DATA-DRIVEN AND TASK-SPECIFIC SCORING FUNCTIONS FOR PREDICTING LIGAND BINDING POSES AND AFFINITY AND FOR SCREENING ENRICHMENT

By

Hossam M. Ashtawy

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Electrical Engineering – Doctor of Philosophy

2017

ABSTRACT

DATA-DRIVEN AND TASK-SPECIFIC SCORING FUNCTIONS FOR PREDICTING LIGAND BINDING POSES AND AFFINITY AND FOR SCREENING ENRICHMENT

By

Hossam M. Ashtawy

Molecular modeling has become an essential tool to assist in early stages of drug discovery and development. Molecular docking, scoring, and virtual screening are three such modeling tasks of particular importance in computer-aided drug discovery. They are used to computationally simulate the interaction between small drug-like molecules, known as *ligands*, and a target protein whose activity is to be altered. Scoring functions (SF) are typically employed to predict the binding conformation (docking task), binary activity label (screening task), and binding affinity (scoring task) of ligands against a critical protein in the disease's pathway. In most molecular docking software packages available today, a generic binding affinity-based (BA-based) SF is invoked for the three tasks to solve three different, but related, prediction problems. The vast majority of these predictive models are knowledge-based, empirical, or force-field scoring functions. The fourth family of SFs that has gained popularity recently and showed potential of improved accuracy is based on machine-learning (ML) approaches. Despite intense efforts in developing conventional and current ML SFs, their limited predictive accuracies in these three tasks have been a major roadblock toward cost-effective drug discovery. Therefore, in this work we present (i) novel task-specific and multi-task SFs employing large ensembles of deep neural networks (NN) and other state-of-the-art ML algorithms in conjunction with (ii) data-driven multi-perspective descriptors (features) for accurate characterization of protein-ligand complexes (PLCs) extracted using our Descriptor Data Bank (DDB) platform.

We assess the docking, screening, scoring, and ranking accuracies of the proposed task-specific SFs with DDB descriptors as well as several conventional approaches in the context of the 2007 and 2014 PDBbind benchmark that encompasses a diverse set of high-quality PLCs. Our approaches substantially outperform conventional SFs based on BA and single-perspective descriptors in all

tests. In terms of scoring accuracy, we find that the ensemble NN SFs, BsN-Score and BgN-Score, have more than 34% better correlation (0.844 and 0.840 vs. 0.627) between predicted and measured BAs compared to that achieved by X-Score, a top performing conventional SF. We further find that ensemble NN models surpass SFs based on other state-of-the-art ML algorithms. Similar results have been obtained for the ranking task. Within clusters of PLCs with different ligands bound to the same target protein, we find that the best ensemble NN SF is able to rank the ligands correctly 64.6% of the time compared to 57.8% obtained by X-Score. A substantial improvement in the docking task has also been achieved by our proposed docking-specific SFs. We find that the docking NN SF, BsN-Dock, has a success rate of 95% in identifying poses that are within 2 Å RMSD from the native poses of 65 different protein families. This is in comparison to a success rate of only 82% achieved by the best conventional SF, ChemPLP, employed in the commercial docking software GOLD. As for the ability to distinguish active molecules from inactives, our screening-specific SFs showed excellent improvements over the conventional approaches. The proposed SF BsN-Screen achieved a screening enrichment factor of 33.90 as opposed to 19.54 obtained from the best conventional SF, GlideScore, employed in the docking software Glide. For all tasks, we observed that the proposed task-specific SFs benefit more than their conventional counterparts from increases in the number of descriptors and training PLCs. They also perform better on novel proteins that they were never trained on before. In addition to the three task-specific SFs, we propose a novel multi-task deep neural network (MT-Net) that is trained on data from three tasks to simultaneously predict binding poses, affinities, and activity labels. MT-Net is composed of shared hidden layers for the three tasks to learn common features, task-specific hidden layers for higher feature representation, and three outputs for the three tasks. We show that the performance of MT-Net is superior to conventional SFs and competitive with other ML approaches. Based on current results and potential improvements, we believe our proposed ideas will have a transformative impact on the accuracy and outcomes of molecular docking and virtual screening.

To the memory of my uncle and role model, Hassan Abuhafa (1955-2015).

ACKNOWLEDGMENTS

I am indebted to many people who played a significant role in the completion of this dissertation. First and foremost, I would like to extend my deepest thanks and appreciation to my research advisor and mentor Prof. Nihar Mahapatra for his time, ideas, patience, and guidance that made my Ph.D. experience productive and stimulating. He taught me how to think critically and pay attention to detail. I would also like to thank my committee members Prof. Jin Chen, Prof. Fathi Salem, and Prof. Yanni Sun for their time in reviewing this work and insightful questions and feedback.

It goes without saying that none of this would have been possible without the support of my parents, Hasna and Mohamed, and the help of my wife Galya. I would like to thank them all for their encouragement and patience. I would also like to thank my 20-month-old daughter, Danya, for the joy that she brings into my life and the motivation she gives me to press forward.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

Protein-ligand binding affinity is the principal determinant of many vital processes, such as cellular signaling, gene regulation, metabolism, and immunity, that depend upon proteins binding to some substrate molecule [1]. Consequently, it is extensively used in molecular docking for rational drug design. Due to prohibitive costs and delays associated with experimental drug discovery, academia and pharmaceutical and biotechnology companies rely on virtual screening using computational molecular docking [2, 3, 4]. Typically, this involves docking tens of thousands to millions of ligand candidates into a target protein receptor's binding site and using a suitable scoring function (SF) to evaluate the binding affinity of each candidate to identify the top candidates and their poses as drug leads and then to perform lead optimization [3]; it is also used for target identification [5]. Besides drug discovery, the bioactive molecules thus identified can be used as chemical probes to investigate the biochemical role of a target of interest [6]. Molecular docking also has applications in many structural bioinformatics problems, such as protein structure [7] and function prediction [8]. It has become attractive because of the ever-increasing number of available receptor protein structures and putative ligand drug candidates in publicly-accessible databases, such as Protein Data Bank (PDB) [9], PDBbind [10], Cambridge Structural Database (CSD) [11], and corporate repositories.

Drug design involves two main steps: first, the enzyme, receptor, or other protein responsible for a disease of interest is identified; second, a small molecule or *ligand* is found or designed that will bind to the target protein, modulate its behavior, and provide therapeutic benefit to the patient. Typically, *high-throughput screening* (HTS) facilities with automated devices and robots are used to synthesize and screen ligands against a target protein. However, due to the large number of ligands that need to be screened, HTS is not fast and cost-effective enough as a lead identification

method in the initial phases of drug discovery [12]. Therefore, computational methods referred to as *virtual screening* are employed to complement HTS by narrowing down the number of ligands to be physically screened. In virtual screening, information such as structure and physicochemical properties of a ligand, protein, or both, are used to estimate *binding affinity* (or *binding free energy*), which represents the strength of association between the ligand and its receptor protein at the binding site. The most popular approach to predicting binding affinity (BA) in virtual screening is *structure-based* in which physicochemical interactions between a ligand and receptor are deduced from the 3D structures of both molecules. This *in silico* method is also known as *protein-based* as opposed to the alternative approach, *ligand-based*, in which only ligands that are biochemically similar to the ones known to bind to the target are screened. Since ligand-based screening does not directly take information about the target into account, it may not as easily identify novel chemicals as hits. Therefore, it is the method of choice when the 3D structure of the target is not available. With the unprecedented growth in the number of available 3D structures of protein-ligand complexes (PLCs) in the last decade, interest in the protein-based approach has increased. It is more accurate than the ligand-based approach due to the inclusion of shape and volume information extracted from the protein's 3D structure during the screening process [13, 14]. Figure 1.1 depicts a simplified view of the drug design process that involves both techniques.

The focus of the dissertation will be on protein-based drug design, wherein ligands are placed into the active site of the receptor during computational molecular docking. The 3D structure of a ligand, when bound to a protein, is known as *ligand active conformation*. *Binding mode* refers to the orientation of a ligand relative to the target and the protein-ligand conformation in the bound state. A *binding pose* is simply a candidate binding mode. In molecular *docking*, a large number of binding poses are generated and then evaluated using a *scoring function (SF)*, which is a mathematical or predictive model that produces a score representing the binding stability of the pose. The outcome of the docking run, therefore, is a ligand's top pose ranked according to its predicted binding score as shown in Figure 1.2. As shown in the figure, the molecule's bioactivity against the target protein is determined in the following screening step. Typically,

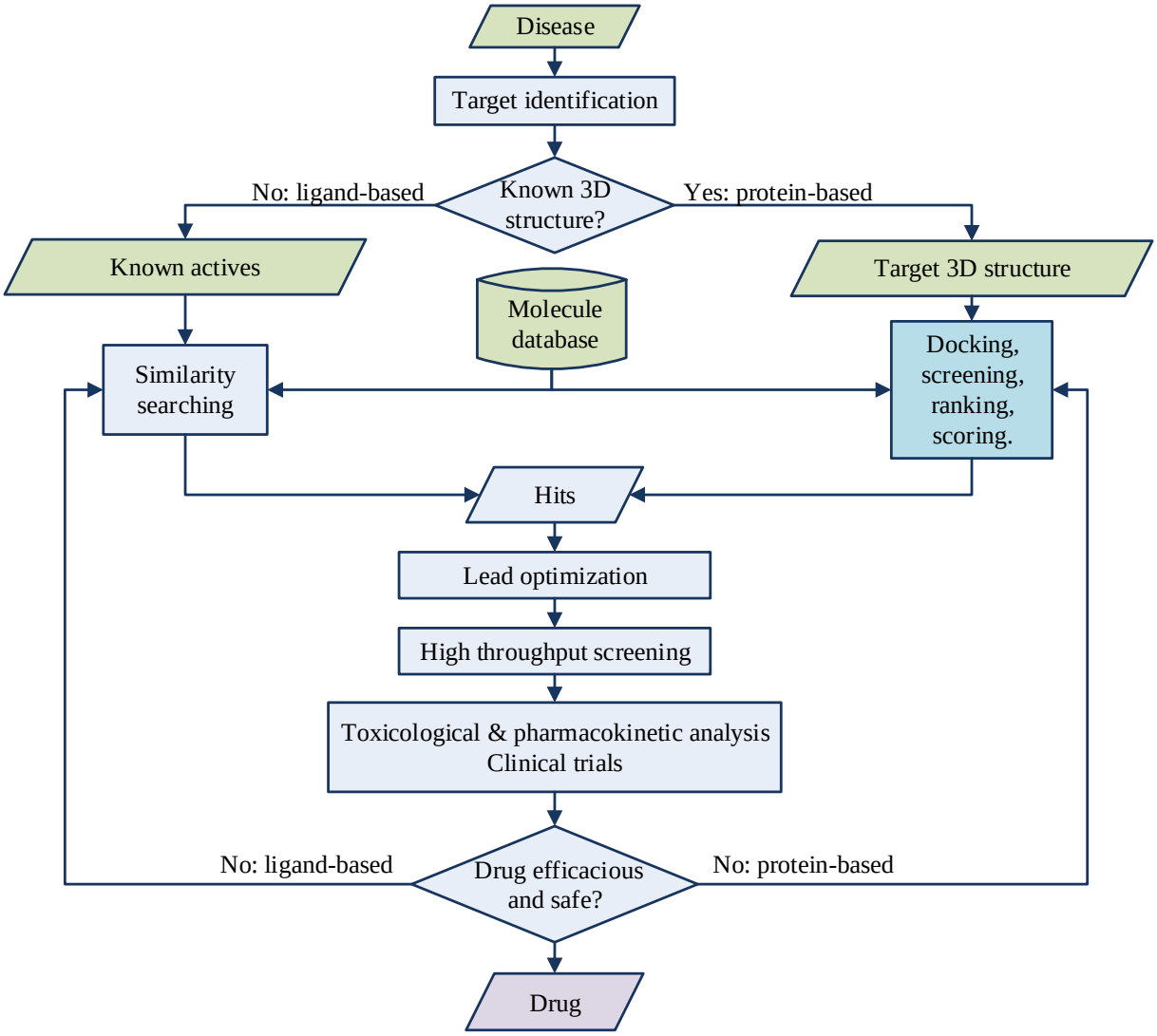


Figure 1.1: The drug design process.

these docking and screening steps are performed iteratively over a database containing thousands to millions of ligand candidates. After predicting their binding modes and bioactivities, ranking and scoring rounds are usually performed to rank ligands according to their predicted binding free energies. The top-ranked ligand, considered the most promising drug candidate, is synthesized and physically screened using HTS.

AIDS treatment provides a classic example of the promise of molecular docking. From the identification of HIV protease (receptor protein), an enzyme critical in the HIV virus' life cycle, as a drug target in 1988, and the determination of its X-ray crystallographic structure in 1989, to

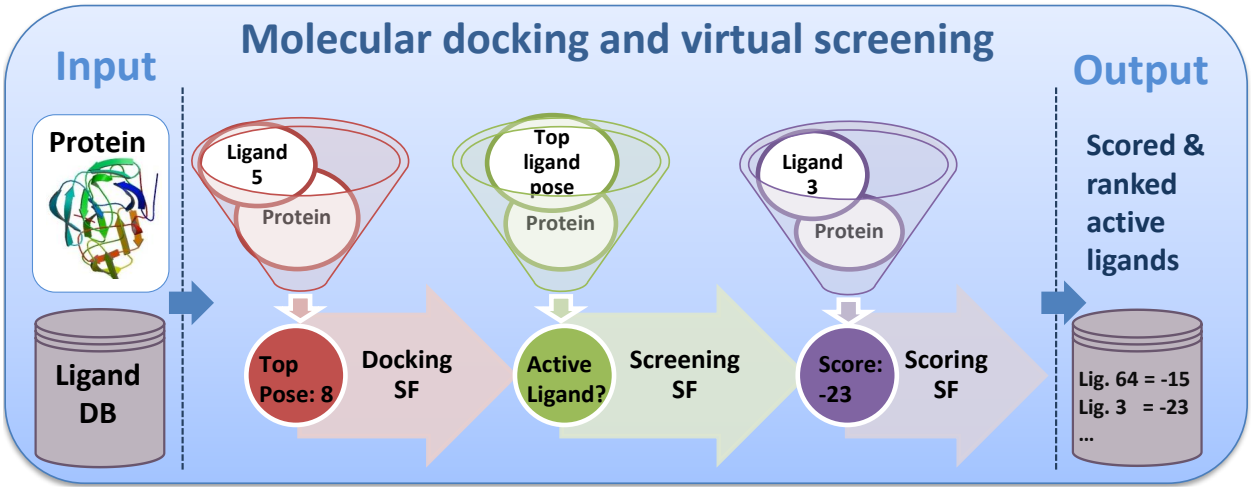


Figure 1.2: Protein-ligand docking, screening, scoring, and ranking workflow.

early 1996, it took less than 8 years to have three FDA-approved protease inhibitor drugs (ligands) on the market [15]. This contrasts sharply with the 10 to 15 years and hundreds of millions of dollars it typically takes using the traditional trial-and-error drug development process. Figure 1.3 shows the cost and time associated with the main stages of drug discovery based on data collected by PARAXEL several years ago [16].

1.2 Scoring function requirements

The most important steps in structure-based drug design are identifying the correct conformation of ligands at their respective binding sites, discriminating active ligands from those that do not bind, predicting their binding affinities and using these scores to rank ligands against each other. These core steps affect the outcome of the entire drug search campaign. That is because predictions of scoring functions determine which binding orientation/conformation is deemed the best, the ligand's bio-activity, which ligand from a database is considered likely to be the most effective drug, and the estimated absolute binding affinity. Correspondingly, four main capabilities that a reliable scoring function should have are: (i) the ability to identify the correct binding mode of a ligand from among a set of (computationally-generated) poses, (ii) the ability to identify whether a ligand is active or not, (iii) the ability to produce binding scores that are (linearly) correlated to the experimentally-determined binding affinities of protein-ligand complexes (PLCs) with known 3D

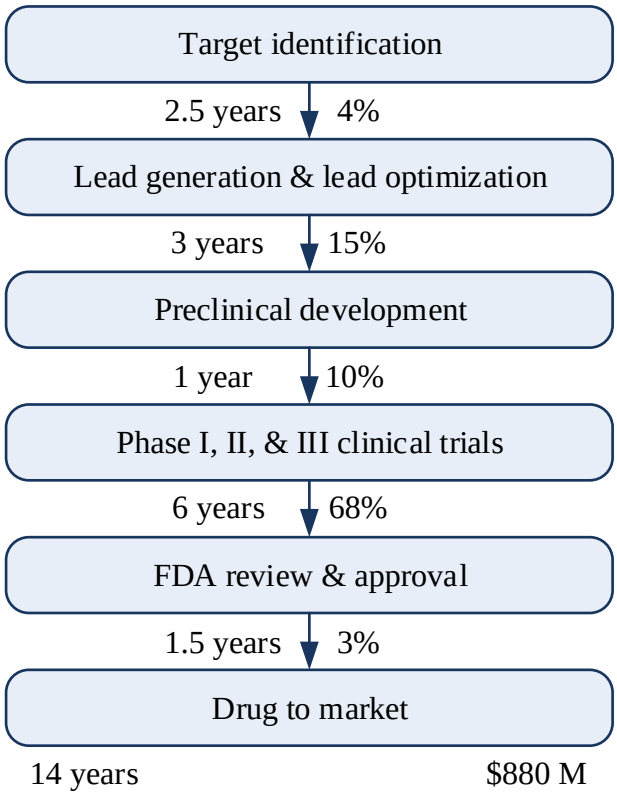


Figure 1.3: Time and cost involved in drug discovery.

structures, and, finally (iv) the ability to correctly rank a given set of ligands, with known binding modes when bound to the same protein. These four performance attributes were referred to by Cheng et al. as *docking power*, *screening power*, *scoring power*, and *ranking power*, respectively [17]. We refer to the corresponding problems as *ligand docking problem*, *ligand screening problem*, *ligand scoring problem*, and *ligand ranking problem* and SFs used to solve the three problems as *ligand docking SF*, *ligand screening SF*, *ligand scoring SF*, and *ligand ranking SF*, respectively. We also refer to the corresponding proposed models as *docking-specific SF*, *screening-specific SF*, *scoring-specific SF*, and *ranking-specific SF*, respectively. In practice and in all existing work, a single SF is trained to predict protein-ligand BA and then used in all the ligand docking, screening, and ranking stages to identify the top pose, bioactivity, and top ligand, respectively. Other requirements of SFs are that they should be fast enough for efficient virtual screening and be interpretable.

It should be noted that the time taken to train SFs is not critical for high-throughput since it is performed offline. Their prediction speed, however, is relevant for fast virtual screening applications. Therefore, computationally-intensive approaches such molecular dynamics and Monte Carlo simulations are not suitable for virtual screening [17]. The task-specific ensemble machine learning SFs we propose are very computationally efficient to apply, easily parallelizable on multi-cores, and also have excellent descriptive and interpretive power [18].

1.3 Motivation: Limitations of existing scoring functions for accurate ligand docking, screening, scoring, and ranking

Traditionally, the ligand docking, screening, scoring, and ranking steps depicted in Figure 1.2 are performed using the same SF built in the molecular modeling software package such as GOLD [19, 20] and Discovery Studio [21]. Such functions predict BA that is in turn used to rank-order hundreds of generated poses (docking) for thousands to millions of different ligands (ranking) for a certain target protein. We have collected scoring, ranking, docking, and screening performance of seventeen conventional SFs used in academic and commercial molecular docking packages. This set of SFs was evaluated on one of the most well-regarded benchmark datasets, viz. PDBbind core test set *Cr* [10], which is derived from PDB and comprises 65 three-complex clusters corresponding to diverse protein families and drug-like molecules.

Table 1.1 lists the scoring performance of the SFs in terms of Pearson correlation coefficient between the predicted and true values of BA. Their standard deviation (SD) values from experimental BA are listed in the second column. In the next columns, we show ranking power in terms of $R_{1\text{-}2\text{-}3}$, $R_{1\text{-}3\text{-}2}$, and $R_{1\text{-}X\text{-}X}$ success rates. The docking power statistics are reported in the subsequent three columns in terms of the S_1^1, S_2^1, and S_3^1 success rates. The last three columns list the screening power in terms of the top 1%, 5%, and 10% enrichment factors. These scoring, ranking, docking, and screening statistics as well as details of the SFs, protein-ligand complexes, and features used are discussed in more detail in the following chapters, where we will see that the higher these values are (except SD), the more accurate an SF is in terms of scoring, ranking, docking, and

Table 1.1: Performance of conventional scoring functions in scoring, ranking, docking, and screening the ligands of 65 different protein families

Scoring Functions	Scoring Power		Ranking Power (in %)			Docking Power (in %)			Screening Power (E.F.)		
	R_p	SD	$R_{1\text{-}2\text{-}3}$	$R_{1\text{-}3\text{-}2}$	$R_{1\text{-}X\text{-}X}$	S^1_1	S^1_2	S^1_3	top 1%	top 5%	top 10%
X-Score::HMScore	***0.644***	***1.83***	***54.7***	***15.6***	***70.3***	51.3	68.5	78.0	5.1	2.3	1.5
$\text{DrugScore}^{\text{CSD}}$	0.569	1.96	51.6	21.9	73.4	62.8	74.3	81.4	2.6	1.4	1.6
SYBYL::ChemScore	0.555	1.98	46.9	25.0	71.9	40.5	60.1	71.4	5.3	2.4	2.2
DS::PLP1	0.545	2.00	54.7	15.6	70.3	64.9	75.3	84.2	6.9	4.3	3.0
GOLD::ASP	0.534	2.02	43.8	21.9	65.6	***69.5***	***82.4***	***88.6***	12.4	6.2	3.8
AffiScore	0.521	2.06	38.5	12.3	50.8	31.8	46.2	52.3	3.6	2.0	1.9
SYBYL::G-Score	0.492	2.08	46.9	25.0	71.9	25.2	41.5	56.3	1.9	1.3	1.4
DS::LUDI3	0.487	2.09	45.3	21.9	67.2	41.6	57.3	67.2	12.5	4.3	2.8
DS::LigScore2	0.464	2.12	35.9	25.0	60.9	53.9	71.5	80.4	15.9	6.2	4.1
GlidScore::XP	0.457	2.14	34.4	29.7	64.1	54.6	73.2	84.7	***19.5***	***6.3***	***4.1***
DS::PMF	0.445	2.14	40.6	18.8	59.4	32.4	43.7	53.4	4.9	2.2	2.6
GOLD::ChemScore	0.441	2.15	35.9	21.9	57.8	54.6	70.5	79.2	18.9	6.8	4.1
SYBYL::D-Score	0.392	2.19	46.9	20.3	67.2	15.3	30.8	47.6	2.3	1.8	1.6
DS::Jain	0.316	2.24	42.2	31.3	73.4	25.7	44.8	64.6	5.9	2.5	1.8
GOLD::GoldScore	0.295	2.29	23.4	26.6	50.0	51.7	69.0	80.9	7.8	4.5	3.2
SYBYL::PMF-Score	0.268	2.29	39.1	21.9	60.9	37.2	48.1	56.8	5.4	2.2	1.9
SYBYL::F-Score	0.216	2.35	29.7	25.0	54.7	54.6	64.4	73.8	1.9	1.3	1.4

Numbers highlighted in bold correspond to the top performing scoring function in each of the three performance metrics: scoring power (R_p), ranking power ($R_{1\text{-}2\text{-}3}$), docking power ($S^1_1\%$), and screening power ($\text{E.F.}_{1\%}$). Scoring functions are evaluated on the core set of PDBbind 2007 to estimate their scoring, ranking, and docking performance. The screening accuracy is obtained by testing scoring functions on the core set of PDBbind 2014. More details about these two benchmarks are provided in Section 2.1.

screening performance.

The numbers indicate that the predictions of almost all SFs have mediocre to weak correlation with true binding affinity values. This trend translates to poor scoring power for all SFs except X-Score whose predictions seem to show moderate linear correlation with experimental values. However, due to uncertainties associated in BA prediction, a relatively higher scoring power of an SF at this correlation level (0.644) does not necessarily lead to corresponding improvements in ranking, docking, and screening performances. This is evident from Table 1.1. The top performing SF, X-Score, has a relatively better BA prediction accuracy ($R_p = 0.644$), but fails to identify poses that are within 1 Å from the native one for almost 49% of the proteins. Its ranking performance is also low where only 55% of protein families had their ligands ranked correctly. A similar mediocre accuracy was also obtained for X-Score when its screening performance was evaluated. The table shows that this empirical SF achieved an enrichment factor of 5.1 when an ideal SF

could result in up to 66 of enrichment to the database on which we tested it. The best performing scoring function in correctly classifying ligands as actives or inactives, GlideScore::XP, is not even among the top ten in scoring performance. Similarly, the most accurate SF in terms of docking power, GOLD::ASP, comes only fifth in scoring and eighth in ranking performance. SF design for solving the ligand scoring, ranking, docking, and screening problems has been investigated for over two decades now [17], but according to the 2007 PDBbind benchmark, the best conventional SF technique has a scoring power of only 0.644 (in terms of R_p), whereas our proposed boosted NN SF attains a much higher scoring power of 0.816, which is a significant result as we will see in Chapter 4. Furthermore, our top ML SF has the best ranking power of $R_{1\text{-}2\text{-}3} = 64.3$ vs. 54.7 for the best conventional SF. In terms of docking and screening power, however, the proposed scoring ML SFs lag behind conventional ones. Thus, as we will see later, using BA-based ML SFs has clearly worked well for scoring in both relative (to conventional SFs) and absolute terms, has worked in relative terms in the case of ranking power, but not so in absolute terms, and has worked in neither absolute nor relative terms in the case of docking nor screening power. Therefore, ligand scoring SF design that is BA based is clearly not sufficient to solve the ligand docking and screening problems.

Note that in SBDD all four of the problems, ligand scoring, docking, screening, and ranking, are critical to solve well. Solving ligand scoring and ranking well, but not the docking and screening problems well will lead to ineffective virtual screening—this is what we noticed from the results above. Solving the ligand docking, screening, and ranking problems well will lead to effective relative screening of ligands, but if ligand scoring is not done properly, there will be uncertainty in the binding affinity (and hence stability) of the identified ligand(s). The proposed research seeks to address the ligand docking, screening, scoring, and ranking problems using completely novel approaches whose performance on multiple tests is promising and show potential for transformative impact.

1.4 Key contributions

To address the ligand docking, screening, scoring, and ranking problems, we propose several innovative and potentially transformative concepts to advance the state-of-the-art in SF design for these problems in the following two directions:

- **Descriptor Data Bank (DDB): A platform for multi-perspective modeling of protein-ligand interactions.** Protein-ligand (PL) interactions are essential biochemical processes for all living organisms. They play a key role in molecular recognition, molecular binding, signal transmission, and cell metabolism, to name a few. Accurate modeling of such interactions remains an unsolved challenge despite advances in molecular biology and the growing number of solved protein-ligand structures. Decades of efforts have been devoted to understand these intermolecular forces in the context of protein-based drug design. Such efforts have resulted in a large number of hypotheses and perspectives of interaction factors scattered in the literature that would have substantially improved binding affinity prediction accuracy of protein-ligand complexes had they been utilized collectively. Typically, only a small subset of hypotheses for few interactions are utilized as descriptors in a scoring function (SF) without making use of existing solutions that may complement the proposed perspectives of the modeled intermolecular forces. In this work, we present Descriptor Data Bank (DDB), an online platform for facilitating multi-perspective modeling of PL interactions. DDB is an open-access hub for depositing, hosting, executing, and sharing descriptor extraction tools and data for a large number of interaction modeling hypotheses. The platform also implements a machine-learning (ML) toolbox for automatic descriptor filtering & analysis, and SF fitting & prediction. To enable local access to many of DDB's utilities, a command-line-based program and a PyMOL plug-in have also been developed and can be freely downloaded for offline and standalone multi-perspective modeling. We seed DDB with 16 diverse descriptor extraction tools developed in-house and collected from the literature. The tools combined generate over 2700 descriptors that characterize (i) proteins, (ii)

ligands, and (iii) protein-ligand complexes. The in-house descriptors we extract are protein-specific which are based on pair-wise primary and tertiary alignment of protein structures followed by clustering and triangulation. We built and used DDB's ML library to fit SFs to the in-house descriptors and those collected from the literature. We then evaluated them on several data sets that were constructed to reflect real-world drug screening scenarios. We found that multi-perspective SFs that were constructed using large and diverse number of DDB interaction models (descriptors) outperformed their single-perspective counterparts in all evaluation scenarios with an average improvement of more than 15%. We also found that our proposed target-specific descriptors improve upon the accuracy of SFs that were trained without them. In addition, DDB's filtering module was able to exclude noisy and irrelevant descriptors when artificial noise was added as new features. We also observed that the tree-based ensemble ML SFs implemented in DDB are robust even with the presence of noisy and decoy descriptors. In Chapter 3, we describe DDB in more detail and in the following chapters we provide results that demonstrate the excellent utility of multi-perspective interaction modeling.

- **Task-specific scoring functions:** Conventional empirical SFs rest on the hypothesis that a linear regression model is capable of computing binding affinity from a set of molecular descriptors. Additionally, these BA-based regression SFs can be used for ligand scoring (with a lot of room for improvement), but they are not suited for ligand docking, screening, and ranking (as we saw earlier in Sec. 1.3). To address this, we propose three task-specific approaches to SF design: (1) *nonparameteric* models capable of learning the underlying unknown function of protein-ligand binding from the data itself with minimal assumptions about the data distribution. We will show in Chapter 4 the excellent accuracy of such models in ligand scoring and ranking problems. (2) A docking SF that performs RMSD-based regression, where RMSD captures how close a given ligand binding pose is to the native pose, and enables us to leverage 3D structure data about all poses, native and non-native, instead of just the native pose as in the case of BA-based docking SFs. It also does not rely

on BA values which have inherent experimental measurement noise. Pose prediction results provided in Chapter 5 show how this makes a significant difference in docking performance. A further enhancement considers both RMSD and BA in SF design. (3) A screening SF that performs classification to discriminate between active and inactive ligands instead of BA-based regression in conventional SFs that make use of data only about active ligands. This family of screening SFs are presented in Chapter 6 along with preliminary results showing excellent accuracy in enriching databases of large numbers of ligands for a diverse set of protein families.

- **Multi-task scoring functions:** In addition to the three task-specific models, we also propose a novel multi-task scoring function for simultaneously predicting ligand binding pose, its activity classification label (active/inactive), and its binding affinity with a target protein. The scoring function, MT-Net, is based on wide and deep multi-task neural network with (i) general hidden layers shared among the three tasks to learn higher-level physiochemical features important for the three tasks, (ii) three specialized sets of hidden layers to learn features for each task separately and simultaneously in parallel, and (iii) three corresponding output layers for binding mode prediction (docking), ligand activity prediction (screening), and binding affinity prediction (scoring). When fitted to our current training sets, MT-Net performed equally well when compared to single-task neural network SF on out-of-sample test sets carefully designed to evaluate screening, docking, and scoring accuracy. Due to their high-capacity, we anticipate the accuracy of scoring functions based on multi-task deep networks to improve beyond those based on single-task NNs when larger and more diverse datasets are used for training. Currently, we are developing a pipeline for automated data retrieval from biochemical databases to continuously train the network on new protein-ligand complexes and screening assays as they become available. Details about the network's architecture and its training algorithm are provided in Chapter 7.

In Chapter 8, we summarize the findings of our work and discuss our future work that we believe will lead to further improvements in the performance of the proposed scoring functions.

CHAPTER 2

MATERIALS AND METHODS

In this chapter, we discuss the materials and methods used to build and evaluate the scoring functions proposed in the following chapters. First, we will describe their main training and test complexes. Then, we will present the machine-learning methods that we use as basis for the proposed scoring functions and re-construct those published in the literature. We will explain the mathematical models behind these prediction algorithms and the software packages that implement them. Finally, we will discuss several conventional scoring functions that will be compared against the machine-learning models considered in this work.

2.1 Main training and test protein-ligand complexes

We use the molecular database PDBBind [22, 23] for building and testing task-specific scoring functions on protein-ligand complexes characterized by the proposed multi-perspective descriptors from DDB. PDBbind is a selective compilation of the Protein Data Bank (PDB) database [9]. Both databases are publicly accessible and regularly updated. The PDB is periodically mined and only complexes that are suitable for drug discovery are filtered into the PDBbind database. We used the 2007 version of PDBbind during the early stages of this work to build and evaluate several novel machine-learning scoring functions. PDBbind 2007 contains a *refined set* of 1300 protein-ligand complexes with high-resolution 3D structures and binding affinity data. The curators of PDBBind partitioned the refined set into training and test sets as described in more detail in Section 2.1.1. We also considered the 2014 release of PDBbind to build more accurate versions of our boosted and bagged neural network scoring functions which utilize larger number of descriptors extracted using Descriptor Data Bank (DDB). The *refined set* of PDBbind 2014 contains 3446 high-quality protein-ligand complexes with experimental binding affinity data collected from the literature. From this set of complexes, we design three training-test sampling strategies to control the overlap in protein family between training and test data sets. The objective of such strategies is to estimate the

generalization ability of scoring functions on familiar proteins represented in their training data or novel targets that were not encountered before.

2.1.1 PDBbind core test set: novel complexes with known protein targets

From the protein-ligand complexes in the *refined set* of PDBbind (version 2007 and 2014), the curators of the database built a test set with maximum diversity in terms of protein families and binding affinies. The test set was constructed as follows. (i) The refined set was first clustered based on the 90% primary-strucutre similarity of its protein families. (ii) Clusters with four (five for the 2014 version) proteins or more are then examined based on their ligands' binding affinity values. From each of these PLC clusters, three complexes are marked. The PLC whose ligand has the lowest BA, the one with the largest BA, and the one closest to the mean BA of that cluster. (iii) The three PLCs are added to the test set if the marked complexes with the largest and lowest BA values are at least 2 orders of magnitude apart in terms of disassociation K_d or inhibition K_i constants. This resulted in the selection of 65 different protein families where each protein family binds to three different ligands with varying BA values to form a set of 195 unique PLCs. This is called the *core test set* and has been a popular benchmarking test set for many recent docking and scoring systems [17, 24, 25, 26, 27]. Due to its popularity, the 2015 and 2016 core sets have been kept identical to their predecessor in PDBbind 2014 so that results of different studies can be compared. In addition to the core test set of PDBbind 2007, we too consider the 2014/2015/2016 version of the core test set *Cr* in our evaluation of the proposed SFs. The corresponding training set for *Cr*, referred to as the *primary training set* and denoted by *Pr*, was formed by removing all *Cr* complexes from the total 1300 and 3446 complexes in the refined sets of PDBbind 2007 and 2014, respectively. As a result, *Pr* contains (1300 - 195 =) 1105 and (3446-195 =) 3251 complexes that are disjoint from *Cr* complexes. In our experiments on PDBbind 2014 we fixed the base sizes of training and test sets to 3000 and 195 PLCs, respectively. The 3000 PLCs are randomly sampled from the 3251 set to help evaluate the variance of ML SFs. In subsequent chapters, we present results for two sets of our task-specific scoring functions. One set of models are trained and

evaluated on complexes from the primary and core test sets of PDBbind 2007. The second set uses the 2014 PDBbind primary training and core test set complexes for construction and validation. This set of SFs are also evaluated based on cross-validation and leave clusters out as described in the next two sections. Scoring functions based on PDBbind 2007 were not evaluated using cross-validation and leave-clusters-out since the models were published before we developed these additional benchmarking strategies.

The core test set complexes are considered known targets to scoring functions due to the overlap between their training and test proteins. More specifically, for each protein in the 65 protein clusters of *Cr*, there is at least one identical protein present in the primary *Pr* training data, albeit bound to a different ligand.

2.1.2 Multi-fold cross-validation: novel complexes with known and novel protein targets

Although the training *Pr* and test *Cr* sets are disjoint at the protein-ligand complex level, they overlap at the protein family level. As mentioned above, each of the three PLC clusters in *Cr* were sampled from larger (≥ 5) protein-family clusters in the refined set and the remaining two or more PLCs are now part of *Pr*. More stringent test sets were also developed in terms of the protein-family overlap with training sets. One testing technique is based on 10-fold cross-validation (*CV*) where the refined set of PDBbind 2014 is shuffled randomly and then partitioned into 10 non-overlapping sets of equal size. Upon training and validation, one out of the ten sets is used for testing and the remaining nine are combined for training. Once the training and test completes for this fold, the same process is repeated for the other nine folds one at a time. In a typical 10-fold *CV* experiment on a dataset of 3446 PLCs, the training and test sizes of the 10 folds are ($3446 \times 9/10 \sim$) 3101 and ($3446 \times 1/10 \sim$) 345 complexes, respectively. In order to neutralize the dataset size effect on our results and have similar sizes to *Pr* and *Cr*, we sample 3000 from the 3101 training complexes and 195 PLCs from the corresponding 345 complexes in each fold. The performance results on the 10 test folds are then averaged and reported in the Results section.

Unlike the overlap in *Cr*, due to the randomness of *CV*, every protein family is not necessarily

present in both the training and test sets across the ten folds. Therefore, some test proteins are actually novel for the scoring function while others may be present in the training data. If it is present, the protein will be bound to a different ligand in the training set.

2.1.3 Leave clusters out: novel complexes with novel protein targets

Our third approach to constructing the training and test datasets is based on the novel-targets experiment by Ashtawy and Mahapatra [28] and leave-one-cluster-out (*LOCO*) by Kramer et al. [29]. In this approach, the refined set complexes are partitioned based on the protein's primary-structure similarity such that the training and test partitions have zero overlap in protein families. More specifically, datasets were constructed as follows. The 3446 PLCs of the refined set were clustered based on the 90% similarity cut-off of the proteins' primary structures. A total of 1135 clusters were found, of which the largest five are: HIV protease (262 PLCs), carbonic anhydrase (170 PLCs), trypsin (98 PLCs), thrombin (79 PLCs), and factor XA (56 PLCs). There are 9 clusters with size 30 PLCs or larger including the aforementioned five, 43 of size 10 or larger, and 1092 clusters with size less than 10 PLCs, the majority of which (644 clusters) consist of a single PLC. We created 20 different training-test dataset pairs based on this distribution of protein families. For the nine most frequent protein families (with ≥ 30 PLCs), we created nine corresponding test sets each consists of a single family. For each family-specific test set of these nine, we sampled 3000 PLCs from the refined set none of which contains the protein family under test. From the less populous clusters (i.e., those with < 30 PLCs), we randomly sampled eleven 195-PLC test datasets. These test sets could contain complexes from (1135 - 9 =) 1126 different protein families mostly singles (i.e., protein families that occur in one complex only). For each multi-family test set of these eleven, we again sampled another 3000 random PLCs from the refined set none of which contains the protein families under test. The justification for selecting these particular number of datasets and sizes is two-fold. First, we wanted about half of the mean performance on these test sets to result from the single-family test sets and the other half from the multi-family sets. Second, taking the average of a larger number of test sets with reasonable sizes reduces the possibility of

skewing the final performance due to potential outliers that could be obtained on few small test sets.

Test proteins are considered novel here due to the imposed sequence similarity constraint between them and training proteins which must be less than 90%. Therefore, all test proteins in a test cluster are considered novel for scoring functions fitted to training proteins sampled using the *LCO* strategy.

2.2 Machine learning algorithms

We utilize several regression and classification techniques in this work: multiple linear regression (MLR), multivariate adaptive regression splines (MARS), k-nearest neighbors (kNN), support vector machines (SVM), deep neural networks (DNN), random forests (RF), boosted regression trees (BRT), and extreme gradient boosting trees (XGB). These methods benefit from some form of parameter tuning prior to their use in prediction. We optimized these parameter values by performing a grid search over a suitable range in conjunction with 10-fold cross-validation over complexes independent from the test examples that we use in the following chapters. For every machine-learning method, we will be using these optimal values to build ML SFs in the subsequent chapters. Next, we provide a brief description of the theory of these algorithms.

2.2.1 Multiple linear regression (MLR)

Multiple linear regression (MLR) model is built based on the assumption that there exists a linear function that relates a set of explanatory variables X (descriptors characterizing protein-ligand complexes) and the response variable Y. This linear relationship has the form:

$$\hat{Y} = f(X) = \beta_0 + \sum_{j=1}^{P} \beta_j X_j. \tag{2.1}$$

The coefficients $\beta_0, \ldots, \beta_P$ are estimated from training data by minimizing a loss function, such as the sum of squared errors (referred to as the *method of least squares*), that measures the overall deviations of fitted values from experimentally-observed ones. Despite the fact that MLR

models are simple to construct and fast in prediction, they tend to suffer from overfitting when tackling high-dimensional data (i.e., datasets with larger values of P/N, where P is the number of features and N is the number of training records) [30]. Moreover, these models have a relatively large bias error due to their rigidity. For large data sets with few dimensions, MLR models are also prone to underfitting the data. The reason for developing simple linear models in this study is two-fold. First, they are useful in estimating the behavior of conventional empirical scoring functions. Second, they serve as baseline benchmarks for the nonlinear regression models we investigate. In our first set of experiments on PDBbind 2007 we use the built-in R language package *stats* to fit MLR models by invoking the function "lm" implemented therein [31]. For the second set of results collected on PDBbind 2014, we fit MLR models using the class "LinearRegression" implemented in the Python library Scikit-learn [32].

2.2.2 Multivariate adaptive regression splines (MARS)

MARS is an extension of MLR that can model nonlinearity in the data and account for interactions between related features [33]. Nonlinearity is modeled by partitioning the feature space into S regions, each of which is fitted by a nonlinear term known as a *basis function*. The MARS model is a weighed sum of an intercept (which is the mean of the response values in the training data) and the S basis functions. Every basis function is the product of one or more *hinge functions* that are generated in pairs from the training data. Each pair is characterized by an input variable X_j and a knot variable k. The knot variable acts as a pivot around which the hinge function produces a positive value on one side and zero on the other side. More formally, a hinge function takes the form:

$$\begin{aligned} &h(X_j,k)_+ = \max(0, X_j - k) \\ &\text{or} \\ &h(X_j,k)_- = \max(0, k - X_j). \end{aligned} \tag{2.2}$$

The $+$ (or $-$) sign in the subscript denotes that the function has the value $X_j - k$ (or $k - X_j$) to the right (or left) of the knot and 0 to its left (or right). The MARS algorithm we use adaptively

determines both the number of knots and their values from the training data. It also employs the method of least-squares to optimize the weights β_i's in the final MARS model which can be written as:

$$\hat{Y} = f(X) = \beta_0 + \sum_{i=1}^{S} \beta_i H_i(X), \tag{2.3}$$

where $\hat{Y}$ is the estimated biological activity for the complex X and H_i is the i-th basis function which is a product of up to d hinge functions; the degree parameter d models interaction between up to d features that appear in the basis functions.

During the training phase of a MARS model, a greedy search is performed in both forward (terms added) and backward (terms eliminated) passes. In the forward pass, the MARS algorithm first builds a complex model consisting of a large number of terms. To prevent the model from overfitting, this step is followed by a pruning round where several terms are dropped. An iterative backward elimination procedure based on generalized cross-validation (GCV) is typically performed to determine which terms are to be removed. Terms that have the least effect in increasing the model's residual sum of squares are dropped. Additionally, models with large number of terms and higher degree are penalized according to a penalty criterion. The penalty criterion and the maximum degree of MARS terms are user-defined parameters. A grid search using 10-fold cross validation has been conducted to tune these parameters. The R language package *earth* was used to construct our MARS models on complexes from PDBbind 2007 [34]. To avoid clutter and permit easier interpretation of the results obtained by other scoring functions, we do not report results of MARS models on the 2014 release of PDBbind. However, our experiments show that their performance are better than MARS scoring functions based on PDBbind 2007 due to the larger size of training data in PDBbind 2014 and the use of multi-perspective descriptors from DDB. The Python library Py-earth was used to conduct the (unreported) experiments of MARS models on PDBbind 2014 [35].

2.2.3 k-nearest neighbors (kNN)

An intuitive approach to predicting a certain biological property (e.g. binding affinity) of a protein-ligand complex is to search the training database for a similar protein-ligand complex and output the value of its response variable as an estimation for the complex in question. The most similar complex is referred to as the *nearest neighbor* of the protein-ligand compound whose biological property we are attempting to predict. This simple algorithm is usually augmented to take into account the biological property values of the k nearest neighbors, instead of just the nearest one. In the kNN algorithm we use here, each neighbor contributes to the final estimated property by an amount that is inversely proportional to its Minkowski distance from the target complex [36]. The Minkowski distance between two complexes X_i and X_j is defined as:

$$D(X_i, X_j) = \left(\sum_{p=1}^{P} |x_{i,p} - x_{j,p}|^q \right)^{1/q}. \tag{2.4}$$

These distances are then normalized and transformed with an arbitrary kernel function $K(.)$ into weights. The final weighted kNN model is formulated as:

$$\hat{Y} = f(X) = \sum_{i=1}^{k} (K(D(X, X_i)) \cdot Y_i), \tag{2.5}$$

where X and X_i are the target and the i-th neighbor complexes, respectively, characterized by P features, and Y_i is the experimental biological property of the i-th neighbor. In this work, we apply the *optimal* kernel formulated by Samworth in [37] to weight the neighbors according to their Minkowski distances D defined in Equation 2.4. The parameter q in Equation 2.4 determines the distance function to be calculated. Manhattan distance is obtained when $q = 1$, whereas Euclidean distance is realized when q is set to 2. A grid search using 10-fold cross validation has been applied to find optimal values of the number of neighbors k and the distance parameter q.

