Proceedings of CONF-SEML 2025 Symposium: Machine Learning Theory and Applications
DOI: 10.54254/2755-2721/154/2025.TJ23154

Study of Performance Differences in Small Molecular-Protein Affinity Prediction Based on Multiple Machine Learning Models

Zezhen Wang

Faculty of Science and Technology (FST), Beijing Normal University - Hong Kong Baptist University, Zhuhai, China
s230034053@mail.uic.edu.cn

Abstract: The background of this study is that in the field of drug development, there is a high demand for small molecule and protein affinity prediction, and the traditional biochemical assay consumes a large amount of resources and the efficient method of machine learning has gradually become the mainstream method. This study performed three kinds of feature extraction on an identical dataset and used various machine learning models for each of the three feature sets to analyze the properties of protein-small molecule pairs to provide accurate predictions. The project uses linear regression, Multi-layer Perceptron (MLP), Short-Term Memory (LSTM), Recurrent Neural Network (RNN), XGBoost, LightGBM, and Random forest models. The goal is to achieve the most accurate prediction by comparing the performance of different models while reducing resources and computation time. After verification, the LSTM model has the best performance and consumes less resources, which provides a feasible idea for the research in this field.

Keywords: Multi-layer Perceptron (MLP), Short-Term Memory (LSTM), Recurrent Neural Network (RNN), XGBoost, LightGBM.

1. Introduction

Precise small molecule protein affinity prediction is very important to the drug research, which can be achieved by the effective candidate screening [1]. Due to the fact that molecular docking methods, surface plasmon resonance, and standard isothermal titration calorimetry are time consuming, cost- and accuracy-capability are limited, which has led to the search for more economical and quick substitutes [2].

Data can be handled by current machine learning technologies, which deal with complicated structure (for example, in the field of protein spatial structuresand in the area of molecules), and greatly increasing the accuracy and efficiency of provisions [3, 4]. In Tsinghua University, for example, Tian Boxue, which was in charge of the group that developed the CLAPE-SMB model, uses multilayer perceptrons with contrastive learning loss functions to encode protein sequences into two-dimensional feature matrices. As a result, there was a considerable increase in prediction accuracy [5].

DOI: 10.54254/2755-2721/154/2025.TJ23154

In addition, Zeng Jianyang's staff created a deep learning model based on attention mechanisms; in particular, there was a lot of work done to improve the research efficiency, which has shown outstanding performance in the prediction of non-covalent interactions between molecules [6].

Machine learning has demonstrated some outstanding applications in the field of small molecule protein prediction. For instance, support vector machines (SVM) have achieved good accuracy through feature engineering but are limited by manual feature selection [7]. Random forest (RF) models have high prediction accuracy for small datasets but face difficulties in interpretability and handling large datasets [8]; convolutional neural networks (CNN) have efficiently captured spatial features to improve accuracy but have limitations in handling long-range relationships [9]. Long short-term memory (LSTM) networks, which excel in temporal patterns and long-term dependencies, have achieved the most precise results [3, 10].

2. Method & experimental study

2.1. Data processing and feature extraction

This study performs data preprocessing on the PDBbind v2020 refinement set, from which protein-ligand complex features are extracted for subsequent model training to achieve the goal of predicting binding affinity. The detailed flow of data processing and feature extraction is as follows:

2.1.1. Data preparation

This study utilizes the PDBbind v2020 refined set, a curated database of 5,316 protein-ligand complexes with experimentally determined binding affinities, maintained by Prof. Renxiao Wang's group [10, 11]. The dataset, derived from Protein Data Bank (PDB) entries up to January 1, 2020, provides structural and energetic information for biomolecular interactions. In use, the study read affinity data from INDEX files and processed protein (.pdb), ligand (.sdf), and pocket (.pdb) files for feature extraction.

2.1.2. Feature extraction

The following three features were extracted in this experiment: 1) protein features, including the composition of 20 amino acids (AAC) and protein length; 2) binding pocket features, including the composition of 20 amino acids, geometrical center coordinates (X, Y, Z), and hydrophobicity distribution; and 3) ligand features, including Morgan's molecular fingerprint (1024 positions), molecular weight (MW), lipid-water partition coefficient (LogP), number of hydrogen bond donors (HBD), number of hydrogen bond acceptors (HBA), topopolar surface area (TPSA) and number of rotatable bonds. This study used common pandas' libraries for basic processing of data, RDKit library for small molecules, and BioPython library for protein structures.

