Using Machine Learning Surrogate Models to Increase Throughput of Virtual Screening with Docking

Vinay Nair

Received September 24, 2024
Accepted January 14, 2025
Electronic access February 15, 2025

Virtual screening is an important step in modern computational drug discovery used to find molecules that can bind to a certain target. Currently, physics-based molecular docking is used in order to simulate how a molecule may bind to a target protein to narrow the amount of molecules needed to be experimentally tested. However, this process has a low throughput, especially when dealing with the rapidly increasing size of ligand libraries. I introduce a machine learning surrogate model able to classify whether or not a given ligand will bind to a protein with an 80x increase in throughput than current docking programs when trained on 10% of the dataset, allowing for the finding of binding molecules without the need to dock the entire library. I also introduce a separate model able to score binding affinity for a ligand and protein pair with a 20% increase in throughput than current docking programs when trained on 40% of the dataset with an average Spearman rank correlation of 0.693. The improvements in throughput compared to smina were statistically significant (p ¡ 0.05) across the different targets. These models can be used in conjunction to be able to score the affinity of a ligand to a target protein after finding that it binds. The models are able to determine the binding of a ligand to a target with a throughput much higher than traditional docking softwares, being capable of screening 48 billion molecules in approximately 8,700 hours using 1,000 computers. They are as accurate as molecular docking when using at least 10% of the data for training for the classification model and at least 40% of the data for training for the scoring model. With these models, the throughput of virtual screening can be increased substantially, allowing for the screening of larger libraries than can be done with molecular docking.

Keywords: Virtual screening, molecular docking, active ligands, decoys

Introduction

Research and development for drug discovery is a costly process, on average costing $879.3 million to develop a single new drug [1]. A large part of this cost comes from searching for molecules needed to address the target symptom. To help alleviate a symptom or disease, a molecule can be bound to a target protein responsible for the symptom. Scientists must find molecules that bind to the target receptor and can have the needed effect on the protein in order to help alleviate the target symptom or disease. To find these targets, it is necessary to screen libraries of molecules in order to find molecules that can bind to the target receptor [2].

Virtual screening is a technique used in drug discovery that computationally screens libraries of molecules against a certain target in order to find molecules that bind to the target. It can screen libraries with a hit rate 100-fold to 1000-fold higher than that of physical screening, where the libraries of molecules would be physically tested for whether or not they bind [3]. Virtual screening can be done using molecular docking programs. These programs fit molecules to the target and predict the binding energy, allowing them to find if the molecule is an active binder, which can bind to the target, or a non-binder, which cannot bind to the target. With molecular docking tools, novel ligands can be found that are able to bind to a certain target [4]. Docking programs such as AutoDock Vina and smina are able to screen large libraries of molecules with a much higher throughput than physical screening. These programs use computational representations of molecules and calculate their binding to proteins with their geometric shape. This allows for the screening of larger libraries than what was able to be screened before, allowing scientists to narrow down the amount of possible binders in the library that need to be experimentally tested [5,6].

Recently, virtual screening libraries have rapidly increased in size. Modern libraries can contain billions of molecules to be screened for a single target. The REAL Compounds space by Enamine contains over 48 billion compounds that can be screened [7]. As libraries rapidly increase in size, the time required to dock the libraries increases as well [8]. Using smina with an average time to dock a single molecule at 30 seconds [9], the total time taken to dock would be $N \times T$, where N is the number of molecules and T is the average time taken to dock one molecule. Docking the REAL Compounds space would take over 400 million hours when running on one computer

and over 400,000 hours when running on 1000 computers. In order to combat this, surrogate models can be created in order to increase the throughput of screening large libraries. This can rapidly decrease the time needed to screen a library against a certain target.

Surrogate models are models that approximate an outcome instead of computing it in order to accelerate the speed of finding the outcome [10]. These models have been used for tasks such as agent-based simulations [11] and physics models [12] in order to increase throughput and decrease computation time of simulations. Surrogate models use machine learning, a technique that uses artificial intelligence to learn about a data set and predict information about new data13. These predictions can be used to approximate outcomes for a surrogate model.