The R language package *kknn* was used to construct our kNN models on complexes from PDBbind 2007 [38]. To avoid clutter and permit easier interpretation of the results obtained by other scoring functions, we do not report results of kNN models on the 2014 release of PDBbind. However, our experiments show that their performance are better than kNN scoring functions based

on PDBbind 2007 due to the larger size of training data in PDBbind 2014 and the use of multi-perspective descriptors from DDB. In the unreported experiments on PDBbind 2014, we build our *k*NN regressors and classifiers using the classes "KNeighborsRegressor" and "KNeighborsClassifier" implemented in the Python library Scikit-learn [32].

2.2.4 Support vector machines (SVM)

Support Vector Machines is a supervised learning technique developed by Vapnic et al. to solve classification and regression problems [39]. The first step in constructing an SVM model is typically to transform the input data to a higher dimensional space using a nonlinear mapping function known as the *kernel* function [40]. After mapping to the new feature space, a linear model $f(X)$ is constructed. In regression formulation, the SVM algorithm induces from the training data an optimal hyperplane defined by a vector w and a bias term b that approximates the underlying unknown relationship between a set of physicochemical descriptors (X) and the protein-ligand interaction of interest (Y). The linear model $f(X)$ can be written more formally as:

$$\hat{Y} = f(X) = \langle w, G(X) \rangle + b, \tag{2.6}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product between w and the output of the kernel function $G(X)$, which is then added to the model's bias b. The norm of w is minimized to ensure a small w in order for the model to be as "flat" as possible so that it generalizes well to previously unseen data. At the same time, the model should fit the training data well by not allowing its predictions to deviate more than ε from true values. This is achieved by minimizing the ε-insensitive loss function which is defined as:

$$L_{\varepsilon}(X) = \max(0, |Y - f(X)| - \varepsilon). \tag{2.7}$$

This loss function defines a region such that data points lying outside of it are penalized. The penalty is introduced by the slack variables ζ and ζ^*. The SVM training process can therefore be formulated as the following optimization problem:

$$\text{minimize} \quad \frac{\|w\|^2}{2} + C\sum_{i=1}^{N}(\zeta_i + \zeta_i^*)$$

$$\text{subject to} \quad \begin{cases} y_i - \langle w, G(X_i)\rangle - b & \leq \quad \varepsilon + \zeta_i \\ \langle w, G(X_i)\rangle + b - y_i & \leq \quad \varepsilon + \zeta_i^* \\ \zeta_i, \zeta_i^* & \geq \quad 0, \end{cases} \tag{2.8}$$

where $C > 0$ is a user-defined variable that can be employed to regularize against overfitting. The parameter serves as a trade-off between model complexity and its flatness by scaling up or down the penalty imposed on examples whose deviations are larger than ε. Equation 2.8 can be solved more easily by applying quadratic programming on its dual formulation as follows:

$$\begin{aligned} \max_{\alpha,\alpha^*}(w(\alpha,\alpha^*)) = \max_{\alpha,\alpha^*}\Big(\sum_{i=1}^{N}(\alpha_i^*(y_i-\varepsilon) - \alpha_i(y_i+\varepsilon)) \\ -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\Big((\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)K(X_i,X_j)\Big)\Big), \end{aligned} \tag{2.9}$$

where α_i and α_i^* are the standard Lagrange multipliers that can be obtained by solving Equation 2.9 under the constraints:

$$\begin{aligned} &\textstyle\sum_{i=1}^{N}(\alpha_i - \alpha_i^*) = 0 \\ &\text{and} \\ &\alpha_i, \alpha_i^* = [0, C]. \end{aligned} \tag{2.10}$$

After obtaining the Lagrange multipliers we can calculate the terms of Equation 2.6 as follows:

$$\begin{aligned} &\langle w, G(X)\rangle = \textstyle\sum_{i=1}^{N}\left(\left(\alpha_i - \alpha_i^*\right)K(X,X_i)\right) \\ &\text{and} \\ &b = -\tfrac{1}{2}\textstyle\sum_{i=1}^{N}\left(\left(\alpha_i - \alpha_i^*\right)(K(X_i,X_r) + K(X_i,X_s))\right), \end{aligned} \tag{2.11}$$

where $K(X,\tilde{X})$ is the kernel function G. We choose K to be the radial basis function (RBF) that is calculated as:

$$K(X,\tilde{X}) = e^{\frac{\|X-\tilde{X}\|^2}{2\sigma^2}}, \tag{2.12}$$

where σ is the kernel width. The data points X_r and X_s in Equation 2.11 are any arbitrary *support vectors* located on both sides of the margin. Support vectors are data instances located outside the ε-insensitive region, and as this region gets wider (i.e., larger ε value), fewer support vectors are obtained and the model becomes flatter. The three parameters of the SVM model, namely C, ε, and σ, are optimized via a grid search using a 10-fold cross validation.

The R language package *e1071* was used to construct our SVM models on complexes from PDBbind 2007 [41]. To avoid clutter and permit easier interpretation of the results obtained by other scoring functions, we do not report results of SVM models on the 2014 release of PDBbind. However, our experiments show that their performance are better than SVM scoring functions based on PDBbind 2007 due to the larger size of training data in PDBbind 2014 and the use of multi-perspective descriptors from DDB. In the unreported experiments on PDBbind 2014, we build our SVM regressors and classifiers using the classes "SVR" and "SVC" implemented in the Python library Scikit-learn [32].

2.2.5 Deep Neural Networks (DNN)

Artificial Neural Networks are computational algorithms inspired from networks of biological neurons to solve prediction problems in computer vision, natural language processing, drug discovery, etc. The neural network we fit here consists of an input layer for the raw descriptors X_1, L hidden layers for extracting higher-level representations for the molecular interactions and forces, and an output layer for prediction as depicted in Figure 2.1.

Upon prediction, new representation of the raw descriptors are first extracted from the hidden layers as follows:

$$X_{l+1} = H(\mathbf{W}_l X_l + B_l), \quad l = 1, \ldots, L \tag{2.13}$$

where $\mathbf{W}_l$ and B_l are respectively the weight matrix and bias for the l-th hidden layer, and H is the activation function associated with it which is selected to be a rectified linear unit (ReLu) in this work. The final output of the network is simply the following transformations for the features

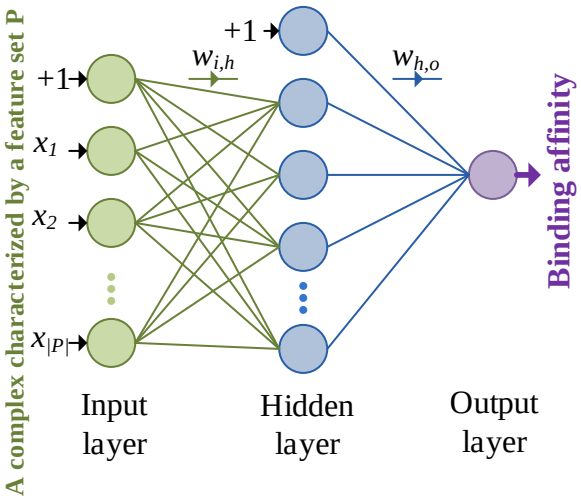


Figure 2.1: A deep neural network for predicting the binding affinity of protein-ligand complexes characterized by a set of descriptors.

$\mathbf{X}_{L+1}$ generated by the L-th hidden layer:

$$\hat{Y} = O(\mathbf{W}_o X_{L+1} + B_o), \tag{2.14}$$

where O is the output function which is logistic for the screening problem and linear for docking and scoring. The weights of the network were learned from the training data using stochastic gradient descent.

In our first set of experiments on PDBbind 2007, we use the R language package *nnet* to fit boosted and bagged ensembles of one-hidden-layer NN models [42]. For the second set of results collected on PDBbind 2014, we fit deeper and wider ensemble NN models based on the classes "DNNRegressor" and "DNNClassifier" implemented in the Python library TensorFlow [43].

2.2.6 Random forests (RF)

A Random Forests model consists of an ensemble of prediction trees that are typically on the order of several hundred to few thousand [44]. Each tree is fitted to bootstrap data consisting of N samples chosen randomly with replacement from N training samples. When a tree is constructed, at each node a binary split is made on the "best" descriptor out of *mtry* features randomly sampled from the P-dimensional feature space ($mtry \leq P$). Each tree is grown independently to its full size

without pruning. When the task is to score a new protein-ligand complex, the output is computed as the average of the predictions of the comprising individual trees. This mechanism can be formally expressed as:

$$\hat{Y} = F(X) = \frac{1}{T}\sum_{t=1}^{T} f_t(X) = \frac{1}{T}\sum_{t=1}^{T} \hat{Y}^{(t)}, \tag{2.15}$$

where X is a protein-ligand complex whose biological property needs to be predicted, T is the number of generated trees, and f_t is the t-th tree in the ensemble. Upon building an RF model, the parameter that typically requires tuning is *mtry*. We choose the optimal value that minimizes the out-of-bag mean squared error (OOBMSE). Out-of-bag (OOB) refers to complexes that are not sampled from the original training set when bootstrap sets are drawn. On average, about 34% of the training set is left out in each tree-generating iteration. These out-of-sample examples are typically utilized to tune *mtry* and estimate the generalization error of the RF model as follows:

$$\text{OOBMSE} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \bar{\hat{y}}_i^{\text{OOB}}\right)^2, \tag{2.16}$$

where $\bar{\hat{y}}_i^{\text{OOB}}$ is the average of the predictions for the i-th complex over all the trees in which it is out of bag.

In our first set of experiments on PDBbind 2007, we use the R language package *randomForest* to fit Random Forest scoring functions [45]. For the second set of results collected on PDBbind 2014, we fit Random Forest regressors and classifiers using the classes "RandomForestRegressor" and "RandomForestClassifier" implemented in the Python library Scikit-learn [32].

2.2.7 Boosted regression trees (BRT)

BRT is an ensemble machine-learning technique based on a stage-wise fitting of regression trees. As the name implies, the technique attempts to minimize the overall loss by boosting the examples with the highest predicted errors, i.e., by fitting regression trees to residuals resulting from the existing ensemble trees. There are several different implementations of the boosting concept in the literature. The differences mainly arise from the employed loss function and treatment of the most erroneous predictions. In this work, we employ the stochastic gradient boosting strategy proposed

by Friedman [46] that builds on an earlier technique known as *AdaBoost* developed by Freund et al. [47]. The employed BRT approach builds a stage-wise model as listed in Algorithm 1 below.

Algorithm 1 BRT algorithm

1: *Input:* training data $D = \{X_i, y_i\}_{i=1}^{N}$
2: Start with an initial guess (e.g., the mean of y):
$$F(X) = f_0(X) = \arg\min_{\theta} \sum_{i=1}^{N} \psi(y_i, \theta)$$
3: **for** $t = 1$ to T **do**
4: Sample D without replacement (SWOR):
$$\left\{X_i^{(t)}, y_i^{(t)}\right\}_{i=1}^{\tilde{N}} = \text{SWOR}(D, 50\%) \text{ (i.e., } \tilde{N} = N/2)$$
5: Calculate current residuals:
$$\left\{\hat{y}_i^{(t)}\right\}_{i=1}^{\tilde{N}} = \left\{y_i^{(t)} - F_{t-1}(X_i^{(t)})\right\}_{i=1}^{\tilde{N}}$$
6: Construct a new training set: $D^{(t)} = \left\{X_i^{(t)}, \hat{y}_i^{(t)}\right\}_{i=1}^{\tilde{N}}$
7: Fit an S-terminal-node tree model f_t to $D^{(t)}$: $f_t = \{R_s^{(t)}\}_{s=1}^{S}$
8: Find the optimal values ($\{\theta_s^{(t)}\}_{s=1}^{S}$) of the S regions of f_t ($\{R_s^{(t)}\}_{s=1}^{S}$) that minimize the loss function ψ: $\theta_s^{(t)} = \arg\min_{\theta} \sum_{X_i^{(t)} \in R_s^{(t)}} \psi(y_i^t, F_{t-1}(X_i^{(t)}) + f_t(X_i^{(t)}, \theta))$
9: Add the tree f_t to the model $F(X)$: $F_t(X) = F_{t-1}(X) + \lambda f_t(X, \theta^{(t)})$
10: **end for**

Step 2 of Algorithm 1 shows that the first term in the model is a simple learner that minimizes the overall loss function ψ. Given the loss criterion ψ we use here is a least-squares function, the outcome of this step is basically the mean value of the biological activities of interest (y_i) of the complexes in the training set D. In each subsequent stage t, a tree f_t is fitted to the residuals $\hat{Y}^{(t)}$ obtained by applying the model $F_{t-1}(X)$ on a random subset of the training data $D^{(t)}$ (Steps 4-7). Split variables and values of the internal nodes of the tree f_t are decided from the current residuals $\hat{Y}^{(t)}$. In contrast, the constant values $\theta^{(t)}$ at the S disjoint terminal regions $R^{(t)}$ (i.e., leaves) are optimized so that they minimize the loss of the entire model. Generation of trees continues as long as their number does not exceed a predefined limit T. At the end of each tree-generating iteration, the BRT model is updated according to the formula in Step 9. A tree joins the ensemble with a shrunk version of itself. The variable that controls its contribution to the final model is known as the *shrinkage* or *learning rate* λ. The smaller the value of the learning rate, the smoother the

building process of a BRT model. It is safer to move slowly down the gradient given the fact that boosting is a form of functional gradient descent. However, the reduction in the learning rate should be compensated by increasing the number of fitted trees by almost the same factor to achieve a comparable level of performance. In our experiments, we fixed the shrinkage at 0.005 and determined the corresponding optimal number of trees using cross-validation.

The R language package *gbm* was used to construct our BRT models on complexes from PDBbind 2007 [48]. Since we primarily use Python language to conduct our experiments on PDBBind 2014 datasets, a choice had to be made between the Python libraries Scikit-learn and XGboost to fit BRT models. Scikit-learn provides the classes *ensemble.GradientBoostingRegressor* and *ensemble.GradientBoostingClassifier* while XGBoost implements *XGBRegressor* and *XGBClassifier* to fit boosted regression and classification trees. The accuracies of models trained using Scikit-learn and XGboost algorithms are more or less identical according to several experiments we conducted on the 2014 version of PDBBind. However, the run time of our experiments are noticeably shorter using XGBoost training algorithms. Therefore, we choose it over Scikit-learn's API to build several scoring functions in the following chapters. XGBoost builds a faster variant of boosted decision trees based on a more recent training algorithm developed by Chen et al. [49]. More details about XGBoost are provided in the following section.

2.2.8 Extreme Gradient Boosting (XGB)

In addition to BRT, we fit a faster variant of boosted trees scoring functions to PDBbind 2014 protein-ligand complexes using the Extreme Gradient Boosting algorithm [49]. Similar to BRT scoring functions discussed in the previous section, the XGB model consists of an ensemble of T trees whose individual outputs make up the final predicted score $\hat{y}$ according to the equation:

$$\hat{Y} = F(X) = \sum_{t=1}^{T} f_t(X),$$

where $X \in \mathbb{R}^P$ is a protein-ligand complex characterized by P molecular descriptors, and f_t is the t-th tree in the ensemble. The trees in XGB are also grown sequentially one after the other in

a greedy fashion. In each tree-fitting step, the training algorithm minimizes the overall loss by boosting the examples with the highest predicted error so far. Each tree in the ensemble is fitted to residuals made by its predecessor trees. The training algorithm implements a regularized stochastic gradient boosting strategy proposed by Chen et al. [49] similar to BRT's algorithm described in the previous section. More specifically, the t-th tree in the ensemble is constructed to minimize the regularized loss $L(t)$ as follows:

$$L(t) = \sum_{i=1}^{N} \psi(y_i, \hat{y}_i^{(t-1)} + f_t(X_i)) + \Omega(f_t),$$

where ψ is the loss function that accounts for the error between the true output y_i and its estimated value by the current (f_t) and past ($\hat{y}_i^{(t-1)}$) trees in the ensemble. The regularization term Ω uses the size of the tree in terms of the number of its nodes, $n^{(t)}$, and the values of its leave nodes, $\theta^{(t)}$, to increase the loss as follows:

$$\Omega(f_t) = \gamma n^{(t)} + 0.5\lambda \left\| \theta^{(t)} \right\|^2.$$

We use the Python library XGBoost with its default parameter values for $\gamma = 0$ and $\lambda = 1$ to build models for the docking, screening, and scoring tasks. The remaining parameters such as the number of trees, their maximum depth, the subsample ratio of the training instances and features were optimized using grid search for each of the three tasks. XGBoost-based scoring function will be trained on their respective task-specific data sets as will be discussed in more detail in the following chapters. Scoring functions based on XGBoost algorithm and fitted to PDBBind 2014 complexes will have the prefix *BT* (for Boosted Trees for regression or classification) as in BT-Score, BT-Dock, and BT-Screen. On the other hand, scoring functions based on boosted trees trained using the R language package *gbm* will start with the prefix *BRT* as in BRT::X or BRT::XAR that are trained and evaluated on protein-ligand complexes from the 2007 version of PDBbind.

2.3 Conventional scoring functions under assessment

The proposed multi-perspective descriptors in DDB and task-specific scoring functions are assessed on benchmark datasets and tests carefully designed to reflect real-world drug discovery and design scenarios. Some of the benchmark datasets and testing strategies have also been used in the literature to profile the performance of other scoring functions developed in academia or used in commercial software packages. In order to study how the proposed approaches compare with other scoring functions, we focus on recent studies in which several scoring methods have been tested on the PDBBind benchmarks we use [17, 27, 50, 25]. From those studies, we compare a total of 19 popular conventional scoring functions to our proposed ML approaches on the 2007 version PDBbind. The 19 functions are either used in mainstream commercial docking tools and/or have been developed in academia. Sixteen of the SFs (all except AffiScore, RF-Score, NNScore) were recently compared against each other in two studies conducted by Cheng et al. [17] and [25]. This set, listed in Table 2.1, includes five SFs in the Discovery Studio software version 2.0 [21]: LigScore, PLP, PMF, Jain, and LUDI. Five SFs in SYBYL software (version 7.2) [51]: D-Score, PMF-Score, G-Score, ChemScore, and F-Score. GOLD software version 3.2 [52] contributes three SFs: GoldScore, ChemScore, and ASP. GlideScore [53] in Schrödinger software version 8.0 [54]. Besides, two standalone SFs developed in academia are also assessed, namely, DrugScore [55] and X-Score version 1.2 [56]. In addition to the performance of the 16 scoring functions originally collected by Cheng et al. [17] and Li et al. [25] studies, we consider three additional scoring functions published in different work. The first is AffiScore, an empirical scoring function used in the academic docking software SLIDE [57]. The other two are the machine-learning scoring functions RF-Score [27] and NNScore [50] based on the ensemble trees algorithm Random Forests and neural networks, respectively. We use AffiScore, RF-Score, and NNScore as benchmark methods and sources of descriptors in our multi-perspective interaction modeling platform DDB and multi-task scoring functions (see Chapter 3 for more details).

Since molecular modeling software programs often provide multiple scoring choices, we use notation to identify both the modeling program and the SF used. For example, when referring to the

Table 2.1: The 19 conventional scoring functions and the molecular docking software in which they are implemented

Scoring function (SF)	Software	Type of SF	Reference
Jain	Discovery Studio	Empirical	[58]
LigScore		Knowledge based	[59]
Ludi		Empirical	[60]
PLP		Empirical	[61]
PMF		Knowledge based	[62]
ChemScore	SYBYL	Empirical	[63]
D-Score		Force-field based	[64]
G-Score		Force-field based	[52]
F-Score		Empirical	[51]
PMF-Score[1]		Knowledge based	[62]
ASP	GOLD	Empirical	[19]
ChemScore[2]		Empirical	[63]
GoldScore[3]		Force-field based	[52]
GlideScore	Glide	Empirical	[53]
AffiScore	SLIDE	Empirical	[57]
DrugScore	*Standalone*	Knowledge based	[55]
X-Score	*Standalone*	Empirical	[56]
NNScore	*Standalone*	Machine-learning	[50]
RF-Score	*Standalone*	Machine-learning	[27]

[1] SYBYL's implementation of PMF
[2] GOLD's implementation of ChemScore
[3] GOLD's implementation of G-Score

SF LigScore1 in Discovery Studio software, the notation DS::LigScore1 is used. Considering all SFs possible from the modeling tools mentioned in this subsection would result in large numbers of them. That is because modeling tools typically supply more than one version/option for the same SF. Each such variation could be regarded as a different SF. LigScore, for example, comes in two different flavors (LigScore1 and LigScore2). For brevity, we only report the version and/or option that yields the best performance on the PDBbind benchmark.

CHAPTER 3

DESCRIPTOR DATA BANK (DDB): A PLATFORM FOR MULTI-PERSPECTIVE MODELING OF PROTEIN-LIGAND INTERACTIONS

Non-covalent binding of protein-ligand complexes involves numerous physiochemical and structural factors that bring the molecules together such as hydrogen bonding, hydrophobic interactions, electrostatic charges, π-effects, van der Waals forces, molecular weight, 3D structures of the molecules and their sizes, etc. Such factors are too complex to model correctly and to determine their contribution to the binding process whether considered individually or collectively in groups of interacting forces. Accurate modeling of protein-ligand interactions remains an unsolved challenge despite advances in molecular biology and the growing number of solved protein-ligand structures. That is in part due to the limitation of current scoring functions in capturing and modeling protein-ligand interactions. Most of the knowledge-based, force-field, and to some extent empirical scoring functions in use today attempt to reduce a very complex system of intermolecular forces into a small number of terms that are combined in a simple additive manner to calculate the free energy of binding. The empirical approach improves upon its knowledge-based and force-field counterparts by relying on experimental data and linear regression to express binding affinity as a weighted sum of energetic terms. Recent studies have shown that empirical scoring functions have better accuracy on average in predicting binding affinity when compared to knowledge-based and force-field techniques [17, 25]. However, the empirical linear model is still too rigid and simple to reliably model the free energy of such a complex system. Their rigidity prevents them from fully taking advantage of new experimental data whose growth is only increasing. A new family of scoring functions based on machine learning has been introduced to the field in the past few years to fix the shortcomings of the simple empirical approaches [27, 50, 65, 24]. These scoring functions are typically based on highly non-linear predictive models such as ensemble trees and neural networks fitted to protein-ligand complexes with known experimental activity data. The complexes are typically characterized by larger number of descriptors than those used by the three

other categories of scoring functions. For example, the ensemble neural network SFs BgN-Score and BsN-Score (proposed in the following chapters) employ more than 86 descriptors that encompass several types and versions of intermolecular forces between the protein and the ligand [24]. These neural network scoring functions have shown substantial improvement in prediction accuracy when compared to 16 other conventional models. Several other studies have found that this new category of SFs are significantly more accurate than knowledge-based, force-field and empirical approaches [65, 17, 27, 50]. In this work, we propose a novel approach to further improve the accuracy of machine-learning scoring functions.

3.1 Improving machine-learning scoring functions

Recent machine-learning scoring functions include almost all the energy terms and potentials that knowledge-based (KB), force-field (FF), and empirical (E) SFs use in one way or another. For example, the frequency of different pairwise atomic types similar to those used in knowledge-based SFs are utilized in ML SFs [27, 50, 65, 24]. Van der Waals interactions, electrostatic contacts, solvation terms, hydrogen bonding, and hydrophobic forces that are employed in force-field and empirical approaches have also been considered in ML scoring functions [28]. In fact, all types and forms of energy terms that have been developed in the literature for empirical SFs can be, in principle, directly included as individual descriptors in ML SFs given enough training data. In this respect, ML SFs can be regarded as a superset of KB, FF, and E SFs. As opposed to linear regression models used in empirical SFs, ML approaches have higher capacity of learning and can better handle high-dimensional data due to their superior bias-variance error trade-off. The more interactions they model, and with more experimental training data, ML SFs have every potential to make near perfect predictions. Not only can they predict the bioactivity of interest for the target complex based on similar molecules in their training data, they can also rely on the large and diverse number of specialized descriptors to search for similar atomic environments and contacts encountered during training. Therefore, with large enough data and interaction modeling techniques (descriptors), ML SFs could truly one day become general purpose SFs valid for almost

all protein families and chemical compounds even the ones they never seen before during training. Other fields such as computer vision and natural language processing have benefited greatly from using machine-learning approaches to perform at par or surpass human precision on test subjects novel to them [66, 67]. The main reason for the successful application of ML in these and other domains with challenging problems is two-fold: (1) effective ML algorithms with high-capacity of learning, (2) the quantity, quality, and diversity of data on which they are trained, (3) availability of high-performance computers. We believe the domain of drug discovery and design has many of the ingredients to be another success story for machine-learning applications.

Many of the necessary building blocks for highly effective machine-learning scoring functions are available today. The field of machine and statistical learning is rapidly growing with new theories and algorithms that can effectively model regression, classification, ranking, clustering problems, etc. The computational resources on which these predictive models can be developed and deployed are becoming more powerful and cost-effective. A machine-learning scoring function that requires expensive computer infrastructure to design a few years ago can now be built on a multi-processor PC or outsourced to public computing cloud in a matter of minutes in exchange for small fees.

As is the case for other machine-learning models [66, 67], the accuracy of ML SFs has the potential to substantially improve as biochemical data continue to grow in size, quality, and diversity [28]. Thousands of new, high-quality, 3D structures of proteins and protein-ligand complexes are resolved and publicly released every year [9]. Millions of annotated drug-like compounds are being added to public databases periodically [68, 69]. Hundreds of millions of assay results and experimental bioactivity data for large numbers of proteins and ligands are also maintained in free databases [70, 71]. Open-access binding affinity databases of protein-ligand complexes for training and benchmarking are also growing in number and size every year [22, 72, 73, 74, 10]. However, despite such advances in data availability, we still do not know how to comprehensively characterize the raw proteins, ligands, and protein-ligand complexes in these databases into descriptors (features) suitable for producing accurate ML SFs. Developers of ML SFs typically build in-house

tools to extract descriptors which is the most time consuming stage in SF design. In this work, we propose a platform that provides SF designers with the tools necessary to easily and efficiently calculate a large number of diverse descriptors.

3.2 The need for a library of descriptor extraction tools

Almost all databases of small compounds, proteins, or protein-ligand complexes host molecular data in their raw form. These molecules must be characterized comprehensively by informative descriptors before applying ML algorithms. Descriptors are models[1] for different molecular properties and protein-ligand interactions such as hydrogen-bonding, hydrophobic forces, steric clashes, electrostatic contacts, etc. Such interactions are very complex to represent correctly and there is no consensus on deriving a standard model for them. As a result, a large number of hypotheses have been proposed in the literature to characterize the same types of intermolecular forces. A hypothesis is basically a single perspective for the interaction it attempts to model as a descriptor. Developers of empirical and recent ML SFs generally rely on their experience, intuition, and analysis of data sets to derive a hypothetical model for each descriptor. Despite the relatively large number of descriptors that could be computed efficiently from proteins, ligands, and protein-ligand complexes, most existing SFs utilize only a very small set of orthogonal descriptors characterizing the various interaction factors. E.g., among popular commercial and academia-developed SFs, D-Score [64]: uses 2 descriptors, LigScore [59], PLP [61], G-Score [51]: 3, LUDI [60], ChemScore-2 [63]: 4, Jain [58], F-Score [60], AffiScore [75]: 5, X-Score [56]: 6, GoldScore [52]: 7, ChemScore-1 [52]: 9, and GlideScore [53]: 10 descriptors. Furthermore, it is uncommon for an empirical SF to include more than one hypothetical model for each type of interaction it attempts to take into account. Also, the type of the interactions employed by current scoring functions vary from one model to another. As an example, the empirical SF ChemPLP [76] accounts for the protein's solvation and flexibility characteristics among other factors. The SF X-Score, on the other hand, models neither of these interactions. Instead, it includes a hydrophobicity energy

[1] Descriptor models should not be confused with machine learning models that we use to build scoring functions for prediction.

term which is not present in ChemPLP. In addition to employing different interactions, two SFs that account for the same type of interactions may model them differently. As an example, hydrogen bonding and hydrophobic contacts are modeled differently by X-Score [56], AffiScore [75], and RF-Score [27]. The variation in the characterization of the same interactions are usually due to differences in the theoretical model of the interactions, the parameters of the model, the level of detail or granularity of the modeling algorithm, etc. The lack of sufficient understanding of the forces that bring proteins and ligands together is the main reason behind such discrepancies.

Although there is an overlap in the types of interactions the features in various SFs attempt to capture, they still represent different ways or perspectives of capturing protein-ligand binding. Current empirical SFs, such as X-Score, ChemPLP, and AffiScore, treat the hypothetical models of interactions as competing perspectives instead of utilizing them collectively as co-operative descriptors. This is evident from the small number of descriptors that is normally considered by traditional SFs. An important reason existing SFs use very few features is to avoid overfitting in the simple, often linear, models employed. Also, the interpretation of the linear SF becomes harder due to high-dimensionality and possible multicollinearity as more factors are added especially when they account for similar interactions. More importantly, implementing interaction terms from the literature is a very challenging task unless the modeling tool or software is easily accessible or the interaction model is simple and well described. Even with the availability of the software, obtaining and installing it could still be a challenge. To solve these problems, in this work we introduce a descriptor-specific platform that enables easy access, execution, and sharing of descriptor extraction tools via simple portals.

3.3 Methods

The main design goal of the descriptor library is to facilitate the extraction of high-quality and comprehensive descriptors for the three molecular entities: ligands, proteins, and protein-ligand complexes (PLCs). Based on our experience working with empirical and machine-learning scoring functions, we observed that the vast majority of them include descriptors specific to ligands

and/or PLCs. Protein-specific descriptors are under-represented in the feature space and we felt the need to fix this imbalance. Therefore, in this section, we present our novel approach for deriving similarity descriptors based on the primary and tertiary structures of proteins. We then describe the proposed Descriptor Data Bank (DDB) framework that facilitates integrating descriptors from various types and sources for multi-perspective modeling of protein-ligand interactions.

3.3.1 Receptor-specific descriptors based on structure similarity and triangulation

The well-known *similar property principle* in chemoinformatics states that "similar compounds have similar properties", i.e., molecules or compounds that have similarity in the descriptor space (which may be based on functional groups, structure, etc.) are likely to have similar biological activity [77]. This principle underpins the similarity-based virtual screening approach employed in ligand-based approached wherein compounds similar to a query compound (a known active ligand) are retrieved from a compound database for enrichment [78]. We propose target-specific descriptors based on the primary and tertiary structure similarities of receptors.

Homologous proteins are likely to have similar functions and hence triggered by related or the same biological signals. Proteins that are structurally and functionally similar are more likely to be expressed in similar cells. Cellular environmental factors, such as the presence and concentration of different molecules and charged ions, could have a bearing on the binding process. Such factors are not typically measured when the structures of proteins or protein-ligand complexes are solved and their binding affinity determined. We believe extracting protein-similarity based features could implicitly model environmental factors and intelligently encode the location of the target receptor in the protein family space. We are not aware of any SFs in the literature or commercial software that consider such information about the target explicitly or implicitly. Distant homologous proteins that have low sequence similarity may have similar tertiary structures and as a result might bind to analogous ligands. To capture such information, we extract two types of structure-specific descriptors: one based on *receptor primary structure similarity via triangulation* (or REPAST) and a second set based on *receptor tertiary structure similarity via traingulation* (or RETEST).

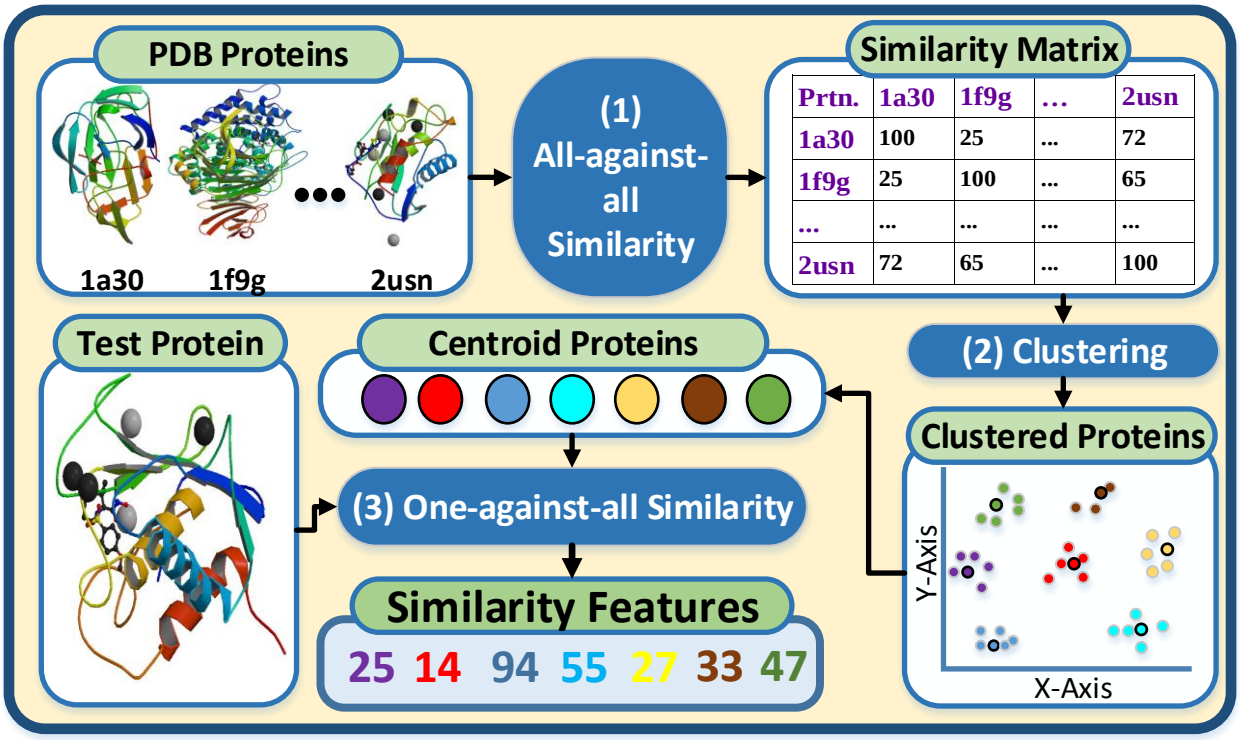


Figure 3.1: The workflow for generating the receptor-based similarity descriptors REPAST and RETEST.

Figure 3.1 illustrates the strategy we follow to extract REPAST and RESTEST descriptors. First, a set of diverse protein families from PDBbind 2007 is prepared for all-against-all pair-wise alignment. We use BLAST [79] for primary sequence alignment and TM-Align [80] to compute the tertiary structural similarity between proteins. The similarity scores associated with the alignment are recorded in a similarity matrix which is then clustered into a predefined number of groups. We use *k-means* to group the training proteins into 80 clusters. The centroid proteins of the clusters are chosen as cluster prototypes or representatives. Finally, for a test PLC, the similarity of the protein is determined w.r.t. the cluster prototypes using BLAST and TM-Align to yield similarity descriptors. The distances (inverse of BLAST and TM-Align similarity scores) of the target protein from the 80 protein prototypes determine its location in the protein family space. The family of the test protein is therefore determined indirectly via triangulation. Two structurally or functionally homologous proteins are likely to have similar REPAST and/or RETEST descriptor vectors and hence separated by a small euclidean distance. Representing large molecules such as proteins into

descriptor vectors is a very efficient and useful technique for ML SFs. In addition to characterizing the protein as a whole molecule, this approach can also be applied to generate sub-molecular descriptors. We plan on extending this strategy in the future to also better characterize the sequence of the receptor's binding site and the secondary structure of the domains near the cavity region.

3.3.2 Multi-perspective modeling of protein-ligand interactions

To date, SF design has primarily focused on hypothesis-driven modeling of the binding mechanism using a set of descriptors specific to each SF. Consequently, a wide variety of descriptor sets representing diverse and overlapping hypotheses or perspectives have been developed over time. These perspectives represent valuable knowledge about the binding mechanism that have never been brought together in one place and systematically analyzed. Given the high impact of PLC repositories such as RCSB PDB [81] and related ones like PDBbind [10] and CSAR [82] on drug discovery research and benchmarking, there is a clear need for such a resource for molecular descriptors representing multiple perspectives. We develop a descriptor data bank (DDB) and resource that will facilitate multi-perspective, data-driven protein-ligand interaction model discovery as depicted in Figure 3.2.

3.3.2.1 Descriptor Data Bank (DDB)

Descriptor Data Bank (DDB) is designed to host and share descriptor calculation tools and data to develop diverse and comprehensive multi-perspective models of protein-ligand interactions. A machine-learning toolbox is also implemented in DDB to provide in-place automatic descriptor filtering and scoring function development. Its multi-modular architecture, illustrated in Figure 3.2, provides a framework for meeting DDB design goals in terms of versatile user-interfaces, services, user programs and data, and system platforms. Descriptor Data Bank is accessible online via the web at *www.descriptordb.com* and can also be installed locally as a standalone PyMOL plug-in and command-line programs. These three separate front-end channels act on user commands and raw data and deliver processed data and results. Data processing, storage, and retrieval takes

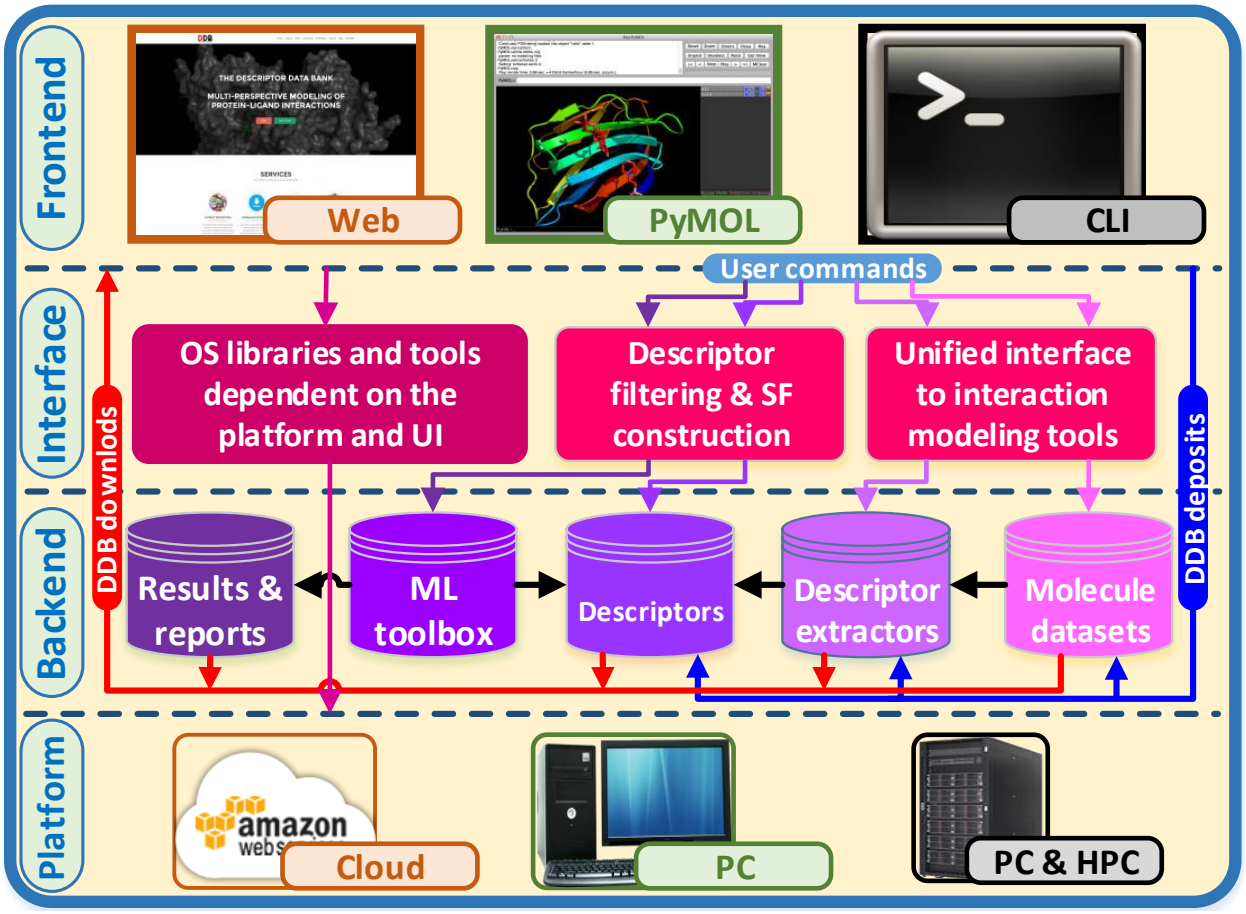


Figure 3.2: The system architecture of Descriptor Data Bank.

place in the back-end subsystem where databases of molecules, descriptor extraction programs, descriptors, ML algorithms, and results are maintained. DDB's front-end and back-end subsystems communicate through a unified interface with two Python-based programs and system-dependent libraries and tools. The two Python programs are for: (i) executing descriptor extraction tools donated by DDB users via well-defined interfaces and (ii) facilitating descriptor filtering and SF development requests and results. We discuss the main features of DDB and its design in the following sections.

3.3.2.2 Descriptor tools and data: deposition, extraction, and sharing

DDB's online platform provides an upload portal to enable users to easily deposit their descriptor extraction programs and data. The descriptor extraction programs can be written in any program-

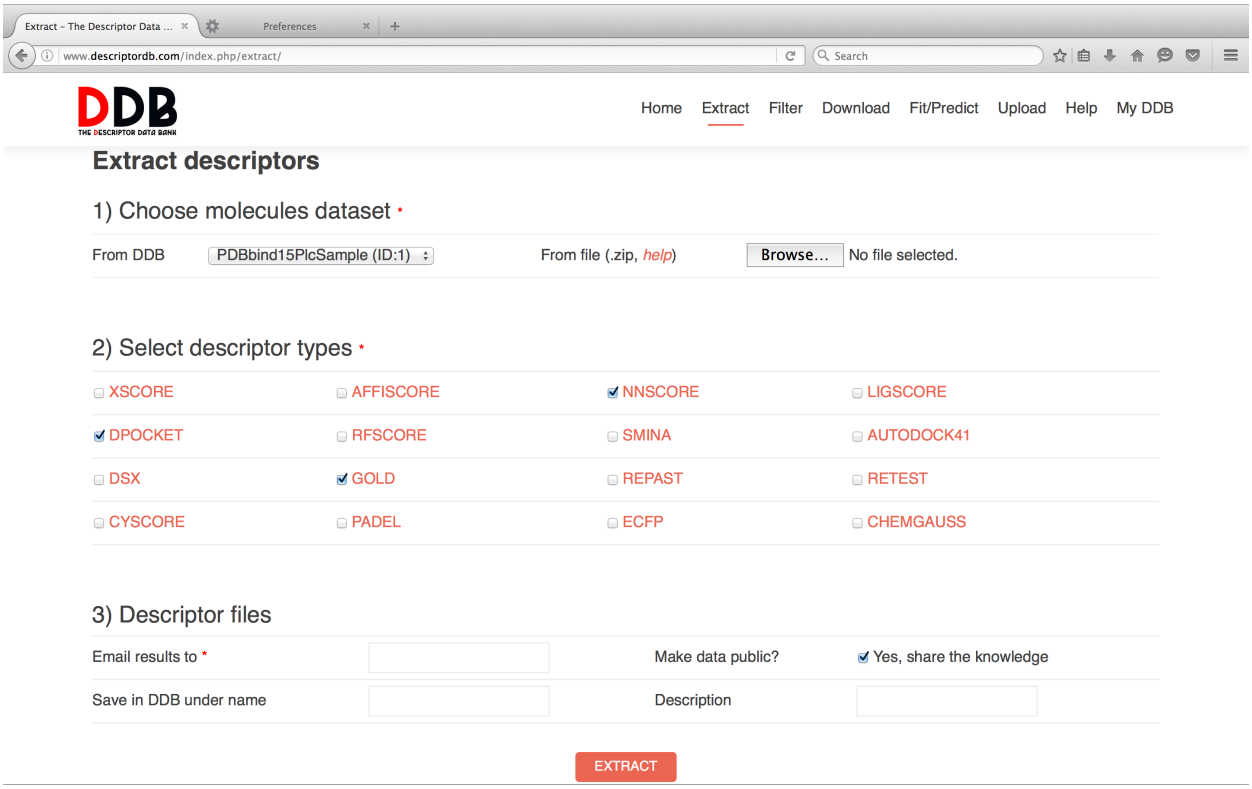


Figure 3.3: The web interface for Descriptor Data Bank.

ming language. To ensure consistency and quality, the donated programs and data must adhere to DDB's packaging and naming conventions which are provided on the website.

When a user deposits a descriptor extraction program, it will be inspected for security and computational resource requirements. If it passes these tests, the program will then be available for use and download. The descriptor extraction tool will appear on the descriptor extraction page (see Figure 3.3) as an additional available descriptor set that can be calculated for a given molecular (P, L, or PLC) dataset. Users will have the choice to use sample molecule datasets hosted on DDB or upload their own. Instructions about data formatting, naming conventions, and upload instructions of molecule datasets can be easily located on the website. Users can also download the deposited descriptor extraction programs, molecule datasets, or raw descriptors via the download page. Figure 3.2 illustrates the flow, storage, and handling of molecule datasets, descriptor extraction programs, and raw descriptors.

