Machine Learning Boosted Docking (HASTEN): An Open-Source Tool To Accelerate Structure-based Virtual Screening Campaigns

Tuomo Kalliokoski[1]*

[1]*Orion Pharma, Orionintie 1A, 02101 Espoo, Finland*

*E-mail: tuomo.kalliokoski@orionpharma.com

Abstract

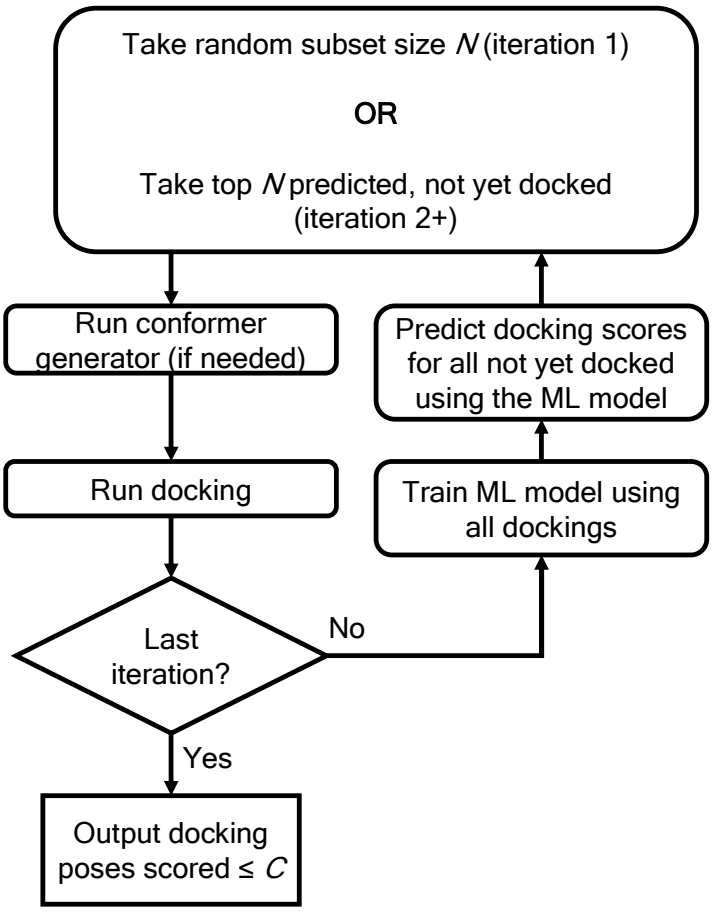
The software macHine leArning booSTed dockiNg (HASTEN) was developed to accelerate structure-based virtual screening using machine learning models. It has been validated using datasets both from literature (12 datasets, each containing three million molecules docked with FRED) and in-house sources (one dataset of four million compounds docked with Glide). HASTEN showed reasonable performance by having the mean recall value of 0.78 of the top one percent scoring molecules after docking 10 % of the dataset for the literature data, whereas excellent recall value of 0.95 was achieved for the in-house data. The program can be used with any docking- and machine learning methodology, and is freely available from https://github.com/TuomoKalliokoski/HASTEN.

Keywords: docking, machine learning, structure-based virtual screening

Structure-based virtual screening (SBVS) is one of the standard tools used in modern drug discovery [1]. In SBVS, from thousands to up to a billion compounds are docked into the structure of a drug target, which is usually a protein structure that has been either experimentally determined or computationally predicted. Each docked compound is scored based on its fit to the target using a scoring function, which enables the ranking of compounds based on their docking score in ascending order. Only the lowest scoring compounds are taken into further consideration, and these molecules are called virtual hits. Although there are some truly impressive reports of large-scale studies using supercomputers [2, 3], all too often the throughput of SBVS remains limited, even with the rapid improvements in computer hardware, due to several factors. These factors can either be scientific, such as the need to use multiple conformations of the target due to its flexibility, or economic, for example the costs associated with cloud computing and software license fees. One alternative to speed up SBVS is to, instead of running real docking calculations, predict the docking score from the structures of molecules alone using a model that has been trained on a dataset of previously calculated docking scores. Such predictions are several order of magnitudes faster than docking calculations. Thanks to the advances in Deep Learning [4, 5], Machine learning (ML) can be used to accelerate SBVS considerably, and enable large screening campaigns without the need for vast supercomputing capabilities [6]. For example, a recently published ML-platform for SBVS called Deep Docking (DD) was shown to be able to speed up the docking screens by up to 50 times.