There are many machine learning algorithms that can be used for a model. Random forests are an algorithm that uses many decision trees for classification or regression tasks. The answers to each decision tree are used to generate the output. The output selected by the most trees is used for classification while the mean output of all the trees is used for regression. Random forest classification is defined by $P(y = 1|x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$, where $h_t(x)$ is the prediction of the t-th decision tree.

Random forest regression is defined by $\hat{y} = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$, where $h_t(x)$ is the output of the t-th decision tree [13]. Random forest models can be used in surrogate models in order to approximate outcomes for hard to compute simulations. Random forest surrogate models have been used to make predictions in fields such as aerospace [14] in order to reduce computation time for complex simulations.

Up until now, machine learning models have been used to replace docking, such as in DiffDock [15]. However, they have often been slow or require large amounts of computational resources and GPU acceleration. While DiffDock and current docking programs such as smina can efficiently screen molecules, the large amounts of computational resources needed make them unsuitable for billion-scale libraries. A high throughput model could be used to make predictions on the interaction between the molecule and target without having to do docking calculations. These predictions can then be used to classify the molecule to determine if it will bind to the target or not or rank molecules according to the predicted probability of their binding. Since these models would not do any docking calculations, they can screen libraries with a much higher throughput than molecular docking. This allows for the screening of libraries with billions of compounds, expediting the discovery of life-saving drugs while reducing cost and time.

In this paper, I build a random forest surrogate model in order to increase throughput of molecular docking without a substantial drop in accuracy. I create a classification model that predicts if a given molecule will bind to a target. I also create a regression model that predicts binding affinity for a given molecule to a target. I investigate whether the model can increase throughput of molecular docking without sacrificing accuracy by testing them against smina, a docking program forked from Autodock Vina. I predict that the models will be able to increase throughput of molecular docking with a slight decrease in accuracy.

Methodology

Benchmarking Set. Both the docking performance and the performance of the surrogate model were tested against the DUD-E benchmarking set. The DUD-E benchmarking set is a directory of varied active binders and decoys for different targets used to assess the performance of molecular docking [16]. The docking and surrogate model performance tests were done against ten targets in the DUD-E diverse subset, ADRB1, AKT1, AMPC, CDK2, CP3A4, CXCR4, GCR, HIVPR, HIVRT, and KIF11. This benchmarking set was chosen because of its diverse dataset of ligand-target pairs compared to other sets, such as ChEMBL and ZINC.

Docking. smina was used to benchmark docking performance. smina is a fork of AutoDock Vina maintained by David Koes at the University of Pittsburgh that improves scoring and minimizes energy. It optimizes for high throughput scoring and can process large numbers of ligands [5]. It is available at https://sourceforge.net/projects/smina/.

Classification Model Training. The machine learning surrogate models were trained to predict the binding of a ligand to a certain protein target. The models were trained on separate DUD-E data of active binders and decoys of one of the 10 DUD-E targets benchmarked. For each target, six models were trained and benchmarked. Each of the six models had a varying amount of the DUD-E data used to train. The sizes were 50%, 40%, 30%, 20%, 10%, and 1% of the DUD-E data. For each size, a random sample of active binders of that size were used for training. The same size of decoys were added to the training set. Before being trained, the training data ligands were given descriptors by the RDKit Descriptors module. The RDKit Descriptors module assigns molecular properties such as weight, surface area, and logP to each molecule, allowing for the model to train from these descriptors and make predictions based on the descriptors [17]. All descriptors provided by the RDKit Descriptors module were used in training the model. Fingerprints were not used in training the model. RDKit Descriptors were chosen as they provide a vast number of molecular properties that can be used to predict binding. The described training data was used to train the random forest classifier for the specific target. Random forests were selected due to their reduced overfitting, high accuracy, and efficiency compared to deep learning models. The classifier was trained with scikit-learn's random forest classifier using default hyperparameters. Scikit-learn is an open-source Python machine learning library with an emphasis on ease of use and performance. It allows for use of different ma-

chine learning algorithms, including logistic regression, nearest neighbors, and random forest[17].

```
def calculate_descriptors(smiles):
  try:
    mol = Chem.MolFromSmiles(smiles)
  except:
    print('Invalid SMILES')
    return
  return pd.Series(Chem.Descriptors.CalcMolDescriptors(mol))
```