The standalone PyMOL plug-in and command-line programs are also capable of descriptor

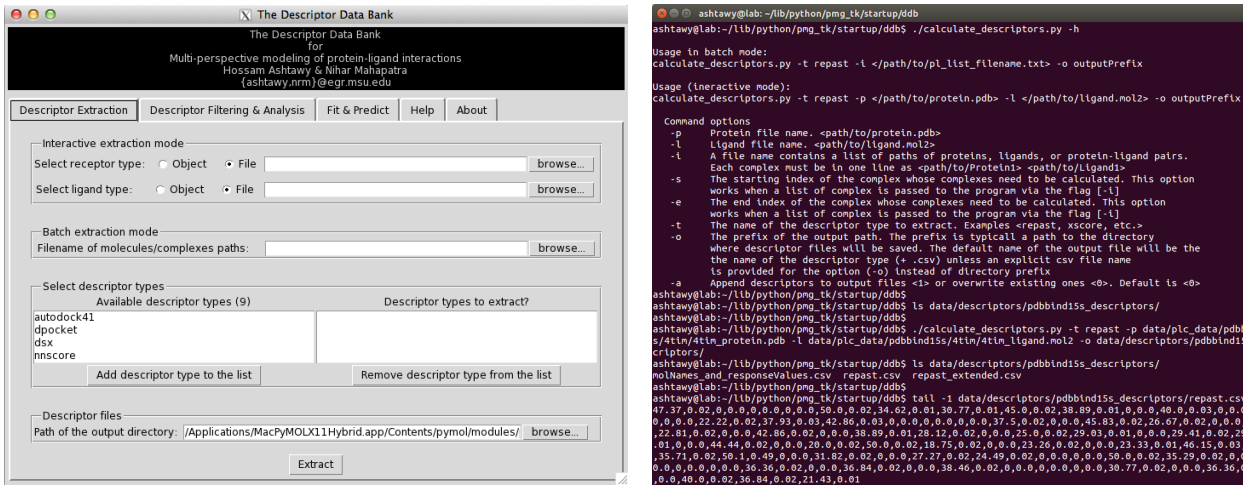


Figure 3.4: The graphical (left) and command-line (right) user interfaces for Descriptor Data Bank.

extraction as shown in Figure 3.4. Users can download descriptor extraction programs from DDB's website to extend the functionality of local DDB and keep it up-to-date. The PyMOL plug-in and command line programs feature, by default, some of the descriptor extraction tools available in DDB. We seed DDB with a rich collection of diverse descriptors and descriptor extraction tools—of all three types: those that depend upon protein only, ligand only, and the protein-ligand complex—from three different sources that together will more fully capture the complex nature of protein-ligand interaction: (a) descriptors used in existing SFs—for some of these (e.g., the AutoDock Vina [83], NNScore [50], and Cyscore [84] descriptors) we have access to descriptors extraction tools and others we will implement ourselves based on descriptions in the literature; (b) the new molecular and sub-molecular (in the future) similarity descriptors such as REPAST and RETEST discussed earlier; and (c) molecular descriptors and physicochemical properties that have been employed in ligand-based screening and QSAR such as PaDEL [85] and ECFP [86].

3.3.2.3 Descriptor filtering and analysis

Due to its open-access nature, there is a possibility that some of the descriptors hosted on DDB may be noisy or irrelevant for drug design and discovery applications. Improper modeling of some interactions, bugs in the donated descriptor extraction program, or any other artifacts can intro-

Table 3.1: The 16 descriptor types and scoring functions and the molecular docking software in which they are implemented

Descriptor set/SF	Symbol[1]	Molecule(s)[2]	Software	Type of SF	Reference
AffiScore	a	PLC	SLIDE	Empirical	[58]
AutoDock Vina	u	PLC	AutoDock Vina	Empirical	[83]
ChemGauss	h	PLC	OpenEye	Gaussian	[87]
Cyscore	c	PLC	*Standalone*	Empirical	[84]
DPOCKET	f	PLC	FPOCKET	*Descriptors*	[88]
DSX	d	Preds.	*Standalone*	Knowledge based	[89]
ECFP	e	L	RDkit	*Descriptors*	[86]
GOLD	g	PLC	GOLD	*Descriptors*	[20]
LigScore	l	Preds.	IMP	Knowledge based	[59]
NNScore	n	PLC	*Standalone*	Machine-learning	[50]
PaDEL	p	L	*Standalone*	*Descriptors*	[85]
REPAST	b	P	*Standalone*	*Descriptors*	descriptordb.com [90]
RETEST	t	P	*Standalone*	*Descriptors*	descriptordb.com [90]
RF-Score	r	PLC	*Standalone*	Machine-learning	[27]
SMINA	s	PLC	*Standalone*	Empirical	[91]
X-Score	x	PLC	*Standalone*	Empirical	[56]

[1] The symbol of the SF or extraction tool is used to identify the source of descriptors in Figure 3.7.

[2] The entity characterized by the descriptors: P for protein, L for ligand, and PLC for protein-ligand complex. When access to the descriptors of SFs in not available, we use their predictions (denoted by Preds. in the table) as a surrogate for them.

duce noise into the computed descriptors. To guard against noisy data and ensure robustness of the DDB's ML SFs, DDB provides an automatic descriptor filtering tool. Users have the option to experiment with DDB's raw descriptors or choose to filter them before fitting an ML SF. Descriptor filters can be also stored in DDB for later use or to share them with other developers to build their SFs. In fact, DDB can build ensemble-based SFs that are less prone to overfitting than their empirical counterparts. They are more forgiving when their training complexes are characterized by a larger number of descriptors and/or the possibility that some of these features are either noisy or not very relevant to the property being predicted. However, the predictive performance of ensemble ML SFs might be enhanced by utilizing only the most relevant and informative descriptors. In addition to the accuracy gain, the other benefit of employing a compact set of features is computational efficiency. Speed is of utmost importance in molecular docking and virtual screening where interaction modeling is perhaps the most time consuming phase in the process. Therefore,

substantial computational efficiency could be achieved saved when using fewer features by ignoring the ones that are slower to extract among redundant descriptors. The gained computational savings could be utilized to explore the conformational space more thoroughly during docking or employed to screen more ligands.

The descriptor filtering algorithm implemented in DDB is based on *embedded feature-subset-selection* in which the search for an optimal subset of descriptors is built into the ML algorithm. The algorithm relies on Random Forest's automatic variable importance calculation in conjunction with in-house iterative backward elimination to remove redundant or noisy descriptors and only retains the most relevant descriptor subset for the given prediction task. Our algorithm employs multi-objective fitness functions that allow task-specific descriptor subset selection. The selected subset of descriptors could be optimized for any or all of the following drug-discovery tasks: identifying the native or near-native binding pose during docking, binding affinity prediction, and enriching compound databases with active compounds. After filtering, DDB will output results in the form of summary tables and graphs. Such reports shed light on the effect of various interaction forces and other factors on the bioactivity of interest. Descriptor filtering and analysis can also be conducted using the PyMOL plug-in and command-line interface (CLI) as shown in Figure 3.4.

We plan to extend the descriptor filtering algorithm such that it selects the descriptors that are faster to calculate when there are equally accurate but redundant features with different computational requirements. Such an improvement will result in the selection of accurate descriptors that are fast to calculate.

3.3.2.4 Machine-learning scoring functions in DDB

We provide several machine-learning algorithms in DDB to develop SFs using descriptors hosted on the platform. The ML algorithms available include Extreme gradient boosting trees (XGBoost) [49], Gradient Boosting Trees (GBT) [92], Deep Neural Networks (DNN) [24], Random Forest (RF) [45], Support Vector Machines (SVM) [40], and Multi-variate Linear Regres-

sion (MLR). These algorithms are implemented in the Python libraries XGBoost [49], TensorFlow [43], and Scikit-learn [32]. DDB provides a unified and simple interface to construct ML SFs and hides the details associated with the underlying ML Python libraries. Users can select the desired learning algorithm, the PLC training and test datasets, and the sets of descriptors to characterize these complexes. Furthermore, users can also choose a pre-calculated descriptor filter to fit the ML algorithm to only the most important descriptors. Upon completion of the ML fitting process, predictions are calculated for the training and test complexes, and summary performance and processing time reports will be produced. The results are saved in DDB for later use, and can be downloaded and shared with other DDB users. The ML SF fitting and prediction capabilities can also be performed locally in the user's PC via the PyMOL plug-in and/or CLI as shown in Figure 3.4.

3.3.2.5 Multi-perspective descriptors extracted for PDBbind complexes

For each PDBbind protein-ligand complex prepared for the docking, screening, and scoring tasks, we extracted 16 descriptor sets covering a wide range of perspectives and interaction hypothetical models. The total number of descriptors we extracted using DDB are 2714 including 320 receptor-specific descriptors from the proposed REPAST and RETEST techniques, 1765 ligand-specefic descriptors using PaDeL and ECFP, and 629 complex-specific interactions using the remaining 12 SFs listed in Tabel 3.1. All the scoring, docking, and screening PLC datasets (derived in Chapters 4, 5, and 6) along with the 16 descriptor types are available on DDB's website (www.descriptordb.com) for use and download.

3.4 Results and discussion

3.4.1 Single vs. multi-perspective modeling of protein-ligand interactions

In this section, we systematically investigate the effect of multi-perspective modeling of protein-ligand interactions on the performance of scoring functions in different applications. We use the training and test datasets described in Sections 2.1 for building and evaluating task-specific ML

Table 3.2: Comparison of the scoring, docking, and screening powers of single and multi-perspective scoring functions using the core test set (Cr), leave-cluster-out (LCO), and cross-validation (CV).

Task (metric)	Best SP (using NNScore descriptors)			MP (without REPAST & RETEST)			MP (with REPAST & RETEST)		
	Cr	CV	LCO	Cr	CV	LCO	Cr	CV	LCO
Scoring (R_p)	0.779	0.773	0.571	0.801	0.815	0.612	0.827	0.825	0.637
Docking ($S\%$)	69.23	68.86	64.24	80.05	81.80	75.48	82.05	81.78	77.89
Screening ($EF_{1\%}$)	28.00	32.92	24.38	33.50	35.82	28.64	34.00	37.33	29.79

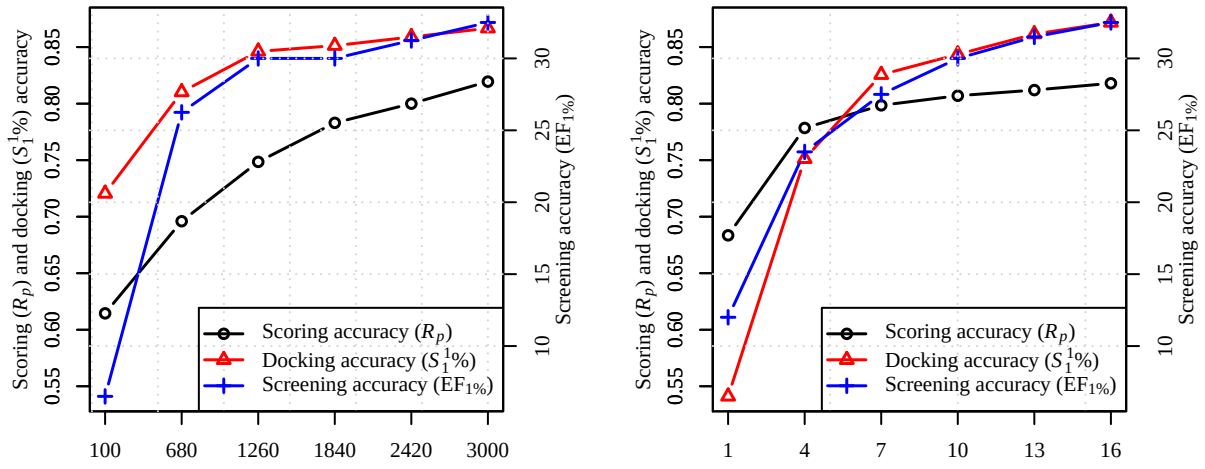
SFs for the scoring (Chapter 4), docking (Chapter 5), and screening (Chapter 6) tasks.

Our proposed data-driven and multi-perspective (MP) modeling of PL interactions is compared against the conventional approach of PLC characterization that depends on single-perspective (SP) modeling. For MP modeling, we built two sets of SFs to evaluate the effectiveness of the proposed protein-based SFs. In the first, we combined *all 2714 descriptors* in the 16 descriptor sets listed in Table 3.1 for each training dataset to build XGBoost SFs specific to the scoring, docking, and screening tasks. We tested the resulting task-specific SFs on the corresponding test PLCs that are also characterized by the same 2714 descriptors. The second set of MP SFs are trained using (16 - 2 =) 14 descriptor types where the 320 protein-based descriptors *(REPAST and RETEST) are excluded* from the total 16 types used to train the full MP version. As for the SP approach, we considered each of the (16 - 4 =) 12 descriptor sets individually to characterize the training PLCs and then built XGBoost SFs for each descriptor set for the three tasks. We excluded the receptor-specific (REPAST & RETEST) and ligand-specific (ECFP & PADEL) descriptor sets from SP SFs since they only characterize the protein or the ligand molecules but not the protein-ligand complex as a whole. We then evaluated all SP-based SFs on out-of-sample validation PLCs whose interactions were modeled using the corresponding descriptor set. We select the descriptor set that is associated with the best overall scoring, docking, and screening accuracies on the out-of-sample validation complexes to build a single-perspective (SP) SF. Our results indicate that the 218 descriptors used by the neural-network SF NNScore are the best set of features for the three tasks.

Table 3.2 shows the scoring, docking, and screening powers of single and multi-perspective SFs on known targets (using the core test set (*Cr*) and cross-validation (*CV*)) and novel proteins via leave-clusters-out (*LCO*) validation. The three rows of the table are for the three different tasks, the three major columns are for different PLC characterization techniques, and the three sub columns list the performance on different training-test sampling strategies. The SP XGBoost SF that obtained the best overall accuracy on validation datasets used the 218 NNScore descriptors. The performance statistics in the table show that both sets of MP XGBoost SFs are significantly and consistently better than the SP SF for the three tasks whether on known (*Cr* and *CV*) or novel (*LCO*) test protein targets. The average improvement of the full MP approach over SP in scoring accuracy (across *Cr*, *CV*, and *LCO* test sets) is more than 8%, and more than 19% for the screening and docking tasks. Also, by comparing the summary statistics in the two right-most major columns we observe that MP SFs that trained with REPAST and RETEST descriptors are consistently more accurate than MP SFs trained without them. This highlights the utility of the proposed receptor-based descriptors in improving overall accuracy especially for the scoring task. It should be noted that all three SP and MP SFs were based on the same ML algorithm XGBoost with the same parameters, trained and tested on the same complexes, and evaluated using the same metrics. The only difference between the three sets of SFs is the descriptors they employ to characterize their training and test PLCs. All other factors that may have an effect on the performance results are fixed. Given the substantial improvement we report in Table 3.2 between SP and the two versions of MP, it is evident that multi-perspective modeling of protein-ligand interactions is a very promising technique in enhancing the accuracy of ML SFs. In addition to XGBoost, we were able to observe similar levels of improvements for other ML algorithms such as Random Forest, boosted and Bagged Ensemble Neural Networks [24], and Support Vector Machines.

3.4.2 Number of perspectives vs. number of training protein-ligand complexes

Public biochemical databases such as PDB [81], PubChem [71], PDBbind [22], Zinc [68], and others are the inspiring successful projects behind the creation of DDB. These public databases



(a) Number of training targets

(b) Number of descriptor sets (multiperspecitves)

Figure 3.5: The scoring accuracy of multi-perspective scoring functions trained on varying numbers of perspectives and known targets.

have had a tremendous impact on advancing virtual screening and other molecular modeling applications. We believe our database will be a very important complement in this space of public biochemical data and will significantly contribute to the advancement of machine-learning SFs. In this section, we quantitatively show how SFs benefit from training on increasing numbers of public complexes that are characterized by an increasing number of descriptor sets. In other words, we demonstrate how the performance of SFs improves as new PLCs and perspectives of their interactions are added to public databases overtime. To mimic the improvement of ML SFs in response to the release of new PLCs, we randomly sample an increasing sizes of training subsets from the 3251 primary training set *Pr*. We randomly select x base PLCs from the 3251 complexes in *Pr*, where $x \in \{100, 680, 1260, 1840, 2420, 3000\}$, and use the selected PLCs to train XGBoost SFs for the three tasks. We then test them on the disjoint core set *Cr* for the scoring, docking, and screening tasks. This process is repeated 100 times to obtain robust average R_p, $S_1^1\%$, and $EF_{1\%}$ values, which are plotted in Figure 3.5(a). In the three tasks, we observe that the performance XGBoost SFs improves as the size of the training dataset increases. The slope of the three curves indicates that the these ML SFs have a potential for improvement as more training data becomes available in the future.

To investigate the effect of increasing number of descriptor sets, we randomly select a maximum of 100 combinations of x descriptor sets from all possible $\binom{16}{x}$ combinations of the 16 types listed in Table 3.1, where $x \in \{1,4,7,10,13,16\}$, and use them to characterize the training set complexes, which we then use to train XGboost SF. These models are subsequently tested on the *Cr* dataset characterized by the same descriptors. For each number of descriptor sets, x, the performance of the 100 SFs are averaged to obtain robust overall R_p, $S_1^1\%$, and $\mathrm{EF}_{1\%}$ statistics, which are plotted in Figure 3.5(b). Similar to the case of increasing training data size in terms of number of new PLCs, the performance of the ML SFs steadily improves by increasing the number of protein-ligand interaction models in all three tasks. It is clear from the right and left panels of Figure 3.5 that the number of different models of protein-ligand interactions are as important as the number of training complexes in improving the quality of prediction. Based on these results, we believe that the public database of protein-ligand interactions we are proposing in this work will be of high value to ML SFs and just as important to their accuracy as the resources of raw structural and experimental data.

3.4.3 Perspective filtering using automatic feature subset selection

In addition to hosting descriptor extraction tools, DDB also offers descriptor filtering and analysis toolbox. The toolbox is developed to fulfill the following needs: (i) automatically guard against noisy and irrelevant descriptors, (ii) provide insight about the effect of different intermolecular forces on the bioactivity of interest, and (iii) distill the information contained in all descriptors into a compact subset of accurate features for applications where high throughput is required. Descriptor filtering is essentially a problem of feature subset selection (FSS) in machine learning. The naïve approach of performing FSS is to search the feature space for the most relevant descriptors using brute force and drop the remaining least important subset. Eliminating the least relevant features from a pool of thousands of descriptors using exhaustive enumeration is impractical because of the combinatorial explosion of the number of all possible subsets ($2^P - 1$, where P is the pool size). Numerous alternative approaches have been proposed in the literature that attempt to

find accurate subset of features in a reasonable time. These approaches can be grouped into three major categories. First are *filter-based* approaches in which one descriptor is considered at a time to examine its relevance score to the property we wish to predict. The descriptors that have the lowest scores (e.g., the lowest correlation with the dependent variable) are filtered out. Although filter techniques are fast, they ignore descriptor interactions with each other since every feature is evaluated separately. The second category of FSS techniques are called *wrapper* methods in which a large number of descriptor subsets with varying sizes and combinations are constructed and examined according to some search heuristic. A classification or regression ML model is typically fitted to each descriptor subset to examine their accuracy on an out-of-sample test set. After evaluating each subset, the wrapper search technique, typically an evolutionary algorithm, explores the descriptor space in the vicinity of the most accurate subsets by generating new combinations of descriptors derived from the best subsets so far. The search continues until a predefined number of search rounds has been performed or a target fitness value has been obtained or reached a plateau. In either case, wrapper search algorithms such as Genetic Algorithms (GA) [93] or Particle Swarm Optimization (PSO) [94] require a large number of descriptor subsets to explore and search rounds to arrive at an optimal descriptor subset. Based on our experiments using GA, we found that it is computationally very expensive to reliably search for the best descriptor subset when the number of descriptors exceed few hundreds.

The third FSS technique is based on the *embedded* paradigm. As the name implies, the search for an optimal subset of descriptors is embedded into, or built in, the ML algorithm itself and therefore it is more computationally efficient than the wrapper approach. It is also more accurate than filter methods since descriptor interactions are considered by the ML model. Due to these properties, we choose the embedded method to be the basis of our FSS algorithm. The algorithm is based on Random Forest's automatic variable importance calculation in conjunction with an in-house approach of iterative backward elimination to remove redundant, noisy, and irrelevant descriptors.

Automatic variable importance calculation is a very useful byproduct in ensemble models. In

addition to utilizing it for filtering, it helps identify the most critical interactions that contribute to the predictive accuracy of the model. Random forest utilizes out-of-bag (OOB[2]) PLCs to calculate the effect of each descriptor on the predicted bioactivity of interest. This is achieved by monitoring the change in the model's accuracy while permuting (shuffling) input descriptors randomly one at a time. With each random permutation of the descriptor values across the OOB complexes (i.e., descriptor noising), the corresponding increase in mean-square-error (MSE) on the OOB examples (OOBMSE (permute)) is measured. By comparing the intact OOBMSE to thus computed OOBMSE (permute), the average increase in MSE is evaluated. For each tree t in the ensemble model, standard OOBMSE is determined as follows: $\text{OOBMSE}^{(t)} = \frac{1}{\tilde{N}} \sum_{i=1}^{\tilde{N}} (y_i - \hat{y}_i^{(t)})^2$, where $\tilde{N}$ is the number of OOB examples of the tree t. The criterion OOBMSE for the same tree when permuting the input variable X^j is defined as: $\text{OOBMSE}_j^{(t)} = \frac{1}{\tilde{N}} \sum_{i=1}^{\tilde{N}} (y_i - \hat{y}_{i,j}^{(t)})^2$. For each descriptor X^j in every tree t in the ensemble, the increase in MSE is calculated as follows: $\Delta\text{OOBMSE}_j^{(t)} = \text{OOBMSE}_j^{(t)} - \text{OOBMSE}^{(t)}$. To calculate the importance ($\mathscr{I}$) of descriptor j, the resulting $\Delta\text{OOBMSE}_j^{(t)}$ values are averaged and normalized over all the trees in the ensemble according to the formula: $\mathscr{I}^j = \mu_{\Delta\text{OOBMSE}_j} / \sigma_{\Delta\text{OOBMSE}_j}$. The overall increase in MSE ($\mathscr{I}^j$) for an input descriptor X^j can be considered its influence on predicting the final binding affinity or any other dependent variable.

Based on variable importance and backward elimination we implement the descriptor filtering approach depicted in Algorithm 2. (i) First, the number of descriptor subsets (M) to test are defined by the user. The sizes of the M feature subsets start from the full dimension P down to 3 with a fixed step of size $\sim (P-3)/(M-1)$. Here, we arbitrarily choose three to be the lowest number of descriptors to test. (ii) Starting from the largest descriptor subset (whose size is $\mathscr{M}(1) = P$), an RF model is built and its relative variable importance $\mathscr{I}^{(1)}$ as well as out-of-sample loss L_1 are recorded. (iii) We then use variable importance scores to select the $\mathscr{M}(2)$ most influential descriptors $V^{(2)}$ and eliminate the remaining less important $\mathscr{M}(1) - \mathscr{M}(2)$ features ($V^{(1)} - V^{(2)}$). Next,

[2]Out-of-bag (OOB) refers to complexes that are not sampled from the training set when bootstrap sets are drawn to fit individual trees in a random forest model—on average, about 34% of the training set complexes are left out (or "out-of-bag") when bootstrap sets are drawn.

Algorithm 2 Descriptor filtering algorithm

Input 1: Training data: $D = \{X_i, y_i\}_{i=1}^{N}$
Input 2: The number of descriptor subset sizes to test: M
Subset sizes $\mathscr{M}$ = *Sequence(from = P, down to = 3, decrement by =* $\sim (P-3)/(M-1)$*)*
Set the initial variable importance values for all P descriptors to some arbitrary value (e.g., 1) as $\mathscr{I}^{(0)} = \{\mathscr{I}_j = 1 \mid 1 \leq j \leq P\}$
for $p = 1$ to M **do**
    $V^{(p)}$ = choose $\mathscr{M}(p)$ descriptors with the largest V. I. values according to $\mathscr{I}^{(p-1)}$
    Characterize PLCs with the descriptor set $V^{(p)}$ as $D_p = \left\{X_i^{(V^{(p)})}, y_i\right\}_{i=1}^{N}$
    Train model F_p on data D_p as $F_p = fit(D_p)$
    Compute variable importance of the set $V^{(p)}$ as $\mathscr{I}^{(p)} =$ Var. Imp.(F_p)
    Calculate the loss of model F_p as $L_p =$ OOBMSE(F_p)
end for
Find the optimal subset of descriptors $V^{(\text{optimal})} = V^{(p^*)}$, where $p^* = \arg\min_p L_p$

Table 3.3: The docking, screening, and scoring accuracies (A) of multi-perspective scoring functions constructed using raw (R) and noisy (R+N) descriptors. The number of descriptors (P) and accuracies (A) are reported when all features are used (Full) and after conducting feature subset selection (FSS).

Task (metric)	Raw descriptors (R)				Raw & noise descriptors (R+N)			
	$P_{\text{R}}^{\text{Full}}$	$A_{\text{R}}^{\text{Full}}$	$P_{\text{R}}^{\text{FSS}}$	$A_{\text{R}}^{\text{FSS}}$	$P_{\text{R+N}}^{\text{Full}}$	$A_{\text{R+N}}^{\text{Full}}$	$P_{\text{R+N}}^{\text{FSS}}$	$A_{\text{R+N}}^{\text{FSS}}$
Scoring (R_p)	2714	0.816	216	0.804	5428	0.772	378	0.823
Docking ($S_1^1\%$)	2714	82.05	270	81.95	5428	79.49	216	81.54
Screening ($\text{EF}_{1\%}$)	2714	34.00	216	34.00	5428	32.00	541	33.50

we build a new RF SF and record its out-of-sample loss L_2 as well as its importance estimates $\mathscr{I}^{(2)}$ of the descriptor subset with size $\mathscr{M}(2)$. (iv) We repeat this process of model construction, out-of-sample loss calculation, variable importance estimation, and backward elimination until all M (where $\mathscr{M}(M) = 3$) descriptor subsets have been tested. Upon completion, the most informative descriptors will be the subset associated with the lowest out-of-sample loss. The variable importance values of the best subset is a good estimate of the contribution of different molecular forces to the system's binding affinity.

We conducted two types of experiments to show the effectiveness of DDB's descriptor filtering

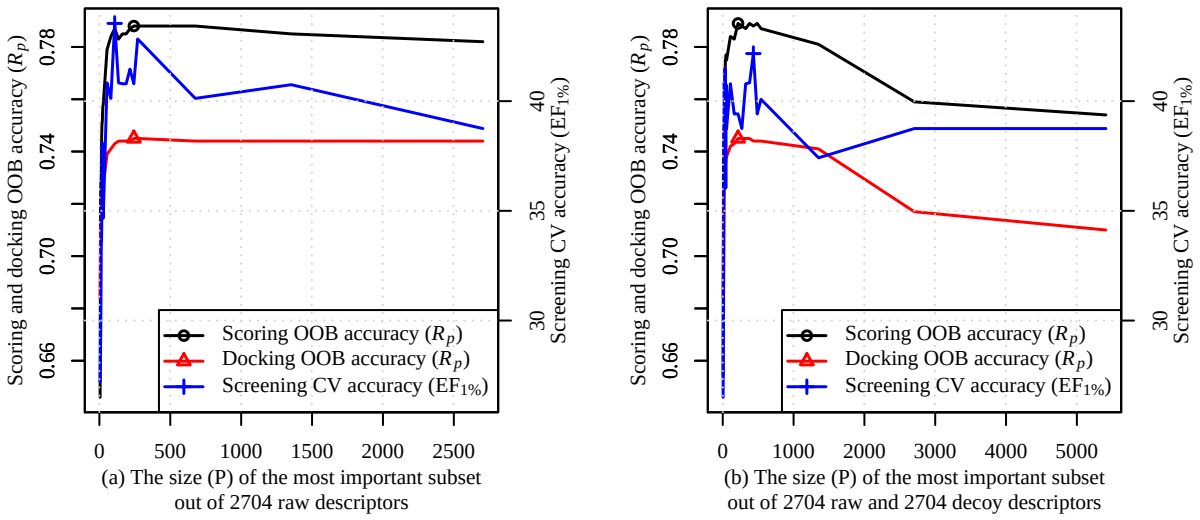


Figure 3.6: The scoring, docking, and screening accuracy during the filtering of raw (left) and noisy (right) descriptors.

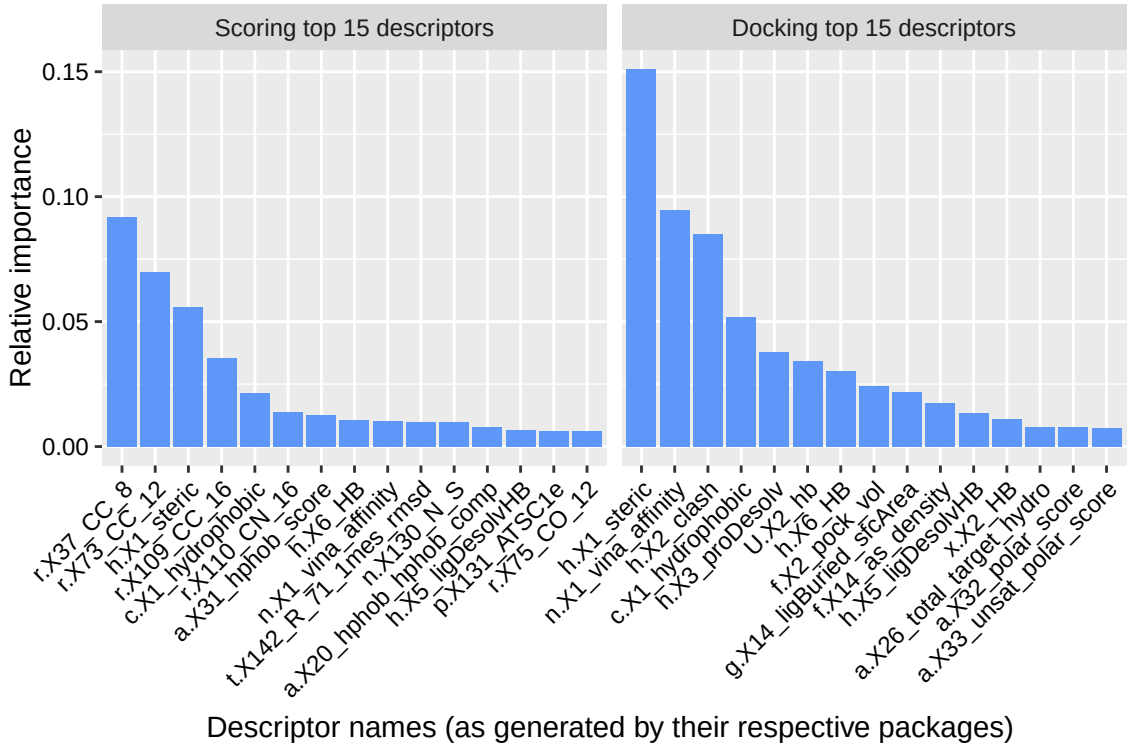


Figure 3.7: The relative influence of the top 15 (out of 2714) descriptors on predicting the ligand's binding affinity (left panel) and poses (right panel) for the core test set complexes. The x-axis shows abbreviated names for the descriptors in which the first letter denotes the symbolic identifier of the SF or tool generating the descriptor (see Table 3.1), the following ".X#" is the original SF's unique index of the descriptor, and the suffix is the descriptor short name as produced by the original SF or extraction tool.

capabilities. In the first, we applied the iterative descriptor importance estimation and backward elimination algorithm on DDB's 2714 original (or raw, denoted by R) descriptors for the scoring, docking, and screening tasks. The goodness of fit metric for the scoring and docking is the MSE between predicted and true labels of OOB PLCs. The true PLC labels for scoring are the experimental BA values and for docking it is the RMSD values of PLC poses. The objective function for the screening task is the average enrichment factor of active ligands on ten out-of-sample test sets. Upon calculating the optimal filtered subset for each task, we used them to build XGBoost SFs whose accuracy statistics are listed in Table 3.3. The second major column of Table 3.3 shows (from left to right) the number (P_R^{Full}) of the original (R) descriptors, the accuracy (A_R^{Full}) of XGBoost SFs trained on the raw descriptors, the number (P_R^{FSS}) of descriptors in the best feature subset, and the accuracy (A_R^{FSS}) of XGBoost SFs trained on this subset of filtered descriptors. We observe that the accuracy of the SFs did not change significantly (A_R^{Full} vs. A_R^{FSS}) despite the use of (270/2714=) 10% of the total number of descriptors (P_R^{Full} vs. P_R^{FSS}). We further found that the best descriptors selected for each task include subsets from 16 descriptor types. Note that the size of the best descriptor subset is comparable to that of NNScore whose 218 descriptors were found to be the best performing when single-perspectives were evaluated individually as shown in Table 3.2. However, the performance of the two descriptor sets (as listed in Table 3.2 and Table 3.3) are quite different and this implies the effectiveness of multi-perspective modeling of interactions and the feature filtering algorithm. Figure 3.6(a) shows the search path for the best subset of descriptors for the three prediction tasks. The *x* axis of the plot is for the descriptor subset size and the *y* axis is for the accuracy of that subset on out-of-sample PLCs. The tick marks (the small square, circle, and triangle) on the plot correspond to the size of the best subsets and their performance. The results in the plot agree with those in the table in the fact that the performance for large descriptor subsets are not very different from those for smaller subsets. The gap in performance for the screening task could be due to the difference in the ML models used for filtering (RF in the plot) and the final SF construction (XGBoost in the table).

The results of the first experiment clearly indicate that valuable computational resources could

be saved by only extracting the most important descriptors without affecting performance. This was one of the objectives of implementing descriptor filtering as part of DDB. The approach of the second experiment is very similar to the first, but the objectives are different. The main objective is to simulate the possibility of depositing noisy descriptors to DDB. Noise could be introduced as a result of improper modeling of some interactions, bugs in the donated descriptor extraction program, or due to any other artifacts. Therefore, in this experiment we evaluate the descriptor filtering algorithm on its ability to remove noise that is added as artificial descriptors. We first added as many as 2714 random noise (N) variables to the 2714 raw (R) descriptors. We then fitted two sets of SFs. The first set of XGBoost SFs are trained directly on the 5428 mixed (R+N) descriptors. For the second, we first applied the descriptor filtering algorithm on the 5428 mixed (R+N) descriptor set. After obtaining the filters for the three tasks, we fitted another set of XGBoost SFs on the resulting filtered descriptors. The results of both experiments are shown in the right-most major column of Table 3.3. Although not very significant, we notice a drop in performance of the SF that was trained on the noisy descriptors compared to its counterpart that was constructed on the original features (i.e., A_R^{Full} vs. A_{R+N}^{Full}). This indicates the resilience of XGBoost SFs to such a large amount of noise (100%) in the data. After applying DDB filters, we notice a rebound in the performance of the ML SF back to normal. We examined the final three subsets of descriptors and we found that none of the 378, 216, and 541 selected descriptors are the noisy set we intentionally added. This clearly shows the accuracy of the embedding-based FSS algorithm in finding and removing noise.

Drug designers and medicinal chemists are often interested in understanding the influence of various intermolecular forces on the ligand's binding affinity and conformation. DDB's filtering and ML toolbox serves this need through its descriptor importance plots such as the ones depicted in Figure 3.7. Out of the 2714 descriptors in DDB, the bar plots illustrate the relative importance of the top 15 descriptors with the largest effect on predicted binding affinities and poses of PLCs in the core test set *Cr*. For binding affinity, RF's variable importance plot suggests that descriptors that have the most predictive power are related to hydrophobic contacts followed by

steric effects, hydrogen bonding, and ligand solvation. Whereas descriptors that characterize the ligand's steric effects and clashes with the protein were found to play the biggest role in estimating its binding pose. The descriptor based on NNScore's term for Vina's (SF) affinity score is among the top predictors of binding affinity and conformation. The volume of the pocket calculated by DPOCKET [88] appears to affect binding pose prediction but does not seem to have the same effect on BA estimation. On the other hand, the number of Carbon-Carbon atom pairs between the protein and ligand within 8 and 12 Å are very relevant for scoring but not docking. The descriptors based on these simple atomic contacts are calculated by the SF RF-Score (descriptor codes with the prefix "r"). The other top 15 scoring and docking descriptors shown in Figure 3.7 are generated by the following diverse set of SFs and characterization tools: ChemGauss (prefix h), AffiScore (a), NNScore (n), PaDEL (p), RETEST (t), DPOCKET (f), AutoDock Vina (U), Cyscore (c), and X-Score (x). In some cases, these tools overlap in features that describe the same interactions but using different modeling approaches. For example, hydrophobic contacts are modeled by RF-Score's CC_8, CC_12 & CC_16 contacts, Cyscore's hydrophobic term, and AffiScore's hydrophobic score. Hydrogen bonding-related descriptors are also generated by X-Score and ChemGauss using two different perspectives. Although the top 15 features capture a diverse collection of intermolecular forces, some of them characterize the same interaction type which highlights the benefits of the proposed multi-perspective modeling paradigm.

3.5 Conclusion

In this chapter, we presented Descriptor Data Bank (DDB), an online platform for facilitating multi-perspective modeling of PL interactions. The online platform is an open-access hub for depositing, hosting, executing, and sharing tools and data arising from a diverse set of interaction modeling hypotheses. In addition to the descriptor extraction tools, the platform also implements a machine-learning (ML) toolbox that drives DDB's automatic descriptor filtering and evaluation utilities as well as scoring function fitting and prediction functionalities. To enable local access to many of DDB's utilities, command-line programs and a PyMOL plug-in have also been devel-

oped and can be freely downloaded for offline multi-perspective modeling. We seed DDB with 16 diverse descriptor extraction tools developed in-house and collected from the literature. The tools altogether generate over 2700 descriptors that characterize (i) proteins, (ii) ligands, and (iii) protein-ligand complexes. The in-house descriptors we propose here, REPAST & RETEST, are target-specific and based on pair-wise sequence and structural alignment of proteins followed by target clustering and triangulation. We built and used the fit/predict service in DDB to fit ML SFs to the in-house descriptors and those collected from the literature. We then evaluated them on several data sets that were constructed to reflect real-world drug screening scenarios. We found that multi-perspective SFs constructed using large and diverse number of DDB interaction models outperformed their single-perspective counterparts in all evaluation scenarios with an average improvement of 15%. We also found that our proposed target-specific descriptors improve upon the accuracy of SFs that were trained without them. In addition, DDB's filtering module was able to exclude noisy and irrelevant descriptors when artificial noise was included as new features. We also observed that the tree-based ensemble ML SFs implemented in DDB are robust even in the presence of noisy and decoy descriptors.

CHAPTER 4

BAGGING AND BOOSTING BASED ENSEMBLE NEURAL NETWORKS SCORING FUNCTIONS FOR ACCURATE BINDING AFFINITY PREDICTION

4.1 Binding affinity

In the context of drug discovery and development, binding affinity is the strength of the non-covalent bond that brings a protein and a ligand together which is typically quantified using the *Dissociation constant* (K_d). It is an equilibrium constant that measures the tendency for a complex to separate into its constituent components (protein and ligand). We utilize it here to describe the degree of tightness of proteins to their binders. That is, by interpreting complexes whose components are more likely to dissociate (high dissociation constants) as loosely bound (low binding affinities) and vice-versa. We can write the binding reaction for a protein-ligand complex as: *Protein-Ligand Complex* $\leftrightarrow$ *Protein* + *Ligand*, and the dissociation constant in terms of molars (mol/L) is given by: $K_d = [P][L]/[PL]$, where $[P]$, $[L]$, and $[PL]$ are the equilibrium molar concentrations for the protein, ligand, and protein-ligand complex, respectively. The higher the concentration of the ligand needed to form a stable protein-ligand complex, the lower the binding affinity will be. Put differently, as the concentration of the ligand increases, the value of K_d also increases and this corresponds to a weaker binding.

In some experimental settings, such as competition binding assays, *inhibitor constant* is used as a measure of binding affinity. Inhibitor constant (commonly denoted by K_i) is defined as the concentration of a competing ligand in a competition assay that would occupy 50% of the receptor in the absence of radioligand. Inhibitor constant is expressed by the following Cheng-Prusoff equation [95]:

$$K_i = \frac{IC_{50}}{1 + [L]/K_d}, \tag{4.1}$$

where IC_{50} is the half maximal inhibitory concentration that measures the effectiveness of the ligand to inhibit a biochemical or biological function by half, K_d is the dissociation constant of

the radioligand, and $[L]$ is the concentration of the ligand L. IC_{50} could also be thought of as the concentration of the ligand required to displace 50% of the radioligand's binding. IC_{50} value is typically determined using dose-response curve [96] and it should be noted that it is highly dependent on experimental conditions under which it is calculated. To overcome such dependency, the inhibitor constant formula K_i is developed such that it has an absolute value. An increase in its value means a higher concentration of an inhibitor needed to block certain cellular activity.

It is customary to logarithmically convert the dissociation and inhibitory constants to numbers that are more convenient to interpret and deal with. These constants are scaled according to the formula:

$$pK_d = -\log(K_d) \quad \text{and} \quad pK_i = -\log(K_i), \tag{4.2}$$

where higher values of pK_d and pK_i reflect an exponentially greater binding affinity. The data we use in our work are restricted to complexes whose either K_d or K_i values are available. Therefore, in many of the regression models that will be built are fitted so that the response variable is either pK_d or pK_i. The binding affinities of the complexes in PDBbind database considered in this work range from 2 to 11.96 in terms of pK_d and pK_i, spanning about 10 orders of magnitude.

4.2 Predicting binding affinity

Unknown binding affinities of protein-ligand complexes are typically estimated by predictive models known as scoring functions. Most existing scoring functions employed in commercial and free molecular docking software fall in one of three main categories: force-field-based [64], empirical [56], or knowledge-based [97] SFs. Many comparative studies have found that these types of SFs are not accurate enough for reliable and successful molecular docking and virtual screening. A recent study examined a total of 16 popular scoring functions in their ability to reproduce experimental binding affinities of 195 protein-ligand complexes that encompass 65 different protein families [17]. Although these SFs are employed in mainstream commercial and academic molecular docking tools, the best performing SF achieved only mediocre accuracy of less than 0.65 in terms of Pearson's correlation between its predictions and measured binding affinities (BAs).

These findings are in agreement with an earlier work by Wang et al. in which a related benchmark and scoring functions were examined [98]. Several of the evaluated SFs were empirical models derived via fitting linear regression equations to training data, but none were based on nonlinear modeling approaches such as Artificial Neural Networks (ANN) or ensemble decision trees.

Artificial neural networks have been previously used in computational drug development, but they have mostly been applied in QSAR modeling problems or in predicting the biological activity of ligands (active or not) against a target protein [99, 100, 101]. Their application in predicting binding affinity has been very rare and only reported in small scale experiments in which just a handful of protein-ligand complexes were used for training and validation [102, 103, 50]. Neural networks' poor generalization performance when trained on small and high-dimensional datasets is perhaps the main reason for their limited use in scoring protein-ligand complexes in commercial docking tools. In this chapter, we propose novel SFs based on an ensemble of neural networks to predict binding affinity of protein-ligand complexes characterized using large and diverse number of descriptors. We train and test our models on hundreds of high-quality protein-ligand complexes and compare their accuracies against conventional and state-of-the-art scoring functions. We show that our NN SFs are resilient to overfitting and generalize well even when predicting BAs of complexes characterized by a large number of descriptors.

4.3 Key contributions

Conventional empirical SFs rest on the hypothesis that a linear regression function is sufficiently capable of modeling protein-ligand binding affinity [63, 56]. Instead of assuming a predetermined theoretical function that models the unknown relationship between different energetic terms and binding affinity, two accurate nonparametric SFs, BgN-Score and BsN-Score, based on ensemble and deep learning approaches are introduced in this chapter. We utilize the multi-perspective molecular descriptors (proposed in Chapter 3) to build the SFs *BgN-Score* and *BsN-Score* by combining a large number of diverse neural networks using bagging and boosting ensemble techniques, respectively. We show that BgN-Score and BsN-Score have scoring powers of

0.840 and 0.844 in terms of Pearson's correlation coefficient between predicted and true binding affinities of PDBbind (version 2014) core test set complexes, respectively. This is in comparison to 0.627 for the best conventional SF, X-Score, on the same benchmark test set. In addition to this substantial 34% improvement, the same ensemble NN SFs are also more accurate than SFs based on a single neural network (0.840 and 0.844 vs. 0.803). We also compare our proposed models to a SF based on other popular machine-learning algorithm Random Forests. We found that our ensemble NN SFs are at least 16% more accurate than RF-Score, an SF based on Random Forests. Scoring functions based on a boosted trees model fitted using the XGBoost algorithm achieved 0.827 correlation with true binding affinity values. Although NN and decision-tree based ensemble approaches are competitive with each other, the significance of NN ensemble SFs introduced in this work is two-fold. First, they represent a way to overcome the overfitting limitations of single neural network models that have been used traditionally in drug-discovery applications [99, 100, 50]. Second, neural networks have the ability to approximate any underlying function smoothly [104, 105, 106] in contrast to decision trees that model functions with step changes across decision boundaries [107].

