n_layers, max_lr` `dropout, depth, max_epochs`
XGBoost	`n_estimators, learning_rate` `max_depth, subsample` `colsample_bytree, min_child_weight`
Random Forest	`n_estimators, max_depth,` `min_samples_split` `min_samples_leaf` `max_features`
FNN	`hidden_dim, n_layers` `dropout, lr, max_epochs`

Boost regressor, 3) a Random Forest regressor, and 4) a FFN with dropout. All models use RDKit descriptions of global molecule properties (Sec. 1). We use Chemprop and PyTorch Lightning for the D-MPNN, the XGBoost library, scikit-learn for the Random Forest regressor, and PyTorch for the FFN (Heid et al., 2024; Falcon & The PyTorch Lightning team, 2019; Chen & Guestrin, 2016; Pedregosa et al., 2011; Paszke et al., 2019). Specific details of the model architecture are determined by hyperparameter tuning.

We use neural networks because they are non-linear universal approximators that can learn the complicated landscape of physicochemical interactions between the solute and solvent. The model needs to combine information between solute and solvent representations, which it achieves through the network architecture and non-linear transformations. Neural networks also work well with high dimensional information, which we generate through our feature representations.

Tree-based methods like XGBoost and random forest are less expensive to train and tune than neural networks, and can perform well in medium-sized dataset regimes and tabular data, which is representative of our training data. Comparing the two models, XGBoost generally performs better with high-dimensional sparse datasets, which is the case for Morgan fingerprints.

Given that we are comparing different architectures, we tune model hyperparameters before evaluating performance to ensure a fair trial. We use Optuna to optimize various hyperparameter pertinent to each model (Table 2) (Akiba et al., 2019). Optuna uses Bayesian optimization to search for a set of hyperparameters that either minimizes the mean square error or maximize the R-squared (prediction accuracy) on the test set. Bayesian optimization uses the performance of past evaluations to construct a probability distribution of the model's cost function, and uses this to suggest new hyperparameters predicted to improve performance, iteratively. Across many iterations, we optimize the choice of hyperparameters. For a larger system requiring distributed optimization, we could use Ray Tune instead (Liaw et al., 2018).

Training the D-MPNN and FNN is considerably more expensive than training the XGBoost and Random Forest regressors, so we tune these two classes of models differently. We tune the neural networks with pruning, which automatically stops training unpromising hyperparameters based on validation set performance. We use the Hyperband algoritm (`HyperbandPruner`) with some `patience` (`PatientPruner`) (Li et al., 2018). Patience is needed because performance on the validation set can be unstable between epochs. The pruner can be "aggressive", so we tune for 200 iterations. In contrast, the XGBoost and Random Forest regressors are less expensive to train, so we tune them with 5-fold cross-validation and without pruning. Empirically, model performance does not improve after 100 iterations of tuning.

For the D-MPNN and FNN, `hidden_dim` is the width of the hidden layer, `n_layers` is the number of hidden layers, `dropout` is the dropout rate, `max_epochs` is the number of epochs the model is trained for, and `lr` is the learning rate. The depth and width of the hidden layers determine model flexibility, which needs to be optimized to improve accuracy without overfitting. The number of epochs has to be optimized for similar reasons. Empirically, the D-MPNN is more robust to overfitting than the FNN: the validation performance of the D-MPNN plateaus after some number of epochs, whereas that of the FNN worsens. Dropout can be theoretically thought of as L2 regularization (Wager et al., 2013), and we implement it to improve model generalization. For the D-MPNN, `depth` corresponds

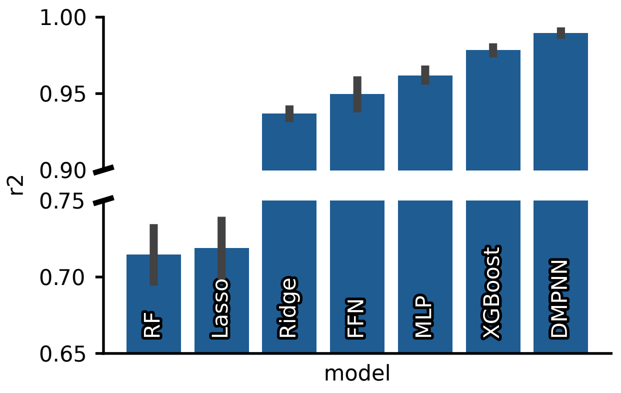

Figure 2. Test set R-squared of the Chemprop D-MPNN with various RDKit descriptor feature masks using 5-fold cross-validation.

to the depth of the message passing layer. Increasing the depth increases the amount of global information in the representation vector, but averages or smooths out the vector, so it needs to be optimized. We optimize the fixed learning rate for the FFN (`lr`) and the maximum learning rate (`max_lr`) of the Noam scheduling scheme for the D-MPNN (Vaswani et al., 2023).