Fig. 1 Assigning descriptors

```
actives = [Chem.MolToSmiles(Chem.AddHs(x)) for x in Chem.SDMolSupplier(fr"{name}_actives_training.sdf")]
actives = random.sample(actives, alen)
decoys = [Chem.MolToSmiles(Chem.AddHs(x)) for x in Chem.SDMolSupplier(fr"{name}_decoys_training.sdf")]
decoys = random.sample(decoys, dlen)
dict = {x:1 for x in actives}
dict.update({x:0 for x in decoys})
key_value_pairs = [{'SMILES': k, 'binds': v} for k, v in dict.items()]

data = pd.DataFrame(key_value_pairs)
data = shuffle(data)

X = data['SMILES'].apply(calculate_descriptors)
y = data['binds']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Fig. 2 Classification model training

Regression Model Training. The regression model was trained to predict the binding affinity score for a ligand and protein. The models were trained on smina docking scores of active binders and decoys from the DUD-E docking set for one of the 10 DUD-E targets. Each target had six models trained and benchmarked with the same sizes as used in the classification model. The trained ligands were given all descriptors by the RDKit Descriptors module[16] and fingerprints were not used in training. The scikit-learn random forest regressor class with default hyperparameters was used to train the model.

```
X = df['SMILES'].apply(calculate_descriptors)
y = df['Binding']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Fig. 3 Regression model training

Testing. smina and the machine learning surrogate model were both tested against the 10 targets in the DUD-E benchmarkings set. Each protein had sets of active ligands and decoys. The set was divided into two parts: a training set for the machine learning model and a testing set used to test both smina and the model. smina was tested on its default settings with exhaustiveness set to 8. Both smina and the machine learning model were run on Google Colab.

Throughput. Throughput was measured as the time it took for each model to score the set of ligands for each target. By using the DUD-E benchmarking set, throughput can be measured for a variety of different targets for each model. When testing the machine learning model, time taken to add descriptors to each molecule, time taken to train the model, and time taken to dock the training set for the regression model will be included in throughput time.

Accuracy. For the classification model, accuracy will be measured by the percentage of ligands correctly identified as an active binder or a decoy by smina and the machine learning model. Since smina outputs a binding score, the identification will be decided by finding if the value is more or less negative than the median score value. If the value is more negative than the median score value, it will be considered an active binder. If the value is less negative than the median score value, it will be considered a decoy binder. The machine learning classification model's binding prediction will be determined by the output given from the classification model. For the regression model, accuracy will be measured using the mean squared error, R2 score, and Spearman rank correlation for the model's output compared to the output determined by smina.

