Cheminformatics & drug design Homework

Xiaolong YANG

Introduction and problem description

Molecular property prediction is a fundamental task in cheminformatics and drug design. Predicting the physicochemical and biological properties of molecules using deep learning methods will save significant labor and costs, makes it more and more popular. At the same time, with the rapid growth of chemical databases and the development of machine learning techniques, computational approaches have become effective tools for predicting molecular properties, helping drug discovery and materials design.

The project used datasets from the MoleculeNet benchmark suite to explore molecular property prediction using machine learning models(Wu *et al.*, 2018). The study is structured into two main parts: regression analysis on a regression dataset and classification analysis on a classification dataset. Through this work, we aim to evaluate the performance of different predictive models, including Random Forests and Message Passing Neural Networks (MPNN), and to gain practical experience in data preprocessing, model training, and result visualization.

Overview of the modeling methods

The project is mainly divided into two parts: regression analysis and classification analysis, with different models used in each part.

For the regression analysis, the dataset contains 4,200 molecules, including their molecular structures represented by SMILES and the experimental results of the octanol/water distribution coefficient. This type of quantitative relationship is suited for regression analysis, allowing the prediction of a molecule’s octanol/water distribution coefficient based on its molecular structure. To obtain the input data for the models, three types of preprocessing were applied to the dataset: Morgan fingerprints, molecular descriptors, and graph-based representations. In the Random Forest model, Morgan fingerprints and molecular descriptors were used as inputs, while in the MPNN, graph-based molecular representations were used.

Random Forests, as an ensemble of decision trees, randomly sample subsets of the training data and use different features to make decisions(Breiman, 2001). They are highly suitable for both regression and classification tasks, which is why they were applied in this project for lipophilicity regression analysis. The initial parameters for the

Random Forest model were set as estimators=10, and RMSE was used for model evaluation. In regression analysis, RMSE measures the error between the predicted values and the actual values, making it a common metric for assessing predictive accuracy. 80% of the dataset was used for training and 20% for validation. K-fold cross-validation (CV) and RandomizedSearchCV were employed for hyperparameter optimization.

The MPNN is a subclass of graph neural networks that learns node features and derives overall graph representations through two main phases: message passing and readout(Gilmer *et al.*, 2017). Due to its ability to directly learn molecular features from molecular graphs, MPNN has been widely applied in chemical property prediction tasks. For this reason, it was used in this project following the Random Forest model to perform lipophilicity prediction. The initial parameters for the MPNN were set as hidden dim = 64, learning rate = 1e-3, batch size = 64, and epochs = 50, with MSE and RMSE used for model evaluation. Various combinations of input parameters were tested, and skip connections and dropout were used for optimizing the model. The model architecture consists of AtomEncoder and BondEncoder layers for feature initialization, NNConv as the message passing layers, and MLP as the readout network. The loss function used is Mean Squared Error (MSE), with the optimizer is Adam. 80% of the dataset was used for training, 10% for validation and 10% for testing.

Results and evaluation

For the regression analysis, when morgan fingerprints were used as input, the model performance was: RMSE on train set = 0.373, test set = 0.850. Using molecular descriptors as input, the performance improved: RMSE on train set = 0.320, test set = 0.756. This indicates that the model performs better with molecular descriptors than with Morgan fingerprints. However, both Random Forest models were overfitting.

To address this, hyperparameter optimization with cross-validation was applied. Using 5-fold cross-validation, multiple values of estimators and max depth were tested (both from 3 to 50)(Kohavi, 1995). In both cases, the optimal parameters were max depth = 50 and estimators = 50. The model performances were = RMSE on train set = 0.346, test set = 0.821 in Morgan fingerprints and RMSE on train set = 0.270, test set = 0.723 in molecular descriptors. These results show that the models were indeed optimized, but the overfitting issue remained.