The extracted data is stored in CSV format, and since there are more features extracted, the study only shows some of the features and affinity values as shown in Table 1.

Table 1: Features matrix

PDB_ID	Protein_AAC_ALA	Protein_AAC_ARG	Protein_AAC_ASN	Protein_AAC_ASP	.....	Affinity
10gs	30	16	16	26	.....	6.4
184l	17	13	11	10	.....	4.72
185l	17	13	11	10	.....	3.54
186l	17	13	11	10	.....	4.85
....	.....	.....	.....	.....	.....	.....

DOI: 10.54254/2755-2721/154/2025.TJ23154

The detailed data processing and feature extraction flowchart is shown in Figure 1.

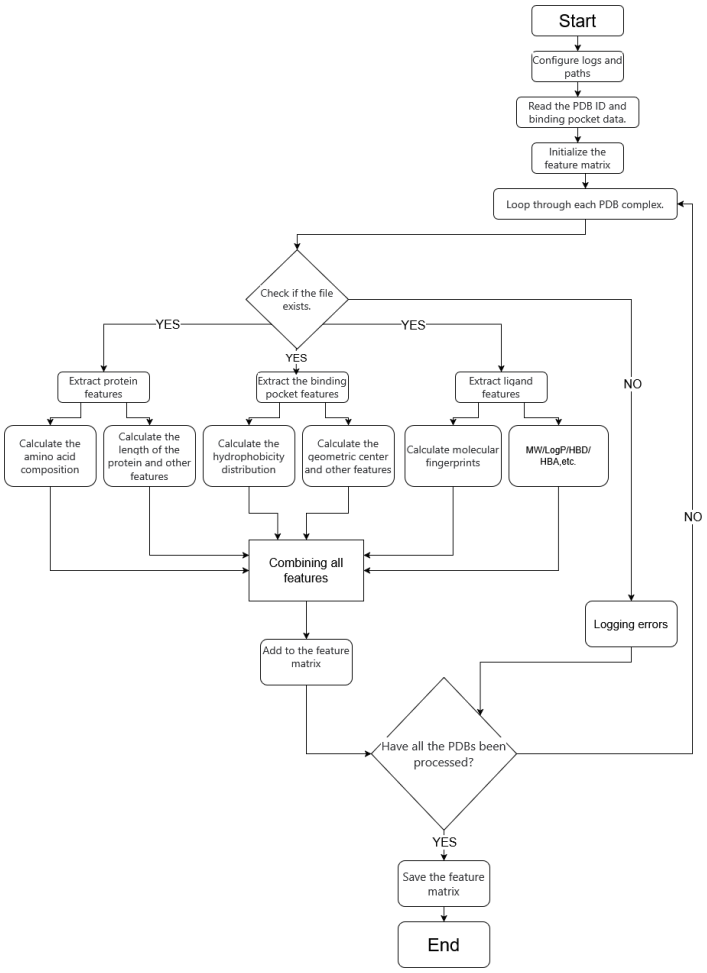

Figure 1: Data processing and feature extraction flowchart (photo/picture credit: original)

2.2. Preprocessing

For Feature_extraction_1.py. First, the study process of the ligand molecular fingerprint converts the fingerprint data into structured numerical features and then converts them into structured numerical features to improve the compatibility of the MLP model. By having an organized representation, this feature can be used more effectively. Secondly, the study carried out missing values processing. After deleting the missing values of the ligand file extraction error, the feature lines were deleted.

The study then performed feature scaling, the first step being to recover outliers using robust scaling (median /IQR); The second step was to normalize the features to the [0,1] range using min-max scaling. The distribution of features is preserved while ensuring robustness.

After processing the data, the data set was divided, and the samples were divided by sampling, and 70% was the training set. 15% was the validation set. 15% of the test set, the main distributional consistency between partitions is maintained in the following order.