The general ML for SBVS process is illustrated in Figure 1. In such approach an random subset of N molecules is first selected from the database to be docked ($N \ll$ the number of molecules in the database). Often three-dimensional conformers have to be generated before the docking step, which adds up to the computational cost of docking. A ML model is then trained on this docking data, together with quickly calculable descriptors for the molecules. This ML model is then used to rank the large number of molecules in the database not yet docked. A new set of N molecules is selected for docking from this ranked list and the process is repeated as long as new good scoring compounds are identified, or there is time left to run the screen using the computational resources at hand. Finally, all molecules with maximum docking score of C are selected as virtual hits. The most time consuming part of the workflow is clearly the docking step, which means that N should be kept as low as possible, while at the same time being able to produce ML models that have adequate accuracy to predict docking scores.

Figure 1. Flowchart on how to apply machine learning to structure-based virtual screening process.



In this study, a software tool called macHine leArning booSTEd dockiNg (HASTEN) was developed. HASTEN enables the easy execution of ML-boosted SBVS-virtual screening campaigns and can be used together with any docking- and ML-methodology. This is the strength of HASTEN when compared to the previously published program DD: HASTEN allows the ML method to be easily replaced depending on the needs of the user. During the preparation of this manuscript, a somewhat similar tool to HASTEN called MolPal was released [7]. One of the differences between HASTEN and MolPal is that there is built-in support for the widely used commercial docking program Glide in HASTEN. This study also provides novel data on the performance of machine-learning boosted docking when it is used together with docking constraints, which are often used in prospective virtual screening. HASTEN is freely available without any restrictions at https://github.com/TuomoKalliokoski/HASTEN.

Upon development, HASTEN was validated using both literature and in-house data. The literature data for 12 targets were retrieved from the study by Gentile et al [6] and the in-house data of a virtual screen for a histone acetyltransferase (HAT) target were from a previously performed prospective screen. The literature data consisted of approximately three million compounds for each target from the ZINC database [8] that were docked using the docking program FRED [9]. Whereas the in-house data consisted of approximately four million compounds from the Enamine REAL database that were docked using the docking program Glide [10], with a core constraint enabled that forced a part of the molecule to be docked to be in a certain reference space. As the usage of the core constraint indicates, all molecules in the in-house dataset therefore shared the same structural feature. The molecules in the in-house dataset that failed to dock, and thus had no docking score were assigned with a docking score of 10.0. The calculations were performed using three CentOS 7 Linux-workstations that had NVIDIA GeForce RTX 2080 Ti as GPU and either AMD Ryzen Threadripper 2970WX CPU with 125 GB RAM or Intel Xeon Gold 5118 CPU with 186 GB of RAM.

The ML models were built using ChemProp, which is a message passing neural network for molecular property prediction. ChemProp is freely available and has been shown to be useful within the context of drug discovery [11, 12]. The default ChemProp settings were used. The datasets were split randomly into training (80 %), validation (10 %) and test sets (10 %) at each iteration step unlike in the previous DD study where compounds were added only to the training set [6]. At each iteration step, 1 % of the top predicted compounds (approximately 30000 to 40000 molecules) were added to the ML model building process. The performance was measured by calculating the recall of the top scoring one percent molecules in the whole dataset after each iteration step. The baseline of random recall is then the same as the percentage of dataset docked, as can been seen in Table 1, where the recall values at first iteration are based on the random selection before the application of the ML model. The effect of iterative training was measured by comparing the recall values at each iteration step to the recall values of the original starting point model.

HASTEN achieved mean recall of 0.78 at 10 percent of the dataset docked for the literature datasets whereas the performance for the in-house dataset was considerably higher with recall of 0.95 at 10 percent of the dataset docked (Table 1). The iterative model training process improved the recall of the top one percent scoring molecules on average 0.15 compared to the initially trained model, which is clearly an improvement worth the modest

additional computational cost associated with the additional model training calculations. No further optimization of the ChemProp settings was done as the performance was deemed already to be satisfactory using the default settings. The increase in recall values does not converge for literature targets at the 10th iteration, whereas for the in-house dataset there is a modest increase in recall from the eighth iteration. This suggests that in a real screening situation, the number of virtual hits should be monitored after each iteration step to avoid running unnecessary iterations or on the other hand, stopping too early.