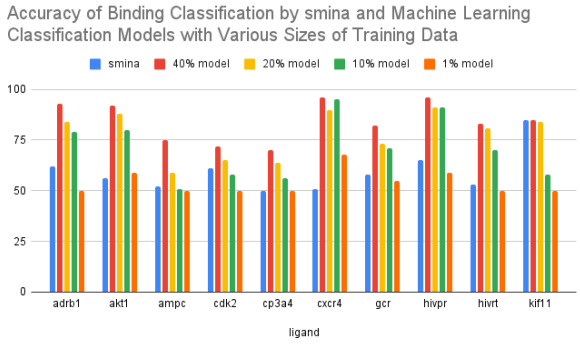
Results

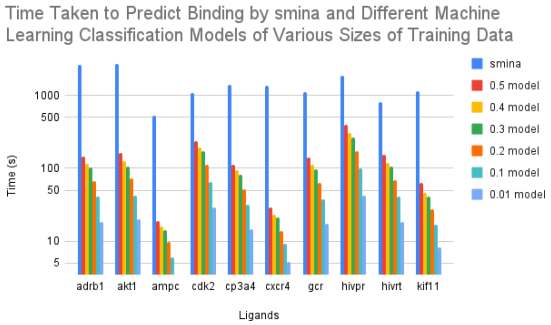
The classification model trained with 10% of the dataset achieved an accuracy of 70.9% $\pm$ 4.5%, outperforming smina's 59.2% $\pm$ 3% while having an 80 times increased throughput on average compared to smina. The differences in throughput were statistically significant (p=0.000215¡0.05) across all proteins. The classification model achieved an average precision of 100%, an average recall of 41.8% $\pm$ 9.1%, and an average F1 score of 53% $\pm$ 9.4%. Across all training sizes, the model had much more false negatives than false positives, having little to no false positives for each protein. Throughput across proteins was more consistent with the machine learning model while smina was fastest when scoring AMPC and slowest when scoring AKT1. When trained on 50% of the data, the accuracy increased to 85.7% $\pm$ 2.5% while still having a consistent 10 times increased throughput on average across proteins compared to smina. The differences in throughput were statistically significant (p=0.000149¡0.05) across all proteins. The classification model achieved an average precision of 100%, an average recall of 71.4% $\pm$ 5.0%, and an average F1 score of 82.2% $\pm$ 3.5%.

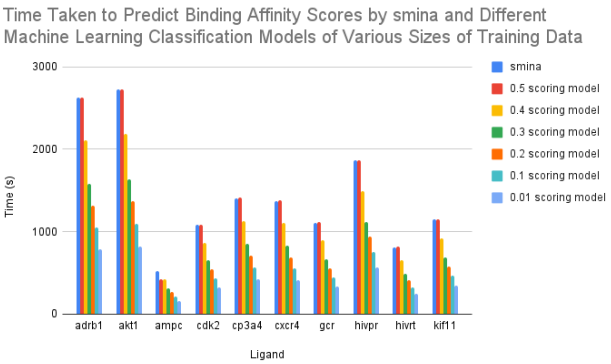
The regression model trained with 40% of the dataset achieved a strong correlation to smina with a consistently higher throughput across proteins, performing 20% faster on average in testing the entire dataset. The differences in throughput were statistically significant (p=0.000142$<$0.05) across all proteins. The average mean squared error of the regression model compared to smina is 2.11 $\pm$ 0.91 and the average R^2 value compared to smina is 0.47 $\pm$ 0.045. When trained with less than 40% of the data, the correlation between score given by the scoring model and the score given by smina reduced. When given less than 10% of the data to train, the model's accuracy became weaker.

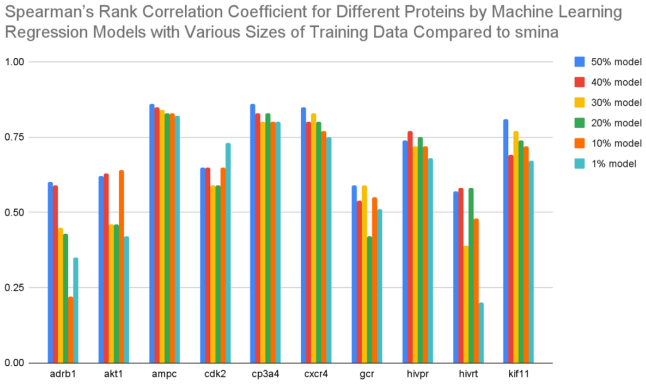
smina was most accurate when scoring the KIF11 protein and

least accurate when scoring the CP3A4 protein. The machine learning classification models were most accurate when scoring the AKT1 and HIVPR proteins and least accurate when scoring the AMPC and CP3A4 proteins. The machine learning regression models had binding affinity scores most correlated with smina when scoring the AMPC and CP3A4 proteins and least accurate when scoring the GCR and HIVRT proteins.









Discussion

The goal of this problem was to demonstrate a higher throughput alternative to molecular docking in order to accelerate the process of virtual screening. The model addresses the low throughput of molecular docking, allowing for high-throughput screening of billion-scale libraries previously computationally infeasible.