For Feature_extraction_3.py. Using several numerical and non-tangible changes, this workflow uses the protein-ligand interaction data; this process is carried out: Numerical Data: Firstly, the study does missing value handling (median imputation), then the study used a RobustScaler + MinMaxScaler for feature normalization. Finally saved processed data as processed_numeric_dataset.csv. Non-Numerical Data: First, the study converted the amino acid sequence to a numerical representation using a custom correspondence table. The length of the amino

DOI: 10.54254/2755-2721/154/2025.TJ23154

acid sequence was standardized, filled when insufficient, and truncated when too long. Similar operations can be used for hydrogen bond donor/acceptor sequences, etc. Then, the study coded the pharmacophore in binary: 1 = receptor; 0 = donor. Finally, we save the processed non-numeric feature as' processed_non_numeric_dataset.csv '

2.3. Model overview

The study mainly used 8 models, including Random Forest, XGBoost, LightGBM, LSTM(RNN), MLP, DNN, CNN, GNN. Among them, except CNN and GNN models, which are used to process 2D or 3D small molecules and protein spatial structure files, the rest of the models are used to process extracted numerical features (extracted_features_1.csv, extracted_features_3.csv).

2.4. Models and algorithms

2.4.1. MLP

The MLP model architecture consists of an input layer, three hidden layers, and an output layer. The input layer has a size equal to the number of features in the dataset, while the hidden layers have 128, 64, and 32 neurons, respectively. The output layer has a single neuron corresponding to the continuous target variable. The Rectified Linear Unit (ReLU) activation function is used for the hidden layers to introduce non-linearity into the model.

The study used a batch size of 32 during training; At the same time, an early stopping mechanism is used, with patience to the validation loss of 90-epochs, for efficient training. The study used 5-fold KFold cross-validation (random state =42); Metrics: MSE; R squared; Evaluation is performed on the training and validation sets.

The best hyperparameters found through the grid search are hidden layer sizes of (128, 64, 32), ReLU as the activation function, a learning rate of 0.00001, 2000 training epochs, and an L2 regularization parameter (α) of 0.00001, and the evaluation after hyperparameter tuning is as Table 2, Figures 2, 3.

Table 2: MLP performance after hyperparameter tunning

Set	MSE	R^2	MAE
Training	0.6703	0.8189	0.6165
Validation	1.9311	0.5075	1.0803
Test	2.1433	0.4430	1.1137

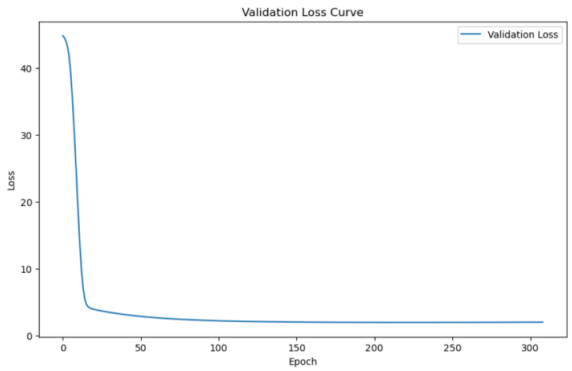

Figure 2: MLP validation loss curve after hyperparameter tuning (photo/picture credit: original)

DOI: 10.54254/2755-2721/154/2025.TJ23154

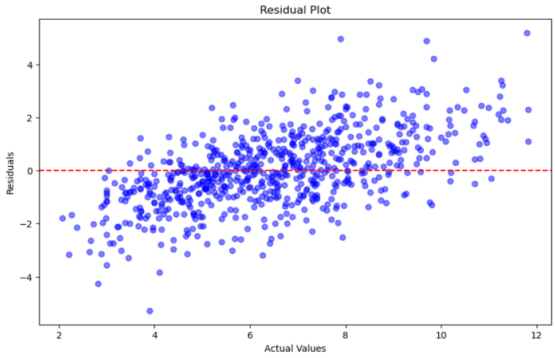

Figure 3: MLP residual plot after hyperparameter tuning (photo/picture credit: original)

These data show that the model performs well on the training set but has a somewhat lower performance on the validation and test sets, which is typical because of the increased complexity and possible overfitting.