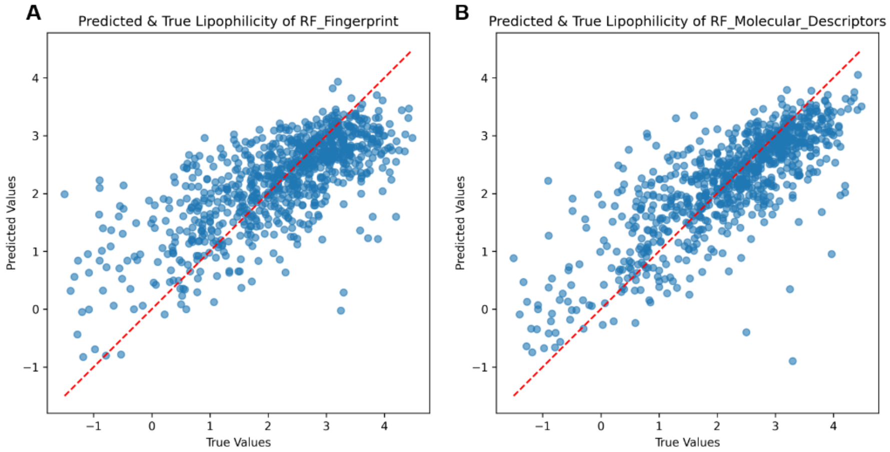

Figure 1 Figure A and Figure B show the comparison between the predicted and true values from the Random Forest model using Morgan fingerprints and molecular descriptors, respectively. Points lying on the red dashed line indicate cases where the predicted values are identical to the true values.

As shown in **Figure 1**, the model predicts molecules with higher lipophilicity better than those with lower lipophilicity. For low lipophilicity molecules, the model tends to predict higher values. This is likely due to the small size of the dataset, and the low ability of Random Forests to produce continuous outputs. The presence of noise in the data also is the reason of overfitting. Data normalization may improve Random Forest performance, but try other models, such as Gradient Boosting or Bagging combined with Ridge Regression, may be a more suitable solution.

The graph features were used as inputs for the MPNN model. With the initial parameters, the model performance was Test MSE = 0.5143 and Test RMSE = 0.7172, slightly better compared to the Random Forest model. After testing multiple parameter combinations, the optimal parameters were learning rate = 1e-3, batch size = 64, and epochs = 100.

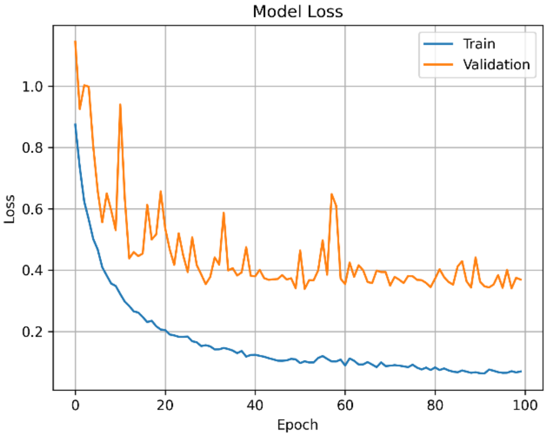

Figure 2 Training and validation loss curves of the MPNN model. The blue line represents the training loss, and the orange line represents the validation loss. Both curves reach equilibrium after around 40 epochs.

As shown in **Figure 2**, the model had converged, and increasing the number of epochs or fine-tuning the parameters did not improve performance. A smaller learning rate, smaller batch size, or more epochs led to overfitting, while the opposite settings led to underfitting, as summarized in Table 1. Therefore, new optimization techniques like skip connections and dropout were used. Skip connections allow the input features to be added to the output features after several layers, helping preserve original node information and reduce overfitting. Dropout randomly removes half of the hidden neurons during training, forcing the network to learn more generalized features and reduce overfitting(Srivastava *et al.*, 2014).

Table 1

Epoch	**Learning Rate**	**Batch Size**	**Skip Connection**	**Dropout**	**Test MSE**	**Test RMSE**
50	1e-3	64	×	0	0.5143	0.7172
50	1e-3	64	√	0	0.7099	0.8426
50	1e-3	64	√	0.1	0.3427	0.5854
100	1e-3	64	×	0	0.3142	0.5605
100	1e-3	64	√	0	0.3551	0.5959
100	1e-3	64	√	0.1	0.5199	0.7211
200	1e-4	32	×	0	0.3964	0.6296
200	1e-4	32	√	0	0.4206	0.6485
200	1e-4	32	√	0.1	0.5014	0.7081

However, as shown in **Table 1**, the skip connections and dropout did not resolve the overfitting problem. This may be due to the small dataset size and the simplicity of the model architecture, so the skip connections and dropout discarded important feature information instead.

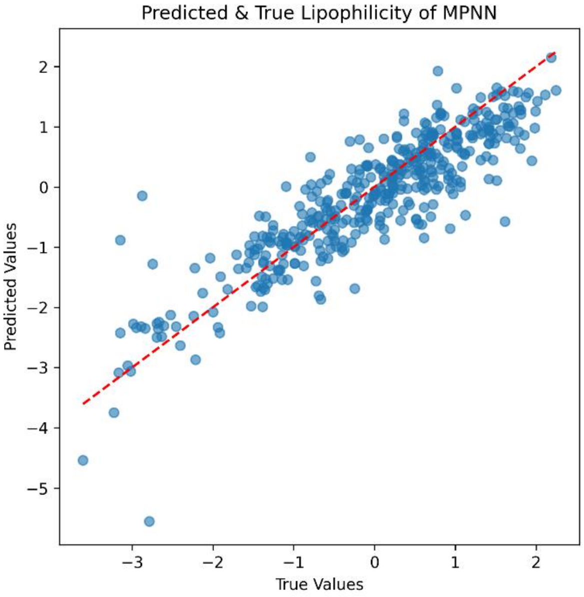

Figure 3 The comparison between the predicted and true values from MPNN using graph-based features. Points lying on the red dashed line indicate cases where the predicted values match the true values.