For both machine learning models, when more data was used to train the model, training time increased, causing an increase to the time taken to score the dataset. The classification model had about a 25 second increase in runtime per 10% increase in training size. The regression model, which required the training set to be docked before training, had about a 300 second increase in runtime per 10% increase in training size. For all the different sizes of training data used except for the 50% training size regression model, the model was able to score the dataset with a throughput much higher than that of smina. With a large dataset of 50% of the set used to train, the classification model was over 10 times faster than smina. When the size of the training set was reduced to a 10% section of the dataset, the model was over 80 times faster at screening the entire dataset.

The classification model was able to reach performance on par with the performance of smina when trained on at least 10% of the dataset. On average, this model took about .657 seconds per ligand, making it able to screen the REAL compounds library of 48 billion compounds in about 8,700 hours when using 1,000 computers, over 45 times faster than the time necessary for smina to dock the compounds. When trained on more data, the classification model was able to outperform smina in identifying active binders and decoys, having an accuracy proportional to the square root of the amount of data given, with diminishing returns above a training size of 40%. When trained on less data, the classification model began to have lower accuracy than smina. While the classification model was able to outperform smina with little data, the regression scoring model required to be trained with at least 40% of the dataset in order to have comparable performance to smina. With this amount of data,

Table 1 Accuracy of Binding Classification for Different Models

	ADRB1	AKT1	AMPC	CDK2	CP3A4	CXCR4	GCR	HIVPR	HIVRT	KIF11
smina	0.62	0.56	0.52	0.61	0.50	0.51	0.58	0.65	0.53	0.84
50% model	0.94	0.95	0.77	0.79	0.73	0.95	0.85	0.95	0.81	0.83
40% model	0.93	0.92	0.75	0.72	0.70	0.96	0.82	0.96	0.83	0.85
30% model	0.89	0.92	0.63	0.71	0.68	0.96	0.76	0.94	0.74	0.82
20% model	0.84	0.88	**0.59**	**0.65**	0.64	0.90	0.73	0.91	0.81	**0.84**
10% model	**0.79**	**0.80**	0.51	0.58	**0.56**	0.95	**0.71**	**0.91**	**0.70**	0.58
1% model	0.50	0.59	0.50	0.50	0.50	**0.68**	0.55	0.59	0.50	0.50

Accuracy of the binding classification of smina and machine learning classification models trained on different sizes of subsets of empirical binding data. The model that was trained on the least amount of training data needed to match smina is bolded.

Table 2 Mean Squared Error Scores for Different Models

	ADRB1	AKT1	AMPC	CDK2	CP3A4	CXCR4	GCR	HIVPR	HIVRT	KIF11
50% scoring model	0.76	5.41	0.22	0.51	0.65	0.24	4.76	0.57	1.65	0.68
40% scoring model	0.74	8.91	0.23	0.53	0.73	0.33	6.62	0.61	1.49	0.93
30% scoring model	0.89	6.32	0.29	0.59	0.83	0.31	7.47	0.72	1.93	0.78
20% scoring model	0.97	6.82	0.29	0.59	0.71	0.28	6.32	0.61	1.79	0.83
10% scoring model	1.11	4.18	0.29	0.55	1.01	0.38	6.27	0.66	1.82	0.91
1% scoring model	1.09	6.59	0.28	0.59	1.04	0.37	6.37	0.74	2.39	0.96

Mean squared error scores of binding affinity scores from machine learning regression models trained on different sizes of subsets of empirical binding data compared to scores given by smina.

Table 3 R^2 Scores for Different Models

	ADRB1	AKT1	AMPC	CDK2	CP3A4	CXCR4	GCR	HIVPR	HIVRT	KIF11
50% scoring model	0.35	0.60	0.73	0.44	0.70	0.72	0.46	0.55	0.31	0.61
40% scoring model	0.37	0.35	0.72	0.42	0.66	0.62	0.25	0.52	0.38	0.46
30% scoring model	0.24	0.54	0.65	0.34	0.62	0.65	0.16	0.43	0.20	0.55
20% scoring model	0.17	0.50	0.65	0.34	0.67	0.68	0.29	0.52	0.26	0.52
10% scoring model	0.05	0.70	0.65	0.39	0.53	0.57	0.29	0.48	0.24	0.48
1% scoring model	0.07	0.52	0.66	0.35	0.52	0.57	0.28	0.42	0.008	0.44

R^2 scores of binding affinity scores from machine learning regression models trained on different sizes of subsets of empirical binding data compared to scores given by smina.