For the tree-based methods, `n_estimators` is the number of trees in the ensemble and `max_depth` is the depth of the trees, which affects the model's accuracy and risk of overfitting. All three of `min_child_weight`, `min_samples_split`, and `min_child_weight` limits the tree depth, and we tune them for similar reasons. The `subsample` parameter in XGBoost corresponds to the random training split used by each tree in the ensemble. Because XGBoost is a gradient-based method, we also need to tune the learning rate.

4. Model Evaluation and Selection

In the present section, we benchmark model performance using 5-fold cross validation. In addition to our tuned D-MPNN, FNN, XGBoost, and Random Forest models, we include from scikit-learn linear regression, Lasso regression, Ridge regression, and a MLP acting on Morgan fingerprints (Fig. 2).

The performance of the models reflects the high-dimensional and sparse nature of circular fingerprints. Linear regression performs worse than a horizontal line (negative r-squared, omitted) because there are many variables (bits) and they are sparse and similarly valued. Regularization improves the linear model's performance. Ridge regression (L2 norm) performs better than Lasso (L1 norm) because many variables are important and they are similarly valued (0 or 1). For the tree-based methods, XGBoost significantly outperforms Random Forest. We expect XGBoost to perform better on sparse datasets because the model uses a sparsity-aware split finding algorithm (Chen & Guestrin, 2016), whereas Random Forest does not.

Our PyTorch FFN and scikit-learn MLP perform worse than XGBoost on circular fingerprints. This is due to the tabular nature of the fingerprints and RDKit descriptors. Fingerprints encode, in table format, the presence or absence of functional groups in the molecular structure. We expect XGBoost to perform better on tabular data than neural networks due to the inductive biases that go into the algorithm. In contrast, neural networks are more suited to learn non-linear interactions and representations in the dataset. By using fingerprints as the representation of our molecules, we have imposed a tabular representation on the neural network, and it loses to XGBoost in this imposed task.

Interestingly, the out-of-the-box scikit-learn MLP performs better than our PyTorch model. This could be due to suboptimal tuning of our model despite using Optuna. Optuna prunes the trials very aggresively for these datasets, and with more time, we would disable the pruner and do hyperparameter optimization for the full epoch. Scikit-learn also implements a regularization term on Adam and uses early stopping, which could have improved the model's performance.

The Chemprop D-MPNN is the best performing model. This model learned its own feature representation from the structure of the molecule. We expect it to perform better because within-molecule atomic bonding and interactions giving rise to molecular and macromolecular properties is a highly complicated physicochemical process that is not captured by fingerprints. The inductive bias from Chemprop is better suited to this problem of molecular property prediction.

To better understand how the global features calculated using RDKit descriptors affects model performance, we masked each feature for both the solvent and solute and retrained the D-MPNN under 5-fold cross validation. RDKit descriptors improve the model's performance by 1%, which was important in exceeding `benchmark_2`. Due to the error, it is unclear which individual feature masks impact model performance more significantly.

Finally, we trained an ensemble of 20 predictors using a random 90% subset of the dataset as training data, and submitted their predictions to Kaggle. We used such a split because we were more concerned with our tuned model's performance but still wanted to ensure generalization. This ensemble achieved an R-squared of 98.9 on the 50% test data made available to us.

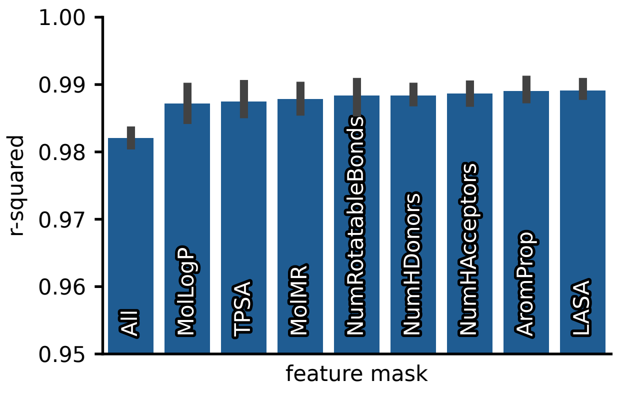

Figure 3. Test set R-squared of the Chemprop D-MPNN with various RDKit descriptor feature masks using 5-fold cross-validation.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. Optuna: A Next-generation Hyperparameter Optimization Framework, July 2019.