The final optimized MPNN performance was Test MSE = 0.3142 and Test RMSE = 0.5605, which is noticeably better than the Random Forest model. As shown in Figure 3, the MPNN's predictions show higher accuracy compared to the Random Forest results (**Figure 3**).

Discussion

In this project, traditional machine learning and deep learning methods were explored for lipophilicity prediction, using Random Forests and MPNNs. While optimized Random Forests showed low training RMSE, the high test RMSE indicated limited predictive ability for molecular structural features. In contrast, MPNNs captured deeper molecular relationships through message-passing networks than Random Forests. Techniques such as dropout and skip connections did not further improve performance, suggesting that larger datasets and deeper networks may be needed for further optimization. It would be also useful to apply data standardization to the input data. Another approach to improve prediction accuracy and model stability would be to average the predictions of multiple models. For instance, five models with different hyperparameters and optimization methods could be trained, and their predictions integrated to decrease noise and reduce variance. However, this method would increase computational demands. Given that the dataset used in this project is relatively small, such an ensemble approach should still be feasible and could potentially increase predictive accuracy.

Through this work, I gained hands-on experience in molecular feature extraction (descriptors, Morgan fingerprints, and graph representations), model construction, hyperparameter tuning, and evaluation. The project deepened my understanding of graph neural networks, training visualization, and the practical application of machine learning and deep learning in Cheminformatics. Through teamwork, I also learned the importance of task allocation. A well-structured division of work can greatly enhance project development efficiency and code readability. My biggest takeaway from this project is understanding how to practically extract the necessary information from a dataset and use it to build prediction tasks. Model development is not only about parameter optimization, it also requires selecting appropriate optimization strategies based on the dataset type and prediction objectives to improve predictive accuracy.

Reflection

In this project, the responsibilities between the two team members were clearly defined, one was in charge of the regression analysis, while the other focused on the classification analysis. The clear division of tasks allowed us to work on the project simultaneously without interfering with each other, improving efficiency and making it possible to explore more models. And the models and evaluation methods used for regression and classification tasks differ significantly, dividing the project into two main parts, regression and classification, is a logical approach. This design allows users to choose the suitable model based on their dataset and prediction objective, making the project more user-friendly and adaptable.

However, this separation also resulted in limited collaboration between the two members. Additionally, differences in coding habits and the coding styles reduced the generalizability across modules. For example, in the data reading and preprocessing parts, although there was some collaboration, the use of different datasets led to redundant code and decreased reusability. To improve efficiency and user experience, it would be better to decide a unified coding standard for shared modules before starting the project.

Use of AI Assistants

In this project, the AI assistant provided support in several parts:

1. **Code debug:** The most frequent use of AI was for debugging. When facing bugs that could not be resolved through online tutorials, GPT was used for assistance.

2. **Code development:** AI was used to assist in writing code for certain parts of

the project. For example, GPT was asked how to pass my dataset into class during feature extraction. For result visualization, GPT provided a plotting code framework and guidance on how to save visualizations to folders. In the MPNN, GPT was used for recording loss values for loss visualization. These AI usages are documented with comments in the code.

3. **Concept explanation and analysis:** AI was used to explain basic concepts, such as the functions of different layers in the MPNN model, which help with model modifications. Additionally, AI assisted in analyzing results, for instance evaluating whether the RMSE could be further optimized.
4. **Report modification:** AI was not used to write the report directly. Instead, it was used to improve grammar and sentence structure for clarity and readability.

References

Breiman L. 2001. Random Forests. Machine Learning **45**, 5–32.

Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE. 2017. Neural Message Passing for Quantum Chemistry. Proceedings of the 34th International Conference on Machine Learning. PMLR, 1263–1272.

Kohavi R. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. IJCAI'95. Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1137–1143.

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**, 1929–1958.

Wu Z, Ramsundar B, Feinberg EN, Gomes J, Geniesse C, Pappu AS, Leswing K, Pande V. 2018. MoleculeNet: a benchmark for molecular machine learning. Chemical Science **9**, 513–530.