The literature dataset contained targets that were nuclear receptors, kinases, G-protein coupled receptors and ion channels. Ion channel set 6D6T (gamma-aminobutyric acid receptor type A, GABAA) had clearly the lowest recall and this set was also the worst performing target in the previous DD study [6]. The authors commented on the poorer retrieval rate with GABAA that *"These results clearly suggest that, like any other computational tool, DD shows performances that are target-dependent"* which could also partially explain the major difference in performance between the literature targets and the in-house target observed here. An additional explanation for the difference could be that the initial training dataset for the in-house target was 33 % larger than the training datasets used for the literature targets. However, this may not to be the case here as the recall remained at the same level of 0.95 when repeating the HASTEN procedure on a dataset that was sampled down to three million compounds (HAT 3 mil) from the original four million compound dataset (HAT 4 mil) (Table 1).

Table 1. The recall values of top one percent scoring molecules at different fractions of database docked.

Literature datasets docked with FRED																		
% Docked	*1 %*	*2 %*	*3 %*		*4 %*		*5 %*		*6 %*		*7 %*		*8 %*		*9 %*		*10 %*	
Model	*Random*	*Iter 1*	*Iter 2*	*Iter 1*	*Iter 3*	*Iter 1*	*Iter 4*	*Iter 1*	*Iter 5*	*Iter 1*	*Iter 6*	*Iter 1*	*Iter 7*	*Iter 1*	*Iter 8*	*Iter 1*	*Iter 9*	*Iter 1*
1ERR	0.01	0.224	0.393	0.334	0.516	0.414	0.600	0.476	0.665	0.529	0.718	0.573	0.758	0.609	0.789	0.641	0.817	0.670
1T7R	0.01	0.244	0.451	0.367	0.569	0.451	0.649	0.517	0.712	0.570	0.760	0.616	0.799	0.656	0.829	0.688	0.851	0.716
2ZV2	0.009	0.162	0.308	0.253	0.414	0.319	0.504	0.376	0.574	0.425	0.630	0.467	0.672	0.505	0.708	0.538	0.740	0.569
4AG8	0.01	0.231	0.416	0.351	0.545	0.437	0.629	0.505	0.692	0.559	0.739	0.605	0.780	0.644	0.811	0.678	0.836	0.705
4F8H	0.01	0.203	0.415	0.308	0.553	0.383	0.644	0.446	0.707	0.498	0.754	0.541	0.792	0.579	0.825	0.613	0.849	0.641
4R06	0.01	0.209	0.361	0.310	0.471	0.380	0.548	0.437	0.606	0.486	0.652	0.525	0.692	0.561	0.726	0.591	0.758	0.620
4YAY	0.01	0.167	0.341	0.258	0.463	0.331	0.544	0.388	0.608	0.438	0.658	0.480	0.700	0.519	0.733	0.553	0.760	0.583
5EK0	0.009	0.242	0.420	0.350	0.541	0.429	0.621	0.491	0.683	0.542	0.730	0.583	0.766	0.620	0.797	0.650	0.825	0.676
5L2S	0.01	0.173	0.321	0.267	0.429	0.338	0.519	0.396	0.579	0.446	0.633	0.488	0.676	0.525	0.711	0.559	0.743	0.591
5MZJ	0.01	0.170	0.325	0.265	0.435	0.335	0.528	0.393	0.594	0.443	0.649	0.486	0.691	0.523	0.731	0.554	0.761	0.583
6D6T	0.01	0.108	0.223	0.173	0.322	0.227	0.402	0.274	0.468	0.314	0.523	0.349	0.572	0.380	0.607	0.413	0.640	0.440
6IIU	0.01	0.152	0.291	0.237	0.395	0.303	0.481	0.358	0.553	0.403	0.610	0.444	0.652	0,480	0.688	0.513	0.720	0.542
Mean	0.01	0.190	0.355	0.289	0.471	0.362	0.556	0.421	0.620	0.471	0.671	0.513	0.713	0.550	0.746	0.583	0.775	0.611
Median	0.01	0.188	0.351	0.287	0.467	0.359	0.546	0.416	0.607	0.466	0.655	0.507	0.696	0.543	0.732	0.575	0.760	0.605
In-house dataset docked with core-constrained Glide																		
% Docked	*1 %*	*2 %*	*3 %*		*4 %*		*5 %*		*6 %*		*7 %*		*8 %*		*9 %*		*10 %*	
Model	*Random*	*Iter 1*	*Iter 2*	*Iter 1*	*Iter 3*	*Iter 1*	*Iter 4*	*Iter 1*	*Iter 5*	*Iter 1*	*Iter 6*	*Iter 1*	*Iter 7*	*Iter 1*	*Iter 8*	*Iter 1*	*Iter 9*	*Iter 1*
HAT 4 mil	0.009	0.337	0.668	0.494	0.795	0.594	0.847	0.659	0.879	0.708	0.903	0.743	0.930	0.771	0.945	0.795	0.952	0.813
HAT 3 mil	0.01	0.336	0.649	0.506	0.794	0.606	0.855	0.672	0.884	0.720	0.911	0.754	0.923	0.781	0.937	0.801	0.947	0.818