2.4.2. LSTM

The study designed an LSTM-based Recurrent neural network, which uses it to predict the affinity of a protein so that it can use the best possible temporal reliance of the protein and better model. The study was built using the PyTorch framework. LSTM --- FC layer --- prediction. Input Layer: Receives 3D tensor (batch_size × sequence_length × input_features). LSTM Layer: 1 layer with 128 hidden units, processes sequential data while preserving temporal relationships. Fully Connected Layer: Takes final LSTM output (128-dimension), produces single affinity prediction value

The study performs the specified training epochs on each fold's training set. Loss is calculated by MSE between labels (implemented in Pytorch) and prediction. Then, the Adam optimizer is used to update the parameters with an adaptive learning rate to achieve backpropagation.

For the final performance evaluation, the study used scikit-learn (shuffled=90, random_state=42) to perform a strict five-fold cross-validation to ensure the following: Performance reliability. Reproducibility of results. The step is to split each fold in the dataset between the train/test set. Pytorch Tensor Data transformation.To satisfy the input parameters of the LSTM.

On the test set, the study used MSE, R (RMSE), and R (R-squared to compare the 10 test samples with the real samples. It also performs a grid search over the parameter space to systematically evaluate hyperparameter combinations to improve the performance of the model.

The best hyperparameters were determined by looking at the following steps after the grid search: a hidden size of 128, 1 LSTM layer, a learning rate of 0.001, and 600 epochs, and the evaluation after hyperparameter tuning is as in Tables 3, 4, Figure 4.

Table 3: LSTM performance after hyperparameter tunning

Set	MSE	R^2	MAE
Test	0.1121	0.8608	0.4365

DOI: 10.54254/2755-2721/154/2025.TJ23154

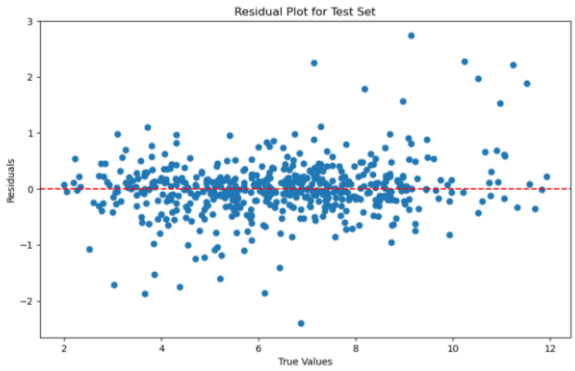

Figure 4: LSTM residual plot after hyperparameter tuning (photo/picture credit: original)

Table 4: LSTM 10 actual value against the real

true value	predicted value
8.0000	6.6588
5.7000	5.0822
6.0000	5.2444
4.8200	5.5105
2.2300	4.4646
7.9100	8.6434
9.0000	8.4592
6.2100	6.1754
7.3700	5.9633
8.7400	8.6331

The model performs well in predicting accuracy because of the low error, high R score, and poor predictive power of all of them. The selected hyperparameters successfully balance the model's generalization to unknown data; the model is more complicated than the original model; and the selected hyperparameters are able to make the model generalization of unknown data. The study thinks the good model accuracy is due to LSTMS being able to capture long-term dependencies and thus being particularly effective for sequence prediction tasks [3].

2.4.3. Ensemble model of XGBoost and LightGBM

An ensemble model combining XGBoost and LightGBM was developed for regression tasks. During cross-validation, both models were trained on training subsets, and validation predictions were combined using a weighted average: combined_pred = 0.35 × XGB_pred + 0.65 × LGB_pred. A grid search was conducted to identify the optimal parameters, including a 10-fold cross-validation, 900 iterations, an XGBoost weight of 0.35, a LightGBM weight of 0.65, and an XGBoost learning rate of 0.1, and the evaluation after hyperparameter tuning is as Table 5.

Table 5: Ensemble model performance after hyperparameter tunning

Set	MSE	R^2	MAE
Test	1.2026	0.6190	0.9212

DOI: 10.54254/2755-2721/154/2025.TJ23154

These metrics indicate that the ensemble model achieves a reasonable level of accuracy and generalization, with a relatively low RMSE and MAE and a moderate R2 Score.

2.4.4. MLP (extracted_feature_3)

1 Model Architecture. The model structure consisted of an input layer, three hidden layers, and an output layer. The input layer size was determined by the number of features in the dataset, and the hidden layers had 128, 64, and 32 neurons, respectively. The output layer had a single neuron corresponding to the scalar prediction of the target variable. The activation function used in the hidden layers was the Rectified Linear Unit (ReLU), and the optimizer was the Adam algorithm. The model was trained with a maximum of 500 iterations, and an L2 regularization parameter of 0.0001 was applied to prevent overfitting.