We seek to advance structure-based drug design by designing SFs that significantly improve upon the protein-ligand binding affinity prediction accuracy of conventional SFs. Our approach is to couple the modeling power of ensemble algorithms and deep learning with training datasets comprising hundreds of protein-ligand complexes with known high-resolution 3D crystal structures and experimentally-determined binding affinities and a variety of features characterizing the complexes. We will compare the predictive accuracies of BgN-Score, BsN-Score, single deep neural network SF (DNN-Score), and other machine-learning approaches as well as existing conventional SFs of all three types, force-field, empirical, and knowledge-based, on diverse and homogeneous sets of protein families. The remainder of the chapter is organized as follows. We first describe the training and test datasets used to construct and evaluate several neural network and machine-learning scoring functions. The next section covers some of the limitations of artificial neural networks and our approach to tackling them. Then we present our proposed SFs based on

these models. Next, we show results comparing the scoring and ranking powers of conventional and the proposed SFs on diverse and homogeneous test sets. We also compare their performance on novel drug targets and analyze how they are impacted by training set size and the number of descriptor sets characterizing protein-ligand complexes. Finally, we summarize our results and conclude this chapter.

4.4 Methodology

4.4.1 Protein-ligand complex datasets and multi-perspective characterization

For most of our experiments in this work, we use two versions of PDBbind to train and test the proposed scoring functions. We build and evaluate the proposed scoring functions on complexes from the database PDBbind versions 2007 and 2014 as outlined in Section 2.1. These two versions were used in recent studies to evaluate the performance of several academic and commercial scoring functions [17, 25, 27, 50]. In order to objectively compare the predictive accuracy of the proposed scoring functions to many existing popular SFs, we too use the 2007 and 2014 PDBbind benchmark. The experiments on the two versions were conducted during different stages of this work and therefore there are some changes concerning the machine-learning methods (addition of XGBoost) and their libraries (python & R), the descriptors we extract for each protein-ligand complex, as well as other variations that will be highlighted in the text.

4.4.1.1 The PDBbind 2007 benchmark

We use the 2007 version of PDBbind to train and test scoring functions based on the machine learning algorithms NN, RF, BRT, SVM, *k*NN, MARS, and MLR. For each of the 1300 protein-ligand complexes in PDBbind 2007, we extracted physicochemical features used in the empirical SFs X-Score [56] (a set of 6 features denoted by X) and AffiScore [108] (a set of 30 features denoted by A) and calculated by GOLD [52] (a set of 14 features denoted by G), and geometrical features used in the ML SF RF-Score [27] (a 36-feature set denoted by R). The software packages that calculate X-Score, AffiScore (from SLIDE), and RF-Score features were available to us in an open-source form

from their authors and a full list of these features is provided in the appendix. The GOLD docking suite provides a utility that calculates a set of general descriptors for both molecules as separate entities and in a complex form. The full set of these features can be easily accessed and calculated via the *Descriptors* menu in GOLD. By considering all fifteen combinations of these four types of features (i.e., X, A, R, G, $X \cup A$, $X \cup R$, $X \cup G$, $A \cup R$, $A \cup G$, $R \cup G$, $X \cup A \cup R$, $X \cup A \cup G$, $X \cup R \cup G$, $A \cup R \cup G$, and $X \cup A \cup R \cup G$), we generated 15 corresponding versions of scoring functions. Each scoring function is trained and evaluated on complexes characterized by one of the 15 descriptor combinations. We denoted these models using the name of the machine-learning algorithm (as a prefix) and the feature combintion (as a suffix). The scoring function *BsN-Score::XR*, for example, denotes the boosted ensemble neural network model fitted to complexes characterized using the set of descriptors $X \cup R$ (referred to simply as XR).

4.4.1.2 The PDBbind 2014 benchmark

In addition to the 2007 release of PDBbind, we also take advantage of PDBbind 2014 which includes 3446 high-quality complexes—about 160% more complexes than the 1300 PLCs in PDBbind 2007. We use protein-ligand datasets from PDBBind 2014 to train and test machine-learning scoring functions based on NN, RF, XGB, and MLR. All protein-ligand complexes are characterized using the proposed multi-perspective descriptors in Descriptor Data Bank (DDB). The set includes more than 2700 descriptors extracted using 16 different scoring functions and molecular modeling tools developed in house and collected from the literature which are now part of DDB. The descriptor sets are comprised of the types X, A, R, G, which were also considered in our experiments on PDBbind 2007 and 12 other types (listed in Table 3.1) extracted for PDBbind 2014 complexes only. We only fit one version of each machine-learning scoring function to all 16 feature types instead of creating as many versions as the number of combinations of the 16 descriptor types. We distinguish scoring functions using their underlying machine-learning algorithms. For example, BsN-Score refers to the scoring function built using the proposed boosted neural network model and fitted to more than 2700 multi-perspective descriptors from DDB.

4.4.2 Limitations of neural networks and our approach to tackling them

Although multi-layer (or deep) ANN models can theoretically approximate any nonlinear continuous function, their application in drug-discovery related problems has always been complicated by several challenges. Data arising from bioinformatics and cheminformatics processes are typically high-dimensional. Since ANN models cannot handle large number of features efficiently when data is scarce, a pre-processing step prior to fitting the data using an ANN model is usually necessary. Feature subset selection using evolutionary algorithms or dimension reduction using, say, principal component analysis (PCA), is commonly applied to overcome this problem. However, valuable experimental information may be discarded when only a small subset of features is selected to build a prediction model. The dimensionality-reduction approach is also complicated by the fact that the underlying data distribution is unknown and hence making the right choice of which dimensionality-reduction technique to apply is a tricky problem in itself. In addition to these pre-processing issues, training ANN models is also a challenging task because their weights can not be guaranteed to converge to optimal values. This causes NN models to suffer from high variance errors which translate to unreliable SFs.

The aforementioned problems can be avoided and state-of-the-art performance can be achieved by combining predictions of hundreds of diverse and nonlinear NN models. We propose here ensemble methods based on ANNs. The ensemble itself is trained on all the features, although each network in the ensemble is fitted to only a subset of the features. This approach relieves us from carrying out feature subset selection or dimensionality reduction prior to training. In fact, the performance of the ensemble can even be improved by describing the data with more relevant features. Moreover, it is no longer critical to tune the weights of each network in the ensemble to optimal values as it is the case for a single NN model. Suboptimal weight tuning of individual networks could contribute to decreasing the inter-correlation between them, which translates to a diverse ensemble and therefore a more accurate model [45].

Our proposed NN ensemble models are inspired from Random Forests [45] and Boosted Regression Trees [92] techniques in the formation of the ensembles. So far, the focus in ensemble

learning has been more or less biased towards using decision trees as base learners in forming ensembles. Choosing trees as base learners is mainly due to their high flexibility and variance (low stability). High variance decreases inter-correlation between trees and therefore increases the overall ensemble model's accuracy. Instead of using decision trees as base learners, we employ here multi-layered perceptron, or deep, neural networks (DNN). DNN shares several characteristics with prediction trees. They are non-parametric, nonlinear, and have high variance. Moreover, both techniques are very fast in prediction. Deep neural networks, however, have the ability of modeling any arbitrary boundary smoothly while decision trees can only learn rectangular-shaped boundaries. Decision trees are typically pruned after training to avoid overfitting, whereas DNN uses regularization while the network weights are optimized during learning. We next describe our two new ensemble NN models.

4.4.3 BgN-Score: Ensemble neural networks through bagging

Bootstrap aggregation, or bagging for short, is a popular approach to construct an ensemble learning model. As the name implies and as indicated in the third step of Algorithm 3, the ensemble is composed of neural networks that are fitted to bootstrap samples from the training data. To further increase the diversity of the ensemble and decrease its training time, the inputs to each network l are a random subset (p_l) of the total P features extracted for every protein-ligand complex (see Step 4). Feature sampling has proven effective in building tree-based ensemble algorithms such as Random Forests [45]. When the task is to predict the binding affinity of a new protein-ligand complex, the output is the aggregated average of the predictions of the comprising individual networks as shown in Algorithm 3 and depicted in Figure 4.1. This mechanism can be formally expressed as:

$$\hat{y} = f(\mathbf{x}^P) = \frac{1}{L}\sum_{l=1}^{L} f_l(\mathbf{x}^{p_l}) = \frac{1}{L}\sum_{l=1}^{L} \hat{y}_l, \tag{4.3}$$

where $\mathbf{x}^P \in \Re^{|P|}$ is a feature vector representing a protein-ligand complex characterized by a feature set P, $f(\mathbf{x}^P)$ is the function that maps it to binding affinity $\hat{y} \in \Re$, $\mathbf{x}^{p_l} \in \Re^{|p_l|}$ is the same

complex but characterized by a random subset p_l of features ($|p_l| < |P|$), L is the number of networks in the ensemble, and $\hat{y}_l$ is the prediction of each network l in the ensemble which is calculated at the output neuron according to Equation 2.14. The resulting bagging-based ensemble SF is referred to as *BgN-Score*.

Algorithm 3 Algorithm for building BgN-Score: an ensemble NN SF using bagging

1: *Input:* training data $\mathbf{D} = \{\mathbf{X}^P, \mathbf{Y}\}$, where $\mathbf{X}^P = \{\mathbf{x}_1^P, \ldots, \mathbf{x}_N^P\}$, $\mathbf{Y} = \{y_1, \ldots, y_N\}$, and N is the number of training complexes.
2: **for** $l = 1$ to L **do**
3: Draw a bootstrap sample $\mathbf{X}_l^P$ from $\mathbf{X}^P$.
4: Characterize the complexes in the bootstrap sample $\mathbf{X}_l^P$ using a random subset p_l of features: $\mathbf{X}_l^{p_l}$.
5: From $\mathbf{Y}$, draw the measured binding affinities of the complexes in the sample $\mathbf{X}_l^P$: $\mathbf{Y}_l$.
6: Construct a new training set: $\mathbf{D}_l = \{\mathbf{X}_l^{p_l}, \mathbf{Y}_l\}$.
7: Learn the current binding affinities by training an FFBP NN model f_l on $\mathbf{D}_l$.
8: **end for**
9: The final prediction of a protein-ligand complex $\mathbf{x}^P$ is: $\hat{y} = f(\mathbf{x}^P) = \frac{1}{L}\sum_{l=1}^{L} f_l(\mathbf{x}^{p_l}) = \frac{1}{L}\sum_{l=1}^{L} \hat{y}_l$

4.4.4 BsN-Score: Ensemble neural networks through boosting

Boosting is an ensemble machine-learning technique based on a stage-wise fitting of base learners. The technique attempts to minimize the overall loss by boosting the complexes having highest predicted errors, i.e., by fitting NNs to (accumulated) residuals made by previous networks in the ensemble model. There are several different implementations of the boosting concept in the literature. The differences mainly arise from the employed loss functions and treatment of most erroneous predictions. Our proposed NN boosting algorithm in this work is a modified version of the boosting strategy developed by Cao et al. [109] and Friedman [92] in that we perform random feature subset sampling. This approach builds a stage-wise model as listed in Algorithm 4 and illustrated in Figure 4.2. The algorithm starts by fitting the first network to all training complexes. A small fraction ($\nu < 1$) of the first network's predictions is used to calculate the first iteration of residuals $\mathbf{Y}_1^{res}$ as shown in Step 3 of Algorithm 4. Step 3 also shows that the network f_1 is the first term in the boosting additive model. In each subsequent stage l, a network is trained on

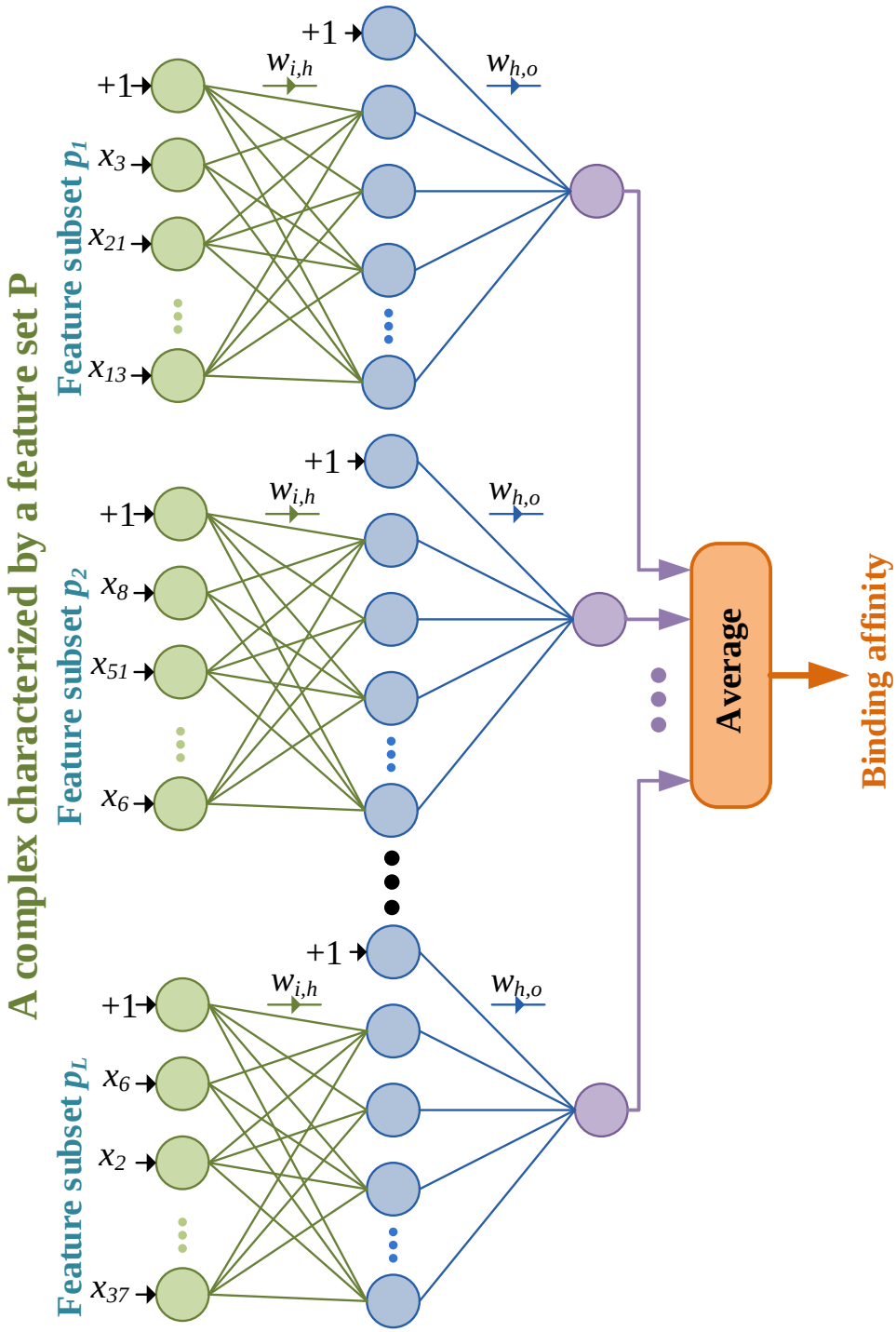


Figure 4.1: BgN-Score: ensemble neural network SF using bagging approach.

a bootstrap sample of the training complexes described by a random subset p_l of features (Steps 5 and 6). The values of the dependent variable of the training data for the network l are the current residuals corresponding to the sampled protein ligand complexes. The residuals for a

network at each stage are the differences between previous stage residuals and a small fraction of its predictions. This fraction is controlled by the shrinkage parameter $\nu < 1$ to avoid any overfitting. Network generation continues as long as the number of networks does not exceed a predefined limit L. Each network joins the ensemble with a shrunk version of itself. In our experiments, we fixed the shrinkage parameter to 0.001 which gave the lowest out-of-sample error. We refer to this boosting-based ensemble SF as *BsN-Score*.

Algorithm 4 Algorithm for building BsN-Score: an ensemble NN SF using boosting

1: *Input:* training data $\mathbf{D} = \{\mathbf{X}^P, \mathbf{Y}\}$, where $\mathbf{X}^P = \{\mathbf{x}_1^P, \ldots, \mathbf{x}_N^P\}$, $\mathbf{Y} = \{y_1, \ldots, y_N\}$, and N is the number of training complexes.
2: Construct $\mathbf{D}_1 = \{\mathbf{X}^{p_1}, \mathbf{Y}\}$ from $\mathbf{X}^P$ by selecting a random subset p_1 of features.
3: Train an FFBP NN model f_1 on $\mathbf{D}_1$ and use it to predict BAs ($\hat{\mathbf{Y}}_1$) of the complexes $\mathbf{X}^{p_1}$. Calculate the residuals: $\mathbf{Y}_1^{res} = \mathbf{Y} - \nu\hat{\mathbf{Y}}_1$.
4: **for** $l = 2$ to L **do**
5: Draw a bootstrap sample $\mathbf{X}_l^P$ from $\mathbf{X}^P$.
6: Characterize the complexes in the bootstrap sample $\mathbf{X}_l^P$ using a random subset p_l of features: $\mathbf{X}_l^{p_l}$.
7: From $\mathbf{Y}_{l-1}^{res}$, draw the residuals corresponding to the complexes in the sample $\mathbf{X}_l^P$: $\mathbf{Y}_{l-1}^{res*}$.
8: Construct a new training set: $\mathbf{D}_l = \{\mathbf{X}_l^{p_l}, \mathbf{Y}_{l-1}^{res*}\}$.
9: Learn the current residuals by training an FFBP NN model f_l on $\mathbf{D}_l$.
10: Calculate the predictions $\hat{\mathbf{Y}}_l$ of the NN model f_l on all $\mathbf{X}^{p_l}$ training complexes in the original training set $\mathbf{D}$.
11: Update the residuals: $\mathbf{Y}_l^{res} = \mathbf{Y}_{l-1}^{res} - \nu\hat{\mathbf{Y}}_l$
12: **end for**
13: The final prediction of a protein-ligand complex $\mathbf{x}^P$ is: $\hat{y} = f(\mathbf{x}^P) = \sum_{l=1}^{L} \nu f_l(\mathbf{x}^{p_l}) = \sum_{l=1}^{L} \nu\hat{y}_l$

4.4.5 Improving speed and accuracy of ensemble neural networks via transfer learning

The bagged and boosted ensemble neural networks proposed in this work involves building large number of wide and deep networks to accurately model the complex binding problem. Before training, the weights of each network in these ensemble models are initialized randomly and independently from one another. Tens of thousands of training steps are then applied to each network to learn the residuals made by the previous networks. We propose a more efficient boosting algorithm in which optimized weights from a trained network are used as initial weights for the next network

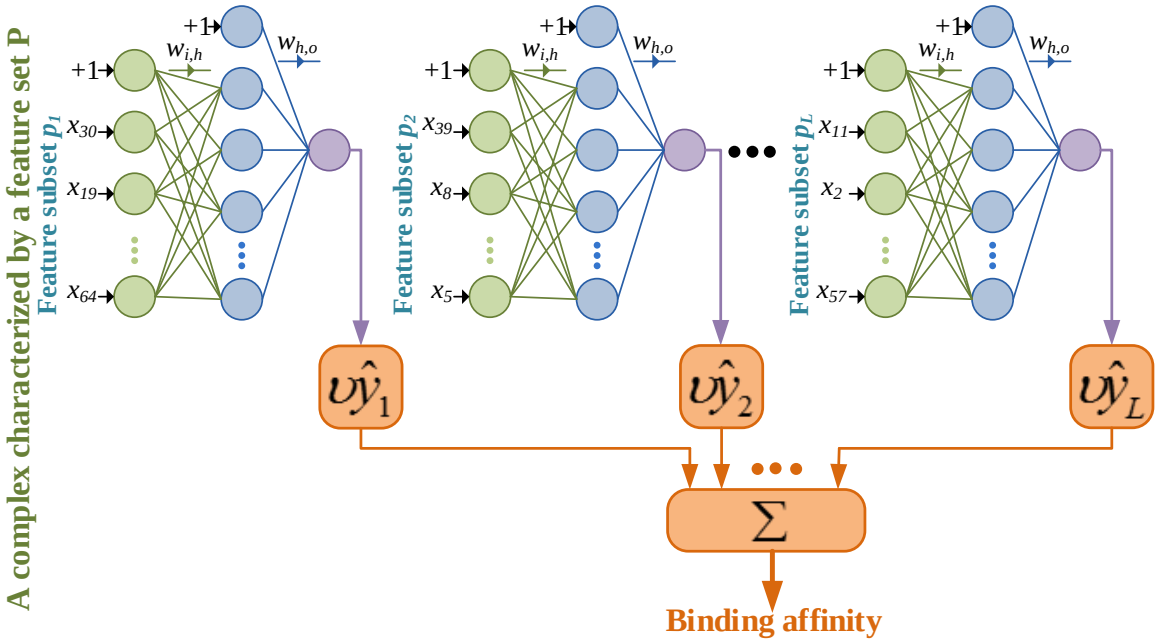


Figure 4.2: BsN-Score: ensemble neural network SF using boosting approach.

to be trained ($W_l^i = W_{l-1}^o$). The process of transferring knowledge from one model to another on a related problem is known as *transfer learning*. The networks whose weights are initialized with values transferred from optimized NNs are faster to train than initializing them randomly. Also, they tend to improve upon the performance of the networks from which the weights are initialized.

Depending on the particular training data, our neural networks typically require tens of thousands of training steps (S) for their weights to converge. In each step, a random subset of training data are predicted by the network with its current weights. The prediction errors made on these examples are used by the back-propagation algorithm to update the weights. These random subsets of protein-ligand complexes are known as mini-batches and their sizes are set to $b = 50$ data points in our experiments. For a boosted ensemble model with L neural networks, the training time is proportional to the number of steps S, mini-batch size b, and the number of networks L as $T \propto SbL$. Using transfer learning, we could cut the number of training steps to few thousand ($s < S$) for networks whose initial weights are transferred from their predecessors ($W_l^i = W_{l-1}^o$). If we initialize weights randomly once after every t networks, we could have L/t networks initialized randomly and as many as $L - L/t$ nets initialized with optimal or suboptimal weights. As a

result, the training time T^* of a boosting model with weight transfer is defined by the equation $T^* \propto SbL/t + sb(L - L/t)$. Since the number of training steps (s) of optimally or sub-optimally initialized networks is a fraction f (typically 10 to 15%) of those initialized randomly, we could substitute fS for s in the equation defining T^*. After this substitution ($s = fS$), we could easily calculate the speedup from transfer learning:

$$\textbf{speedup} = \frac{T^*}{T} = \frac{t}{1 + f(t-1)}.$$

For a boosted neural network without weight transfer (i.e., when $t = 1$), we gain no speedup. The speedup is also one when the number of training steps is equal for both types of initializing approaches (i.e., $s = S$ and therefore $f = 1$). On the other hand, with setting $t = 30$ and $f = 0.2$, we get a speedup of more than 4x. For a regular boosted neural network that takes 8 hours to train, we could train it in less than 2 hours with transfer learning without affecting accuracy.

In our experiments, we attempt to strike a balance between model diversity and speed of training via transfer learning. The networks whose weights are initialized randomly tend to be less correlated amongst each other (i.e., more diverse) than those that share their initial weights. However, the diverse ensemble model is slower to train than the latter. Therefore, our BsN-Score and BgN-Score SFs are composed of 300 networks, 20% of which with weights randomly initialized and the other 80% started with optimized weights. Networks benefiting from transfer learning must use the same input descriptors (p_l) that the optimal-weight-donating networks utilize $p_l = p_{l-1}$. Otherwise, the transferred initial weights would be effectively random and the training process would be slower.

4.4.6 Neural networks and other machine-learning scoring functions

In order to investigate the effectiveness of ensemble NN SFs in comparison to traditional NN models and ensemble decision-tree models, as well as linear regression, we trained and tested BgN-Score, BsN-Score, a single deep neural network SF referred to as *DNN-Score*, RF, BRT, XGB, SVM, *k*NN, MARS, and MLR SFs on training and test datasets sampled from PDBbind complexes

according the procedures described in Section 2.1. Our neural network and other machine-learning SFs are trained and tested on complexes characterized by the multi-perspective descriptors derived using the proposed DDB platform from 4 and 16 (in case of PDBbind 2007 and PDBbind 2014, respectively) different sources as described in Chapter 3. The parameters of these SFs were tuned in a consistent manner to optimize the mean-squared prediction errors on validation complexes sampled without replacement from the training set and independent of the test sets. Out-of-bag instances were used as validation complexes for BgN-Score and RF, while a ten-fold cross-validation was conducted for BsN-Score, DNN-Score, and other ML SFs. Out-of-bag (OOB) refers to complexes that are not sampled from the training set when bootstrap sets are drawn to fit individual NNs in BgN-Score models or decision trees in RF—on average, about 34% of the training set complexes are left out (or "out-of-bag") when bootstrap sets are drawn. The parameters that are tuned and their optimized values are as follows:

1. L: the number of base learners (neural networks in ensemble NN SFs) was set to 300.

2. $|p|$: the size of the feature subset p randomly selected from the overall set of features P while constructing each neural network in ensemble NN SFs. This was set to $\lceil 33\% \rceil$ of the total number of feature for ensemble NN SFs. The number of input neurons for DNN-Score is set to the full set of input descriptors since this model does not sample the input features. All NN SFs have one output neuron per network that produces the binding affinity score.

3. $H+1$: the number of hidden-layer neurons in NN SFs was set to 150%, 100% and 50% of the number of neurons in the input layer.

4. The activation function for hidden layers is the rectified linear unit [110].

5. ν: the shrinkage parameter for BsN-Score models was set to 0.001.

6. The initial weights of each layer were randomly sampled from a normal distribution with zero mean and variance of $1/(n_i+n_o)$; where n_i and n_o are the number of input and output neurons for the layer being initialized, respectively [111].

7. Adaptive Moment Estimation (also known as Adam) variant of stochastic gradient descent algorithm was used to update the weights during training [112]. We use TensorFlow's implementation for Adam optimization algorithm [43].

8. A total of 50000 training batches (parameter optimization steps) and a batch size of 50 random complexes were used to optimize the weights for each network in the ensemble and single NN SFs. The training process of a network is terminated earlier if the moving-average of validation loss (RMSE) does not decrease for 1000 successive steps. Twenty-percent of the training data of each network is held aside before the training starts as an out-of-sample validation set to enable early stopping.

9. The number of regression trees for Random Forest model was set to 3000.

10. During fitting each tree in the ensemble, a random subset of 12 are considered when expanding each node in the tree. A random subset is sampled for each node from the 36 RF-Score's geometric descriptors.

The parameter optimal values of the other ML SFs trained on PDBbind 2007 are listed in Table 4.1. We distinguish them from each other using the notation *ML model::tools used to calculate features*. For instance, BsN-Score::XA implies that the SF is a boosted ensemble neural networks model that is trained and tested on complex sets described by *XA* features. For brevity, for each of DNN-Score, BgN-Score, BsN-Score, RF, BRT, SVM, *k*NN, MARS, and MLR models, we report results only for the feature combination (out of the fifteen possible) that yields the best performance on the validation complexes sampled without replacement from the training data and independent of the test set.

Table 4.1: Optimal parameter values for MARS, *k*NN, SVM, RF, and BRT models

Model	Parameter	Feature set														
		X	A	R	G	XA	XR	XG	AR	AG	RG	XAR	XAG	XRG	ARG	XARG
MARS	*Degree*	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	Penalty	2	6	5	6	7	2	6	7	6	5	6	7	6	5	6
*k*NN	*k*	15	13	14	16	9	19	17	19	18	17	18	19	17	18	19
	q	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
SVM	*C*	2	2	1	1	1	4	1	2	2	1	1	2	2	1	2
	ε	0.5	0	0.250	0.250	0.125	0.125	0.250	0.250	0.250	0.125	0.250	0.125	0.125	0.125	0.250
	σ	1	0.25	0.125	0.250	0.250	0.031	0.031	0.031	0.125	0.031	0.125	0.031	0.031	0.031	0.031
RF	*mtry*	3	18	8	7	31	5	8	10	16	17	14	20	21	25	35
BRT	*Interaction depth*	15	17	18	16	19	15	18	19	17	16	16	20	18	17	20
	Number of trees	1114	1523	1573	1208	1371	2113	1610	2950	2181	2303	2213	2590	2854	2921	2859

4.5 Results and discussion

4.5.1 Evaluation of scoring functions in binding affinity prediction and ligand ranking

Scoring power of SFs quantifies their ability to accurately predict protein-ligand binding affinity or reproduce it for complexes with known experimental BA data. The similarity between the predicted and measured BAs are calculated using Pearson's (R_p) and Spearman's (R_s) correlation coefficients, the standard deviation (SD) of errors, and the root-mean square-error (RMSE). Pearson's correlation coefficient measures the linear relationship between two variables as follows:

$$R_p = \frac{\sum_{i=1}^{N}\left[\left(\hat{y}_i-\bar{\hat{y}}\right)\left(y_i-\bar{y}\right)\right]}{\sqrt{\sum_{i=1}^{N}\left(\hat{y}_i-\bar{\hat{y}}\right)^2\sum_{i=1}^{N}\left(y_i-\bar{y}\right)^2}},$$

where N is the number of complexes and $\hat{y}_i$ and y_i are the predicted and measured binding affinities of the i-th complex, respectively. The average values of the predicted and experimentally measured affinities for all complexes are $\bar{\hat{y}}$ and $\bar{y}$, respectively. Spearman's correlation coefficient is used to evaluate the correlation between the predicted and measured BAs in terms of their ranks and it is defined as follows:

$$R_s = 1-\frac{6\sum_{i=1}^{N}d_i^2}{N\left(N^2-1\right)},$$

where d_i is the difference in ranks of the predicted and measured affinities of the i-th complex.

The SF that achieves the highest correlation coefficient (maximum is one) for some dataset is considered more accurate than its counterparts that realize smaller R_p and/or R_s values (minimum

is negative one). Another measure of scoring power we report here is the standard deviation (SD) of errors between predicted and measured BAs (in $-\log K_i$ or $-\log K_d$ units). To calculate this statistic for a given SF, a linear model that correlates predicted scores $\hat{Y}$ to the measured ones Y is first evaluated: $Y = \beta_0 + \beta_1 \hat{Y}$, where β_0 and β_1 are the intercept and the slope of the model, respectively. The SD statistic can then be computed as follows [98]:

$$\mathrm{SD} = \sqrt{\frac{\sum_{i=1}^{N} \left(y_i - \left(\beta_0 + \beta_1 \hat{y}_i \right) \right)^2}{N-2}}.$$

The root-mean square-error (RMSE) of the predicted scores is calculated as:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} \left(y_i - \hat{y}_i \right)^2}{N}}.$$

SFs that yield smaller SD and RMSE values usually realize higher R_p and R_s values, and therefore have higher scoring power than models with large SD and RMSE statistics.

Calculating the ranking power for an SF on the core test set is straightforward due to its construction (see Section 2.1.1 for more information about this set). First, each protein-ligand complex is scored. Then, for each protein cluster (i.e., the three protein-ligand complexes associated with a common protein), complexes are ordered according to their predicted binding affinity scores. Any given cluster is considered correctly ranked if its predicted-affinity-based order matches its corresponding measured-affinity-based order. We denote this order by "1-2-3" which implies that the strongest-binding ligand is ranked as the first candidate drug, the second strongest binding one is ranked second by the SF, and the weakest binder is ranked last—the number in the ordering corresponds to the true measured rank and the position in the ordering corresponds to the predicted rank. The percentage of clusters with "1-2-3" ordering is referred to as the *1-2-3 ranking rate*, denoted by $\mathrm{R}_{\text{1-2-3}}$, and is used as a measure of the *ranking power* of a given scoring function as in [17]. In order to more fully capture ranking accuracy, we also report rates for other rankings and use similar notation for them. For example, "1-3-2" implies that the actual second- and third-ranked ligands are incorrectly ranked while the first one is correctly identified as the strongest binder among the three candidates—the percentage of such clusters with such an ordering is referred to as the *1-3-2*

ranking rate and denoted by $R_{1\text{-}3\text{-}2}$. We also report the success rates of different SFs in identifying the strongest binder for each cluster (i.e., their *1-X-X ranking rates*, $R_{1\text{-}X\text{-}X}$). We also use the Spearman correlation coefficient (denoted by R_s) to measure ranking accuracy on test samples other than the core test set.

4.5.2 Ensemble neural networks vs. other approaches on a diverse test set

In this section, we examine the binding affinity prediction performance of several machine-learning and conventional scoring functions on protein-ligand complexes obtained from two versions of PDBbind. In the first, we evaluate the performance of 9 machine-learning scoring functions (including neural network models) and 17 conventional approaches on the 2007 release of PDBBind. This set of experiments was conducted prior to the development of our multi-perspective descriptor extraction platform DDB. Therefore, we only use a subset of 4 descriptor types based on X-Score, RF-Score, Affi-Score, and GOLD type features. In the second set of experiments, we train and evaluate scoring functions on complexes from PDBbind 2014 characterized by over 2700 descriptors extracted from 16 different sources (perspectives) in DDB.

4.5.2.1 Scoring performance on diverse protein families from PDBbind 2007

We trained three neural network SFs (DNN-Score, BgN-Score, and BsN-Score) and six other machine-learning scoring models (RF, BRT, SVM, *k*NN, MARS, MLR) on the primary (*Pr*) training set and evaluated their scoring performance on an independent test set of 195 diverse protein-ligand complexes from 65 different protein families (refer to Section 2.1 for more details about these sets). Both the training and test set complexes are obtained from the 2007 version of PDBbind.

Table 4.2 lists the scoring and ranking powers of these models and the same performance statistics of the seventeen conventional SFs evaluated on the same test set. We also report the scoring performances of NN and the other ML SFs on the training set *Pr* by using out-of-sample validation to show how close the predicted BAs are to the experimentally-measured ones in terms of

RMSE. Therefore, this statistic indicates whether SFs deemed accurate on training data will also be reliable scoring models on the test set *Cr*. This measure was not calculated for the conventional SFs (except X-Score) since we do not have access to the BA values of their training sets. All the scoring and ranking power metrics indicate that our proposed SFs and the ensemble decision-tree-based models are the most accurate in predicting the binding affinities and the correct ranks of the independent test set complexes and out-of-sample training data. BsN-Score outperforms the most accurate conventional SF, X-Score::HMScore, by at least 26% in terms of Pearson's correlation coefficients, which is 0.816 and 0.644 for both SFs, respectively. BgN-Score also achieves excellent performance of 0.808 which is about 25% improvement over X-Score::HMScore.

As for the ranking power, we notice that all SFs are able to identify the top binder (see 1-X-X ranking rate) in a majority of the clusters. However, ordering the three ligands correctly within a cluster (1-2-3 ranking rate) is much more challenging for scoring models. In fact, less than half of the SFs are capable of correctly ranking 50% or more of the protein-family clusters. Ensemble learning based scoring models (i.e. BsN-Score, BgN-Score, BRT, and RF) top the list by achieving 1-2-3 ranking rates of 60.0% (39 clusters) for BgN-Score and upto 64.6% (42 clusters) achieved by BsN-Score. SVM model fitted to XAR features comes in fifth place by identifying the correct order of 37 clusters. BsN-Score, BgN-Score, BRT, RF, and SVM also yield the highest R_s and R_p values on the core set and lowest RMSE values on the training data compared to the other models.

There are two main reasons for the superior performance of ensemble SFs. First, our proposed multi-perspective interaction modeling result in more numerous and varied features that more fully characterize protein-ligand interactions. Thus we find that BsN-Score, BgN-Score, BRT, and RF SFs employing all four features types considered (X, A, R, and G features) are more accurate than the same SFs employing fewer features. Second, the learning model of ensemble SFs is nonlinear and flexible and can exploit a large number of features while being resilient to overfitting. Thus we find that DNN-Score::X (for which $R_p = 0.675$) is more accurate compared to the versions of DNN-Score employing one of A, R, or G features only as well as DNN-Score::XARG (for which $R_p = 0.517$) because single neural network models overfit the training complexes when characterized by

Table 4.2: Comparison of the scoring and ranking powers of the proposed and 17 conventional scoring functions on the diverse protein-ligand complexes of PDBbind 2007 core (*Cr*) test set.

Scoring Functions	N[1]	Scoring Power					Ranking Power (in %)		
		R_p[2]	R_s[3]	SD[4]	$\text{RMSE}_{\text{test}}$[5]	$\text{RMSE}_{\text{train}}$[6]	$\text{R}_{\text{1-2-3}}$	$\text{R}_{\text{1-3-2}}$	$\text{R}_{\text{1-X-X}}$
BsN-Score::XARG	195	0.816	0.800	1.38	1.386	1.366	64.6	13.8	78.4
BRT::XARG	195	0.810	0.798	1.40	1.485	1.425	63.1	10.8	69.3
BgN-Score::XARG	195	0.808	0.799	1.40	1.449	1.403	60.0	15.4	75.4
RF::XARG	195	0.804	0.790	1.43	1.498	1.442	63.8	6.2	70.0
SVM::XAR	195	0.773	0.793	1.51	1.490	1.580	56.9	21.9	78.8
*k*NN::XA	195	0.740	0.731	1.61	1.52 0	1.600	44.6	18.5	63.1
MARS::XAR	195	0.710	0.740	1.68	1.66 0	1.680	44.6	15.4	60.0
MLR::XA	195	0.689	0.730	1.73	1.700	1.790	41.5	21.5	63.0
DNN-Score::X	195	0.675	0.685	1.76	1.760	1.704	33.8	29.2	63.0
X-Score::HMScore	195	0.644	0.705	1.83	1.865	1.730	54.7	15.6	70.3
$\text{DrugScore}^{\text{CSD}}$	195	0.569	0.627	1.96	–	–	51.6	21.9	73.4
SYBYL::ChemScore	195	0.555	0.585	1.98	–	–	46.9	25.0	71.9
DS::PLP1	195	0.545	0.588	2.00	–	–	54.7	15.6	70.3
GOLD::ASP	195	0.534	0.577	2.02	–	–	43.8	21.9	65.6
AffiScore	195	0.521	0.505	2.06	–	–	38.5	12.3	50.8
SYBYL::G-Score	195	0.492	0.536	2.08	–	–	46.9	25.0	71.9
DS::LUDI3	195	0.487	0.478	2.09	–	–	45.3	21.9	67.2
DS::LigScore2	193	0.464	0.507	2.12	–	–	35.9	25.0	60.9
GlidScore-XP	178	0.457	0.435	2.14	–	–	34.4	29.7	64.1
DS::PMF	193	0.445	0.448	2.14	–	–	40.6	18.8	59.4
GOLD::ChemScore	178	0.441	0.452	2.15	–	–	35.9	21.9	57.8
SYBYL::D-Score	195	0.392	0.447	2.19	–	–	46.9	20.3	67.2
DS::Jain	189	0.316	0.346	2.24	–	–	42.2	31.3	73.4
GOLD::GoldScore	169	0.295	0.322	2.29	–	–	23.4	26.6	50.0
SYBYL::PMF-Score	190	0.268	0.273	2.29	–	–	39.1	21.9	60.9
SYBYL::F-Score	185	0.216	0.243	2.35	–	–	29.7	25.0	54.7

[1] Number of complexes in *Cr* with positive (favorable) binding scores using this SF [17].

[2] R_p is the Pearson correlation coefficient between predicted and measured BA values of complexes in *Cr*.

[3] R_s is the Spearman correlation coefficient between predicted and measured BA values of complexes in *Cr*.

[4] SD is the standard deviation of errors between predicted and measured BA values of complexes in *Cr* based on Equation 3 in [98].

[5] RMSE is the root-mean-square of errors between predicted and measured BA values of the test complexes in *Cr*. Test RMSE is not available for conventional SFs except for X-Score::HMScore that we have re-constructed.

[6] RMSE is the root-mean-square of errors between predicted and measured BA values of out-of-sample complexes in the training set *Pr*. Training RMSE is not available for conventional SFs except for X-Score::HMScore that we have re-constructed.

a large number of features. We attempted to decrease the effect of overfitting by conducting feature reduction using PCA which helped increase the performance of DNN-Score::XARG to $R_p = 0.667$. However, the predictions of DNN-Score::XARG are still substantially less accurate than those of BsN-Score::XARG and BgN-Score::XARG even though the first 10 principle components we used to calculate the 10 new features explain more than 99.7% of the total variance in the raw XARG features. Further, the significance of the ensemble modeling approach can be gauged from the fact that even with a single type of feature, BsN-Score::A and BgN-Score::A yield accuracies of R_p = 0.780 and 0.775, respectively, which are within $\sim 4\%$ of the accuracies of BsN-Score::XARG and BgN-Score::XARG.

Table 4.2 also shows that the ensemble NNs SFs BsN-Score::XARG and BgN-Score::XARG are more accurate than their counterparts of decision-trees-based ensemble SFs, BRT::XARG and RF::XARG, though the latter comes a close second and forth (R_p = 0.816 and $R_{1\text{-}2\text{-}3}$ = 64.6% achieved by BsN-Score vs. 0.810 and 63.8% realized by BRT and RF, receptively). Note that the boosting model BRT corresponds to BsN-Score and RF is the bagging analog of BgN-Score. We believe the difference in performance between neural-networks and decision-trees based ensemble models, although small, is mainly attributable to the way the base learners of these ensemble models approximate the unknown function. Decision trees model the unknown function by partitioning training data into smaller subsets from which a prediction is calculated. Such a procedure creates a series of non-overlapping regions with axis-parallel decision boundaries. The numerical values associated with each region are typically the average BA of the training data subset belonging to that partition which could be significantly different from the neighboring regions. This could create a rough and abrupt approximation of the unknown function. On the other hand, NNs with hidden units can closely and smoothly model any nonlinear continuous function. In addition, hidden neurons may create new important features that would otherwise be impossible to extract directly from protein-ligand complexes. These two factors minimize the bias error of NN models, but may lead to increased variance or instability as in the case of single neural network SFs. The proposed boosting and bagging ensemble learning approaches greatly reduce the variance error.

Such simultaneous reduction in bias and variance errors makes the ensemble NN SFs the most accurate BA predictors compared to the other scoring functions listed in Table 4.2.

4.5.2.2 Scoring performance on diverse protein families from PDBbind 2014

In the second set of experiments, we use the 2014 release of PDBbind training and test datasets described in Sections 2.1 for building and evaluating three neural network SFs (DNN-Score, BgN-Score, and BsN-Score), the Random Forest SF RF-Score, and the empirical scoring function X-Score. The neural network scoring functions are trained and tested on complexes characterized by 2714 multi-perspective descriptors using the proposed descriptor platform DDB. The Random Forest scoring function RF-Score uses 36 geometrical features of atomic pair-wise counts while the empirical model X-Score employs 6 physiochemical descriptors as energy terms.

After training them, the five SFs are then tested on the out-of-sample (OOS) core test set (*Cr*) of familiar (or known) targets, OOS cross-validation (*CV*) sets of partially familiar targets, and OOS leave-clusters-out (*LCO*) sets of novel targets. In these three training-test sampling scenarios, the protein-ligand complex as a whole is always out-of-sample (OOS) and is never present in the training data of the five SFs. A scoring function is considered familiar with a protein target if it was part of its training data even though the protein is bound to a ligand different from the one it binds to in the test set. Figure 4.3 shows the scoring powers of the five SFs in terms of Pearson's and Spearman's correlation coeffients, and normalized root-mean-square-error (NRMSE[1]). All three scoring power metrics indicate that our proposed SFs are the most accurate in predicting the binding affinities of the OOS complexes in the three training-test sampling scenarios. The average correlation coefficients of ten versions of the five SFs on the familiar targets test set *Cr* reveal that BsN-Score and BgN-Score outperform the most accurate empirical SF, X-Score, by at least 34% in terms of Pearson's correlation coefficients R_p, which is 0.844, 0.840, and 0.627 for the three SFs, respectively. BsN-Score and BgN-Score also achieve better scoring accuracy than RF-Score

[1] $NRMSE = RMSE/(\mathrm{BA}_{\max} - \mathrm{BA}_{\min})$, where $\mathrm{BA}_{\max} = 11.96$ and $\mathrm{BA}_{\min} = 2.00$ are the largest and smallest BA values of the training complexes, respectively.

whose predictions have an average correlations of 0.725 and 0.729 for *Cr* and *CV* tests. The single (i.e., non-ensemble) deep neural network SF, DNN-Score, performs better than RF-Score (0.802 for *Cr* & 0.773 for *CV*) but second to the ensemble DNN SFs BsN-Score and BgN-Score. Similar accuracy trends for the five SFs can also be observed from the other two performance metrics R_s and *NRMSE*.

The scoring accuracy of the five SFs drop substantially when they are tested on protein targets that share no sequence similarity with their training proteins as can be observed from the Leave-Clusters-Out (*LCO*) evaluation scenario. However, BsN-Score and BgN-Score can still re-produce experimental affinity measurements (*pKd* and *pKi*) more accurately than the other three SFs. With correlation coefficient of 0.637, BsN-Score is at least 19% more accurate than DNN-Score, RF-Score, and X-Score.

