Navigating the Molecular Maze: A Python-Powered Approach to Virtual Drug Screening

John Raicu
University of Chicago
johnny.raicu@gmail.com

Valerie Hayot-Sasson, Kyle Chard, Ian Foster
University of Chicago
{vhayot,chard,foster}@uchicago.edu

Abstract

The COVID-19 pandemic has highlighted the power of using computational methods for virtual drug screening. However, the molecular search space is enormous and protein docking methods are still computationally intractable without access to the world's largest supercomputers. Instead, researchers are using AI methods to help guide docking campaigns. In such approaches, a lightweight surrogate model is trained and then used to identify promising candidates for screening. We present ParslDock, a Python-based pipeline using the Parsl parallel programming library and the K-Nearest Neighbors machine learning model to screen a huge molecular space of molecules against arbitrary receptors. We achieved a 38X speedup with ParslDock compared to a brute-force docking approach.

Keywords: Protein Docking, Parsl, KNN, Machine Learning

1 Introduction

Protein docking is a key computational method used in molecular biology to predict the structure of protein complexes formed when two or more proteins interact. Essentially, protein docking involves simulating the process by which proteins fit together or 'dock' to form a stable complex. Docking scoring functions estimate the binding free energy between a ligand and a protein. The lower the score, the more energetically favorable the binding interaction.

A typical docking computation can take over 10 minutes on 1-core; a typical workload involves multiple protein receptors (e.g., we found studies with 15 receptors) with millions of possible ligands to dock to (e.g., we found studies with 13M ligands), yielding a total compute complexity for a brute force approach to be over 32M CPU-hours (3710 years in a serial computation on a single core). [2]

We present ParslDock—a Python-based system to make docking tractable on even modest computing systems. We integrate machine learning (ML) methods to help simplify and speed up the docking search process. We also make use of parallel libraries to make use of high-performance computing (HPC) systems with hundreds of cores.

SC '23, November 12–17, 2023, Denver, CO
2023. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 Proposed Solution

We propose ParslDock, an ML-based system that enables scalable and flexible screening of molecules. We assume that users provide a target receptor (in PDBQT format) and a list of candidate molecules (as SMILES strings). We aim for ParslDock to efficiently use allocated resources (e.g., from parallel or distributed systems) combining ML models and Monte Carlo Simulations.

We used AutoDock Vina (a leading docking implementation) that performs docking and utilizes a scoring function and gradient-based optimization algorithm. [4]

We used the Parsl parallel programming library to execute parts of the application in parallel. The power of Parsl lies in its ability to manage and coordinate the execution of tasks across a range of computing resources, from multicore workstations to HPC systems. [1]

We used a K-Nearest Neighbor (KNN) model. [3] Each molecule (a SMILES string) is converted into a Morgan Fingerprint that is represented as a bit-vector. We computed the euclidean distance matrix in the KNN as an all-to-all distance matrix between every molecule to every other molecule's bit-vector representation. Our ultimate goal is to find a good mapping between fingerprints and docking score.

The ParslDock pipeline (Figure 1) is composed of 7 stages:

1. Dataset 4M: Input data is a target receptor (PDBQT) and a random list of 100K molecules (SMILES)
2. Data Format: Data preparation for AutoDock Vina docking; SMILES string to PDB file to PDBQT file
3. Docking: Executes Monte Carlo simulations between a protein receptor PDBQT file and a ligand PDBQT file, yielding a binding-affinity score output
4. Morgan Fingerprints: Generated 128-bit vector with a depth of 8 from a SMILES string [6]
5. Machine Learning: Morgan Fingerprints and docking scores are paired as the input to KNN model to identify correlations between fingerprints and docking scores
6. Dataset Top-4K: The top 4K ligands identified based on the predicted docking scores (lowest binding-affinity scores) of all 4M molecules
7. Docking: Runs Monte Carlo simulations on the Top-4K subset of data to obtain a final list of top molecules ordered by docking scores

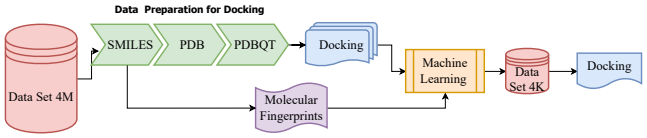


Figure 1. ParslDock pipeline

Most of these steps can be run concurrently across a set of input molecules. We implement each step as a Parsl app and establish dependencies between each step. This allows us to scale out execution of each step across input data. All our code and datasets can be found on Github. [5, 7]

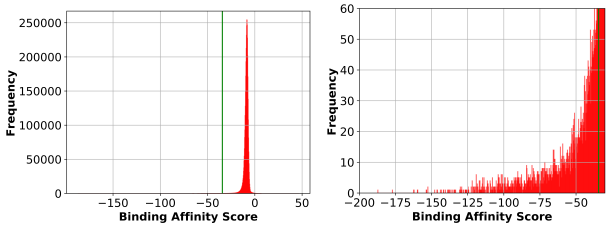
3 Evaluation

We conduct experiments on two testbeds: an 8-core laptop (8c-laptop) and a 192-core server (192c-server).

We used 4 million ligands from the 8 million ligand dataset and performed docking against the 1IEP receptor protein. This resulted in a new dataset that included 4M scores that are represented in the Figure 2. The scores follow a normal distribution. Our goal was to identify the top 0.1% of the ligands based on the docking score (green line at -34.32).