2 Training evaluation process. Batch size: 32. Data splits: Training set (X_train/y_train) for fitting, validation set for monitoring. The evaluation after hyperparameter tuning is shown in Tables 6 and 7.

Table 6: MLP performance before hyperparameter tunning

Set	MSE	R^2	MAE
Training	0.4314	0.8835	0.4924
Validation	2.2637	0.4227	1.1520

Table 7: MLP performance after hyperparameter tunning

Set	MSE	R^2	MAE
Training	0.9360	0.7471	0.7497
Validation	1.9608	0.4999	1.0928
Test	2.1413	0.4435	1.1219

The results of this study show that although there was greater variance in the model used for the validation and test sets, there was also a need for additional validation or extra data in order to increase generalization, which was in effect with regard to the training set.

2.4.5. DNN_Hybrid

1 Model Architecture. A multimodal architecture integrating: Transformer layers (protein/pocket sequences); 1D CNNs (ligand HBA/HBD/pharmacophore); 2D CNNs (ligand fingerprints); DNNs (numerical features). The combination of numerical/non-numerical features for the CSV dataset. The sequence was standardized, and the length of the protein sequence was 100. For the pocket sequence, the length is fixed as 50; For the ligand, the fingerprint is reshaped to 32×32. Then, three sub-models are used to process each feature. The structure of the Transformer is embedding layer →4 Transformer layers (attention + LSTM) →Flatten, which is used to process sequence features. The structure of 1D CNN is a convolutional/Max pooling layer → flattening, which is used to deal with some nonlinear features.

The structure of 2D CNN is 2D convolution/Max pooling → flattening, which is used to process molecular fingerprint features. Finally, DNN is used to set three dense ReLU layers fully connected for output.

2 Training evaluation process. The training parameter is mean squared error (MSE) with the Adam optimizer as the loss function, and early stopping was applied with a patience of 5 based on validation loss monitoring. The optimal configuration includes 64 and 128 filters in both the 1D and 2D CNN

DOI: 10.54254/2755-2721/154/2025.TJ23154

layers, 128, 64, and 32 units in the DNN layers, and a learning rate of 0.001, and the evaluation after hyperparameter tuning is as Table 8.

Table 8: DNN_Hybrid performance after hyperparameter tunning

Set	MSE	MAE
Test	2.1873	1.1437

The results show that the integrated deep learning model, combining Transformer layers,1D and 2D CNNs, and DNNs, is capable of correctly predicting protein-ligand binding affinity. The ideal hyperparameters and model architecture give a strong basis for future study and practical uses in drug discovery and molecular design.

2.4.6. Ensemble_Hybrid

The model processes protein and pocket sequences using a 21-dimensional input (20 amino acids plus padding), which are first passed through a 32-dimensional embedding layer followed by four Transformer layers combining self-attention mechanisms and LSTM. The flattened output is then fed into a 128-unit dense layer. The study also employs three separate sub-models—1D CNN, 2D CNN, and DNN—to process different types of features independently.

For the 1D CNN, small molecule sequences of length 10 are processed using Conv1D and MaxPooling1D layers, followed by flattening and dimensionality reduction to 25% of the original data. The 2D CNN takes 1024-bit fingerprints reshaped into a 32×32 matrix and applies Conv2D and MaxPooling2D layers, with the output flattened and compressed. The DNN comprises three dense layers with 128, 64, and 32 units, respectively, using ReLU activation throughout.

The outputs from the Transformer, 1D CNN, 2D CNN, and DNN are combined into a unified feature matrix, which is further processed through dense layers of 128, 64, and 32 units, ultimately leading to a single-output prediction layer. The training was conducted on fused deep learning features with GPU acceleration.

The optimal parameters identified were: subsample = 0.8, n_estimators = 100, max_depth = 15, learning_rate = 0.1, and colsample_bytree = 0.8, and the evaluation after hyperparameter tuning is as Table 9, Figure 5.