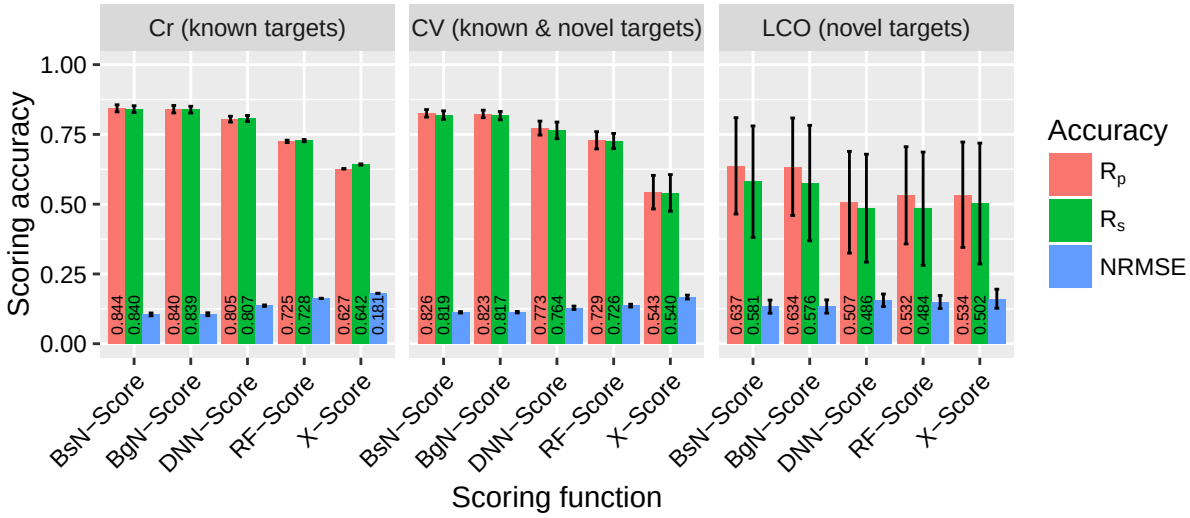


Figure 4.3: The scoring accuracy of the proposed and conventional scoring functions when evaluated on test complexes with proteins that are either fully represented (*Cr*), partially represented (*CV*), or not represented (*LCO*) in the SFs' training data. Training and test protein-ligand complexes are from the refined set of PDBbind 2014.

The ranking performance of the scoring functions are evaluated on the core test set of PDBbind 2014 and depicted in Figure 4.4. The results in the figure are in line with those summarized in Table 4.2 for the 2007 core test set. We here observe that all five SFs are also able to identify the actual top (1-X-X ranking rate) and bottom (X-X-3) binders in more than two thirds of the protein-

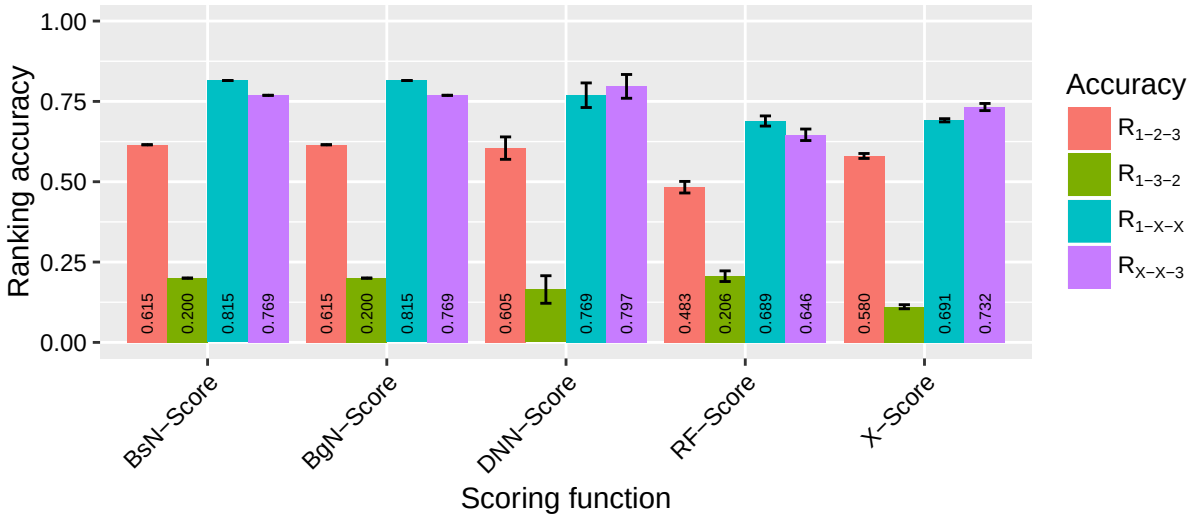


Figure 4.4: The ranking accuracy of the proposed and conventional scoring functions when evaluated on the core (*Cr*) test set complexes of PDBbind 2014.

family clusters. Ordering the three ligands correctly within a cluster (1-2-3 ranking rate) is also much more challenging for scoring models. Neural network SFs top the list again by achieving 1-2-3 ranking rates of 61.5% and 60.0% (40 and 39 clusters) as opposed to 58.0% and 48.3% (38 and 32 clusters) achieved by X-Score and RF-Score, respectively.

The main factors behind the superior performance of BsN-Score and BgN-Score are the learning algorithm and multi-perspective descriptors we use to characterize protein-ligand complexes. Deep neural networks are excellent in modeling very complex functions due to the their non-linearity and capacity. Ensemble learning through boosting and bagging significantly decrease their variance error and overfitting. Second, our proposed multi-perspective interaction modeling result in more numerous and divers features that more fully characterize protein-ligand interactions. Thus we find that BsN-Score, BgN-Score, DNN-Score employing all 16 features types are more accurate than the scoring functions employing only one set of features such as RF-Score and X-Score.

We conducted another set of experiments on PDBbind 2014 complexes in which we compare the performance of the proposed ensemble neural network models to ensemble trees SFs. The neural network and tree based ensemble models were trained and evaluated using the same *Cr*, *CV*,

and *LCO* training-test sampling strategies whose complexes are also characterized by the same 16 descriptor sets. The only difference between the two types of SFs is the base learner they use to construct the ensemble (i.e., a neural network versus a decision tree). Our results indicate that the neural network SFs BsN-Score and BgN-Score are more accurate than their counterparts of decision-trees SFs based on XGBoost and Random Forests (RF). The boosted trees-based SF, we coin as BT-Score, comes a close second to BsN-Score and BgN-Score with $R_p = 0.830$ and $R_{1\text{-}2\text{-}3} = 60\%$ on the core test set *Cr*. Note that the boosting model BT-Score corresponds to BsN-Score and RF is the bagging analog of BgN-Score. The result on the 2014 version of PDBbind confirms our finding on the 2007 release that the performance gap between neural-networks and decision-trees based ensemble models is mainly due to their superior approximation of the unknown function that govern the molecular binding process.

4.5.3 Ensemble neural networks vs. other approaches on homogeneous test sets

4.5.3.1 Scoring performance on four protein families from PDBbind 2007

It has been observed that around 92% of existing drug targets are similar to proteins already present in the Protein Data Bank, which is the primary source of our training and validation complexes [113]. Based on this finding and the similar overlap relationship between training and test set proteins in the core test set experiment in Section 4.5.2.1, we believe that the scoring performance of the SFs listed in Table 4.2 should be expected in typical molecular docking and virtual screening campaigns. For each protein family in PDBbind's core test set, there is at least one protein family in the primary training set of our proposed NN and the six other ML SFs, but the two sets share no protein-ligand complexes when these pairs of compounds are considered as whole biological units. We describe here a more stringent experiment to assess the generalization of the NN, RF, BRT, SVM, *k*NN, MARS and MLR SFs when they are applied to score ligands for novel drug targets. In this experiment, we evaluate the BA predictive accuracy of the NN SFs on four protein families not present in their training set. These protein families are the most frequent in

the 2007 version of PDBbind which includes 112 HIV protease, 73 trypsin, 44 carbonic anhydrase, and 38 thrombin complexes. A test set for each of these protein families was constructed by sampling all complexes formed by that protein from the training (Pr) and the test (Cr) sets. The training complexes corresponding to each of these four test sets are the remaining protein-ligand pairs in Pr. For each protein family, we fitted the proposed models to the corresponding independent training complexes and validated them on the test set complexes that are formed between that type of protein and a unique set of co-crystallized ligands. The prediction accuracy of our proposed models and the top four conventional scoring functions on complexes formed by the four protein types are shown in Table 4.3.

Examining the upper portion of the table for the four families where the test and training sets are disjoint for the NN and the other 6 ML SFs, we notice that the predictive accuracy of all SFs varies from poor to good depending on the protein family. Predictions of all SFs have mediocre correlation with the experimental binding affinities for the ligands that bind to HIV protease proteins. The highest Pearson's correlation value between predicted and true BAs is 0.465, which is the case for the ML scoring function MARS::X. Improper characterization of enthalpic and entropic forces for HIV protease complexes could be the main reason for these erroneous predictions [17]. The significant conformational changes observed during binding as well as the lack of similar proteins in the training set could also result in such inaccurate estimations for BA. The scoring accuracy on the other three protein families is substantially better. The binding affinities for ligands bound to trypsin were predicted with an accuracy of at least $R_p = 0.842$; and again with a scoring function based on MARS. Discovery Studio's empirical SF PLP2 shows the highest accuracy on the carbonic anhydrase dataset with a linear correlation value of 0.800. The most accurate models on the thrombin test set are the NN and several other ML based SFs with R_p values of about 0.700 and better, followed by the conventional scoring functions.

It can be observed that the SF based on a single NN, DNN-Score, performs relatively poorly overall, except in one case. In some of these test sets, a few conventional SFs perform better than the ensemble NN SFs. This behavior can be attributed to the possibility of some overlap between

Table 4.3: Comparison of the scoring and ranking accuracies of scoring functions on four protein-family-specific tests sets derived from the refined set of PDBbind 2007.

HIV protease (N = 112)						Trypsin (N = 73)					
Scoring function	R_p[1]	R_s[1]	SD[1]	RMSE[1]	D[2]	Scoring function	R_p[1]	R_s[1]	SD[1]	RMSE[1]	D[2]
MARS::X	0.465	0.467	1.384	1.405	Y	MARS::XARG	0.842	0.825	0.910	1.269	Y
MLR::G	0.463	0.470	1.452	1.486	Y	*k*NN::XA	0.833	0.782	0.936	0.970	Y
BRT::A	0.403	0.281	1.500	1.621	Y	SYBYL::ChemScore	0.829	0.773	0.950	–	U
BgN-Score::A	0.388	0.340	1.511	1.817	Y	SVM::XA	0.826	0.821	0.953	1.029	Y
*k*NN::XG	0.386	0.306	1.512	1.641	Y	DS::Ludi2	0.823	0.791	0.960	–	U
SVM::G	0.350	0.313	1.535	1.706	Y	X-Score::HSScore	0.817	0.824	0.970	1.401	N
X-Score::HPScore	0.341	0.339	1.540	1.509	N	DS::PLP2	0.797	0.774	1.020	–	U
BsN-Score::XARG	0.290	0.230	1.560	1.705	Y	BsN-Score::AR	0.793	0.727	1.080	1.119	Y
RF::XARG	0.289	0.219	1.519	1.719	Y	MLR::XG	0.783	0.786	1.050	1.370	Y
SYBYL::ChemScore	0.255	0.228	1.580	–	U	BgN-Score::XAR	0.778	0.728	1.060	1.070	Y
DrugScore::PairSurf	0.225	0.170	1.590	–	U	RF::XAR	0.774	0.753	1.070	1.133	Y
DS::PMF04	0.183	0.200	1.610	–	U	BRT::XA	0.771	0.748	1.076	1.128	Y
DNN-Score::X	0.039	0.048	1.640	2.255	Y	DNN-Score::X	0.735	0.672	1.140	1.209	Y
BRT::A	0.989	0.996	0.243	0.263	N	BRT::XAG	0.969	0.961	0.420	0.425	N
BsN-Score::XARG	0.979	0.980	0.405	0.410	N	BsN-Score::XA	0.968	0.961	0.424	0.429	N
BgN-Score::XA	0.968	0.977	0.445	0.598	N	SVM::A	0.965	0.979	0.445	0.454	N
RF::XARG	0.965	0.975	0.440	0.588	N	BgN-Score::XAG	0.950	0.966	0.439	0.475	N
SVM::A	0.964	0.978	0.438	0.440	N	*k*NN::XA	0.936	0.926	0.596	0.610	N
*k*NN::XA	0.842	0.797	0.883	0.905	N	RF::XAG	0.934	0.928	0.600	0.657	N
DNN-Score::X	0.748	0.716	1.080	1.085	N	MARS::XAR	0.861	0.828	0.861	0.953	N
MARS::XR	0.619	0.587	1.287	1.292	N	MLR::XARG	0.806	0.795	1.001	1.136	N
MLR::RG	0.522	0.554	1.398	1.393	N	DNN-Score::X	0.829	0.789	0.940	0.957	N

Carbonic anhydrase (N = 44)						Thrombin (N = 38)					
Scoring function	R_p[1]	R_s[1]	SD[1]	RMSE[1]	D[2]	Scoring function	R_p[1]	R_s[1]	SD[1]	RMSE[1]	D[2]
DS::PLP2	0.800	0.772	0.840	–	U	DNN-Score::X	0.756	0.704	1.380	1.433	Y
MLR::XR	0.729	0.634	0.954	2.969	Y	BRT::XG	0.738	0.722	1.429	1.432	Y
BRT::A	0.721	0.631	0.965	3.508	Y	BsN-Score::X	0.730	0.710	1.449	1.586	Y
MARS::G	0.706	0.651	0.986	2.777	Y	BgN-Score::XAG	0.723	0.726	1.465	1.562	Y
SYBYL::G-Score	0.706	0.646	0.990	–	U	*k*NN::XG	0.713	0.722	1.487	1.586	Y
SYBYL::ChemScore	0.699	0.631	1.000	–	U	MARS::X	0.704	0.644	1.505	1.588	Y
SVM::G	0.695	0.598	1.001	2.919	Y	RF::XARG	0.697	0.693	1.520	1.674	Y
BsN-Score::X	0.674	0.434	1.030	3.418	Y	MLR::XG	0.674	0.624	1.566	1.681	Y
BgN-Score::XA	0.669	0.527	1.030	3.541	Y	DS::PLP1	0.667	0.672	1.580	–	U
*k*NN::XAG	0.661	0.616	1.045	3.793	Y	SYBYL::G-Score	0.667	0.626	1.580	–	U
DNN-Score::X	0.631	0.451	1.080	3.561	Y	X-Score::HSScore	0.666	0.586	1.580	1.737	N
SYBYL::PMF-Score	0.627	0.618	1.090	–	U	DrugScore::Pair	0.651	0.622	1.611	–	U
RF::XARG	0.601	0.374	1.112	3.393	Y	SVM::X	0.647	0.575	1.615	1.607	Y
BRT::A	0.988	0.979	0.214	0.218	N	BRT::XG	0.975	0.989	0.471	0.548	N
BsN-Score::XARG	0.948	0.921	0.441	1.004	N	BsN-Score::XA	0.955	0.976	0.637	0.747	N
SVM::A	0.946	0.974	0.451	0.455	N	SVM::XA	0.927	0.948	0.793	0.820	N
RF::XARG	0.910	0.860	0.57	1.140	N	RF::XARG	0.910	0.934	0.860	1.125	N
*k*NN::RG	0.896	0.827	0.619	0.613	N	BgN-Score::XA	0.901	0.935	0.920	1.113	N
BgN-Score::XARG	0.884	0.766	0.650	1.320	N	*k*NN::XA	0.859	0.873	1.084	1.175	N
MARS::XR	0.785	0.694	0.862	0.974	N	DNN-Score::X	0.761	0.756	1.371	1.374	N
MLR::AR	0.794	0.762	0.845	1.039	N	MARS::XR	0.746	0.691	1.412	1.449	N
DNN-Score::X	0.652	0.310	1.050	1.687	N	MLR::XRG	0.724	0.638	1.463	1.553	N

[1] R_p and R_s are the Pearson and Spearman correlation coefficients between predicted and measured BA values of complexes in this protein-family-specific test set, respectively. SD and RMSE are the standard deviation and the root-mean-square of errors between predicted and measured BA values of complexes in this protein-family-specific test set, respectively.

[2] This indicates whether the test set complexes are disjoint from (D = Y) or overlap with (D = N) the training set complexes for NN and the six other ML models. Any overlap between the training and test data of the conventional SFs is unknown (D = U) to us.

the training complexes of the conventional approaches and the four family-specific test sets. As discussed earlier, the protein families of training and test complexes for the NN and the six other ML models do not overlap and they are completely disjoint. When we retrain ensemble NN and the six other ML SFs on the original training set (*Pr*), which overlaps with the family-specific test sets, and assess their scoring power on the four homogeneous test sets, we notice that the predictions of the proposed SFs are near perfect as listed in the lower portion of Table 4.3.

The results listed in Tables 4.2 and 4.3 show the performance of the proposed and conventional SFs on target proteins when they are partially or fully encountered in their training sets, or completely novel for them. Therefore, we believe that these results are very useful in estimating the accuracy of our scoring models given the number of solved structures of the drug target with other ligands and the availability of their binding data.

We performed another experiment in which the ML SFs were calibrated with training sets containing some known binders to the protein target and assessed their scoring ability on an independent test set of complexes that all feature the protein target, but bound to different ligands. Since HIV protease complexes are the most abundant in our dataset and they have proved to be the most challenging in the previous experiment (see the upper portion of Table 4.3), we focus on them. Also, we only consider ML SFs in this and subsequent experiments since for the conventional 17 SFs we do not have access to their full training set BA values; in these cases, the MLR model, due its linear nature, can be considered representative of empirical SFs [63, 56].

The SFs based on RF, BRT, SVM, *k*NN, MARS, and MLR are trained using X, A, R, and XAR features on a training set that includes a randomly selected (without replacement) $x\%$ of the 112 complexes (more precisely, $\lfloor x \times 112/100 \rfloor$ complexes) from the HIV protease homogeneous test set[2]. The remaining complexes in the training set are the non-HIV-protease complexes in *Pr*, except that as many of them as the number of the added HIV complexes are randomly excluded

[2]This experiment was conducted before considering the G-type features and developing the Neural Network based models, therefore they are not included here. However, we expect the performance trends of BsN-Score and BgN-Score to be similar to that of BRT and RF SFs, respectively.

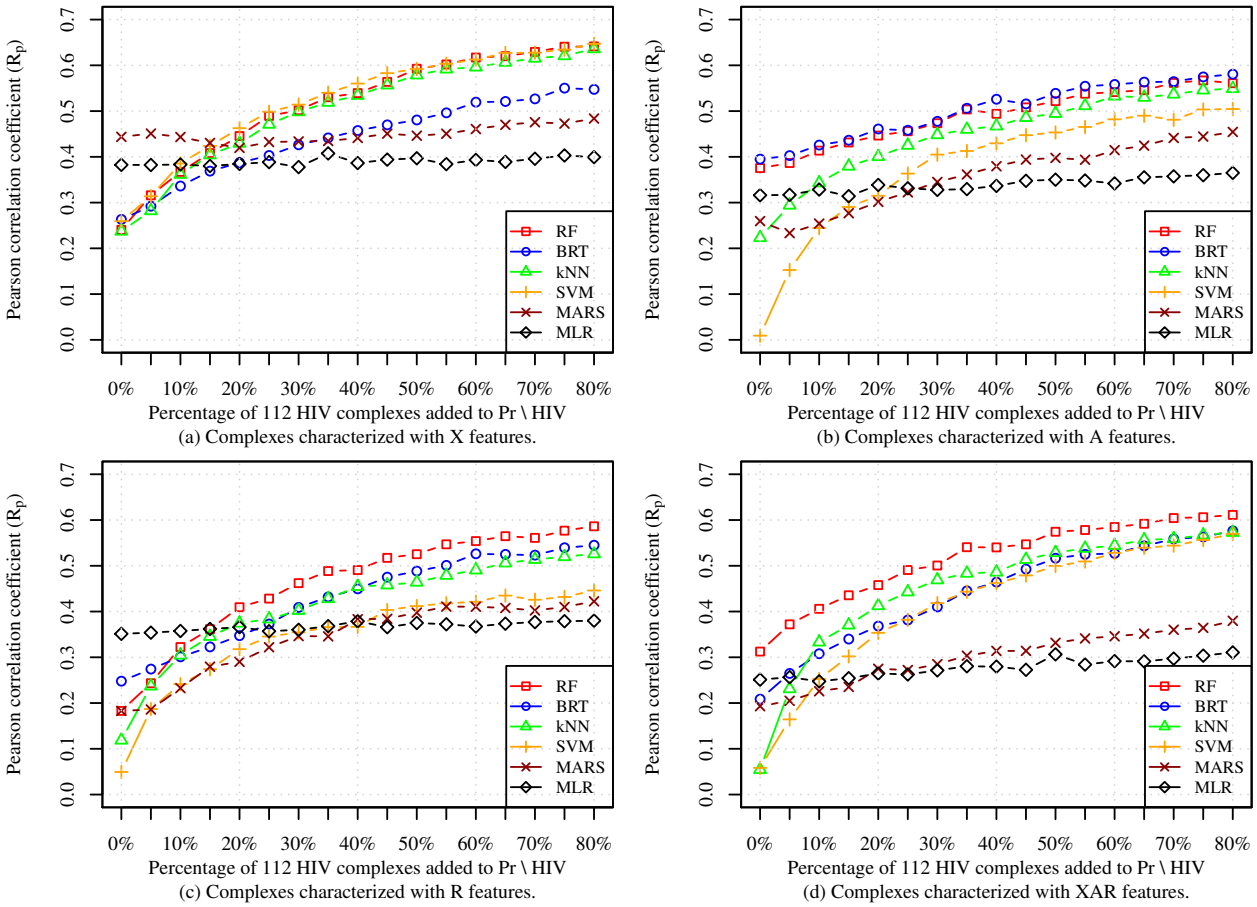


Figure 4.5: Dependence of scoring performance of machine-learning scoring functions on the number of HIV protease complexes in their training set. They are evaluated on out-of-sample HIV protease complexes from the refined set of PDBbind 2007.

from the training set. This means that the size of the training set is fixed and equals the number of non-HIV-protease complexes in *Pr*. Another different 20% of the 112 HIV protease complexes (more precisely, $\lceil 0.2 \times 112 \rceil = 23$ complexes) are randomly selected (without replacement) to form the test set. This process is repeated 1000 times to obtain a robust average R_p scoring measure for various values $(0, 5, 10, \ldots, 80)$ of x (i.e., the percentage of HIV protease complexes) for the six ML models—these are plotted in Figure 4.5.

The performance behavior of ML SFs seems similar across the four feature sets as shown in panels (a) through (d) of Figure 4.5. There are three other important observations that can be made from these four panels. First, when the training dataset includes none to very few HIV protease complexes (roughly up to 5–10% of the total), the simpler linear model MLR ties with or outper-

forms some of the more sophisticated nonlinear techniques such as SVM, *k*NN, BRT, MARS, and RF. This is somewhat similar to that seen earlier in the upper portion of Table 4.3. Second, the RF model clearly performs the best across the entire range of number of HIV complexes in training data characterized with XAR features. Other ML SFs such as *k*NN, SVM, and BRT also show substantial improvement in performance when trained on larger numbers of HIV complexes. Finally, MLR, due to its rigid linear nature that resembles that of empirical SFs, is not able to effectively exploit increasing numbers of HIV complexes in training data to improve its scoring accuracy in contrast to the other flexible models which show moderate (MARS) to significant improvement (*k*NN, RF, BRT, and SVM). For instance, for the XAR feature set (see Figure 4.5-d), the scoring accuracy of some ensemble models improves by more than 90% (RF) or 140% (BRT) when the number of HIV complexes increases from 0% to 80%, whereas the corresponding increase in performance for the linear model does not even exceed 11%. A similar trend can also be observed for the other three feature sets in the other panels.

The analysis of the effectiveness of different regression approaches on specific families of proteins has very practical benefits in virtual screening and drug discovery. Typically, the task in drug design is to conduct computational screening of large numbers of ligands against a certain protein family. The knowledge of the number of known binders to that protein present in the training set and the characteristic plots of several scoring models (similar to the ones in Figure 4.5) can aid in the selection of the most effective SF.

4.5.3.2 Scoring performance on four protein families from PDBbind 2014

The Leave-Clusters-Out (*LCO*) training-test sampling strategy of the 2014 PDBBind complexes was a stringent experiment to assess the generalization of the proposed and conventional scoring functions when they are applied to score ligands for novel drug targets. The difficulty of the prediction was reflected in the performance drop as can be observed in the novel targets panel of Figure 4.3. As explained in Section 2.1.3, the accuracy statistics reported on the right panel of Figure 4.3 are the average across several clusters with different protein families and types. In this

experiment, we again evaluate the BA predictive accuracy of the NN and conventional SFs on the largest four protein families or clusters. These protein families are the most frequent in PDBbind 2014 which include 262 HIV protease, 170 carbonic anhydrase (CAH), 98 trypsin, and 79 thrombin complexes. Similar to the experiment on PDBbind 2007 we described in the previous section, a test set for each of these protein families was also constructed by sampling all complexes formed by that protein from the 2014 PDBbind refined set. The training complexes corresponding to each of these four test sets are the remaining protein-ligand pairs. For each protein family, we fitted the proposed models to the corresponding independent training complexes and validated them on the test set complexes that are formed between that type of protein and a unique set of co-crystallized ligands. The prediction accuracy of our proposed models and the conventional scoring functions on complexes formed by the four protein types are shown in Figure 4.6.

The right panels of the figure depict the scoring performance when the training and test sets are disjoint (i.e., out-of-sample) for the five scoring functions. It can be observed that the predictive accuracy of all SFs varies from poor to good depending on the protein family. Predictions of all SFs have mediocre correlation with the experimental binding affinities for the ligands that bind to HIV protease and carbonic anhydrase proteins. The highest Pearson's correlation value between predicted and true BAs is 0.451 for HIV and 0.524 for CAH, which are obtained by the ensemble NN scoring function BsN-Score. Complexes formed by HIV protease in the 2007 version of PDBbind were also challenging to score accurately. However, many of the examined scoring functions in the previous section were able to correctly re-produce the binding affinity of the 44 carbonic anhydrase complexes of PDBbind 2007. The vast majority of the new complexes in PDBbind 2014 were not part of the 2007 release which are apparently harder to predict. The scoring accuracy on the other two protein families in PDBbind 2014 is substantially better. The binding affinities for ligands bound to trypsin were predicted with an accuracy of at least $R_p = 0.820$; and again with a scoring function based on BsN-Score. BgN-Score performs almost as well followed by RF-Score and X-Score whose accuracies are not far behind. BsN-Score and BgN-Score also show the highest accuracy on the thrombin dataset with a linear correlation value

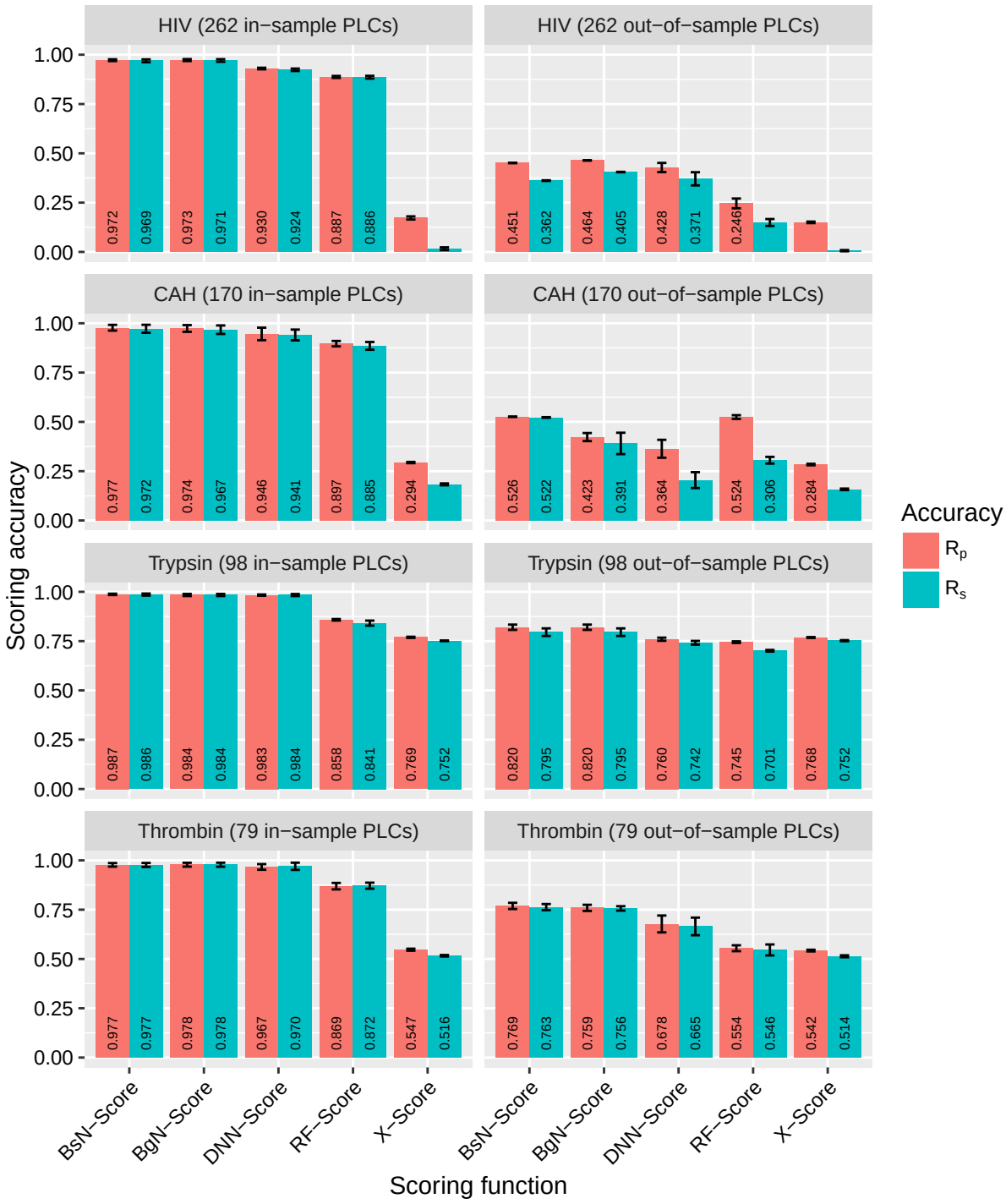


Figure 4.6: Comparison of the scoring and ranking accuracies of scoring functions on four protein-family-specific tests sets derived from the refined set of PDBbind 2014.

of 0.769. The correlation coefficients of RF-Score and X-Score at 0.554 and 0.542 on this novel target are substantially less than those obtained by the three neural-network SFs.

In addition to the out-of-sample evaluation of the five SFs, we also examined their in-sample accuracy to empirically test their capacity in exploiting their knowledge about known protein-ligand complexes and reproducing observed binding affinities. Unsurprisingly, the non-linear SFs, specially those based on NN, show near perfect predictions while X-Score perform very poorly on HIV and CAH complexes as depicted in the left panels of Figure 4.6. This implies that the relationship between the employed descriptors by X-Score and binding affinity is non-linear. In addition, our experiments show that the six X-Score descriptors do not appear to capture all the binding-related interactions between the protein and ligand molecules. We made this conclusion after we obtained mediocre performance from boosted NN model fitted to the six features used by X-Score. However, the performance of this boosted NN model was substantially better than X-Score in which a linear regression model is used.

The results shown in Figures 4.6 and 4.3 summarize the performance of the proposed and conventional SFs on target proteins when they are encountered in their training sets or completely novel for them. Therefore, we believe that these results are very useful in estimating the accuracy of our scoring models given the number of solved structures of the drug target with other ligands and the availability of their binding data.

We performed another experiment in which the ML SFs were calibrated with training sets containing some known binders to the protein target and assessed their scoring ability on an independent (OOS) test set of complexes that all feature the protein target, but bound to different ligands. Since HIV protease and carbonic anhydrase complexes are the most abundant in our dataset and they have proved to be the most challenging in the previous experiment (see the upper portion of Figure 4.3), we focus on them.

Due to the large number of models required to perform this experiment and the computational resources needed to train deep neural networks, the boosting neural network SF, BsN-Score, is replaced with its sister Boosted decision Tree Scoring function, BT-Score. The other two NN SFs, BgN-Score and DNN-Score, are not considered in this experiment due to the same reason. The use of BsN-Score's surrogate SF BT-Score will be performed for the experiments in the following

sections as well since they too involve building a large number of SFs to obtain reliable results and trends. BT-Score will be compared to our versions of the conventional ML SF RF-Score and empirical SF X-Score.

The SF BT-Score is fitted to the 2714 multi-perspective descriptors while RF-Score and X-Score are fitted to their original 36 geometrical and 6 physiochemical features, respectively. Before training an SF based on these models, half of the 260-HIV and 190-CAH PLCs are randomly sampled and held aside for testing while the remaining 130 and 85 complexes are considered for training, respectively. Then, the SFs are trained on 3000 complexes sampled randomly (without replacement) from the PDBbind refined set and $x\%$ of the 130 HIV (or CAH 85) training complexes. The overall size of the training set for any SF was fixed to 3000 complexes regardless of the number of added HIV or CAH complexes (by removing extra non-HIV or non-CAH PLCs from the original 3000 complexes). The scoring and ranking performance of the SFs are evaluated on the OOS 50% HIV (or CAH) test complexes. Then another 50% of the 260 HIV (or 190 CAH) complexes are randomly sampled to form the next training and test sets. This process is repeated 50 times to obtain a robust average scoring (R_p) and ranking (R_s) measures for various values of $x \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ (i.e., the fraction of 130 HIV and 85 CAH complexes) for the three models—these are plotted in Figure 4.7.

The overall scoring and ranking performance trends of the three SFs appear to be similar for both HIV and CAH complexes as shown in Figure 4.7. There are three important observations that can be made from these four panels. First, the XGboost SF, BT-Score, clearly performs the best across the entire range of number of HIV complexes and more than half of the range of CAH complexes in the training data. The Random Forest SF, RF-Score, also shows substantial improvement in performance when trained on larger numbers of HIV and CAH complexes. Second, when the training dataset includes less than half of the CAH complexes, the SF RF-Score slightly outperforms BT-Score in both scoring and ranking tests. Finally, X-Score, due to its rigid linear nature, is not able to effectively exploit increasing numbers of HIV and CAH complexes in training data to improve its scoring and ranking accuracy in contrast to the two other flexible models which show

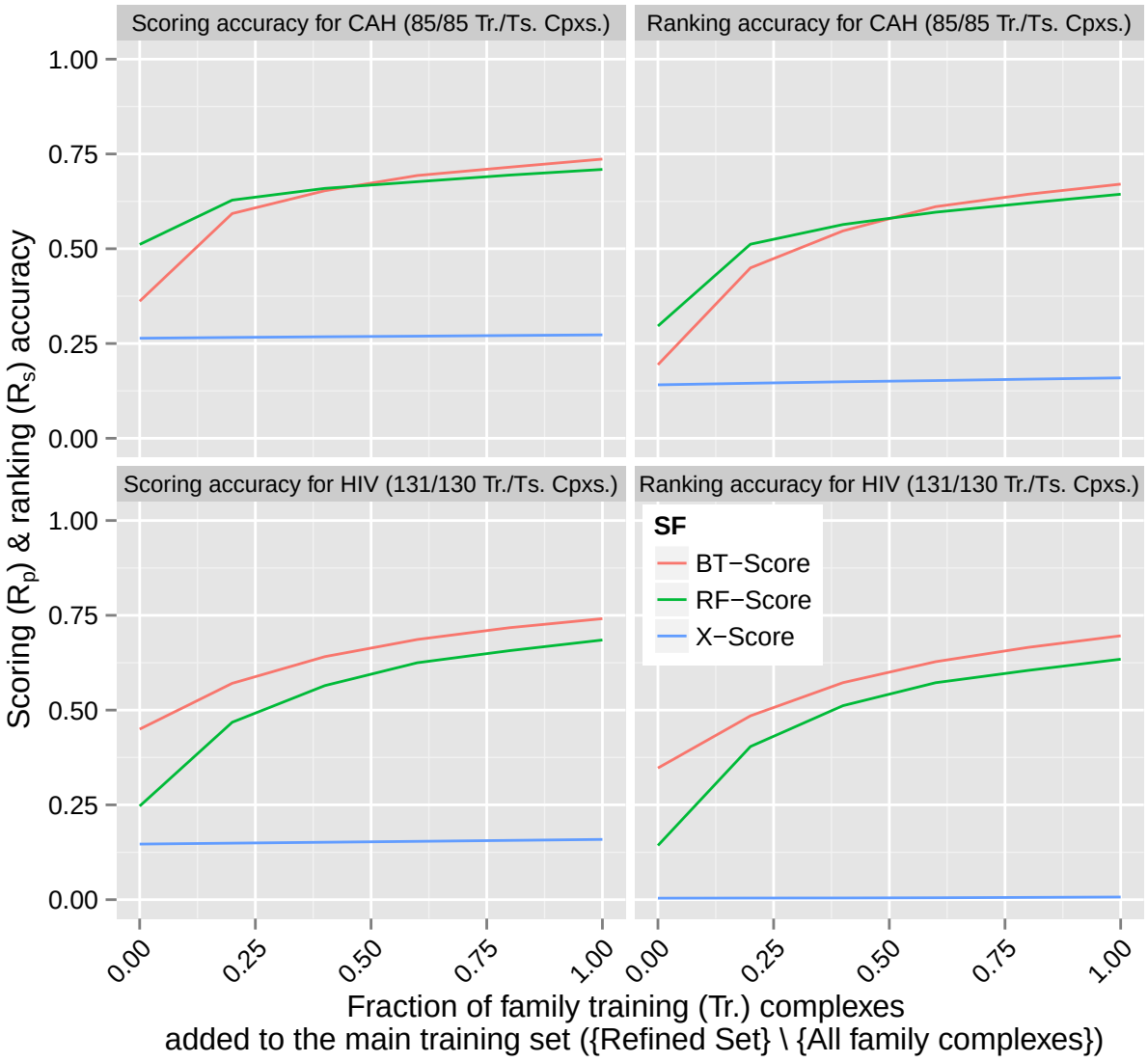


Figure 4.7: Dependence of scoring performance of machine-learning scoring functions on the number of HIV protease (left) and carbonic anhydrase (right) complexes in their training set. They are evaluated on out-of-sample HIV protease and carbonic anhydrase complexes from the refined set of PDBbind 2014.

significant improvement. To put that in perspective, the scoring and ranking accuracy of BT-Score improve by more than 140% when the number of HIV complexes in the training data increases from 0 to 208, whereas the corresponding increase in performance for the linear model X-Score does not even exceed 2%.

The analysis of the effectiveness of different regression approaches on specific families of

proteins has very practical benefits in virtual screening and drug discovery. Typically, the task in drug design is to conduct computational screening of large numbers of ligands against a certain protein family. The knowledge of the number of known binders to that protein present in the training set and the characteristic plots of several scoring models (similar to the ones in Figure 4.7) can aid in the selection of the most effective SF.

4.5.4 Performance of machine-learning scoring functions on novel targets

4.5.4.1 Simulating novel targets using PDBbind 2007

The training-test set pair (*Pr*, *Cr*) of PDBbind 2007 is a useful benchmark when the aim is to evaluate the performance of scoring functions on targets that have some degree of sequence similarity with at least one protein present in the complexes of the training set. This is typically the case since, as it was mentioned earlier, 92% of drug targets are similar to known proteins [113]. When the goal is to assess scoring functions in the context of novel protein targets, however, the training-test set pair (*Pr*, *Cr*) is not that suitable because of the partial overlap in protein families between *Pr* and *Cr*. We considered this issue to some extent in Section 4.5.3.1, where we investigated the performance of SFs on four different protein-specific test sets from PDBbind 2007 after training them on complexes that did not have the protein under consideration. This resulted in a drop in performance of all SFs, especially, in the case of HIV protease as a target. However, even if there are no common proteins between training and test set complexes, different proteins at their binding sites may have sequence and structural similarities, which influence protein-ligand BA. To more rigorously and systematically assess the performance of ML SFs on novel targets, we performed a separate set of experiments in which we limited BLAST sequence similarity between the binding sites of proteins present in the training and test set complexes in the refined set of PDBbind 2007.

Specifically, for each similarity cut-off value S = 30%, 40%, 50%, ..., 100%, we constructed 100 different independent 150-complex test and T-complex training set pairs, trained the scoring models (MLR, MARS, kNN, SVM, BRT, and RF) using two different feature sets (XAR and X

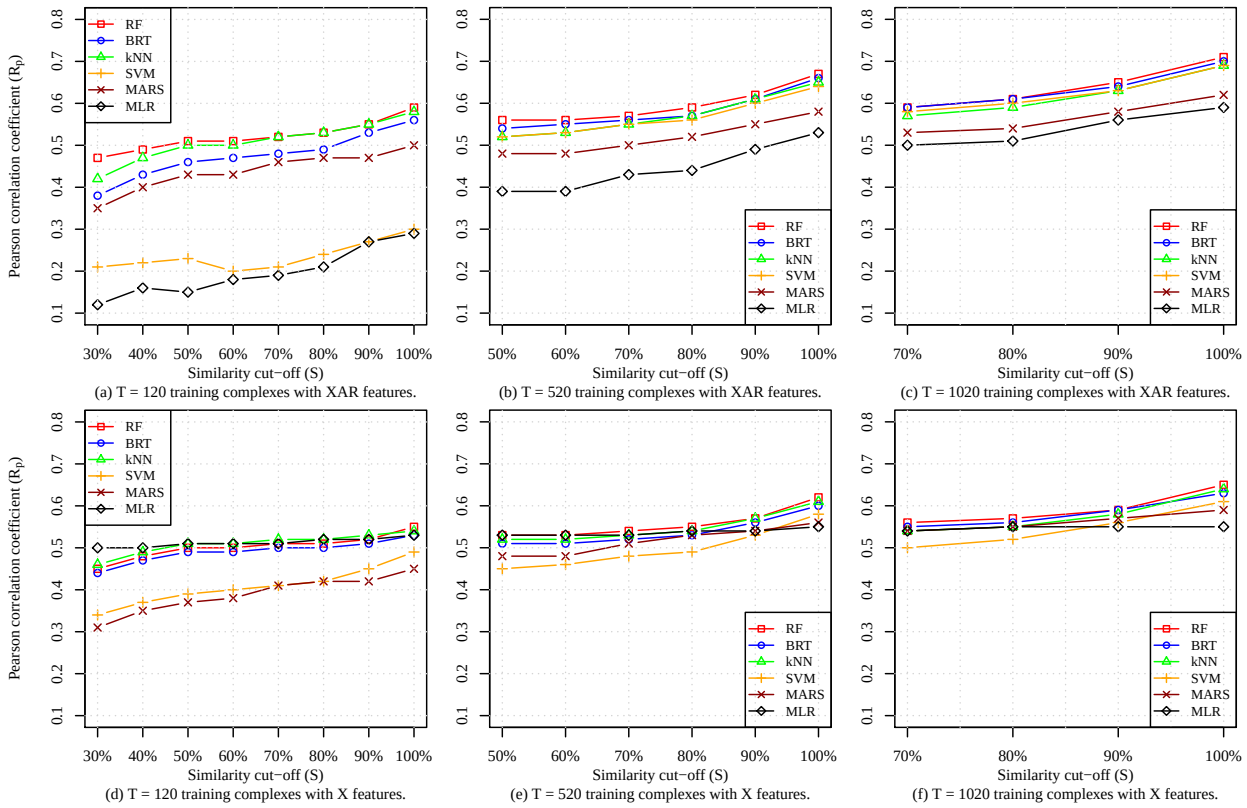


Figure 4.8: Binding affinity prediction accuracy of scoring models as a function of BLAST sequence similarity cutoff between binding sites of proteins in training and test complexes of PDBbind 2007.

features) on the training set and evaluated them on the corresponding test set, and determined their average performance over the 100 rounds to obtain robust results. Since SF prediction performance depends upon both similarity cut-off and training set size and since training set size is constrained by similarity cut-off (a larger S means a larger feasible T), we investigated different ranges of S (30% to 100%, 50% to 100%, and 70% to 100%) and for each range we set T close to the largest feasible value for the smallest S value in that range. Each test and training set pair was constructed as follows. We randomly sampled a test set of 150 protein-ligand complexes without replacement from the refined sets of PDBbind 2007 and 2010 which together provide 2182 high-quality protein-ligand complexes. The remaining (2182-150=) 2032 complexes were randomly scanned until T different training complexes were found that had protein binding site similarity of S% or less with the protein binding sites of all complexes in the test set — if less than T such

complexes were found, then the process was repeated with a new 150-complex test set.

The accuracy of the six scoring models in terms of Pearson correlation coefficient R_p is depicted in Figure 4.8 for a variety of similarity cut-offs and training set sizes. The plots in the first column (Figure 4.8 (a) and (d)) are for similarity cut-offs 30% to 100% in which $T = 120$ complexes is the largest training set size feasible for $S = 30\%$. The upper plot (Figure 4.8 (a)) shows these results when protein-ligand complexes are described by XAR features and the bottom plot when such compounds are characterized by X-Score (X) features alone. When XAR features are used, RF and *k*NN perform the best across the entire range of S when $T = 120$. In the corresponding bottom plot, in which the six X features are considered, it can be easily observed that MLR, *k*NN, RF, and BRT are the top four performing models. In both feature set cases, we see that no ML SF is able to predict the measured BA with good accuracy ($R_p \leq 0.5$ when $S = 30\%$). This can be attributed to three factors. First is the significant sequence, and hence structural, dissimilarity between the binding sites of proteins in training and test complexes. Second is the limited size of the training data. We will show how important the size of training data is in influencing the quality of scoring models in Section 4.5.5. Finally, the scoring model parameters used for this experiment were based on parameter tuning with respect to *Pr* as described in Section 2.2 rather than with respect to the training set used for this experiment, which is not only small but has a different mix of complexes governed by the small similarity cut-off of 30%. This is evident in Figure 4.8 (a) from the inferior performance of several SFs (MLR, SVM, MARS and, to a limited extent, BRT) when XAR features are used instead of X features (which is a subset of XAR features), signifying overfitting or suboptimal parameter values. The reason the RF model had relatively good performance compared to the other generally-competitive ML models for both XAR and X features is most likely because it is immune to overfitting and less sensitive to parameter tuning. Due to the time-intensive nature of tuning parameters of multiple scoring models for the many training scenarios we consider in this work, we used the parameters tuned with respect to *Pr*.