KNN. We selected 100K random ligands and used the docking scores to build a KNN model. We found that using 100K samples yielded highly accurate (99%+) results, this was due to the unbalanced dataset (100 vs 99,900 samples). We balanced the dataset by choosing 100 of each. Larger samples did not significantly improve accuracy. We explored different values of dataset sizes (from 10K to 10M samples), and determined that 100K offers good enough accuracy of 87%. The number of neighbors we settled on for the KNN was 5. Different values of K did not yield significantly and consistently better results.

Fingerprints. We conducted a hyperparameter search of the "depth" (radius around atoms) and "size" (number of bits) of fingerprint. Figure 3 shows the accuracy of the KNN model on a training/testing dataset partitioned in 70% training and 30% testing. The best configuration (yellow) is when size is 128 and depth is 8, achieving an accuracy of 87%. Note that sub-optimal parameters for the Morgan Fingerprint can easily yield under 50% accuracy.



(a) All scores **(b)** Zoomed view

Figure 2. Binding affinity score for docking of 4M molecules.

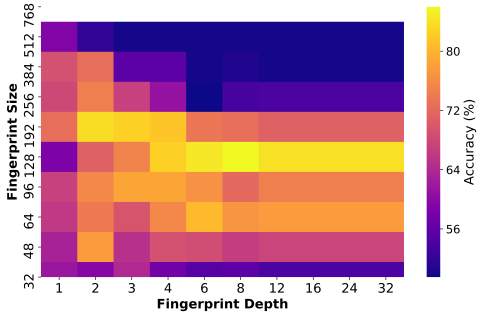


Figure 3. 2D-Heatmap showing KNN accuracy for fingerprint size vs depth

Performance. Figure 4 shows the performance of the entire pipeline on two systems. Most stages used Parsl to parallelize execution across the multicore systems in our testbed (the only exception was the building of the KNN, which was a significant part of the total pipeline). The speedup obtained on ParslDock compared to the brute force docking of the entire 4M ligands was 38X on the laptop and 37X on the server. Note that the brute force results for both systems as well as the ParslDock on the 16-HT system are estimated based on a smaller sub-sample of results. The ParslDock on the 384-HT system was computed in its entirety.

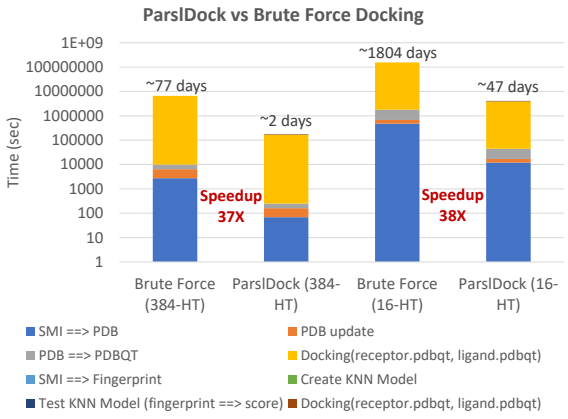


Figure 4. ParslDock and Brute Force Docking end-to-end pipeline execution time

4 Conclusions

Protein docking is a complex process that can be computationally expensive and time-consuming. ParslDock is a Python-powered automated pipeline that uses ML to accelerate the docking process and reduce compute costs by up to 38X. Our performance evaluation showed linear scalability from an 8-core laptop to a 192-core server. With further improvements, we believe we can bring down the computational requirements to the point that ParslDock will be tractable on a modern day personal computer.

References

[1] Yadu Babuji, Anna Woodard, Zhuozhao Li, Daniel S. Katz, Ben Clifford, Rohan Kumar, Lukasz Lacinski, Ryan Chard, Justin M. Wozniak, Ian Foster, Michael Wilde, and Kyle Chard. 2019. Parsl: Pervasive Parallel Programming in Python. In *Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing* (Phoenix, AZ, USA) *(HPDC '19)*. Association for Computing Machinery, New York, NY, USA, 25–36. https://doi.org/10.1145/3307681.3325400

[2] Austin Clyde, Thomas Brettin, Alexander Partin, Hyunseung Yoo, Yadu Babuji, Ben Blaiszik, Andre Merzky, Matteo Turilli, Shantenu Jha, Arvind Ramanathan, et al. 2021. Protein-ligand docking surrogate models: A sars-cov-2 benchmark for deep learning accelerated virtual screening. *arXiv preprint arXiv:2106.07036* (2021).

[3] T. Cover and P. Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13, 1 (1967), 21–27. https://doi.org/10.1109/TIT.1967.1053964

[4] Jerome Eberhardt, Diogo Santos-Martins, Andreas F Tillack, and Stefano Forli. 2021. AutoDock Vina 1.2. 0: New docking methods, expanded force field, and python bindings. *Journal of chemical information and modeling* 61, 8 (2021), 3891–3898.

[5] Valerie Hayot-Sasson, John Raicu, Yadu Babuji, and Kyle Chard. 2023. ParslDock Tutorial. https://github.com/Parsl/parsl-docking-tutorial/blob/main/ParslDock.ipynb.

[6] Lagnajit Pattanaik and Connor W Coley. 2020. Molecular representation: going long on fingerprints. *Chem* 6, 6 (2020), 1204–1207.

[7] John Raicu. 2023. ParslDock Repository. https://github.com/johnny-raicu/ParslDock.