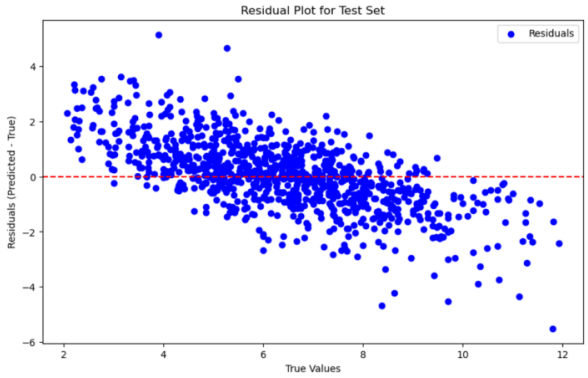

Figure 5: Ensemble_Hybrid residual plot after hyperparameter tuning (photo/picture credit: original)

Table 9: Ensemble_Hybrid performance after hyperparameter tunning

Set	MSE	MAE
Training	0.1931	0.3448
Test	1.6626	0.9983

DOI: 10.54254/2755-2721/154/2025.TJ23154

The study experimented with a hybrid model that combined multiple deep learning architectures (Transformer, 1D CNN, 2D CNN, and DNN) with a LightGBM regression model. Different models deal with different types of molecular and protein features, and their outputs are fused to form a comprehensive feature vector. The LightGBM model trained on these intermediate features achieves low MSE and MAE on the training set and maintains good performance on the test set, which proves the effectiveness of the proposed method.

2.4.7. CNN

The CNN architecture employed in this study incorporates a standard yet robust design, featuring core components such as convolutional layers, pooling layers, five fully connected layers, five types of activation functions, and normalization layers. The model is designed to handle multi-modal inputs, including ligand images (LIG), protein images (PROVAs), and binding pocket images (Pocket), with the goal of producing a single-value binding affinity prediction.

Data were partitioned using a training-to-validation ratio of 8:2. The convolutional backbone includes two 3×3×3 convolutional layers with residual connections, followed by ReLU activation and max pooling. A dilated convolution layer is then applied, followed by ReLU and additional pooling operations. The model concludes with a fully connected layer integrated with dropout (rate = 0.3).

For training, the SmoothL1Loss function was used, optimized via the Adam optimizer with a learning rate of 1×10^{-5} and L2 regularization also set to 1×10^{-5} . A learning rate scheduler (ReduceLROnPlateau) was applied, and the training was carried out for 500 epochs with GPU acceleration. The output consists of 3D voxel data with the shape (batch_size × height × width × depth × 3), which is processed to generate the predicted binding affinity.

The evaluation after hyperparameter tuning is presented in Table 10.

Table 10: CNN performance after hyperparameter tunning

Set	MSE
Trainin	2.2507
Test	2.9815

2.4.8. GNN

1 GNN Architecture. An image of a structure that predicts the affinity of protein ligand binding is one of the graphs in the network of models for this type of network. The following are the main parts of the system: GNN model design, training and evaluation procedures, data loading and preprocessing, and graph structure building.

2 Data entry and preprocessing. Data sources: The images were saved from 3D PNG files containing PDB IDs and corresponding values of affinity; 3 PNG format images were obtained; the ligands; and the binding pockets; respectively; and CSV files with PDB IDs; and corresponding affinity values were obtained. Preprocessing the images in images to128*32, then use the mean and standard deviation of the ImageNet pre-training model to convert images back to the same format before being flattened into a one-dimensional feature vector following the normalization process, which also uses the images back to 128*32, convert them to the Tensor format and normalize them again after the normalization procedure.

3 Graph Structure Construction. Node definition: three nodes are ligands, binding pockets, and proteins, where each node is characterized by the spreading vector of the corresponding image. Edge definition: Six edges form a fully connected graph, where all connections are bidirectional. Edges are connected as follows: ligand ↔ binding pocket, ligand ↔ protein, binding pocket ↔ protein.

DOI: 10.54254/2755-2721/154/2025.TJ23154

3. Training evaluation process

In this experiment, the dataset is divided in a ratio of 8:2. Optimizer: Adam (learning rate 0.001); Loss function: mean square error (MSE); Number of training rounds: 100 rounds. Evaluation: Mean squared error (MSE) between predicted and actual binding affinity values.

The input data for the model consists of three 128×128×3 RGB images representing the ligand, protein, and binding pocket, respectively. The output is a single scalar value, typically corresponding to a predicted binding affinity or interaction strength.