The current version of HASTEN is ready to use with the Glide docking software and ChemProp machine learning tool, but it is trivial to write wrappers to add support for other methods as well. In addition to its use in large virtual screening campaigns, HASTEN could also be used in the further development of machine learning methods for docking, as the software contains a simulation mode allowing the input of pre-docked data that was used to produce the results reported here.

Acknowledgement

The author would like to thank Käthe Dahlström, Heikki Käsnänen and Aino-Leena Turku for their input. William Hennah is acknowledged for revising the language of the manuscript.

References

[1] E. Lionta, G. Spyrou, D. K. Vassilatis Z. Cournia, *Curr. Top. Med. Chem.* **2014**, 14, 1923.

[2] J. Lyu, S. Wang, T. E. Balius, I. Singh, A. Levit, Y S. Moroz, M. J. O'Meara, T. Che, E. Algaa, K. Tolmachova, A. A. Tolmachev, B. K. Shoichet,1, B. L. Roth, J. J. Irwin. *Nature* **2019**, 566, 224.

[3] C. Gorgulla, A. Boeszoermenyi, Z. Wang, P. D. Fischer, P. W. Coote, K. M. Padmanabha Das, Y. S. Malets, D. S. Radchenko, Y. S. Moroz, D. A. Scott, K. Fackeldey, M. Hoffmann, I. Iavniuk, G. Wagner, H. Arthanari. *Nature* **2020**, 580, 663.

[4] J. Shen, C. A. Nicolaou. *Drug Discov. Today Technol.* **2019**, 32, 29.

[5] W. P. Walters, R. Barzilay. *Acc. Chem. Res.* **2021**, 54, 263.

[6] F. Gentile, V. Agrawal, M. Hsing, A. Ton, F. Ban, U. Norinder, M. E. Gleave, A. Cherkasov, *ACS Cent. Sci.* **2020**, 6, 939.

[7] D. E. Graff, E. I. Shakhnovich, C. W. Coley, *arXiv* **2020***,* 2012.07127 [q-bio.QM].

[8] T. Sterling, J. J. Irwin. *J. Chem. Inf Model.* **2015**, 55, 2324.

[9] M. McGann. *J. Chem. Inf. Model.* **2011,** 51, 578.

[10] Schrödinger Release 2020-3: Glide, Schrödinger, LLC, New York, NY, 2020.

[11] K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, A. Palmer, V. Settels, T. Jaakkola, K. Jensen, R. Barzilay, *J. Chem. Inf Model.* **2020**, 59, 3370.

[12] J. M. Stokes, K. Yang, K. Swanson, W. Jin, A. Cubillos-Ruiz, N. M. Donghia, C. R. MacNair, S. French, L. A. Carfrae, Z. Bloom-Ackermann, V. M. Tran, A. Chiappino-Pepe, A. H. Badran, I. W. Andrews, E. J. Chory, G. M. Church, E. D. Brown, T. S. Jaakkola, R. Barzilay, J. J. Collins, *Cell* **2020**, 180, 688.