Table 4 Spearman's Rank Correlation Coefficient for Different Models

	ADRB1	AKT1	AMPC	CDK2	CP3A4	CXCR4	GCR	HIVPR	HIVRT	KIF11
50% scoring model	0.60	0.62	0.86	0.65	0.86	0.85	0.59	0.74	0.57	0.81
40% scoring model	0.59	0.63	0.85	0.65	0.83	0.80	0.54	0.77	0.58	0.69
30% scoring model	0.45	0.46	0.84	0.59	0.80	0.83	0.59	0.72	0.39	0.77
20% scoring model	0.43	0.46	0.83	0.59	0.83	0.80	0.42	0.75	0.58	0.74
10% scoring model	0.22	0.64	0.83	0.65	0.80	0.77	0.55	0.72	0.48	0.72
1% scoring model	0.35	0.42	0.82	0.73	0.80	0.75	0.51	0.68	0.20	0.67

Spearman's rank correlation coefficient of binding affinity scores from machine learning regression models trained on different sizes of subsets of empirical binding data compared to scores given by smina.

the regression model tended to predict a less negative binding affinity for ADBR1, AKT1, CDK2, CP3A4, and HIVRT while predicting a more negative binding affinity for AMPC, CXCR4, GCR, HIVPR, and KIF11. When given less data, the model's accuracy became weaker.

Both the classification and regression scoring model are able to be used to accelerate the performance of virtual screening by providing an alternative to molecular docking without sacrificing accuracy. Both models are able to assess the binding of a ligand to a protein with a much higher throughput than that of traditional molecular docking programs such as smina. When given at least 10% of the dataset for the classification model and 40% of the dataset for the regression model, the models were able to have an accuracy similar to that of smina while having a 80x increased throughput for the classification model and a 20% increased throughput for the regression model. When given more data, the models can have an accuracy higher than smina while still having a higher throughput.

These models were able to accelerate the throughput of screening proteins by avoiding the slow calculations done by molecular docking. Molecular docking programs analyze the structure of the target protein and ligand in order to simulate how they will interact. In contrast, the machine learning model does not do any simulations. Instead, the model predicts the binding of the protein and ligand by analyzing specific attributes of the ligand and comparing them to known actives and decoys. Because the model is not doing any slow simulations, it is understandable that the model is able to outperform molecular docking.

While the models did have an increase in throughput, they are limited as they need to have experimentally confirmed binders as training data for the classification model. While the regression model did not need experimentally confirmed binders, it did need the use of slow docking of training sets to create training data. When used in cases where experimentally confirmed binders are not available, the regression model is required to be used, requiring the docking of training sets. This increases the runtime and computational cost of the machine learning model.

When dealing with libraries of ligands that are constantly growing, screening the growing number of ligands will take more and more time as time goes on. Processing billions of ligands per target using molecular docking is very computationally costly and takes large amounts of time. Current machine learning algorithms such as DiffDock are computationally intensive, making them unfit for large libraries. By employing the machine learning surrogate models to classify and score the binding of molecules to binders, both computational cost and time taken by virtual screening can be substantially reduced. By doing this, large libraries can be screened much faster, making it easier to identify molecules that bind to a certain target when developing drugs with a specific target.

Conclusion

The results in this paper demonstrate a new application of random forest models in accelerating virtual screening with minimal loss in accuracy, allowing for the screening of billion-scale libraries. The machine learning model trained on binding data of various ligands and proteins can classify ligands as actives or decoys and score binding affinity of ligand-protein interactions with a much higher throughput than that of traditional molecular docking tools. The model can classify ligands with a speed up to 80 times faster than that of smina, a traditional molecular docking program, without sacrificing accuracy. The model, scaling up to larger libraries, can screen a library of 48 billion compounds in about 8,700 hours using 1,000 computers. The classification model requires around 10% of the dataset as training data for similar performance to smina while the scoring regression model requires around 40% of the dataset as training data. Because of this, the model can be used as a surrogate model for virtual screening to predict docking and docking scores for the binding of ligands to proteins in much shorter times than traditional docking software, drastically shortening drug discovery timelines.