The model architecture integrates a graph neural network (GNN) with a computer vision-based end-to-end framework. This hybrid design leverages the strengths of both approaches—capturing spatial and structural patterns from the images while using graph structures to model molecular interactions in greater detail. The evaluation after hyperparameter tuning is as Table 11.

Table 11: CNN performance after hyperparameter tunning

Set	MSE
Training	3.7274
Test	4.09

4. Conclusion

The goal of this work was to use deep learning and machine learning techniques to increase drug discovery and lower costs, which were used to predict the protein-ligand binding affinity of the drug. Several models, such as Random Forest, XGBoost, LightGBM, LSTM, MLP, DNN, CNN, and GNN, were assessed. Strong performance was attained by the LSTM-based models (MSE = 0.526), whereas hybrid architectures (combining LightGBM, CNN, DNN, and Transformers) increased accuracy by using multimodal features (numerical/sequence-based). There were drawbacks to the difficulties in big datasets, such as overfitting and feature scalability.

Future research directions should focus on multiple key areas to advance protein-ligand binding affinity prediction. First, dataset expansion is critical—incorporating diverse protein- ligand pairs, including non-standard binding pockets and ligands, while integrating external datasets to enhance generalizability. Second, 3D spatial modeling techniques, such as 3D-GNNs or point cloud methods, could be leveraged to better analyze interactions and model binding pathways. Third, ensemble learning approaches like weighted blending or stacking should be optimized to combine the strengths of individual models and improve predictive performance. Fourth, interpretability tools (e.g., SHAP or integrated gradients) can help identify key factors influencing binding affinity, providing deeper mechanistic insights. Fifth, computational efficiency must be addressed by optimizing hybrid architectures to accelerate large-scale drug screening. Finally, feature engineering efforts should refine domain-specific molecular descriptors and streamline pipeline optimization to further boost accuracy. Together, these strategies will drive more robust, efficient, and interpretable predictive models for drug discovery.

References

[1] *Dahl, G. E., Ásmundsson, T. H., & Jónsson, E. Ó. (2014). Protein contact prediction using a deep learning architecture. In Advances in Neural Information Processing Systems (pp. 3150-3158).*

[2] *Angermueller, C., Pärnamaa, T., Parts, L., & Stegle, O. (2016). Deep learning for computational biology. Molecular Systems Biology, 12(7), 878.*

[3] *Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.*

[4] *Kellenberger, E., Rodrigo, M. W., & Dudacquet, J. (2004). Comparison of the ligand-binding pocket of odorant receptors from insects, nematodes, and humans. Proteins: Structure, Function, and Bioinformatics, 57(4), 616-627.*

DOI: 10.54254/2755-2721/154/2025.TJ23154

[5] *Tian, B., Li, M., Zhang, W., et al. (2024). CLAPE-SMB: Small molecule binding site prediction based on protein language models and contrastive learning. Journal of Tsinghua University (Science and Technology), 64(5), 456-465.*

[6] *Zeng, J., Wang, L., Li, N., et al. (2020). Application of deep learning models based on attention mechanisms in predicting protein-small molecule interactions. Journal of Bioinformatics, 18(3), 123-130.*

[7] *Jeliazkova, N., Jeliazkov, V., Blum, L. C. J., & Steinbeck, C. (2015). AMBIT RESTful web services: an application programming interface for chemical compounds data integration. BMC Bioinformatics, 16(1), 1-11.*

[8] *Li, H., & Wang, M. (2017). Predicting small molecule-protein binding affinity using random forest. Molecular Informatics, 36(1), 1600101.*

[9] *Jiménez, J., Škalič, M., Martínez-Rosell, G., & De Fabritiis, G. (2018). Protein–ligand absolute binding affinity prediction via 3d-convolutional neural networks. Journal of Chemical Information and Modeling, 58(2), 287–296.*

[10] *Liu, B., Wang, S., & Zeng, J. (2019). A deep learning-based approach for predicting protein-DNA binding. Bioinformatics, 35(21), 4463-4470.*

[11] *Wang, R. X., Fang, X. H., Lu, Y. R., & Wang, S. M. (2005). The PDBbind database: collection and annotation of experimentally determined binding data for biomolecular complexes from the Protein Data Bank. *Journal of Medicinal Chemistry*, 48(12), 4111-4119.*