For a given similarity cut-off, not only does the scoring performance of models improve with the size of training dataset, the overfitting/parameter-tuning problem also gets alleviated or mini-

mized. For example, for a similarity cut-off of 50%, BRT::XAR has inferior performance relative to BRT:X when training dataset has 120 complexes (0.46 vs. 0.49), but the opposite is true when the training set has 520 complexes (0.54 vs. 0.51). Also, the performance of RF and the other generally-competitive ML models (BRT and *k*NN) becomes comparable as training dataset size increases, signifying that parameter tuning becomes less of an issue.

To summarize, imposing a sequence similarity cut-off between the binding sites of proteins in training and test set complexes has an expected adverse impact on the accuracy of all scoring models. However, increasing the number of training complexes helps improve accuracy for all similarity cut-offs. RF has the best accuracy considering the entire range of similarity cut-offs. The other generally-competitive ML models (BRT, *k*NN, and SVM) may also provide comparable accuracy if parameter tuning is performed with respect to the training set being considered.

4.5.4.2 Simulating novel targets using PDBbind 2014

In this section, we test the scoring ability of scoring functions on novel targets extracted from the 2014 release of PDBbind. We addressed this issue to some extent using our leave-cluster-out (*LCO*) test sets by restricting the similarity between training and test complexes to 90% or less using BLAST. Figure 4.3 shows the gap between the accuracies of SFs on *Cr* complexes as opposed to *LCO* PLCs. In Section 4.5.3.2, we also investigated the performance of SFs on four different 2014 PDBBind protein-specific test sets after training them on complexes that did not have the protein under consideration. This resulted in a drop in performance of all SFs, especially, in the case of HIV protease and carbonic anhydrase targets. As we explained in the previous section, even if there are no common proteins between training and test set complexes, different proteins at their binding sites may have sequence and structural similarities, which influence protein-ligand BA. To more rigorously and systematically assess the performance of ML SFs on the 2014 release of PDBbind novel targets, we performed another set of experiments in which we limited BLAST sequence similarity between the binding sites of proteins present in the training and test set complexes.

The procedure of simulating the novel targets using PDBbind 2014 has some similarities with

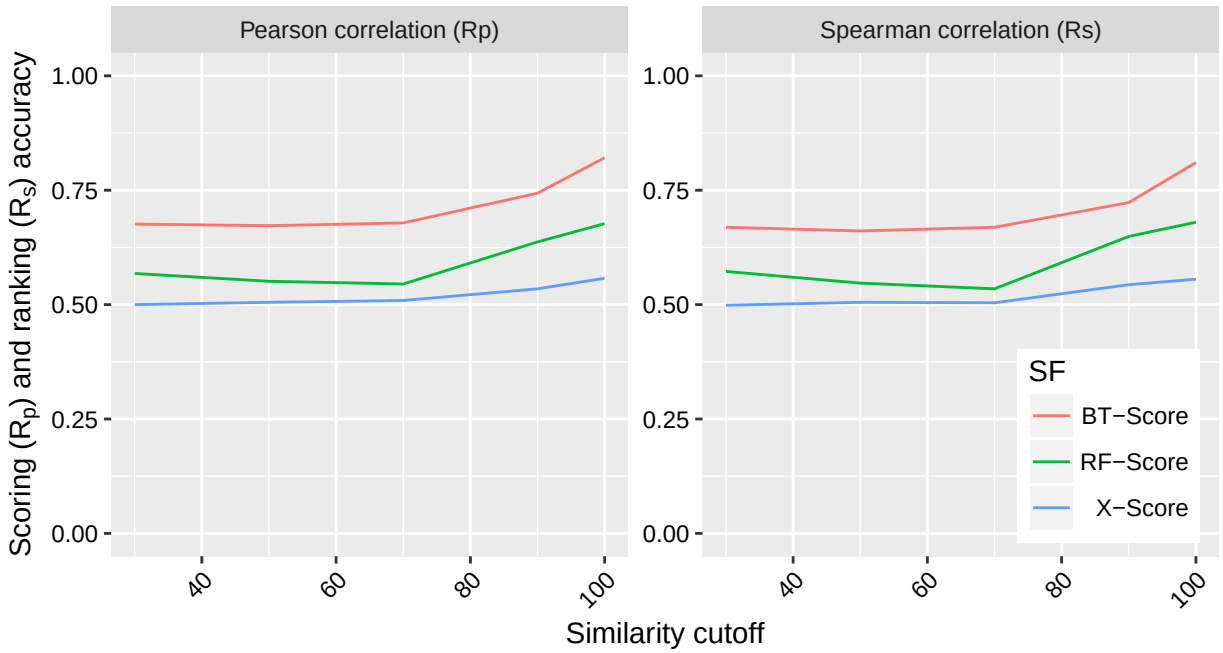


Figure 4.9: Binding affinity prediction accuracy of scoring models as a function of BLAST sequence similarity cutoff between binding sites of proteins in training and test complexes of PDBbind 2014.

the novel targets experiment performed on the 2007 release of PDBbind in Section 4.5.4.2. Specifically, for each similarity cut-off value $S = 30\%, 40\%, 50\%, ..., 100\%$, we constructed 100 different independent 100-complex test and 3000-complex training set pairs, trained the scoring models BT-Score, RF-Score, and X-Score using their respective feature sets (2714 multi-perspective, 36 geometric, and 6 physiochemical descriptors, respectively) on the training set and evaluated them on the corresponding test set, and determined their average performance over the 50 rounds to obtain robust results. Each test and training set pair was constructed as follows. We randomly sampled a test set of 100 protein-ligand complexes without replacement from the 3446 complexes of the 2014 PDBbind refined set. The remaining 3346 complexes were randomly scanned until 3000 different complexes were found that had protein binding site similarity of $S\%$ or less with the protein binding sites of all complexes in the test set — if less than 3000 such complexes were found after multiple trials, then the process was repeated with a new 100-complex test set.

The accuracy of the three scoring models in terms of Pearson (R_p) and Spearman (R_s) correla-

tion coefficients are depicted in Figure 4.9 for a variety of similarity cut-offs. BT-Score produces substantially better predictions than RF-Score and X-Score regardless of the level of similarity between the binding pocket of training and test proteins. BT-Score and RF-Score performance improves steadily as they are trained and tested on proteins with BLAST similarity of 70% or more. On the other hand, the improvement slope of X-Score appears flatter than BT-Score and RF-Score. The slow response of X-Score to increasing the similarity between training and test proteins can be attributed to the high rigidity of their underlying linear model. The small number of descriptors used by X-Score also plays a role in capturing and utilizing the similarity information between hundreds of different training protein families. The SF BT-Score, and to some extent RF-Score, do not suffer from model rigidity and descriptor limitations. We trained and tested boosted regression tree models in conjunction with the six physiochemical descriptors of X-Score and noticed better predictions and sharper responses to train-test similarity than those obtained using the original linear regression model.

Due to the sufficiently large number of training complexes in PDBbind 2014, we notice that the performance of BT-Score is always better than RF-Score and X-Score despite its use of much bigger feature set (2714 vs. 36 and 6 descriptors used by RF-Score and X-Score respectively). In the case of the 2007 PDBbind simulation in the previous section, we noticed that larger number of descriptors has some negative effect on the predictive accuracy of scoring functions when trained on smaller datasets. In this section, our results show that larger training sets sizes make it possible to incorporate more descriptors without incurring model overfitting. In the next section, we will investigate the effect of training set size on the scoring accuracy in more detail.

4.5.5 Impact of the number of training protein-ligand complexes on the scoring and ranking accuracies

4.5.5.1 Simulating increasing training set size using PDBbind 2007 & 2010

Experimental information about 3D structures and binding affinities of new protein-ligand complexes are regularly determined. This contributes to the growing size of public biochemical repos-

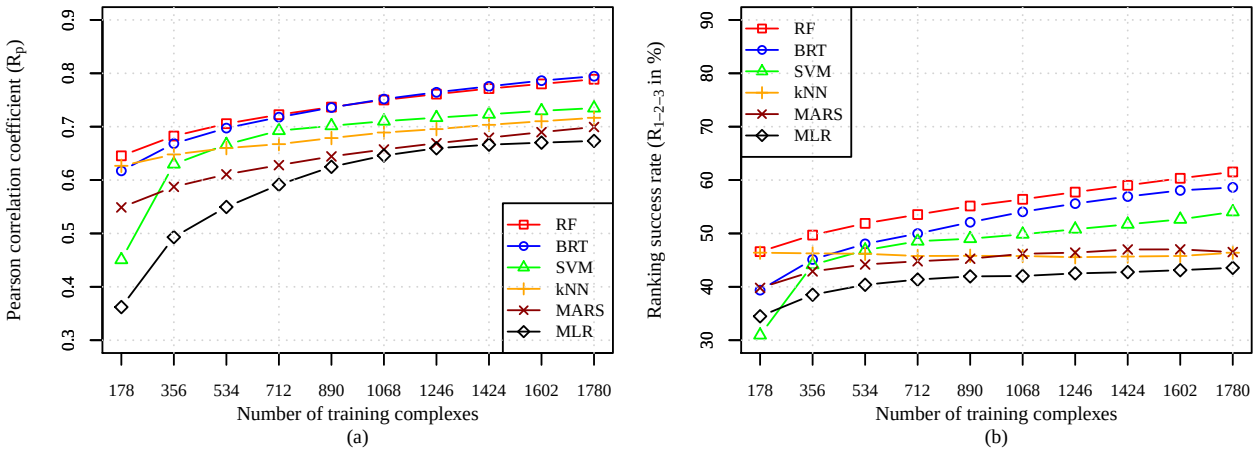


Figure 4.10: Dependence of scoring (left) and ranking (right) accuracies of machine-learning scoring functions on training set size. The models are trained on complexes selected randomly (without replacement) from the 2007 and 2010 refined sets of PDBbind and tested on out-of-sample complexes from the core (*Cr*) test set of PDBbind 2007.

itories and corporate compound banks. To assess the impact that a larger training set size would have on the predictive accuracy of ML SFs, we consider the 2007 and 2010 PDBbind datasets—more than 750 new protein-ligand complexes have been deposited into this database from the year 2007 to 2010[3]. After setting aside the core (*Cr*) test set of PDBbind 2007 for evaluation, we use the remaining complexes of the PDBbind 2007 and 2010 refined sets for training which we denote by *Tr*. For a given number of training complexes x, $x = 1 \times 178, 2 \times 178, \ldots, 10 \times 178$, we select x complexes randomly (without replacement) from *Tr* to train the six ML models. We then test them on the disjoint core set *Cr* which is comprising 195 complexes. This process is repeated 100 times to obtain robust average R_p values, which are plotted in Figure 4.10.

Figure 4.10 shows the scoring and ranking accuracies of RF, BRT, SVM, *k*NN, MARS and MLR SFs when the datasets are characterized using XAR features[4]. In both cases (scoring and ranking), we observe that the performance of almost all models improves as the size of the training

[3]This experiment was conducted prior to considering the 2014 version of PDBbind, therefore complexes from this set are not used in this experiment. However, we expect similar performance trends for all scoring functions.

[4]This simulation was performed before extracting the G-type features and developing the Neural Network based models, therefore they are not included here. The performance trends of BsN-Score and BgN-Score are expected to be similar to that of BRT and RF SFs, respectively.

dataset increases. When the number of features is high ($|XAR| = 72$) and the size of training dataset is relatively small, we notice poor accuracy of some SFs such as MLR and SVM. This is perhaps a consequence of overfitting since the ratio of training dataset size to number of features is small. The performance of MLR and SVM improves substantially when the training dataset size increases and/or lower dimensional data is used. RF and BRT based SFs are almost always the leading models regardless of the size of training set.

From Figure 4.10, we can conclude that most ML SFs have a potential for improvement as more training data becomes available. SF designers can conduct similar experiments to estimate accuracy enhancement when their proposed functions are recalibrated on larger number of data instances.

The non-linear ML approaches fitted to the 6 X-type features (whose accuracy plots are not shown) were also found to be improving as the number of their training complexes increases. MLR::X model on the other hand, which resembles empirical SFs, showed minimal response to increasing the training set size which is an indication of a very small potential for improvement in the future. This is due to the rigidity of linear models whose performance tends to saturate. Many of the 16 conventional SFs considered in this study are empirical and thus they are also most likely to suffer from the same limitation. In fact, the best performing model of those 16 in terms of scoring power, X-Score::HMScore, is very similar to MLR::X. That is because both SFs use almost the same features and both are linearly fit to the same training data. Therefore, one should consider better prediction approaches to derive accurate models from training data available on hand and from future updates.

4.5.5.2 Simulating increasing training set size using PDBbind 2014

In this section we consider a larger number of complexes obtained from the 2014 version of PDBbind to simulate the performance of scoring functions on increasing sizes of training data. More than 2100 new protein-ligand complexes have been deposited into this database from the year 2007

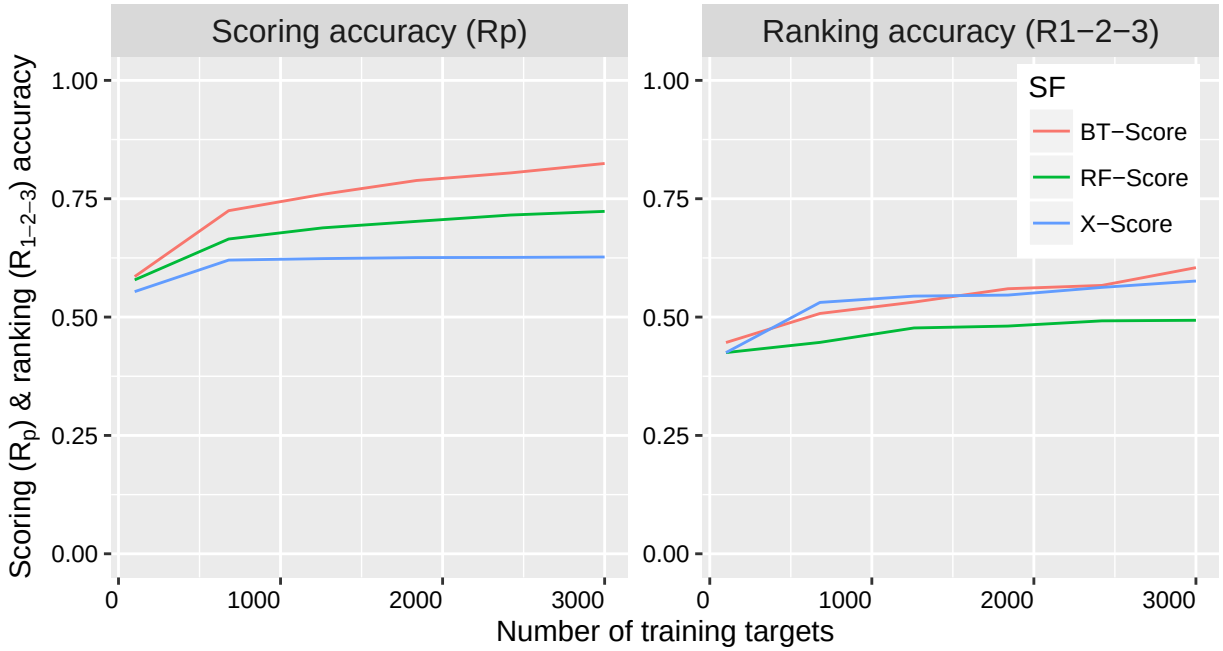


Figure 4.11: Dependence of scoring (left) and ranking (right) accuracies of machine-learning scoring functions on training set size. The models are trained on complexes selected randomly (without replacement) from the refined set of PDBbind 2014 and tested on out-of-sample complexes from the core (*Cr*) test set of PDBbind 2014.

to 2014[5]. To be able to use a greater range of training set sizes, we choose the refined (*Ref*) set of PDBBind 2014 to build and test three different scoring models. For a given number of training complexes x, where $x \in \{100, 680, 1260, 1840, 2420, 3000\}$, we select x complexes randomly (without replacement) from *Ref* to train the three ML models. We then test them on the out-of-sample core set *Cr* comprising 195 complexes. This process is repeated 50 times to obtain robust average scoring (R_p) and ranking ($R_{1\text{-}2\text{-}3}$) values, which are plotted in Figure 4.11.

Figure 4.11 shows the scoring and ranking accuracies of BT-Score, and our versions of RF-Score and X-Score. The figure clearly shows that the scoring and ranking accuracies of BT-Score and RF-Score improve as the size of the training dataset increases. The scoring performance of X-Score is substantially lower than both ensemble SFs and its rate of improvement is almost zero beyond 680 training complexes. Its ranking accuracy, however, is on par with BT-Score and

[5]This experiment was conducted prior to the release of the 2015 and 2016 versions of PDBbind, therefore complexes from the new releases are not used in this experiment. However, we expect similar performance trends for all scoring functions.

both show similar gain in ranking performance as they get exposed to more training complexes. Similar rise in improvement was also achieved using Random Forest (whose accuracy plots are not shown) when it was fitted to our multi-perspective descriptors instead of the RF-Score's original 36 geometric descriptors. Another set of Random Forest models fitted to the 6 features used by X-Score were also found to be improving in scoring accuracy as the number of their training complexes increases. This indicates that ensemble SFs have a potential for improvement as more training data becomes available. The linear model used in X-Score (and other empirical SFs) on the other hand showed minimal rise in scoring power by increasing the training set size which is an indication of a very small potential for improvement in the future. Overall, the improvement plots of BT-Score, RF-Score, and X-Score on PDBbind 2014 complexes are similar to the improvement trends of the corresponding models BRT, RF, and MLR on the PDBbind 2007 benchmark as can be observed by comparing Figures 4.10 and 4.11.

4.5.6 Impact of the type and number of descriptors on the scoring and ranking accuracies

4.5.6.1 Type and number of descriptors characterizing PDBbind 2007 complexes

The BA of a protein-ligand complex depends on many physicochemical interaction factors that are too complex to be accurately captured by any one approach. Therefore, we perform three different experiments to investigate how utilizing different types of features from different scoring tools, X-Score, AffiScore, RF-Score, and GOLD, and considering an increasing number of features affects the performance of the various ML models. We also test how increasing the number of descriptor sets affect the performance of three scoring models. In the first experiment, six ML (RF, BRT, SVM, *k*NN, MARS, and MLR) models were fitted to the PDBbind primary *Pr* training complexes characterized by X, A, R, G, and XARG features and tested on the corresponding core test *Cr* characterized by the same features[6]. Table 4.4 reports the scoring (R_p) and ranking ($R_{1\text{-}2\text{-}3}$)

[6]The training (*Pr*) and test (*Cr*) sets were based on the 2007 PDBBbind datasets. This experiment was conducted before the release of 2014-2016 PDBBind datasets and prior to developing the Neural Network based models, therefore they are not included here. However, we expect the

Table 4.4: Scoring and ranking accuracies of machine-learning scoring functions when protein-ligand complexes are characterized by different descriptor types

Model	Scoring Power (R_p)					Ranking Power ($R_{1\text{-}2\text{-}3}$ in%)				
	X	A	R	G	XARG	X	A	R	G	XARG
BsN-Score	0.694	0.737	0.710	0.728	0.816	52.3	47.7	41.5	41.5	64.3
BgN-Score	0.664	0.783	0.750	0.550	0.808	53.8	53.8	44.6	41.5	60.0
DNN-Score	0.675	0.583	0.503	0.501	0.631	33.8	35.7	35.3	35.1	38.9
RF	0.743	0.763	0.777	0.726	0.804	56.6	46.5	52.1	49.9	63.8
BRT	0.719	0.787	0.746	0.72	0.810	60.6	51.7	50.8	44.6	63.1
SVM	0.693	0.716	0.739	0.603	0.779	44.6	46.2	38.5	36.9	56.9
*k*NN	0.703	0.716	0.717	0.613	0.753	52.3	50.8	44.6	33.8	53.8
MARS	0.641	0.675	0.625	0.591	0.681	41.5	44.6	44.6	35.4	41.5
MLR	0.648	0.681	0.602	0.555	0.684	49.2	44.6	50.8	44.6	50.8

performance statistics for the resulting 36 SFs.

We notice that the scoring and ranking accuracies of almost all models, except MARS and MLR, improve by considering more than one type of features rather than just X, A, R, or G features alone. The results of Table 4.4 are useful in assessing the relative benefit of different types of features for the various ML models.

A pertinent issue when considering a variety of features is how well different SF models exploit an increasing number of features. The features we consider are the X, A, G, and a larger set of geometrical features than the R feature set available from the RF-Score tool. RF-Score counts the number of occurrences of 36 different protein-ligand atom pairs within a distance of 12 Å. In order to have more features of this kind for this experiment, we produce 36 such counts for five contiguous distance intervals of 4 Å each: (0 Å, 4 Å], (4 Å, 8 Å], ..., (16 Å, 20 Å]. This provides us 6 X, 30 A, 14 G, and $(36 \times 5 =)$ 180 geometrical features or a total of 230 features. We randomly select (without replacement) x features from this pool, where $x = 20, 40, 60, \ldots, 220$, and use them to characterize the 2007 and 2010 PDBbind dataset, which we then use to train the six ML models. These models are subsequently tested on the out-of-sample 2007 *Cr* dataset characterized by the same features[7]. This process is repeated 100 times to obtain robust average R_p statistics, which are

performance trends of BsN-Score and BgN-Score to be similar to that of BRT and RF SFs.

[7] This experiment was conducted before the release of 2014-2016 PDBBind datasets and prior

plotted in Figure 4.12.

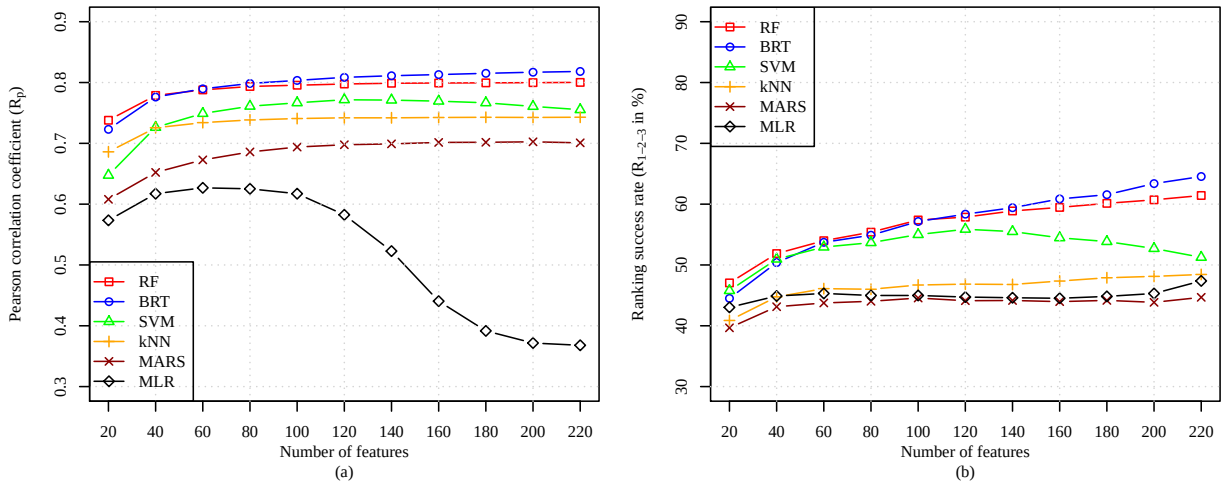


Figure 4.12: Dependence of scoring (left) and ranking (right) accuracies of machine-learning scoring functions on the number of descriptors. The descriptors are drawn randomly (without replacement) from a pool of X, A, R, and G-type features and used to characterize the training complexes of the refined sets of PDBbind 2007 and 2010 and the out-of-sample complexes from the core (*Cr*) test set of PDBbind 2007.

Clearly, the RF, BRT, SVM, *k*NN, MARS, and MLR SFs have very different responses to an increase in the number of features. For several of them, peak performance is attained at 60 (*k*NN and MLR) or 120 (SVM) features and then there generally tends to be a drop in or saturation in performance at larger number of features. Although the features we used are distinct, they have varying degrees of correlation between them. This combined with larger number of features lead to overfitting problems for some of the SFs. Further, ML model meta-parameters are not tuned for every number of features chosen. This especially affects the performance of SVM which we have found to be very sensitive to its parameter values. However, tuning SVM parameters for every number of features is computationally intensive, and therefore we did not attempt to search for the optimal parameter values for every feature set size for it. In contrast to these models, the performance of RF and BRT benefits from increasing number of features. Based on these results, utilizing as many relevant features as possible in conjunction with ensemble based approaches like

to developing the Neural Network based models, therefore they are not included here. However, we expect the performance trends of BsN-Score and BgN-Score to be similar to that of BRT and RF SFs.

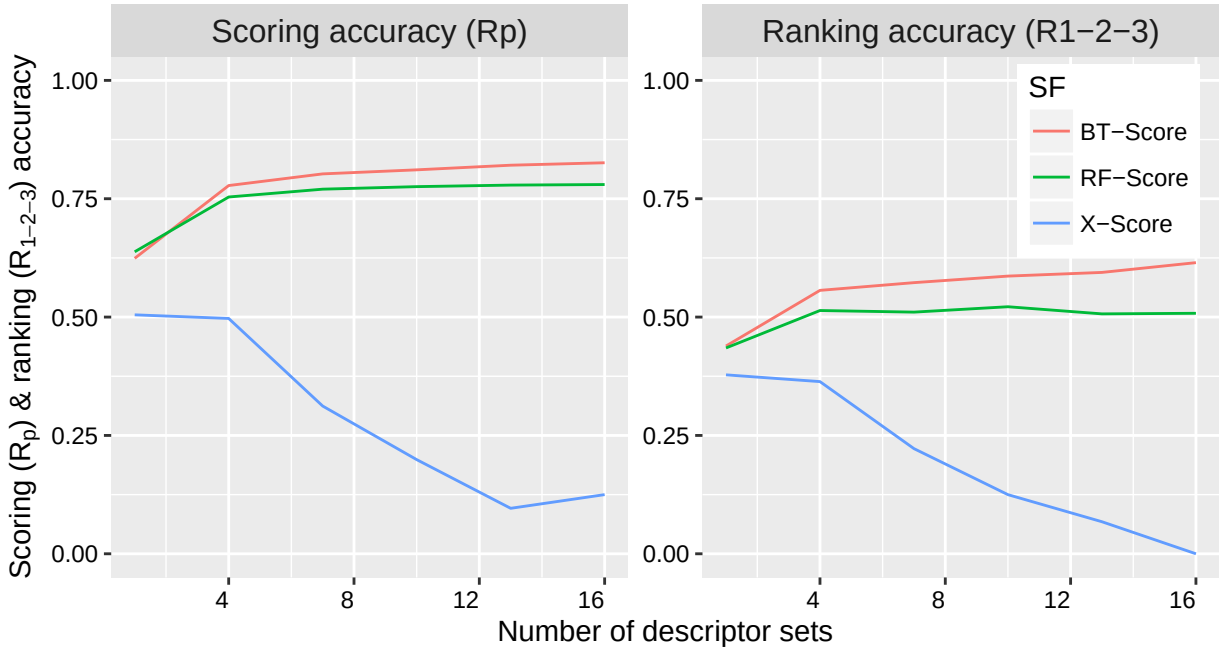


Figure 4.13: Dependence of scoring (left) and ranking (right) accuracies of machine-learning scoring functions on the number of descriptor sets. The descriptor sets are randomly sampled from all feature types in Descriptor Data Bank. The selected descriptor types are used to characterize the training complexes of the refined set of PDBbind 2014 and the out-of-sample complexes from the core (*Cr*) test set of PDBbind 2014.

BRT and RF that are resilient to overfitting appears to be the best option.

4.5.6.2 Number of descriptor sets characterizing PDBbind 2014 complexes

To investigate the effect of increasing the number of descriptor sets, we randomly select a maximum of 100 combinations of x descriptor sets from all possible (16 choose x) combinations of the 16 DDB types listed in Table 3.1, where $x \in \{1,4,7,10,13,16\}$, and use them to characterize the training set complexes in PDBbind 2014 database, which we then use to train the XGboost, Random Forest, and MLR models to build the scoring functions BT-Score, RF-Score, and X-Score, respectively. These models are subsequently tested on the 2014 PDBbind *Cr* benchmark data set characterized by the same descriptors. For each number of descriptor sets, x, the performance of the 100 SFs are averaged to obtain robust overall scoring (R_p) and ranking ($R_{1\text{-}2\text{-}3}$) statistics, which are plotted in Figure 4.12.

Similar to the case of increasing training data in terms of new PLCs and number of raw features in the previous section, the performance of the ensemble SFs are clearly benefiting from increasing the number of protein-ligand interaction models in the scoring and ranking tasks. It is clear from the right and left plots that the number of different models of protein-ligand interactions (descriptor sets) are as important as the number of training complexes in improving the quality of prediction. Based on these results, we believe that the public database of protein-ligand interactions we are proposing in this work will be of high value to ML SFs and just as important to their scoring and ranking accuracy as the resources of raw structural and experimental data.

4.6 Conclusion

Our experiments have shown that the proposed neural networks SFs, BsN-Score and BgN-Score, achieved the best results in reproducing experimental binding affinity for large and diverse number of protein-ligand complexes. We further found that ensemble models based on NNs surpass SFs based on the decision-tree ensemble techniques Boosted Regression Trees and Random Forests as well as other state-of-the-art ML algorithms such as SVM. SFs that were trained on a single neural network, which have traditionally been used in drug-discovery applications, showed linear correlation (R_p) of only 0.627 between observed and predicted binding affinities. On the other hand, BsN-Score and BgN-Score outperform the best of other ML and existing conventional knowledge-based, force-field-based, and empirical SFs ($R_p = 0.844$ and 0.840 vs. 0.725 and 0.627, respectively) and those based on a single neural network whose correlation coefficient is 0.803. Similar results have been obtained for the ranking task. Within clusters of protein-ligand complexes with different ligands bound to the same target protein, we find that the best ML-based SF is able to rank the ligands correctly based on their experimentally-determined binding affinities 61.5% of the time and identify the top binding ligand 81.5% of the time. Given the challenging nature of the ranking problem and that SFs are used to screen millions of ligands, this represents a significant improvement over the the best conventional SF we studied, for which the corresponding ranking performance values are 58.0% and 69.1%. The accuracies of ensemble NN SFs are even

higher when they predict binding affinities for protein-ligand complexes that are related to their training sets. The high predictive accuracy of ensemble SFs BsN-Score and BgN-Score is due to the following three factors: (i) the low bias error of the highly-nonlinear neural network base learners, (ii) the low variance error achieved using bagging and boosting, and (iii) the employed diverse set of multi-perspective descriptors we extract for protein-ligand complexes.

We also observed that the ensemble scoring functions benefit more than their conventional counterparts from increases in the number of descriptors and the size of training dataset. This means that the proposed scoring models will be even more accurate in the future when 3D structures of more protein-ligand complexes are resolved and larger numbers of descriptors are modeled.

CHAPTER 5

DOCKING-SPECIFIC SCORING FUNCTIONS FOR ACCURATE LIGAND POSE PREDICTION

Molecular docking is a computational approach widely used in structure-based drug design to simulate the behavior of drug-like ligands interacting with a target protein to bind with it and form a stable protein-ligand complex. An essential component of molecular docking programs is a scoring function (SF) that is used to identify the most stable binding pose of a ligand, when bound to a receptor protein, from among a large set of candidate poses. Scoring functions employed in most of commercial docking software rank ligand poses based on the binding affinities they predict. Although some scoring functions have achieved some success in terms of identifying the correct poses for many protein targets, their accuracy is still inadequate for reliable virtual screening. When a poor ligand pose is mistakenly predicted as the conformation the ligand would take when it interacts with the protein, the ligand could be incorrectly prioritized over other compounds in the database as a better drug candidate. This possibility is not implausible given the current limitation of binding affinity prediction which conventional scoring function uses to rank-order ligand poses.

In this chapter, we introduce a novel type of ensemble neural network scoring functions optimized specifically for the docking task. Various other nonparametric ML methods inspired from statistical learning theory are also examined in this chapter to model the unknown function that maps structural and physicochemical information of a protein-ligand complex to a corresponding distance to the native pose (in terms of RMSD value). Ours is the first work to develop and perform a comprehensive assessment of the docking accuracies of task-specific and conventional SFs across both diverse and homogeneous test sets of protein families. We show that the best docking-specific SF has a success rate of $\sim$80% compared to $\sim$70% for the best conventional SF when the goal is to find poses within RMSD of 1 Å from the native ones for 195 different protein-ligand complexes in PDBbind 2007 benchmark. Furthermore, ensemble neural network scoring functions optimized for the docking task correctly identified the ligand poses for 95% protein-targets in the

PDBbind 2014 benchmark as opposed to 82% obtained by the best commercial scoring function ChemPLP. In this test, the model's top-scoring pose is considered native or near-native (i.e., correctly identified) if it lies within 2 Å RMSD from the ligand's native conformation observed in the crystallographic structure.

Such a significant improvement ($> 14\%$) in docking power will lead to better quality drug hits and ultimately help reduce costs associated with drug discovery. We seek to advance ligand pose prediction by designing SFs that significantly improve upon the protein-ligand docking performance of conventional SFs. Our approach is to couple the modeling power of flexible machine learning algorithms with training datasets comprising hundreds of protein-ligand complexes with native poses of known high-resolution 3D crystal structures and experimentally-determined binding affinities. In addition, we computationally generate a large number of decoy poses and utilize their RMSD values from the native pose and a variety of features characterizing each complex. We compare the docking accuracies of the proposed ensemble deep neural networks and several other ML models to existing conventional SFs of all three types, force-field, empirical, and knowledge-based, on diverse and independent test sets. We also perform a systematic analysis of the ability of the proposed SFs in identifying native poses of ligands that are docked to novel protein targets. Further, we assess the impact of training set size on the docking performance of the conventional BA-based SFs and the proposed RMSD-based models.

The remainder of the chapter is organized as follows. The next section presents the procedure for decoy generation and formation of training and test datasets. Then, we present results comparing the docking powers of conventional and ML SFs on diverse and homogeneous test sets. We also compare the performance of the ML techniques on novel drug targets and analyze how they are impacted by training set size. Finally, we summarize our findings.

5.1 Decoy generation of ligand poses and formation of training and test complexes

We conducted our experiments on two versions of the protein-ligand complex repository PDBbind. In the first set of experiments, we considered the 2007 version of PDBbind which contains 1300 complexes with experimental binding affinity data. The primary reason for selecting this release was the availability of related studies to this work that used its primary and core datasets for training and benchmarking purposes [17, 27]. This dataset enables us to objectively compare the performance of our docking-specific ML scoring models to conventional SFs published by Cheng et al. [17] and Ballester et al. [27] since the proposed and traditional models are tested on the same benchmark using the same evaluation criteria. Upon the development of our boosted and bagged deep-learning SFs for docking, the 2014 version of PDBbind was released with about 250% more protein ligand complexes than the 1300 PLCs in the 2007 PDBbind. Shortly after that, another comparative study of SFs was also published which uses the 2014 PDBbind as a benchmark for several popular scoring functions. To evaluate the accuracy of our deep neural network based SFs and compare them consistently with the SFs published therein, we too use the same release of PDBbind and the evaluation criteria employed in the latter comparative study. It should be noted that both comparative studies based on the 2007 and 2014 releases of PDBbind are similar in scope and methodology. In this chapter, we will show results on both versions for every experiment we conduct.

5.1.1 Generating decoy poses for protein-ligand complexes in PDBBind 2007

The proteins of both the 2007 PDBbind primary (*Pr*) training and core (*Cr*) test sets form complexes with ligands that were observed bound to them during 3D structure identification. These ligands are commonly known as native ligands and the conformation in which they were found at their respective binding sites are referred to as true or native poses. In order to assess the docking power of SFs in distinguishing true poses from random ones, a decoy set was generated for each protein-ligand complex in *Pr* and *Cr*. We utilize the decoy set produced for the core set *Cr*

by Cheng et al. [17] using four popular docking tools: LigandFit in Discovery Studio, Surflex in SYBYL, FlexX in SYBYL (currently in LeadIT [114]), and GOLD. From each tool, a diverse set of binding poses was generated by controlling docking parameters as described in [17]. This process generated a total of $\sim$2000 poses for each protein-ligand complex from the four docking protocols combined. Binding poses that are more than 10 Å away, in terms of RMSD (root-mean-square deviation), from the native pose are discarded. The remaining poses are then grouped into ten 1 Å bins based on their RMSD values from the native binding pose. Binding poses within each bin were further clustered into ten clusters based on their similarities [17]. From each such sub-cluster, the pose with the lowest noncovalent interaction energy with the protein was selected as a representative of that cluster and the remaining poses in that cluster were discarded. Therefore, at the end of this process, decoy sets consisting of (10 bins $\times$ 10 representatives $=$) 100 diverse poses were generated for each protein-ligand complex. Since we have access to the original *Cr* decoy set, we used it as is and we followed the same procedure to generate the decoy set for the training data *Pr*. Since we did not have access to Discovery Studio software, we did not use LigandFit protocol for the training data. In order to keep the size of the training set reasonable, we generated 50 decoys for each protein-ligand complex instead of 100 as it is the case for *Cr* complexes. Due to geometrical constraints during decoy generation, the final number of resultant decoys for some complexes does not add up exactly to 50 for *Pr* and 100 for *Cr*. It should be noted that the decoys in the training set are completely independent of those in the test set since both datasets share no ligands from which these decoys are generated.

As noted earlier, in the ligand docking problem with which we are concerned in this chapter, the task is to identify the correct binding mode of a ligand from among a set of (computationally-generated) poses. The closer, in terms of RMSD, a pose is to the experimentally-determined native pose, the better [17]. We develop two types of ML SFs in this work to identify poses close to the native one. The first type are trained on training complexes with (known experimentally-determined) binding affinity (BA) as the response variable. To assess their docking accuracy, their predicted BA on a separate set of test complexes is used to distinguish promising poses from less

promising ones. Note that for the test complexes, the experimentally-determined BA and actual RMSD values are not used during BA prediction; the actual RMSD value for test complexes is only used to assess docking accuracy. In all previous work, BA has been used for identifying near-native poses, which carries with it the implicit assumption that higher predicted BA implies lower RMSD for a pose. We believe a better approach is to model RMSD instead of BA. Therefore, the second set of SFs we build are trained on training complexes with (known) RMSD as the response variable. The accuracy of this approach, as in the case of BA-based SFs, is assessed on a separate set of test complexes by ranking poses according to predicted RMSD values: the lower the predicted RMSD, the more likely a pose is closer to the native pose. Note that for the test complexes the experimental BA and actual RMSD values are not used during RMSD prediction; the actual RMSD value is used only for docking accuracy assessment after prediction. RMSD-based SFs have three advantages over BA-based SFs. First, RMSD-based SFs model the same parameter (RMSD) that is used for pose ranking instead of relying on a related parameter (BA). Second, BA-based SFs are trained on experimental BA values, which are inherently noisy, whereas RMSD-based SFs use computationally-determined RMSD values during training which makes them less error prone. Third, during training, multiple decoys with different RMSD values can be computationally generated per complex. Therefore, the number of training records that can be utilized by RMSD-based SFs is the product of the number of different training complexes and the average number of computationally-generated poses per training complex. This training set size can be much larger compared to that available to BA-based SFs which is limited to as many training records as the number of different training complexes because BA values can be experimentally determined only for native poses, not for decoys. As it will be shown later, our novel RMSD-based approach provides significantly superior accuracy compared to conventional BA-based prediction.

For the two types of SFs, two versions of training and test data sets are created. The first version uses BA as the dependent variable ($Y = \text{BA}$) and the size of Pr remains fixed at 1105 while Cr includes 16554 complex conformations because it consists of native poses and a decoy set for each

native pose. The dependent variable of the second version is RMSD (Y = RMSD) and because both training and test sets consist of native and decoy poses, the size of Pr expands to 39,085 while Cr still retains the same 16554 complex conformations.

For all protein-ligand complexes, for both native poses and computationally-generated decoys, we extracted X, A, R, and G features. By considering all fifteen combinations of these four types of features (i.e., X, A, R, G, $X \cup A$, $X \cup R$, $X \cup G$, $A \cup R$, $A \cup G$, $R \cup G$, $X \cup A \cup R$, $X \cup A \cup G$, $X \cup R \cup G$, $A \cup R \cup G$, and $X \cup A \cup R \cup G$), we generated 30 versions of scoring functions. Each scoring function is trained and evaluated on complexes characterized by one of the 15 descriptor combinations and use RMSD or binding affinity (BA) as a dependent variable (Y). We denoted these models using the name of the machine-learning algorithm (as a prefix) and the feature combintion (as a suffix). The scoring function *BsN-Score::XR*, for example, denotes the boosted ensemble neural network model fitted to complexes characterized using the set of descriptors $X \cup R$ (referred to simply as XR). RMSD and BA-based scoring functions will be distinguished from each other in the text when necessary.

5.1.2 Generating decoy poses for protein-ligand complexes in PDBBind 2014

The methodology of generating computational poses for the 2014 version of PDBbind is very similar to the process discussed in the previous section for PDBbind 2007. The main differences are the software packages used to perform the docking and the number of poses we generate for each protein-ligand complex. For this release, we use the open-source software AutoDock Vina [83] to conduct docking instead of Surflex in SYBYL [51], FlexX in LeadIT [114], and GOLD [52] due to the expiration of their licenses during the time we conducted this study. We also retain the full number of 100 computational poses for each protein-ligand complex in PDBbind 2014 instead of 50 as in the case of PDBbind 2007 since we have access to more computational resources. Figure 5.1 summarizes the steps we performed to generate the training poses and calculate their distance from their respective native conformations for PDBbind 2014. First, we randomly translated and rotated the native pose and then docked it into the binding site of the target which is

marked by the original location of the crystallized ligand. We use AutoDock Vina to generate 100 binding modes with its *exhaustiveness* search argument was set to 25. We repeated this random re-orientation and docking process 20 times to obtain a total of $(100 \times 20 =)$ 2000 diverse poses around the native conformation. We then calculated the symmetry-corrected root-mean-square distance of each computationally-generated pose to the native conformations. All poses with RMSD greater than 10 Å away from the native conformation were discarded. The remaining poses were then clustered into 10 one-Angstrom-wide bins where each bin contains 10 representative poses uniformly-spaced from each other. Therefore, for every native PLC we obtained a total of 100 poses spanning the [0-10Å] range where each pose is approximately 0.1 Å from its closest two neighboring conformations. The pose with 0 Å is simply the native conformation.

The pose-generation strategy was applied to each protein-ligand complex in our training and test datasets formed based on the sampling strategies we describe in Section 2.1. The multi-perspective descriptors were then extracted for each native, near-native, and decoy complex. Similar to the two versions of the six ML SFs evaluated on PDBbind 2007, here we also fit two versions of the proposed ensemble of deep neural networks scoring functions. Namely, we build the docking-specific SFs BsN-Dock and BgN-Dock that we train on PLCs with native and computationally-generated poses characterized by the 2714 DDB's multi-perspective descriptors and labeled by the RMSD values. We also construct their binding-affinity-based counterparts BsN-Score and BgN-Score that are fitted to complexes of native poses only and characterized by the same 2714 multi-perspective descriptors but labeled using their measured binding affinity values.

5.2 Conventional scoring functions

The proposed docking-specific models presented in this work are evaluated on the 2007 and 2014 versions of PDBbind and their performance is compared against popular SFs from the literature whose docking accuracy has been reported on the same benchmarks [17, 25].

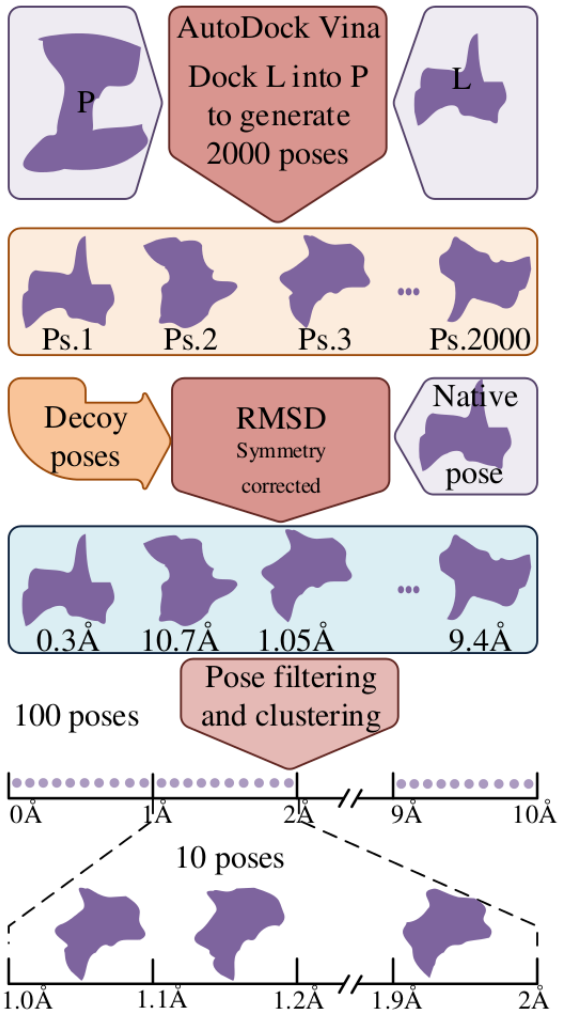


Figure 5.1: The decoy generation process of ligand poses to train and test scoring functions on complexes from the refined set of PDBbind 2014.