While the model in its current state does not require much data in order to classify ligands as active binders or decoys for a certain protein with an accuracy on par with smina, it requires more data in order to score binding affinity for ligand-protein interactions. With a larger set of available binding affinity scores, the percentage of scores needed for training could be reduced. However, larger sets of binding affinity scores require large amounts of molecular docking scores, which take long amounts of time to score. Real-world implementation would require validation of accuracy for unseen protein targets to ensure generalizability across targets. Future developments could explore adaptive learning techniques, updating the model as new docking data becomes available, and integrating deep-learning methods to improve accuracy and scalability. The machine learning method can also be expanded to computational challenges outside of docking tasks, such as prediction molecular properties and other chemistry challenges. Overall, the machine learning model can be used as a surrogate model in order to virtually screen large libraries of ligands and proteins with a much higher throughput than that of molecular docking. This makes it easier to screen large libraries of molecules for drug discovery when attempting to find a ligand for a certain target protein.

Acknowledgment

Thank you for the guidance of Ayush Pandit from Stanford University in the development of this research paper.

References

1 A. Sertkaya, T. Beleche, A. Jessup and B. D. Sommers, *Costs of Drug Development and Research and Development Intensity in the US, 2000-2018*, 2024.

2 J. P. Hughes, S. Rees, S. B. Kalindjian and K. L. Philpott, *Principles of early drug discovery*, 2011.

3 B. K. Shoichet, *Virtual screening of chemical libraries*, 2004.

4 J. M. Paggi, A. Pandit and R. O. Dror, *The Art and Science of Molecular Docking*, 2024.

5 O. Trott and A. J. Olson, *AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading*, 2011.

6 D. R. Koes, M. P. Baumgartner and C. J. Camacho, *Lessons learned in empirical scoring with smina from the CSAR 2011 benchmarking exercise*, 2013.

7 Enamine, *REAL Compounds*, 2024.

8 J. Lyu, J. J. Irwin and B. K. Shoichet, *Modeling the expansion of virtual screening libraries*, 2023.

9 K. M. Wong, H. K. Tai and S. W. I. Siu, *GWOVina: A grey wolf optimization approach to rigid and flexible receptor docking*, 2021.

10 B. Lei, T. Q. Kirk and A. Bhattacharya, *Bayesian optimization with adaptive surrogate models for automated experimental design*, 2021.

11 C. Angione, E. Silverman and E. Yaneske, *Using machine learning as a surrogate model for agent-based simulations*, 2022.

12 F. T. Zahura, J. L. Goodall, J. M. Sadler, Y. Shen, M. M. Morsy and M. Behl, *Training Machine Learning Surrogate Models From a High-Fidelity Physics-Based Model: Application for Real-Time Street-Scale Flood Prediction in an Urban Coastal Community*, 2020.

13 L. Breiman, *Random Forests*, 2001.

14 S. K. Dasari, A. Cheddad and P. Andersson, *Random Forest Surrogate Models to Support Design Space Exploration in Aerospace Use-Case*, 2019.

15 G. Corso, H. Stark, B. Jing, R. Barzilay and T. Jaakkola, *DiffDock: Diffusion steps, twists, and turns for molecular docking*, arXiv preprint arXiv:2210.01776, 2022.

16 M. M. Mysinger, M. Carchia, J. J. Irwin and B. K. Shoichet, *Directory of Useful Decoys, Enhanced (DUD-E): Better Ligands and Decoys for Better Benchmarking*, 2012.

17 G. Landrum, *Descriptor Calculation*, 2019.

Appendix

GitHub repository: https://github.com/Jaden-McCarney/Models

The three files located in the GitHub repository are the code for the two linear and neural network models as well as the scores table, which contains the average meal ADP and average SAT section scores for each district in New York.