Capecchi, A., Probst, D., and Reymond, J.-L. One molecular fingerprint to rule them all: Drugs, biomolecules, and the metabolome. *Journal of Cheminformatics*, 12(1):43, June 2020. ISSN 1758-2946. doi: 10.1186/s13321-020-00445-4.

Chen, T. and Guestrin, C. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, August 2016. doi: 10.1145/2939672.2939785.

Ertl, P., Rohde, B., and Selzer, P. Fast Calculation of Molecular Polar Surface Area as a Sum of Fragment-Based Contributions and Its Application to the Prediction of Drug Transport Properties. *Journal of Medicinal Chemistry*, 43(20):3714–3717, October 2000. ISSN 0022-2623. doi: 10.1021/jm000942e.

Falcon, W. and The PyTorch Lightning team. PyTorch lightning, March 2019.

Heid, E., Greenman, K. P., Chung, Y., Li, S.-C., Graff, D. E., Vermeire, F. H., Wu, H., Green, W. H., and McGill, C. J. Chemprop: A Machine Learning Package for Chemical Property Prediction. *Journal of Chemical Information and Modeling*, 64(1):9–17, January 2024. ISSN 1549-9596. doi: 10.1021/acs.jcim.3c01250.

Labute, P. A widely applicable set of descriptors. *Journal of Molecular Graphics and Modelling*, 18 (4):464–477, January 2000. ISSN 1093-3263. doi: 10.1016/S1093-3263(00)00068-1.

Landrum, G., Tosco, P., Kelley, B., Rodriguez, R., Cosgrove, D., Vianello, R., sriniker, gedeck, Jones, G., NadineSchneider, Kawashima, E., Nealschneider, D., Dalke, A., Swain, M., Cole, B., Turk, S., Savelev, A., Vaucher, A., Wójcikowski, M., Take, I., Scalfani, V. F., Walker, R., Ujihara, K., Probst, D., godin, g., Pahl, A., Lehtivarjo, J., Berenger, F., jasondbiggs, and strets123. RDKit: Open-source cheminformatics. Zenodo, May 2024.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization, June 2018.

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., and Stoica, I. Tune: A Research Platform for Distributed Model Selection and Training, July 2018.

O'Boyle, N. M. and Sayle, R. A. Comparing structural fingerprints using a literature-based similarity benchmark. *Journal of Cheminformatics*, 8:36, July 2016. ISSN 1758-2946. doi: 10.1186/s13321-016-0148-0.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library, December 2019.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, É. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12 (85):2825–2830, 2011. ISSN 1533-7928.

Ritchie, T. J., Macdonald, S. J. F., Young, R. J., and Pickett, S. D. The impact of aromatic ring count on compound developability: Further insights by examining carbo- and hetero-aromatic and -aliphatic ring types. *Drug Discovery Today*, 16(3):164–171, February 2011. ISSN 1359-6446. doi: 10.1016/j.drudis.2010.11.014.

Rogers, D. and Hahn, M. Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, May 2010. ISSN 1549-9596. doi: 10.1021/ci100050t.

van der Maaten, L. and Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. ISSN 1533-7928.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention Is All You Need, August 2023.

Wager, S., Wang, S., and Liang, P. Dropout Training as Adaptive Regularization, November 2013.

Wildman, S. A. and Crippen, G. M. Prediction of Physicochemical Parameters by Atomic Contributions. *Journal of Chemical Information and Computer Sciences*, 39(5):868–873, September 1999. ISSN 0095-2338. doi: 10.1021/ci990307l.

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., Palmer, A., Settels, V., Jaakkola, T., Jensen, K., and Barzilay, R. Analyzing Learned Molecular Representations for Property Prediction. *Journal of Chemical Information and Modeling*, 59

(8):3370–3388, August 2019. ISSN 1549-9596. doi:
10.1021/acs.jcim.9b00237.