5.2.1 Conventional scoring functions evaluated on the PDBbind 2007 benchmark

A total of sixteen popular conventional SFs are compared to ML SFs on the 2007 version of PDBbind. The sixteen functions are either used in mainstream commercial docking tools and/or have been developed in academia. The functions were recently compared against each other in a study conducted by Cheng et al. [17]. This set includes five SFs in the Discovery Studio software [21]: LigScore, PLP, PMF, Jain, and LUDI. Five SFs in SYBYL software [51]: D-Score, PMF-Score, G-Score, ChemScore, and F-Score. GOLD software [52] contributes three

SFs: GoldScore, ChemScore, and ASP. GlideScore in the Schrödinger software [54]. Besides, two standalone scoring functions developed in academia are also assessed, namely, DrugScore [55] and X-Score [56]. Some of the SFs have several options or versions, these include LigScore (LigScore1 and LigScore2), PLP (PLP1 and PLP2), and LUDI (LUDI1, LUDI2, and LUDI3) in Discovery Studio; GlideScore (GlideScore-SP and GlideScore-XP) in the Schrödinger software; DrugScore (DrugScore-PDB and DrugScore-CSD); and X-Score (HPScore, HMScore, and HSScore). For brevity, we only report the version and/or option that yields the best performance on the PDBbind benchmark that was considered by Cheng et al.

5.2.2 Conventional scoring functions evaluated on the PDBbind 2014 benchmark

In addition to the conventional methods evaluated on the core test set of PDBbind 2007, we compare the docking accuracy of the proposed deep neural networks SFs to five conventional models on the 2014 release of PDBbind. Three out of the five SFs are the empirical ChemPLP and ChemScore implemented in Gold [52] and the empirical GlideScore-SP featured in the modeling software Schrödinger [54]. These three were the top performing models among 18 other scoring functions in identifying the correct binding of PDBbind 2014 complexes as reported by Li et al. in a recent study [25]. The team also found X-Score to be the top performing in correctly predicting binding affinity and therefore we consider it here too. The ML SF RF-Score is also reported here due to its popularity in the literature and resemblance to the proposed bagged deep neural networks [27].

5.3 Docking-specific machine-learning scoring functions

5.3.1 Generic ML scoring functions evaluated on the PDBbind 2007 benchmark

For the 2007 version of PDBBind, we utilize a total of six regression techniques in our study: multiple linear regression (MLR), multivariate adaptive regression splines (MARS), *k*-nearest neighbors (*k*NN), support vector machines (SVM), random forests (RF), and boosted regression trees (BRT) [44]. We use the optimal parameters listed in Table 5.1 for both the BA and RMSD versions. The

experiments presented in this chapter for the six ML methods on complexes from PDBbind 2007 were conducted prior to the development of our proposed ensemble neural networks SFs, therefore we include them separately in this chapter. However, we expect the performance trends of BsN-Score and BgN-Score to be similar to that of BRT and RF SFs, respectively. The absolute performance of NN SFs may in fact be higher as we demonstrated in the previous chapter when the task was to predict binding affinities.

The six ML techniques are implemented in the following R language packages that we use [31]: the package *stats* readily available in R for MLR, *earth* for MARS [34], *kknn* for *k*NN [38], *e1071* for SVM [41], *randomForest* for RF [45], and *gbm* for BRT [48]. These methods benefit from some form of parameter tuning prior to their use in prediction. For example, the most important parameters in MARS are the number of terms (or basis functions) in the model, the *degree* of each term, and the *penalty* associated with adding new terms. Here we only tune the degree and penalty parameters and leave the final number of terms of MARS models to be automatically selected by the MARS algorithm implementation we use [34]. The *k*NN method has two parameters that require optimization: the neighborhood size k and the degree of Minkowski distance q [38]. For the SVM model, we have three parameters to optimize: the complexity constant C, the width of the ε-insensitive zone ε, and the width σ of the radial basis function that is used as a *kernel* [41]. RF algorithm has effectively only one important parameter *mtry* which determines the number of features to be randomly selected at each node split when growing the forest's trees [45]. The number of unpruned trees in the forest was fixed at 2000. BRT, on the other hand, has several parameters in addition to the most important two we tune: the *number of trees* and the *interaction depth* between the features [48]. The number of trees is optimized automatically using a cross-validation scheme internally implemented in the BRT algorithm [48]. The number of trees is tuned simultaneously with the interaction depth that controls their sizes. The shrinkage (or learning) rate of the BRT algorithm is set to 0.005 in all our experiments.

The values of the aforementioned parameters were selected so as to optimize the mean-squared errors on validation complexes sampled without replacement from the training set and independent

of the test data. Out-of-bag instances were used as validation complexes to select the optimal value for the RF parameter *mtry*. Out-of-bag (OOB) refers to complexes that are not sampled from the training set when bootstrap sets are drawn to fit individual trees in RF models. The parameter values for MARS, *k*NN, SVM, and BRT were optimized by performing a grid search over a suitable range in conjunction with 10-fold cross-validation over the training set *Pr*. The resulting optimal parameter values are provided in Table 5.1. This optimization was performed on the primary training (*Pr*) data for all 15 descriptors set. For every machine-learning method, we will be using these values to build ML SFs in the subsequent experiments.

Table 5.1: Optimal parameter values for MARS, *k*NN, SVM, RF, and BRT models for the docking task

Model	Parameter	Descriptor set														
		X	A	R	G	XA	XR	XG	AR	AG	RG	XAR	XAG	XRG	ARG	XARG
MARS	*Degree*	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	Penalty	2	6	5	6	7	2	6	7	6	5	6	7	6	5	6
*k*NN	*k*	15	13	14	16	9	19	17	19	18	17	18	19	17	18	19
	q	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
SVM	*C*	2	2	1	1	1	4	1	2	2	1	1	2	2	1	2
	ε	0.5	0	0.250	0.250	0.125	0.125	0.250	0.250	0.250	0.125	0.250	0.125	0.125	0.125	0.250
	σ	1	0.25	0.125	0.250	0.250	0.031	0.031	0.031	0.125	0.031	0.125	0.031	0.031	0.031	0.031
RF	*mtry*	3	18	8	7	31	5	8	10	16	17	14	20	21	25	35
BRT	*Interaction depth*	15	17	18	16	19	15	18	19	17	16	16	20	18	17	20
	Number of trees	1114	1523	1573	1208	1371	2113	1610	2950	2181	2303	2213	2590	2854	2921	2859

5.3.2 Ensemble deep neural networks evaluated on the PDBbind 2014 benchmark

We trained and tested the ensemble neural network SFs, BsN-Dock, BgN-Dock, BsN-Score, BgN-Score, and an updated (in-house) versions of RF-Score and X-Score on protein-ligand complexes sampled from PDBbind 2014 according the procedures described in Section 2.1. A boosted regression tree SF, BT-Dock, based on XGBoost algorithm was also constructed to substitute the neural network SFs for computationally expensive experiments that require fitting large number of models. Our neural network SFs and BT-Dock are trained and tested on complexes characterized by the multi-perspective descriptors derived using the proposed DDB platform from 16

different sources as described in Chapter 3. The parameters of these SFs were tuned in a consistent manner to optimize the mean-squared prediction errors on validation complexes sampled without replacement from the training set and independent of the test sets. Out-of-bag instances were used as validation complexes for BgN-Score while a ten-fold cross-validation was conducted for BsN-Score. The parameters that are tuned and their optimized values are described in Section 4.4.6. The Python language libraries TensorFlow, XGBoost, and Scikit-learn were used to construct BsN-Dock, BgN-Dock, BsN-Score, BgN-Score, BT-Dock, and our versions of RF-Score and X-Score.

5.4 Results and discussion

5.4.1 Evaluation of the docking power of scoring functions

In contrast to our work presented in the previous chapter for improving and examining scoring and ranking accuracies of different families of SFs, this chapter is devoted to enhancing and comparing SFs in terms of their docking powers. Docking power measures the ability of an SF to distinguish a promising binding mode from a less promising one. Typically, generated conformations are ranked in non-ascending order according to their predicted binding affinity (BA). Ligand poses that are very close to the experimentally-determined ones should be ranked high. Closeness is measured in terms of RMSD (in Å) from the true binding pose. Generally, in docking, a pose whose RMSD is within 2 Å from the true pose is considered a success or a hit.

In this chapter, we use comparison criteria similar to those used by Cheng et al. to compare the docking accuracies of sixteen popular conventional SFs. Doing so ensures fair comparison of ML SFs to those examined in that study in which each SF was assessed in terms of its ability to find the pose that is closest to the native one. More specifically, docking ability is expressed in terms of a success rate statistic S that accounts for the percentage of times an SF is able to find a pose whose RMSD is within a predefined cutoff value C by only considering the N topmost poses ranked by their predicted scores. Since success rates for various C (e.g., 0, 1, 2, and 3 Å) and N (e.g., 1, 2, 3, and 5) values are reported in this study, we use the notation S_C^N to distinguish between these

different statistics. For example, S_1^2 is the percentage of protein-ligand complexes whose either one of the two best scoring poses are within 1 Å from the true pose of a given complex. It should be noted that S_0^1 is the most stringent docking measure in which an SF is considered successful only if the best scoring pose is the native pose. By the same token and based on the *C* and *N* values listed earlier, the least strict docking performance statistic is S_3^5 in which an SF is considered successful if at least one of the five best scoring poses is within 3 Å from the true pose.

5.4.2 Docking-specific scoring functions vs. conventional approaches on a diverse test set

5.4.2.1 Docking performance on diverse protein families from PDBbind 2007

In this experiment, we compare the docking performance of the six ML SFs to the sixteen conventional approaches on the core test *Cr* of PDBbind 2007 that comprises thousands of protein-ligand complex conformations corresponding to 195 different native poses in 65 diverse protein families. As mentioned earlier, we conducted two experiments. In the first, BA values predicted using the conventional and ML SFs were used to rank poses in a non-ascending order for each complex in *Cr*. In the other experiment, RMSD-based ML models directly predicted RMSD values that are used to rank in non-descending order the poses for the given complex.

By examining the true RMSD values of the best *N* scoring ligands using the two prediction approaches, success rates of SFs are computed; these are shown in Figure 5.2. Panels (a) and (b) in the figure show the success rates S_1^1, S_2^1, and S_3^1 for all 22 SFs. The SFs, as in the other panels, are sorted in non-ascending order from the most stringent docking test statistic value to the least stringent one. In the top two panels, for example, success rates are ranked based on S_1^1, then S_2^1 in case of a tie in S_1^1, and finally S_3^1 if two or more SFs tie in S_2^1. In both BA- and RMSD-based scoring, we find that the 22 SFs vary significantly in their docking performance. The top three BA-based SFs, GOLD::ASP, DS::PLP1, and DrugScorePDB::PairSurf, have success rates of more than 60% in terms of S_1^1 measure. That is in comparison to the BA-based ML SFs, the best of which has an S_1^1 value barely exceeding 50% (Fig. 5.2(a)). On the other hand, the other six ML SFs that

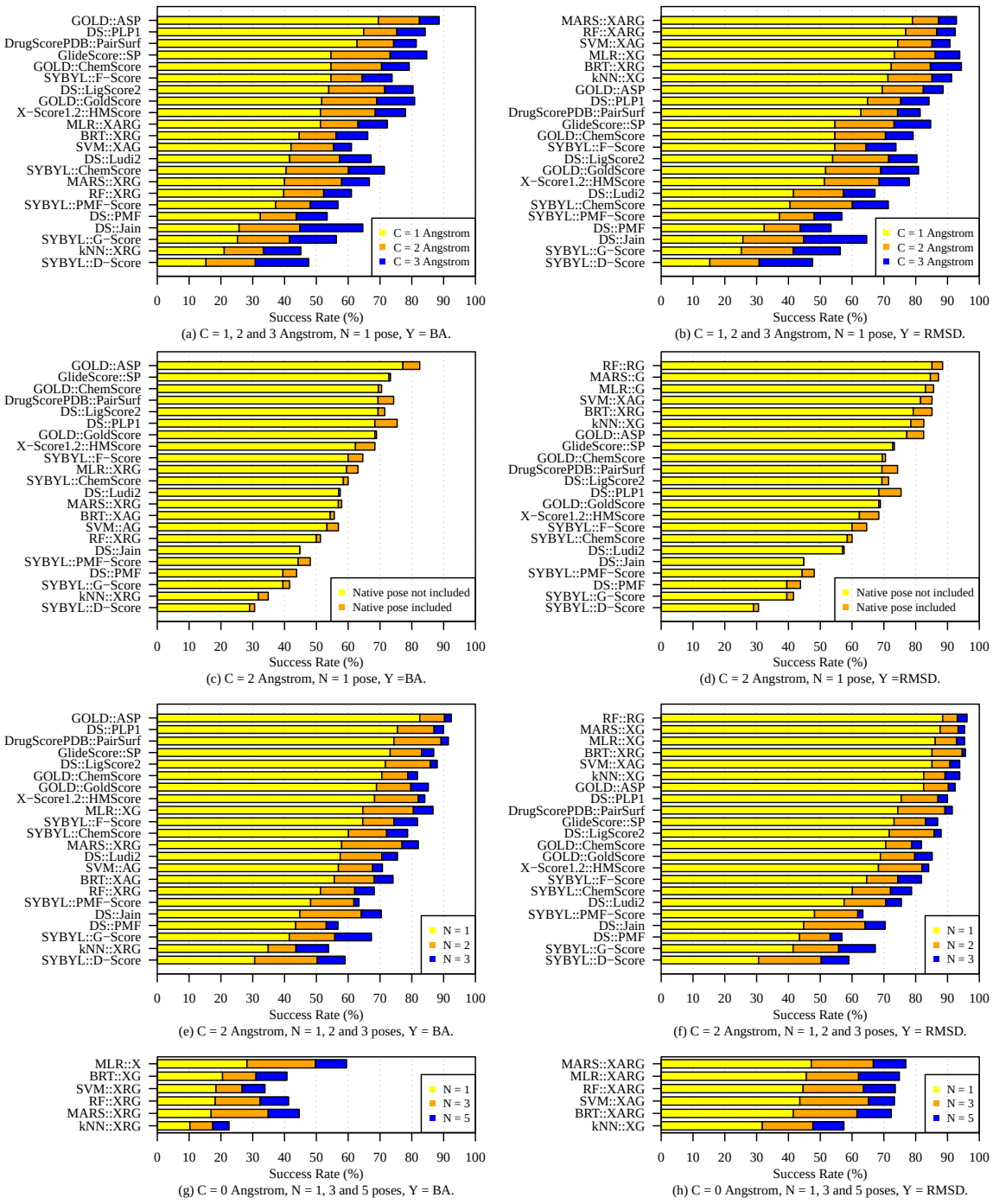


Figure 5.2: Success rates of scoring functions in identifying binding poses that are closest to native conformations of complexes in PDBbind 2007. The results show these rates by examining the top N scoring ligands that lie within an RMSD cut-off of C Å from their respective native poses. Panels on the left show success rates when BA-based scoring is used and the ones on the right show the same results when docking-specific SFs predicted *RMSD* values directly. Accuracies of conventional SFs are re-depicted on right panels for comparison convenience.

directly predict RMSD values achieve success rates of over 70% as shown in Figure 5.2(b). The top performing of these ML SFs, MARS::XARG, has a success rate of ∼80%. This is a significant improvement ($> 14\%$) over the best conventional SF, the empirical GOLD::ASP, whose S_1^1 value is ∼70%. Similar conclusions can also be made for the less stringent docking performance measures S_2^1 and S_3^1 in which the RMSD cut-off constraint is relaxed to 2 Å and 3 Å, respectively.

The success rates plotted in the top two panels (Figure 5.2 (a) and (b)) are reported when native poses are included in the decoy sets. Panels (c) and (d) of the same figure show the impact of removing the native poses on docking success rates of all SFs. It is clear that the performance of almost all SFs does not radically decrease by examining the difference in their S_2^1 statistics which ranges from 0 to ∼5%. This is due to the fact that some of the poses in the decoy sets are actually very close to the native ones. As a result, the impact of allowing native poses in the decoy sets is insignificant in most cases and therefore we include such poses in all other tests in the paper.

In reality, more than one pose is usually used from the outcomes of a docking run in the next stages of drug design for further experimentation. It is useful therefore to assess docking accuracy of SFs when more than one pose is considered (i.e., $N > 1$). Figure 5.2 (e) and (f) show the success rates of SFs when the RMSD values of the best 1, 2, and 3 scoring poses are examined. These rates correspond, respectively, to S_1^2, S_2^2, and S_3^2. The plots show a significant boost in performance for almost all SFs. By comparing S_1^2 to S_3^2, we observe a jump in accuracy from 82% to 92% for GOLD::ASP and from 87% to 96% for RF::RG that models RMSD values directly. Such results signify the importance of examining an ensemble of top scoring poses because there is a very good chance it contains relevant conformations and hence good drug candidates.

Upon developing RMSD-based ML scoring models, we noticed excellent improvement over their binding-affinity-based counterparts as shown in Figure 5.2. We conducted an experiment to investigate whether they will maintain a similar level of accuracy when ML SFs are examined for their ability to pinpoint the native poses from their respective 100-pose decoy sets. The bottom two panels, (g) and (h), plot the success rates in terms of S_0^1, S_0^3, and S_0^5 for the six ML SFs. By examining the five best scoring poses, we notice that the top BA-based SF, MLR::X, was able to

distinguish native binding poses in ∼60% of the 195 decoy sets whereas the top RMSD-based SF, MARS::XARG, achieved a success rate of S_0^5 = 77% on the same protein-ligand complexes. It should be noted that both sets of ML SFs, the BA- and RMSD-based, were trained and tested on completely disjoint test sets. Therefore, this gap in performance is largely due to the explicit modeling of RMSD values and the corresponding abundance of training data which includes information from both native and computationally-generated poses.

5.4.2.2 Docking performance on diverse protein families from PDBbind 2014

In this section, the docking-specific SFs based on boosted and bagged deep neural networks, BsN-Dock and BgN-Dock, are compared against their scoring-specific counterparts BsN-Score and BgN-Score as well as the ML SF RF-Score and the empirical SFs X-Score, ChemPLP, ChemScore, and GlideScore-SP on PDBbind 2014. The docking-specific NN SFs are RMSD-based while the remaining models are trained on complexes labeled with binding affinity data. As in the previous section, all nine SFs were evaluated based on their ability to find the native or near-native binding pose for each of the 195 protein-ligand complex in the core test set *Cr*. In addition, six of these SFs were further tested using the stratified cross-validation (*CV*) and leave-clusters-out (*LCO*) approaches to assess their generalization abilities in identifying the correct binding poses for known and novel protein targets.

Figure 5.3 shows the docking accuracy of the RMSD-based SFs BsN-Dock and BgN-Dock as well as the BA-based models BsN-Score, BgN-Score, RF-Score, and X-Score in finding native or near-native poses for ligands docked to known protein targets in the core (*Cr*) and cross-validation (*CV*) test sets and novel proteins in the leave-clusters-out (*LCO*) test sets. More specifically, for each SF and training-test sampling protocol (*Cr, CV*, and *LCO*), we report the SF's success rates (S_C^N%) in identifying the binding pose that is within C Å from the native conformations by considering the top N scoring poses after ordering the hundred poses for each protein-ligand complex based on the score predicted by the SF. We denote these success rates by S_1^1, S_1^2, and S_1^3 in Figure 5.3 and we order SFs based on their average S_1^1 value across the 195 PLCs in the core test set

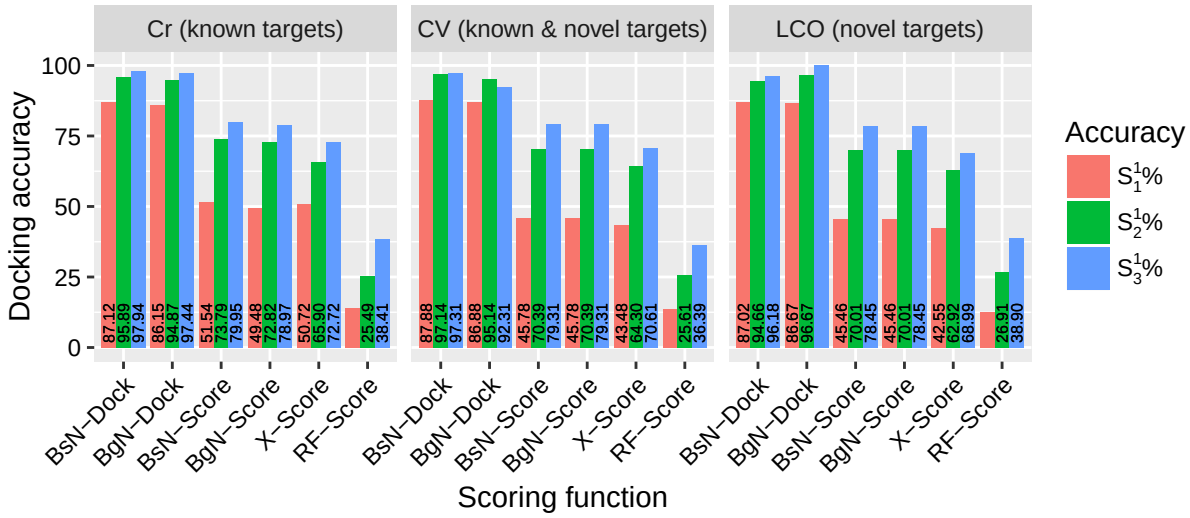


Figure 5.3: The docking accuracy of docking-specific (proposed) and binding-affinity-based (conventional) scoring functions when evaluated on test complexes with proteins that are either fully represented (*Cr*), partially represented (*CV*), or not represented (*LCO*) in the SFs' training data. The docking accuracy is expressed in terms of the success rates (S_1^1, S_2^1, and S_3^1) of SFs in identifying binding poses that are closest to native ones for protein-ligand complexes derived from the refined set of PDBbind 2014.

(*Cr*). The figure clearly shows superior docking performance for the RMSD-based SFs BsN-Dock and BgN-Dock on known and native targets alike. BsN-Dock and BgN-Dock are able to correctly find the native or near-native ligand pose for 87% of the test protein targets by only examining their top-scoring poses (i.e., by allowing them to output only one pose for each PLC). It should be noted that there are 90 poses on average for each protein ligand complex in the test set that are more than 1 Å RMSD away from the actual native conformation which for this test ($C = 1$) considered to be non-native (or decoy) poses. Compare this to the BA-based SFs RF-Score and X-Score whose success rates are less than 45%. This gap in performance, 42% in absolute points, is due to three main reasons. First is the superior ensemble deep learning models used to train BsN-Dock and BgN-Dock SFs as opposed to Random Forests (RF-Score) and linear-regression (X-Score). Second, BsN-Dock and BgN-Dock are trained on large number of native and computationally generated poses and optimized to directly reduce the RMSD between generated poses and the native ones. Third, BsN-Dock and BgN-Dock capture more important protein-ligand interactions through a comprehensive and diverse set of multi-perspective descriptors. RF-Score

and X-Score, on the other hand, are only trained on native protein-ligand complexes characterized by smaller subset of descriptors and optimized to reduce the error between actual and estimated binding affinity data. BsN-Score and BgN-Score use the same underlying deep learning-based algorithms and multi-perspective descriptors employed by BsN-Dock and BgN-Dock which contribute to their improved performance in comparison to RF-Score and X-Score. Their substantial lag behind BsN-Dock and BgN-Dock, however, is attributed to the use of native protein-ligand complexes with BA labels as training data instead of both native and computationally-generated poses labeled with RMSD values on which BsN-Dock and BgN-Dock are trained. Therefore, enriching the original complexes in PDBbind 2014 with computationally generated poses and replacing their binding affinity labels with our RMSD values boosted the docking performance of our ensemble deep learning SFs by at least 25 absolute percentage points in terms of S_1^1 success rate (BsN-Dock = 87% vs. BsN-Score = 51%).

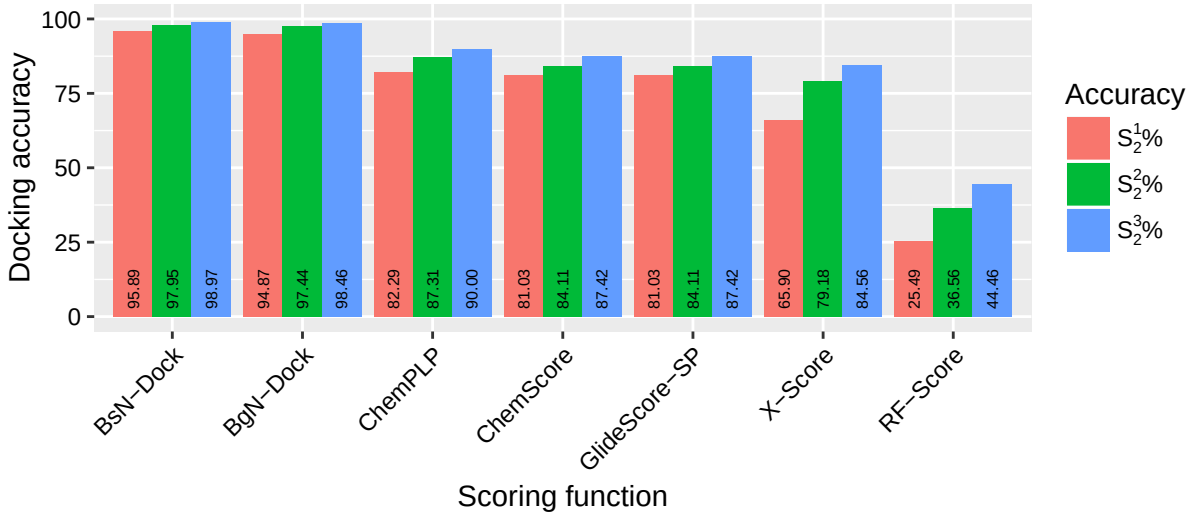


Figure 5.4: The docking accuracy of docking-specific and best binding-affinity-based (conventional) scoring functions on native and decoy conformations of ligands docked to diverse proteins from the core test set of PDBbind 2014. The docking accuracy is expressed in terms of the success rates (S_2^1, S_2^2, and S_2^3) of SFs in identifying binding poses that are closest to the native conformation for each protein-ligand complex in the test set.

BsN-Dock and BgN-Dock are also compared against the empirical SFs ChemPLP and ChemScore used in the commercial software GOLD and GlideScore-SP employed in the other popular

commercial molecular modeling suite Schrödinger. These three SFs were the top performing predictors of binding poses according to a recent comparative study by Li et al. [25]. In that comparative study, the docking accuracies of ChemPLP, ChemScore, and GlideScore-SP, as well as 18 other popular SFs, were tested on the core test set of PDBbind 2014 which we also use to evaluate our models. Their performance was not reported for cross-validation and leave-clusters-out tests and therefore we do not show them for all SFs in Figure 5.4. The figure shows the docking accuracy in terms of the success rate of identifying the poses that are within 2 Å from the native poses by only examining the top 1, 2, and 3 scoring poses out of a 100 total conformations for each protein-ligand complex. BsN-Dock and BgN-Dock are at least 13 absolute percentage points more accurate than ChemPLP and 30 percentage points ahead of X-Score in terms of S_2^1 success rate. This gap in performance confirms the superiority of BsN-Dock and BgN-Dock even against the best approaches used in commercial tools to solve real-world problems.

5.4.3 Docking-specific scoring functions vs. conventional approaches on homogeneous test sets

5.4.3.1 Docking performance on four protein families from PDBbind 2007

In the previous section, performance of 6 ML and 16 conventional SFs were assessed on the diverse test set *Cr* of PDBbind 2007. The core set consists of more than sixty different protein families each of which is related to a subset of protein families in *Pr*. That is, while the training and test set complexes were different (at least for all the ML SFs), proteins present in the core test set were also present in the training set, albeit bound to different ligands. A much more stringent test of SFs is their evaluation on a completely new protein, i.e., when test set complexes all feature a given protein—test set is homogeneous—and training set complexes do not feature that protein. To address this issue, four homogeneous test sets were constructed corresponding to the four most frequently occurring proteins in our data: HIV protease (112 complexes), trypsin (73), carbonic anhydrase (44), and thrombin (38). Each of these protein-specific test sets was formed by extracting complexes containing the protein from *Cr* (one cluster or three complexes) and *Pr* (remaining

complexes). For each test set, we retrained BRT, RF, SVM, *k*NN, MARS, and MLR models on the non-test-set complexes of *Pr*. Figure 5.5 shows the docking performance of resultant BA and RMSD-based ML scoring models on the four protein families. The plots clearly show that success rates of SFs are dependent on the protein family under investigation. It is easier for some SFs to distinguish good poses for HIV protease and thrombin than for carbonic anhydrase. The best performing SFs on HIV protease and thrombin complexes, MLR::XRG and MLR::XG, respectively, achieve success rates of over 95% in terms of S_1^3 as shown in panels (b) and (n), whereas no SF exceeded 65% in success rate in case of carbonic anhydrase as demonstrated in panels (i) and (j). Finding the native poses is even more challenging for all SFs, although we can notice that RMSD-based SFs outperform those models that rank poses using predicted BA. The exception to this is the SF MLR::XAR whose performance exceeds all RMSD-based ML models in terms of the success rate in reproducing native poses as illustrated in panels (c) and (d).

The results also indicate that multivariate linear regression models (MLR), which are basically empirical SFs, are the most accurate across the four families, whereas ensemble learning models, RF and BRT, unlike their good performance in Figure 5.2, appear to be inferior compared to simpler models in Figure 5.5. This can be attributed to the high rigidity of linear models compared to ensemble approaches. In other words, linear models are not as sensitive as ensemble techniques to the presence or absence of certain protein family in the data on which they are trained. On the other hand, RF- and BRT-based SFs are more flexible and adaptive to their training data that in some cases fail to generalize well enough to completely different test proteins as seen in Figure 5.5. In practice, however, it has been observed that more than 92% of today's drug targets are similar to known proteins in PDB [113], an archive of high quality complexes from which our training and test compounds originated. Therefore, if the goal of a docking run is to identify the most stable poses, it is important to consider sophisticated SFs (such as RF and BRT) calibrated with training sets containing some known binders to the target of interest. Simpler models, such as MLR and MARS, tend to be more accurate when docking to novel proteins that are not present in training data.

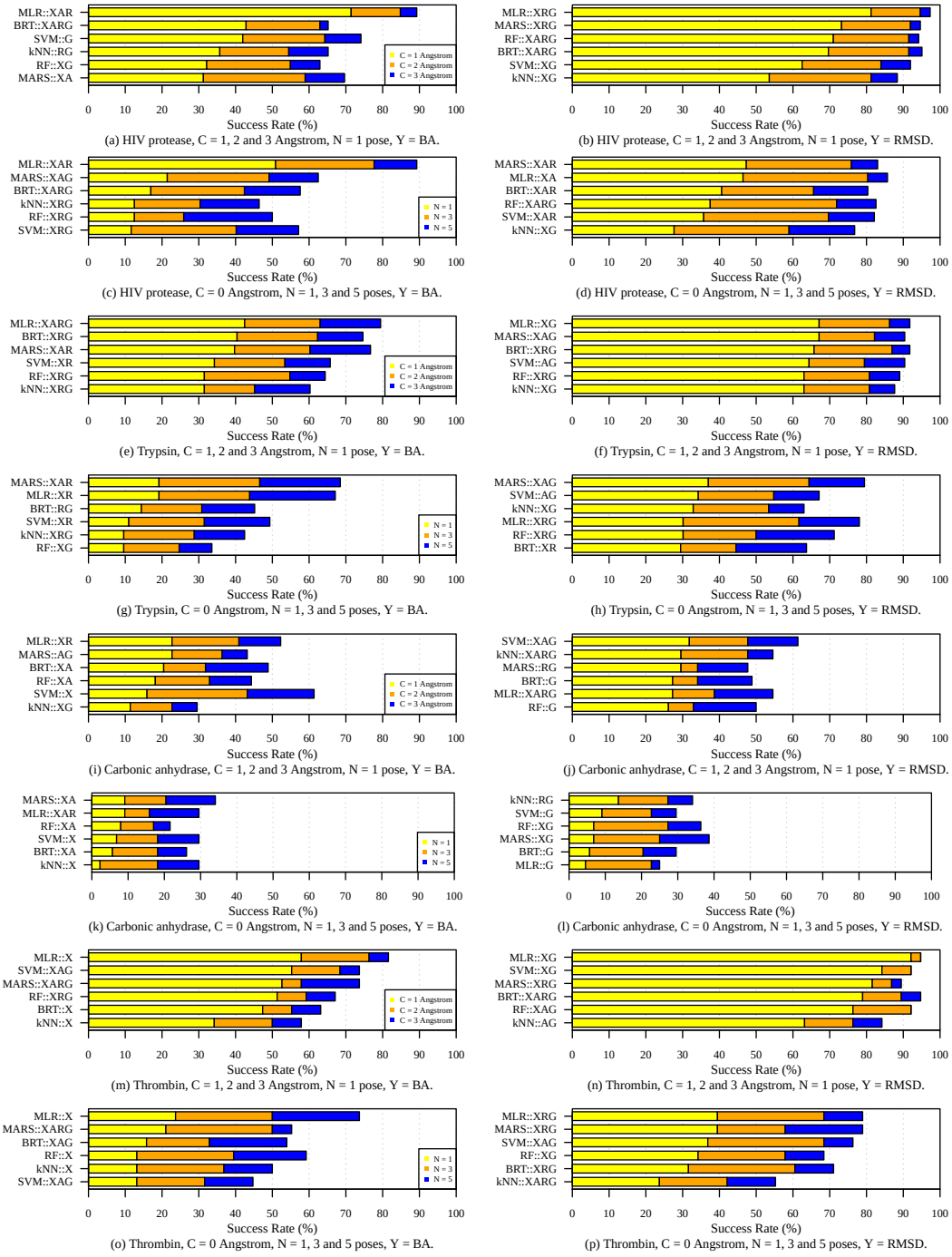


Figure 5.5: Success rates of scoring function in identifying binding poses that are closest to native conformations observed in four protein families in PDBbind 2007: HIV protease (a-d), trypsin (e-h), carbonic anhydrase (i-l), and thrombin (m-p). The results show these rates by examining the top N scoring ligands that lie within an RMSD cut-off of C Å from their respective native poses. Panels on the left show success rates when binding-affinity-based scoring is used and the ones on the right are for RMSD-based SFs.

Sophisticated ML algorithms are not the only critical element in building a capable SF. Features to which they are fitted also play an important role as can be seen in Figure 5.5. By comparing the right panels to the ones on the left, we can notice that X-Score features (X) are almost always present in BA-based SFs while those provided by GOLD (G) are used more to model RMSD explicitly. This implies that X-Score features are more accurate than other feature sets in predicting BA, while GOLD features are the best for estimating RMSD and hence poses close to the native one.

5.4.3.2 Docking performance on four protein families from PDBbind 2014

In this section, we present the docking accuracy of six SFs in identifying the correct binding pose of four protein families from the 2014 version of PDBbind. The four families are the same targets we considered in the previous section. However, the number of complexes formed by these families are larger in PDBbind 2014: 262 HIV protease, 170 carbonic anhydrase (CAH), 98 trypsin, and 79 thrombin complexes. The six SFs we evaluate here are the ensemble deep learning-based models BsN-Dock, BgN-Dock, BsN-Score, BgN-Score, and two popular SFs from the literature RF-Score and X-Score which are based on Random Forests and linear regression, respectively. We build two sets of the ensemble deep neural networks models. BsN-Dock and BgN-Dock are trained on the 2014 refined set complexes with native and computationally generated poses characterized by DDB's multi-perspective descriptors and labeled using RMSD values from native poses. The second set includes BsN-Score and BgN-Score which are trained on the native poses of the PDBbind 2014 refined set complexes without adding any computationally-generated decoy poses. Therefore, BsN-Score and BgN-Score's training complexes are labeled using their measured binding affinity values just like the training data for RF-Score and X-Score. Similar to BsN-Dock and BgN-Dock, BsN-Score and BgN-Score training complexes are also characterized by DDB's multi-perspective. The SF RF-Score uses 36 geometric descriptors based on counts of different protein-ligand atom types, while X-Score employs six physiochemical descriptors.

The docking performance of the six SFs are shown in Figure 5.6 for the four protein families.

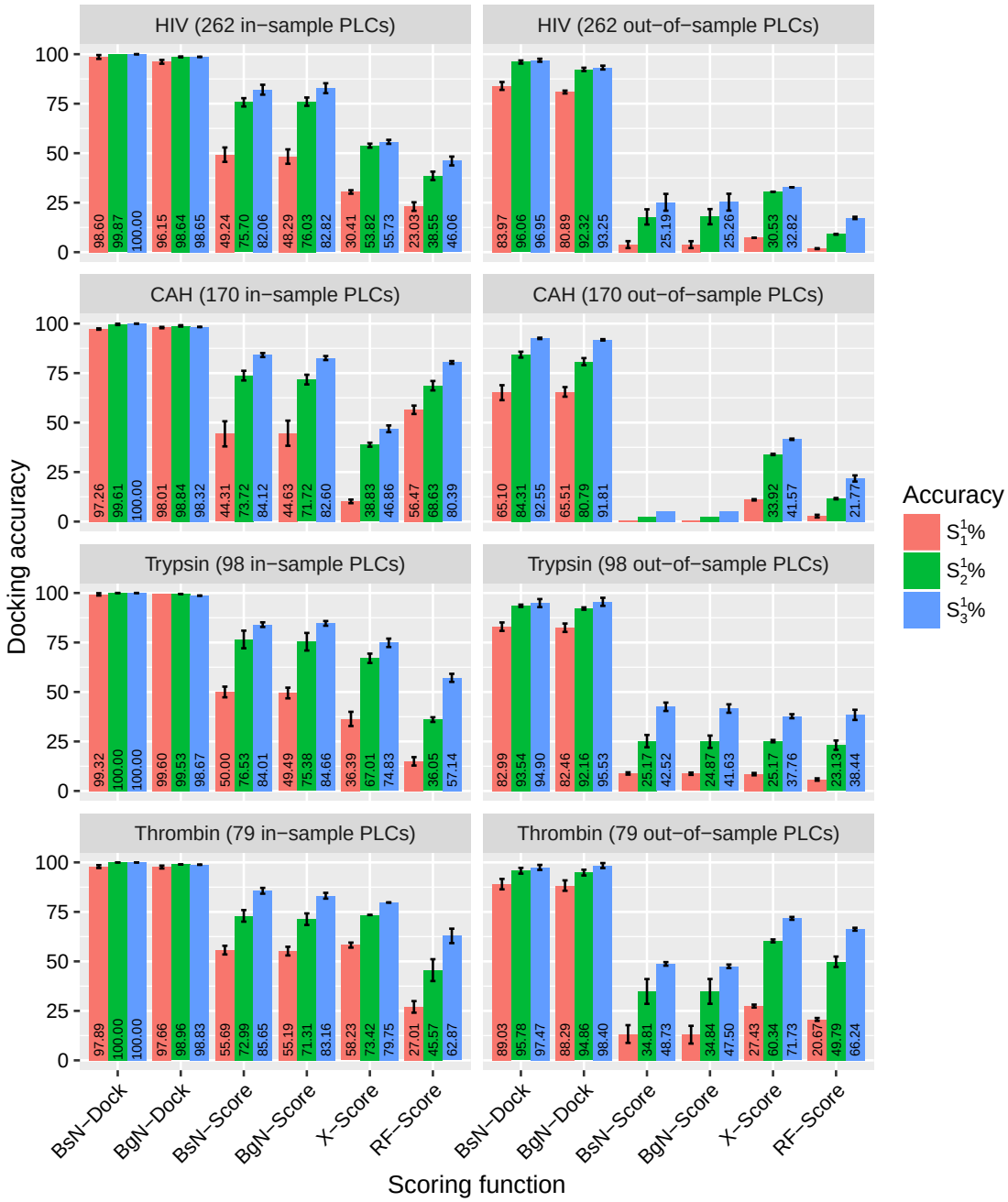


Figure 5.6: Success rates of docking-specific (proposed) and binding-affinity-based (conventional) scoring functions in identifying binding poses that are closest to the native conformations observed in four protein families from PDBbind 2014: HIV protease, carbonic anhydrase (CAH), trypsin, and thrombin. The results show these rates by examining the top $N = 1$ scoring ligands that lie within an RMSD cut-off of $C \in \{1, 2, 3\}$ Å from their respective native poses. Panels on the left show in-sample success rates and the ones on the right show the out-of-sample docking accuracies.

The left panels summarize the in-sample docking accuracies in terms of S_1^1, S_2^1, and S_3^1 success rates. BsN-Dock and BgN-Dock perfectly recognize the correct poses for the four protein families they encountered in their training data. The four other scoring functions failed in more than fifty percent of the cases (S_1^1) even though all the protein-ligand complexes are part of their training data. This implies the difficulty of the task of finding the relationship between BA and RMSD values for BA-based SFs. RMSD-based SFs perform very well in ranking the ligand poses correctly whether the four test protein families are part of their training data or out-of-sample as shown in the right panels of Figure 5.6. We notice a slight drop in performance for BsN-Dock and BgN-Dock in this test when neither the PLCs nor even the proteins are part of their training data. On the other hand, the other four SFs fail completely in identifying the native or near native poses even when considering poses that are 3 Å away as "near-native" (S_3^1). This experiment, yet again, highlights the potential of our RMSD-based approach in identifying binding poses correctly even for novel protein targets that they have never seen before.

5.4.4 Performance of docking-specific scoring functions on novel targets

5.4.4.1 Simulating novel targets using PDBbind 2007

The primary training-core test set pair (*Pr*, *Cr*) sampled from PDBbind 2007 PLCs is a useful benchmark when the aim is to evaluate the performance of SFs on targets that have some degree of sequence similarity with at least one protein present in the complexes of the training set. This is typically the case since, as it was mentioned earlier, 92% of drug targets are similar to known proteins [113]. When the goal is to assess SFs in the context of novel protein targets, however, the training-test set pair (*Pr*, *Cr*) is not that suitable because of the partial overlap in protein families between *Pr* and *Cr*. We considered this issue to some extent in the previous section, where we investigated the docking accuracy of SFs on four different protein-specific test sets after training them on complexes that did not have the protein under consideration. This resulted in a drop in performance of all SFs, especially, in the case of carbonic anhydrase as a target. However, even if

there are no common proteins between training and test set complexes, different proteins at their binding sites may have sequence and structural similarities, which influence docking results. To more rigorously and systematically assess the performance of BA and RMSD-based ML SFs on novel targets, we performed a separate set of experiments in which we limited BLAST sequence similarity between the binding sites of proteins present in the training and test set complexes. Sequence similarity was used to construct the core test set and it was also noted by Ballester and Mitchell as being relevant to testing the efficacy of SFs on a novel target [29].

Specifically, for each similarity cut-off value S = 30%, 40%, 50%, ..., 100%, we constructed 100 different independent 100-complex test and T-complex training set pairs. Two versions were created out of these training and test set pairs. The first version uses BA as a response variable that SFs will be fitted to, predict, and employ to assess poses. The response variable of the other version is the RMSD value of true poses (RMSD = 0 Å) and computer generated decoys (with 10 Å $\leq$ RMSD $>$ 0 Å) of each original protein-ligand complex in every training and test dataset pair. A total of 20 poses per complex have been used in this second version. Then, we trained BA and RMSD scoring models (MLR, MARS, kNN, SVM, RF, and BRT) using XARG features on the training set and evaluated them on the corresponding test set, and determined their average performance over the 100 training-test-set pairs to obtain robust results. Since SF docking performance depends upon both similarity cut-off and training set size and since training set size is constrained by similarity cut-off (a larger S means a larger feasible T), we investigated different ranges of S (30% to 100%, 50% to 100%, and 70% to 100%) and for each range we set T close to the largest feasible value for the smallest S value in that range. Each test and training set pair was constructed as follows. We randomly sampled a test set of 100 protein-ligand complexes without replacement from all complexes at our disposal: 1105 in Pr + 195 in Cr = 1300 complexes. The remaining 1200 complexes were randomly scanned until T different complexes were found that had protein binding site similarity of S% or less with the protein binding sites of all complexes in the test set — if less than T such complexes were found, then the process was repeated with a new 100-complex test set.

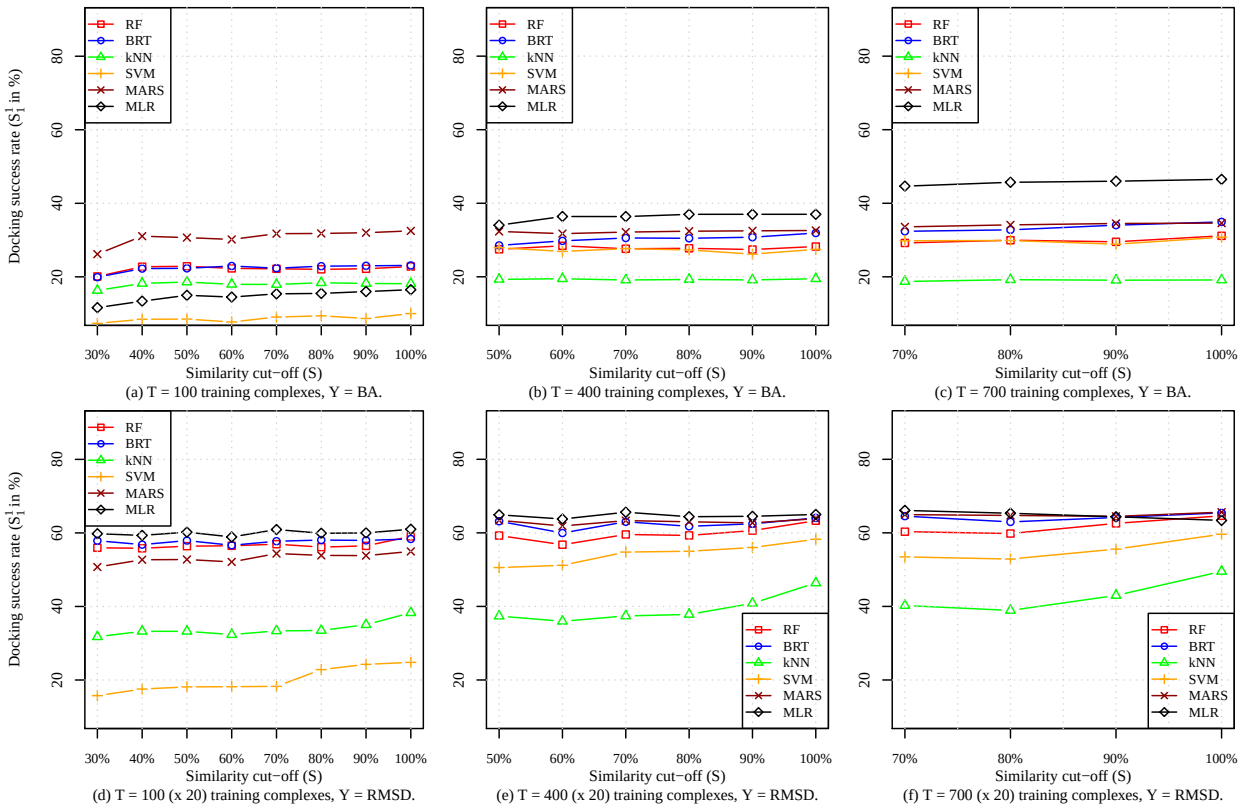


Figure 5.7: Docking accuracy of docking-specific (proposed) and BA-based (conventional) scoring models as a function of BLAST sequence similarity cutoff between binding sites of proteins in training and test complexes. The docking accuracy is expressed in terms of S_1^1 success rate obtained by examining the top $N = 1$ scoring ligand that lies within an RMSD cut-off of $C = 1$ Å from its respective native pose. In panels (a)-(c), a single (native) pose is used per training complex, whereas in panels (d)-(f) 20 randomly-selected poses are used per training complex. Training and test complexes are sampled from the refined set of PDBbind 2007.

The performance of the six scoring models in terms of their S_1^1 docking accuracy is depicted in Figure 5.7 for various similarity cut-offs and training set sizes. The plots in each column ((a) and (d), (b) and (e), and (c) and (f) of Figure 5.7) show docking power results for similarity cut-offs of 30% to 100%, 50% to 100%, and 70% to 100% for which T = 100, 400, and 700 complexes is the largest training set size feasible for S = 30%, 50%, and 70%, respectively. The results in these plots are consistent with those obtained for the four protein families presented in the previous section and illustrated in Figure 5.5. More specifically, we notice that simpler models such as MLR::XARG and MARS::XARG perform the best across almost all values of similarity cut-offs (S = 30%, 50%, or 70%-100%), training set sizes (T = 100, 400, or 700 complexes), and

response variables (Y = BA or RMSD). This is mainly due to their rigidity. The performance of such models do not suffer as much as the more flexible ML SFs when their training and test proteins have low sequence similarity. On the other hand, the SFs based on MLR and MARS are also less responsive to increasing the similarity between protein families in the training and test sets. Unlike the other four non-linear ML SFs, we can observe that the performance curves of MLR and MARS are flat and do not seem to react to having more and more similar training and test proteins. This observation is more clear in the bottom row of plots of Figure 5.7 where the training set sizes are large enough (i.e., 2000 ligand poses or more). Plot (f) shows that the RMSD-based SFs RF and BRT are catching up with MLR and MARS and can eventually surpass them in terms of performance as training set sizes become larger. Similar to RF and BRT, the other non-linear RMSD SFs, namely kNN and SVM, have the sharpest increase in docking performance as similarity cut-off S increases. However, unlike the ensemble SFs RF and BRT, kNN and SVM SFs are the least reliable models when ligand poses need to be scored for novel targets.

To summarize, imposing a sequence similarity cut-off between the binding sites of proteins in training and test set complexes has an expected adverse impact on the accuracy of all scoring models. However, increasing the number of training complexes helps improve accuracy for all similarity cut-offs as we will show in more detail in the next section. Scoring functions based on MLR and MARS have the best accuracy when training set sizes are small which is typically the case when the response variable is binding affinity. The other generally-competitive ML models are RF and BRT whose accuracies surpass all other SFs when evaluated on targets that have some sequence similarity with their training proteins.

5.4.4.2 Simulating novel targets using PDBbind 2014

Similar to the objective of the previous experiment described in Section 5.4.4.1, the aim of this simulation is to compare the docking accuracy of the proposed docking-specific SFs to the generic binding affinity-based approaches on novel protein targets. The scoring functions we consider here are trained and evaluated on data sets from the larger and more recent version of PDBbind.

From the 3446 PLCs of PDBbind 2014, we randomly sample a test set of 100 complexes each of which is associated with 100 decoy and native ligand poses. From the remaining 3346 complexes, we randomly sample 3000 training PLCs where each of which must not have a binding pocket with a BLAST similarity of more than $S\%$ with any of the 100 test set proteins. If we are unable to find such complexes in the 3346 PLCs, we repeat this random sampling process several times until the similarity restriction is met for every similarity cut off value S, where $S \in \{30\%, 50\%, 70\%, 90\%, 100\%\}$. We then build four SFs using the 3000 training complexes and test their docking accuracy on the 100 novel targets whose binding sites have less than $S\%$ BLAST similarity with the binding pockets of the training complexes. This procedure is repeated 50 times for every similarity level S to obtain robust summary values of docking success rates S_1^1 and S_1^2 as shown in Figure 5.8. The four scoring functions tested here are: (i) BT-Dock, the docking-specific boosted decision trees model fitted to native and computationally-generated poses characterized by DDB's multi-perspective descriptors and labeled using RMSD values; (ii) BT-Score, the generic, scoring-specific counterpart of BT-Dock that is also based on a boosted ensemble of decision trees fitted to complexes characterized by DDB's multi-perspective descriptors. However, BT-Score is only trained on the native poses of these complexes labeled with BA-data. The decision to replace deep neural networks with decision trees as base learners for the boosted ensemble model is based on computational considerations. This experiment involves training large number of models to obtain statistically significant results. Since deep neural networks require hefty computational resources, training hundreds of them is impractical for this test and the models in the following sections. We believe BT-Dock and BT-Score resemble BsN-Dock and BsN-Score, respectively, at least with respect to their performance trends across different similarity values. (iii) RF-Score, the Random Forests based scoring functions trained on complexes with native poses characterized by the 36 geometric features and labeled using BA-data. (iv) X-Score, the empirical SF based on a linear regression model fitted to native complexes with binding affinity labels and 6 physiochemical descriptors.

The docking performance trends in Figure 5.8 across the various levels of similarity cutoffs

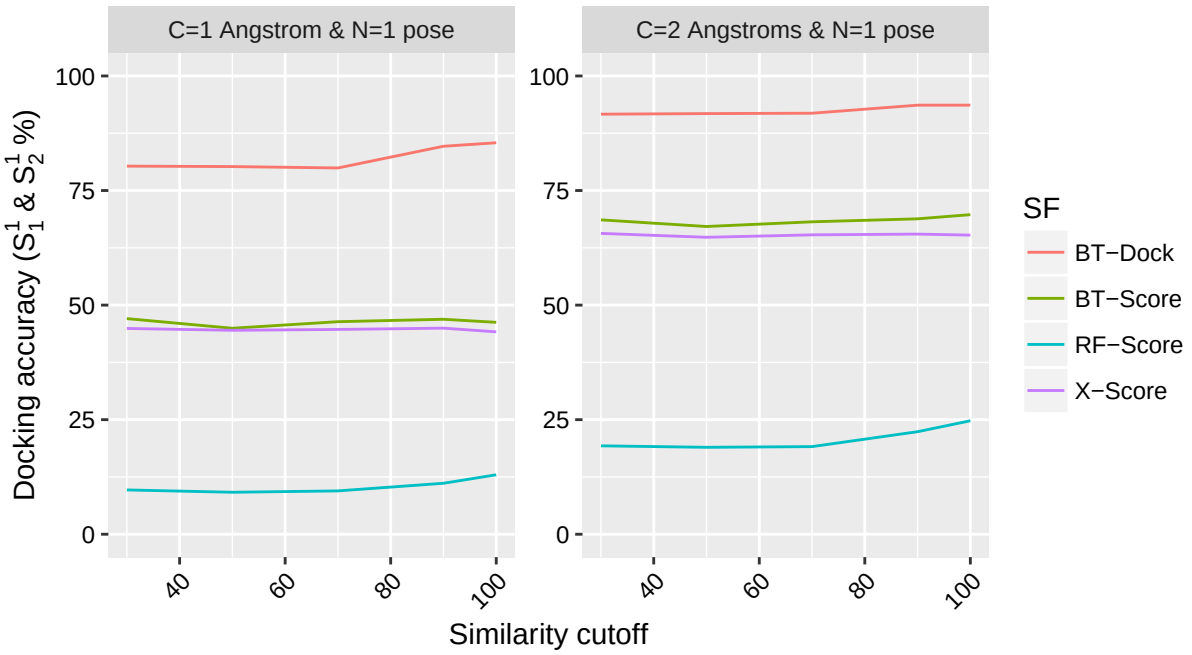


Figure 5.8: Docking accuracy of docking-specific (proposed) and BA-based (conventional) scoring models as a function of BLAST sequence similarity cutoff between binding sites of proteins in training and test complexes. The docking accuracy is expressed in terms of S_1^1 and S_2^1 success rates obtained by examining the top $N = 1$ scoring ligand that lies within an RMSD cut-off of $C \in \{1,2\}$ Å from its respective native pose. The docking-specific SF BT-Dock is trained on 100 computer-generated poses for each of its 3000 native protein-ligand complexes. Training and test complexes are sampled from the refined set of PDBbind 2014.

resemble those obtained using PDBbind 2007 and depicted in Figure 5.7. BT-Dock achieves substantially better ranking accuracy of ligand poses for each novel test protein. The accuracy of BT-Dock gradually becomes better as the binding pocket similarity between its training and test proteins moves past the 70% cutoff. On the other hand, the S_1^1 success rate of BA-based SFs, BT-Score, X-Score, and RF-Score is less than 50% and does not show any sign of improvement even when tested on familiar targets (with BLAST similarity $S \geq 90\%$). It should be noted that the only difference between the RMSD-based SF BT-Dock and its sister BA-based model BT-Score is the use of computational ligand poses during training BT-Dock whose dependent variable is the RMSD distances of ligand poses from their respective novel conformations. This difference in modeling is responsible for attaining the gap in performance shown in Figure 5.8.

5.4.5 Impact of the number of training protein-ligand complexes on the docking accuracy

5.4.5.1 Simulating increasing training set size using PDBbind 2007

An important factor influencing the accuracy of ML SFs is the size of the training dataset. In the case of BA-based ML SFs, training dataset size can be increased by training on a larger set of protein-ligand complexes with known binding affinity values. In the case of RMSD-based SFs, on the other hand, training dataset size can be increased not only by considering a large number of protein-ligand complexes in the training set, but also by using a larger number of computationally-generated ligand poses per complex since each pose provides a new training record because it corresponds to a different combination of features and/or RMSD value. Unlike experimental binding affinity values, which have inherent noise and require additional resources to obtain, RMSD from the native conformation for a new ligand pose is computationally determined and is accurate.

We carried out three different experiments to determine: (i) the response of BA-based ML SFs to increasing number of training protein-ligand complexes, (ii) the response of RMSD-based ML SFs to increasing number of training protein-ligand complexes while the number of poses for each complex is fixed at 50, and (iii) the response of RMSD-based ML SFs to increasing number of computationally-generated poses while the number of protein-ligand complexes is fixed at 1105. In the first two experiments, we built 6 ML SFs, each of which was trained on a randomly sampled $x\%$ of the 1105 protein-ligand complexes in PDBbind 2007 *Pr* dataset, where $x = 10, 20, \ldots, 100$. The dependent variable in the first experiment is binding affinity ($Y = \text{BA}$), and the performance of these BA-based ML SFs is shown in Figure 5.9(a) and partly in Figure 5.9(d) (MLR::XARG). The set of RMSD values from the native pose is used as a dependent variable for ML SFs trained in the second experiment ($Y = \text{RMSD}$). For a given value of x, the number of conformations is fixed at 50 ligand poses for each protein-ligand complex. The docking accuracy of these RMSD-based ML models is shown in Figure 5.9(b). In the third experiment, all 1105 complexes in *Pr* were used for training the RMSD-based ML SFs (i.e., $Y = \text{RMSD}$) with x randomly sampled poses considered per complex, where $x = 2, 6, 10, \ldots, 50$; results for this are reported in Figure 5.9(c) and partly

in Figure 5.9(d) (MARS::XARG). In all three experiments, results reported are the average of 50 random runs in order to ensure all complexes and a variety of poses are equally represented. All training and test complexes in these experiments are characterized by the XARG (=$X \cup A \cup R \cup G$) features.

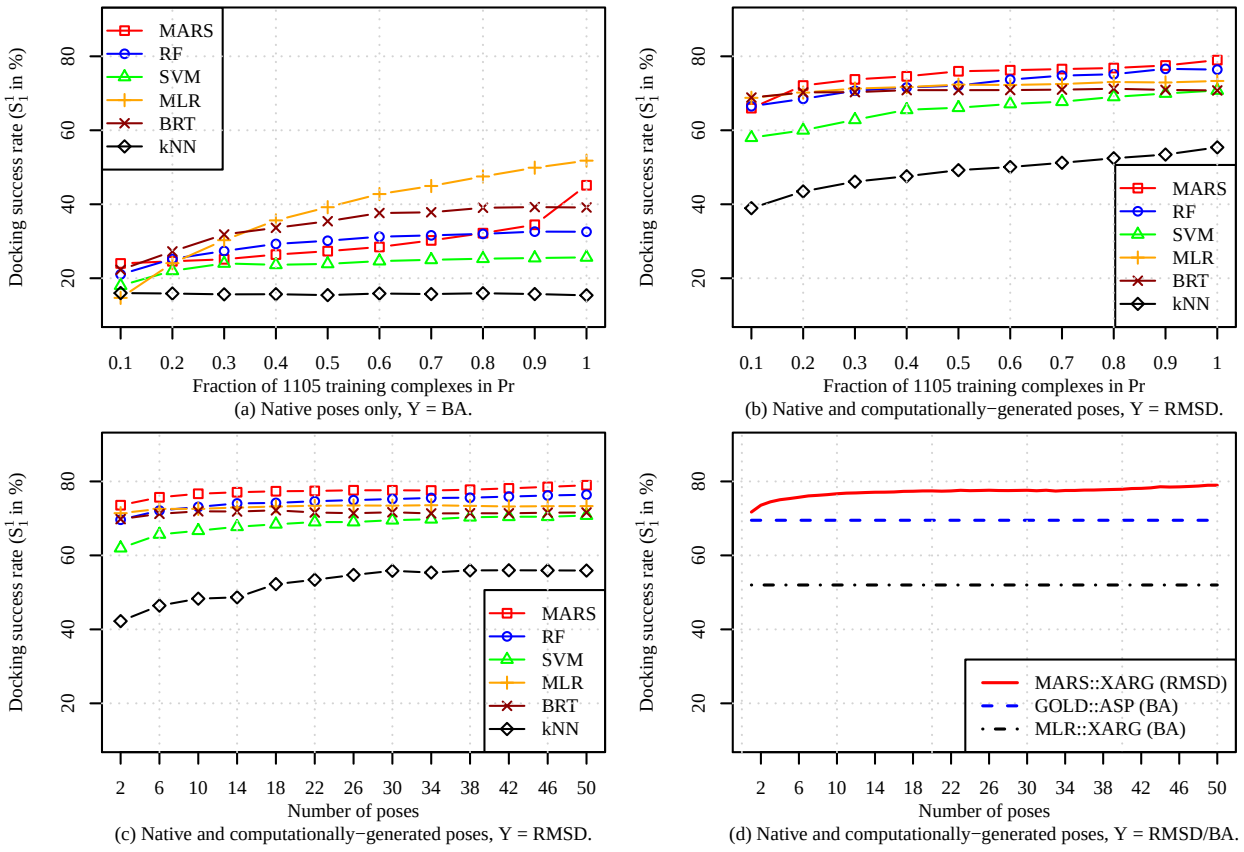


Figure 5.9: Dependence of docking accuracy of docking-specific and binding-affinity-based scoring models on training set size when training complexes are selected randomly (without replacement) from the refined set of PDBbind 2007 and the models are tested on the out-of-sample core (*Cr*) set. The size of the training data was increased by including more protein-ligand complexes ((a) and (b)) or more computationally-generated poses for all complexes ((c) and (d)).

From Figure 5.9(a), it is evident that increasing training dataset size has a positive impact on docking accuracy (measured in terms of S_1^1 success rate), although it is most appreciable in the case of MLR::XARG and MARS::XARG, two of the simpler models, MLR being linear and MARS being piecewise linear. The performance of the other models, which are all highly nonlinear, seems to saturate at 60% of the maximum training dataset size used. The performance of all six models is

quite modest, with MLR::XARG being the only one with docking success rate (slightly) in excess of 50%. The explanation for these results is that binding affinity is not a very good response variable to learn for the docking problem because the models are trained only on native poses (for which binding affinity data is available) although they need to be able to distinguish between native and non-native poses during testing. This means that the training data is not particularly well suited for the task for which these models are used. An additional reason is that experimental binding affinity data, though useful, is inherently noisy. The flexible highly nonlinear models, RF, BRT, SVM, and *k*NN, are susceptible to this noise because the training dataset (arising only from native poses) is not particularly relevant to the test scenario (consisting of both native and non-native poses). Therefore, the more rigid MLR and MARS models fair better in this case.

When RMSD is used as the response variable, the training set consists of data from both native and non-native poses and hence is more relevant to the test scenario and the RMSD values, being computationally determined, are also accurate. Consequently, docking accuracy of all SFs improves dramatically compared to their BA-based counterparts as can be observed by comparing Figure 5.9(a) to Figure 5.9(b) and (c). We also notice that all SFs respond favorably to increasing training set size by either considering more training complexes (Figure 5.9(b)) or more computationally-generated training poses (Figure 5.9(c)). Even for the smallest training set sizes in Figure 5.9(b) and (c), we notice that the docking accuracy of most RMSD-based SFs is about 70% or more, which is far better than the roughly 50% success rate for the largest training set size for the best BA-based SF MLR::XARG.

In Figure 5.9(d), we compare the top performing RMSD SF, MARS::XARG, to the best BA-based SFs, GOLD::ASP and MLR::XARG, to show how docking performance can be improved by just increasing the number of computationally-generated poses, an important feature that RMSD-based SFs possess but which is lacking in their BA-based conventional counterparts. To increase the performance of these BA-based SFs to a comparable level, thousands of protein-ligand complexes with high-quality experimentally-determined binding affinity data need to be collected. Such a requirement is too expensive to meet in practice. Furthermore, RMSD-based SFs with

the same training complexes will still likely outperform BA-based SFs.

5.4.5.2 Simulating increasing training set size using PDBbind 2014

The number of protein ligand complexes in PDBbind has increased from 1300 in 2007 to 3446 in 2014. In this section, we use the updated dataset to re-evaluate the capacity of BA- and RMSD-based SFs in utilizing an increasing number of complexes to improve their performance. Therefore, we follow a very similar experimental setup to the one we describe in the previous subsection (5.4.5.1). This section however, focuses on the docking-specific (RMSD-based) SF BT-Dock and the three BA-based models BT-Score, RF-Score, and X-Score. We conduct two sets of experiments on the 2014 version PDBbind similar to those performed using the 2007 release. In the first, we train the four SFs on training sets of x protein-ligand complexes randomly sampled from the 3446-complex refined set of PDBbind 2014 where $x \in \{100, 680, 1260, 1840, 2420, 3000\}$. The SFs are then tested on 195 PLCs randomly sampled from the remaining $(3446 - x)$ complexes in the *refined* set. The training and test complexes are characterized using the 2714 proposed multi-perspective descriptors for BT-Dock and BT-Score. RF-Score extracts 36 geometrical features of atomic pair-wise counts while X-Score relies on its 6 physiochemical descriptors to characterize the complexes. The three generic SFs BT-Score, RF-Score and X-Score are trained on the native poses of the x training complexes whose labels are the measured binding affinity values. The proposed docking-specific SF BT-Dock is trained on 100 native and computationally-generated poses for each training complex and uses their root-mean-square distances from the native conformations as a dependent variable. For each training set size x, the four SFs are tested on their ability in identifying the native or near-native pose of the test protein-ligand complexes. Each test protein-ligand complex is associated with 100 poses, the vast majority (90%) of them are decoys. The success rates (in terms of S_1^1 and S_1^2) are then reported for each SF and averaged over 50 runs for every training set size x to obtain reliable results. The average success rates for BT-Dock, BT-Score, RF-Score, and X-Score are shown in Figure 5.10

The plot shows that the accuracy of the docking-specific SF BT-Dock is superior to the three

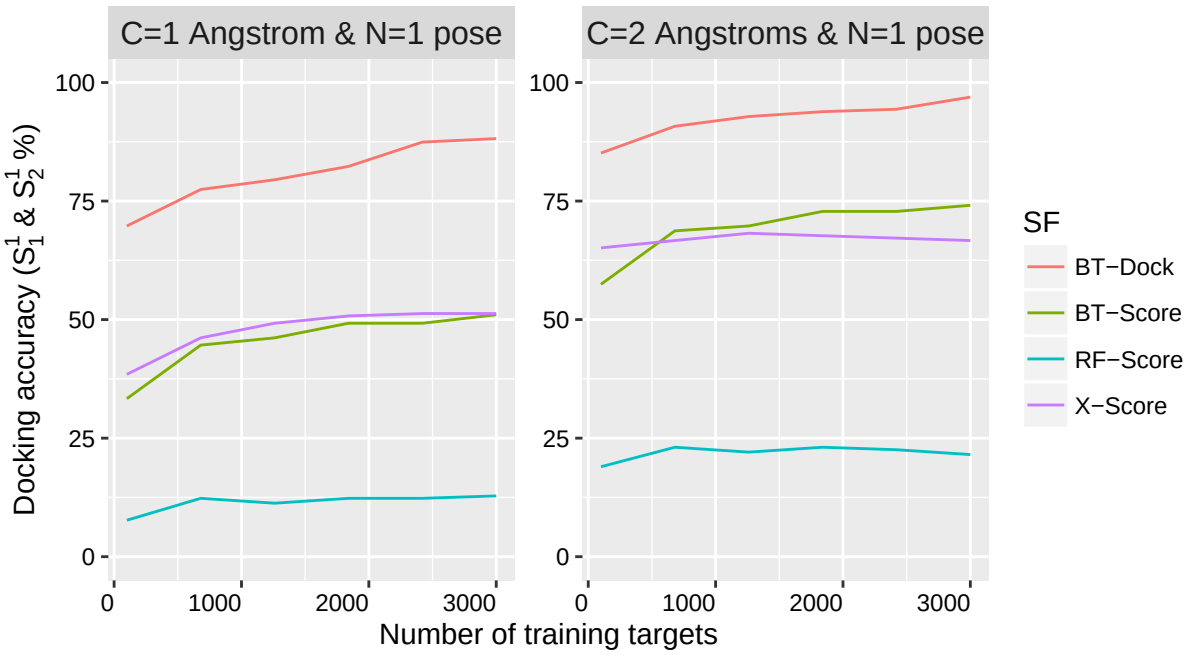


Figure 5.10: Dependence of docking accuracy of docking-specific and binding-affinity-based scoring models on training set size when training complexes are selected randomly (without replacement) from the refined set of PDBbind 2014 and the models are tested on the out-of-sample core (*Cr*) set. The docking accuracy is expressed in terms of S_1^1 and S_2^1 success rates obtained by examining the top $N = 1$ scoring ligand that lies within an RMSD cut-off of $C \in \{1,2\}$ Å from its respective native pose.

BA-based SFs even for 100 training PLCs. Increasing the training data consistently improves BT-Dock and BT-Score followed by X-Score. RF-Score appears to be less responsive to increasing its training data size. We attribute this mainly to the type of features that RF-Score uses to characterize the protein-ligand molecules. When we fitted a Random Forests model (the algorithm that the SF RF-Score uses) to the 2714 multi-perspective descriptors generated using the proposed Descriptor Data Bank (DDB) we observed an improvement trend on par with BT-Score.

In the second experiment, we test how the docking accuracy of the best performing SF BT-Dock reacts to different sizes of the computationally generated poses for each training complex. To answer this question, we trained BT-Dock on a fixed number of 3000 complexes with increasing number of ligand poses x, where $x \in \{10,20,40,60,80,100\}$. The three BA-based SFs BT-Score, RF-Score, and X-Score are trained on a fixed size of 3000 complexes with their native conformations. We use the BA-based SFs in this experiment as baseline models to compare against the

proposed BT-Dock model. The types of descriptors, the labels of PLCs, and the test complexes are the same for every SF as the ones described for the previous experiment. The docking success rates S_1^1 and S_2^1 for the four SFs are summarized in Figure 5.11.

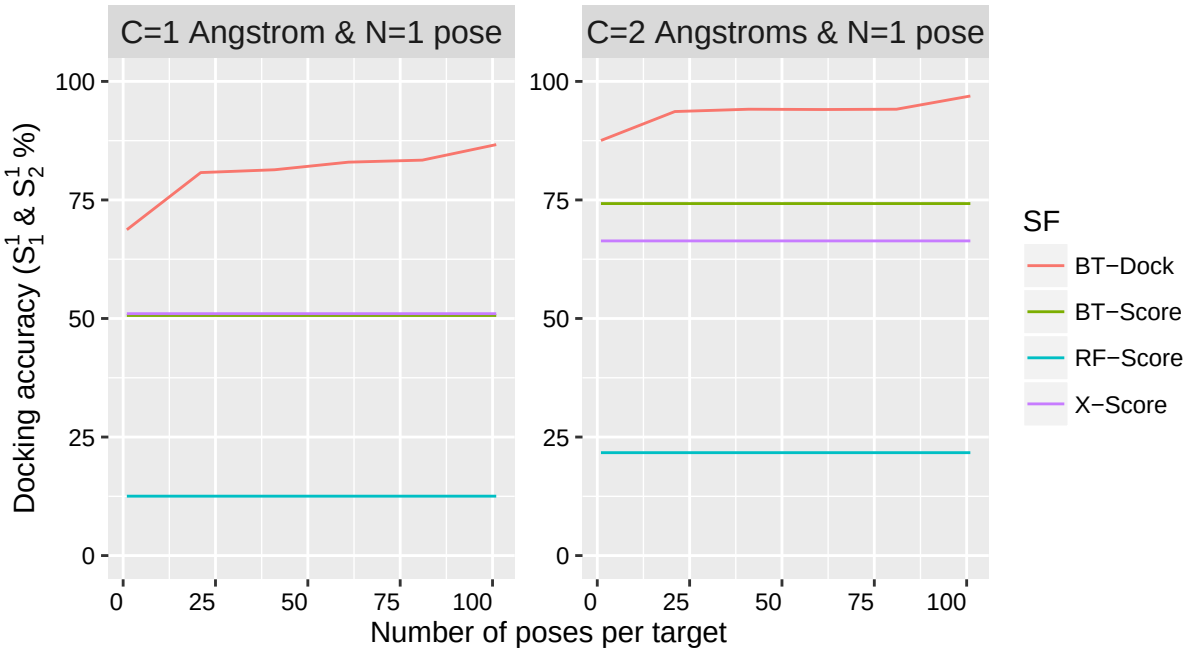


Figure 5.11: Dependence of docking accuracy of the docking-specific model BT-Dock on the number of training poses generated for each native conformation of 3000 protein-ligand complexes selected randomly (without replacement) from the refined set of PDBbind 2014. The three binding-affinity-based SFs BT-Score, RF-Score, and X-Score are only trained on the native conformations. The docking accuracy is expressed in terms of S_1^1 and S_2^1 success rates obtained by examining the top $N = 1$ scoring ligand that lies within an RMSD cut-off of $C \in \{1,2\}$ Å from its respective native pose in the core test set.

The accuracy of BT-Dock is substantially benefiting from generating more computational poses for each training protein-ligand complex. Enriching the performance of an SF without using more native complexes is a very desirable outcome since solving the native structure of a protein-ligand complex and determining its binding affinity value is an expensive task. Therefore, we believe virtual enrichment of training data should be pursued when possible for applications in which experimental data is difficult to obtain.

5.4.6 Impact of the type and number of descriptors on the docking accuracy

5.4.6.1 Type and number of raw descriptors characterizing PDBbind 2007 complexes

The binding pose of a protein-ligand complex depends on many physiochemical interaction factors that are too complex to be accurately captured by any one approach. Therefore, we perform two different experiments to investigate how utilizing different types of features from different scoring tools, X-Score, AffiScore, RF-Score, and GOLD, and considering an increasing number of features affects the performance of the various ML models. In the first experiment, the 6 ML models were trained on the 2007 *Pr* dataset characterized by all 15 combinations of X, A, R, and G feature types and tested on the corresponding PDBbind 2007 core test *Cr* characterized by the same features. Table 5.2 reports the S_1^1 docking success rate for three groups of ML SFs. The first set (Table 5.2 top part) of 90 (6 methods $\times$ 15 feature combinations) BA-based SFs is trained on 1105 *Pr* complexes. The second set (Table 5.2 middle part) of 90 RMSD-based SFs is again trained on the 1105 *Pr* complexes with one randomly sampled pose from 50 poses generated per complex. Therefore, the training set size for these first two groups of SFs is identical and consists of 1105 training records, with the only difference being the response variable that they are trained for, BA in the first case and RMSD in the second case. The final (Table 5.2 bottom part) 90 RMSD-based SFs are trained on 1105 *Pr* complexes, with 50 poses per complex, so that its training set size is $1105 \times 50 = 55,250$ records.

We notice that the S_1^1 value of almost all models improves by considering more than one type of feature rather than just X, A, R, or G features alone. The table also shows that RMSD SFs are substantially more accurate than their BA counterparts for each feature type and ML method. By comparing the 180 RMSD SFs with the corresponding 90 BA SFs across all feature types and ML models, we find that the former are, on average, almost twice as accurate as the BA approaches (50.64% and 57.61% vs. 27.95% — see Table 5.2 rightmost column). In terms of feature types, we note that the most accurate SFs always include X-Score and GOLD features. SFs that are

Table 5.2: Docking success rate S_1^1 (in %) of ML SFs complexes characterized by different descriptors

Y (T)	Model	Descriptor set															
		X	A	R	G	XA	XR	XG	AR	AG	RG	XAR	XAG	XRG	ARG	XARG	Average
BA (1105)	MARS	28.72	20.00	5.13	18.46	34.36	36.92	28.21	21.03	16.41	8.21	37.95	28.72	**40.00**	8.21	36.41	24.58
	RF	23.08	23.08	10.77	24.10	30.52	30.77	37.44	21.54	30.00	25.65	31.03	34.12	**39.75**	25.90	32.31	28.00
	SVM	26.67	30.26	4.62	19.49	30.77	23.08	29.23	20.51	41.54	30.26	26.15	**42.05**	37.44	38.97	41.54	29.51
	MLR	43.08	18.97	9.74	28.21	33.33	46.67	47.18	21.03	40.00	33.85	46.15	49.23	49.74	40.00	**51.28**	***37.23***
	BRT	23.85	25.39	9.24	39.23	30.52	30.26	42.31	22.82	37.69	33.85	33.85	44.36	**44.62**	36.67	43.08	33.18
	*k*NN	17.95	11.28	8.72	16.41	9.23	18.46	15.90	11.28	15.38	17.95	15.38	17.44	**21.03**	14.36	17.44	15.21
	Average	27.22	21.50	8.04	24.32	28.12	31.03	33.38	19.71	30.17	24.96	31.75	35.99	***38.76***	27.35	37.01	27.95
RMSD (1105)	MARS	45.64	42.05	26.15	72.31	54.87	52.31	73.85	33.85	70.26	71.28	52.82	70.26	72.82	67.18	**75.90**	**58.77**
	RF	32.72	32.31	13.85	67.39	39.80	42.26	70.36	32.10	64.82	68.10	41.95	64.82	**72.00**	64.51	67.18	51.61
	SVM	29.74	28.72	6.15	66.15	37.95	30.77	**70.77**	32.31	61.54	56.41	45.64	64.10	62.05	58.97	63.08	47.62
	MLR	44.62	35.90	6.15	70.26	58.46	54.36	72.31	40.51	70.26	65.64	56.92	**75.90**	70.26	67.69	69.23	57.23
	BRT	38.77	34.87	8.10	70.98	52.00	41.13	**74.77**	35.59	67.70	69.02	48.92	71.39	72.51	65.33	67.49	54.57
	*k*NN	31.79	24.10	3.08	55.38	30.77	17.44	**62.56**	15.90	40.00	38.97	22.05	44.62	45.13	37.95	41.03	34.05
	Average	37.21	32.99	10.58	67.08	45.64	39.71	**70.77**	31.71	62.43	61.57	44.72	65.18	65.79	60.27	63.98	50.64
RMSD (1105x50)	MARS	44.10	31.79	5.13	72.82	62.56	49.74	75.38	34.87	73.33	70.77	64.10	76.92	72.31	72.31	**78.97**	59.01
	RF	39.39	48.10	22.46	70.77	61.95	54.77	72.41	50.97	73.64	75.18	63.79	76.00	75.49	75.28	**76.90**	**62.47**
	SVM	36.41	43.08	10.26	66.15	54.87	43.08	70.77	43.08	71.79	65.13	57.95	**74.36**	72.31	69.74	70.77	56.65
	MLR	45.64	36.41	6.15	70.26	56.92	52.31	**73.34**	37.95	71.79	71.28	57.44	72.82	71.79	70.77	73.33	57.88
	BRT	46.15	36.92	13.33	71.59	54.36	54.87	71.59	42.56	70.77	70.77	56.92	70.26	**72.31**	71.28	71.28	58.33
	*k*NN	36.41	46.15	23.59	61.03	51.28	41.03	**71.28**	45.13	60.00	53.33	49.23	61.54	60.51	53.33	55.90	51.32
	Average	41.35	40.41	13.49	68.77	57.01	49.30	**72.46**	42.43	70.22	67.69	58.24	72.05	70.68	69.23	70.63	57.61

fitted to the individual X and G features only are more accurate than their A and R counterparts whether they are BA or RMSD models. By averaging the performance of all ML models across all feature types, we see that the simple linear approach MLR outperforms other more sophisticated ML SFs that are trained to predict binding affinity. MARS outperforms all other RMSD SFs that are trained on the same number of training records (1105) as their BA counterparts. The lower part of the table shows that the ensemble SF RF that predicts the binding pose directly has the highest docking accuracy (62.47%), on the average, across 15 different feature types and MARS::XARG has the highest docking accuracy (78.97%) overall. Comparing the two versions of RMSD SFs in the middle and lower portions of the table, we notice that the largest gainers from increasing training set size are the most nonlinear ML techniques (RF, BRT, SVM and *k*NN). The results of Table 5.2 are useful in assessing the relative benefit of different types of features for the various ML models.

A pertinent issue when considering a variety of features is how well different SF models exploit an increasing number of features. The features we consider are the X, A, G, and a larger set of

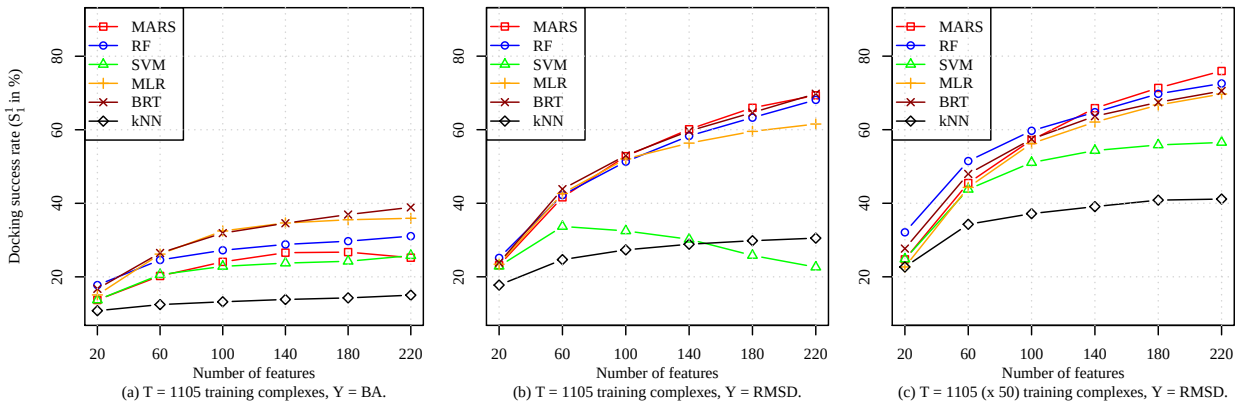


Figure 5.12: Dependence of docking accuracy of docking-specific (proposed) and binding-affinity-based (conventional) scoring models on the number of features. The features are drawn randomly (without replacement) from a pool of X, A, R, and G-type features and used to characterize the training and out-of-sample core *Cr* test set complexes of PDBbind 2007. In panels (a) and (b), a single pose (native pose in (a) and randomly-selected pose in (b)) is used per training complex, whereas in panel (c) 50 randomly-selected poses are used per training complex.

geometrical features than the R feature set available from the RF-Score tool. Recall from the Compound Characterization subsection that RF-Score counts the number of occurrences of 36 different protein-ligand atom pairs within a distance of 12 Å. In order to have more features of this kind for this experiment, we produce 36 such counts for five contiguous distance intervals of 4 Å each: (0 Å, 4 Å], (4 Å, 8 Å], ..., (16 Å, 20 Å]. This provides us 6 X, 30 A, 14 G, and $(36 \times 5 =)$ 180 geometrical features or a total of 220 features. We randomly select (without replacement) x features from this pool, where $x = 20, 60, 100, \ldots, 220$, and use them to characterize the *Pr* dataset, which we then use to train the six ML models. These models are subsequently tested on the core (*Cr*) dataset characterized by the same features. This process is repeated 100 times to obtain robust average S_1^1 statistics, which are plotted in Figure 5.12.

The performance of the BA SFs is depicted in Figure 5.12(a) whereas panels (b) and (c) of the same figure show the docking success rates for the RMSD versions of the scoring models. In order to fairly compare the docking performance of BA and RMSD SFs as number of features increase, we fixed their training set sizes to 1105 complexes as shown in Figure 5.12 (a) and (b). We also show in Figure 5.12(c) the effect of increasing number of features on the docking performance of RMSD SFs when trained on all *Pr* complexes, with 50 poses per complex. The

plots clearly indicate that RMSD SFs benefit the most from characterizing complexes with more descriptors. This is the case regardless of the number of records used to train RMSD SFs (compare plots (b) and (c) in Figure 5.12). The only exception is the RMSD SF based on SVM where it appears to overfit the 1105 training records when they are characterized by more than 60 features. This ML scoring function, however, performs better when trained on larger number of records and shows a slight increase in performance as more features are included in building the model. Other RMSD SFs such as RF, BRT, MLR, and MARS have much sharper slopes than SVM and *k*NN. Compare these SFs to their BA counterparts in Figure 5.12(a) where most of them show none to little improvement as the number of features increases due to overfitting. Not only are they resilient to overfitting, most RMSD SFs improve dramatically by extracting more relevant features. Adding more features may result in highest gains in performance when more training complexes are included as was discussed in the previous subsection.

5.4.6.2 Number of descriptor sets characterizing PDBbind 2014 complexes

Public biochemical databases such as PDB [81], PubChem [71], PDBbind [22], Zinc [68], and others are the inspiring successful projects behind the creation of Descriptor Data Bank (DDB). These public databases have had a tremendous impact on advancing virtual screening and other molecular modeling applications. We believe our descriptor database will be a very important complement in this space of public biochemical data and will significantly contribute to the advancement of machine-learning scoring functions. We showed in Section 5.4.5 the substantial improvement to the docking accuracy of SFs when trained on more complexes obtained from the public database PDBbind 2007 and 2014. In this section, we quantitatively show how SFs benefit from training on public complexes characterized by an increasing number of descriptor sets using the 2014 version of PDBbind. In other words, we demonstrate how the performance of SFs enhance as new perspectives of protein-ligand interactions are added to public databases overtime.

To investigate the effect of increasing the number of descriptor sets, we randomly select a maximum of 100 combinations of x descriptor sets from all possible (16 choose x) combinations of the

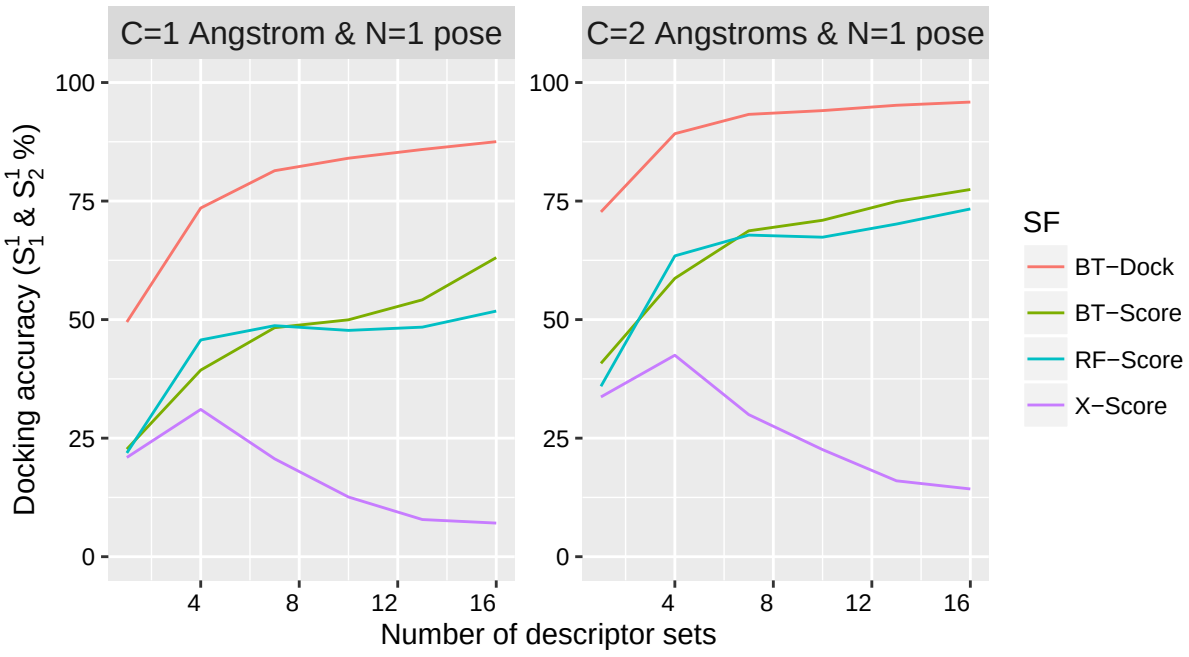


Figure 5.13: Dependence of docking accuracy of docking-specific (proposed) and binding-affinity-based (conventional) scoring models on the number of descriptor (feature) sets. The sets are drawn randomly (without replacement) from a pool of 16 feature types in Descriptor Data Bank (DDB) and used to characterize the training and out-of-sample core *Cr* test set complexes of PDBbind 2014. The docking accuracy is expressed in terms of S_1^1 and S_2^1 success rates obtained by examining the top $N = 1$ scoring ligand that lies within an RMSD cut-off of $C \in \{1,2\}$ Å from its respective native pose in the core test set.

16 DDB types listed in Table 3.1, where $x \in \{1,4,7,10,13,16\}$, and use them to characterize the training set complexes, which we then use to train the RMSD-based SF BT-Dock and its BA-based counterpart BT-Score. We also fit a Random Forests and linear regression models to these descriptors which we call RF-Score and X-Score for notational convenience. We choose these models in particular due to their use in constructing the SFs RF-Score and X-Score from the literature which were originally fitted to different descriptor sets of 36 geometrical features and 6 physiochemical terms, respectively. The four SFs, BT-Dock, BT-Score, RF-Score and X-Score, are subsequently tested on the *Cr* dataset characterized by the same descriptors on which they were trained. For each number of descriptor sets, x, the performance of the 100 SFs are averaged to obtain robust overall docking success rates S_1^1 and S_2^1, which are plotted in Figure 5.13. Similar to the case of increasing training data in terms of new PLCs, the performance of the ensemble SFs BT-Dock, and

RF-Score are clearly benefiting from increasing the number of protein-ligand interaction models. The level of improvement for the three SFs with increasing the number of descriptor sets is on par or better with that observed when increasing their number of training complexes. The linear regression model suffers serious overfitting problems when trained on large and diverse sets of features as can be observed in the plot. A similar behavior was also observed for the MLR model in Figure 5.12(b) when it was fitted to an increasing number of descriptors.

For the proposed RMSD-based SF BT-Dock in particular, it is clear from the plots in Figure 5.13 and 5.10 that the number of different models of protein-ligand interactions are as important as the number of its training complexes in improving the quality of the SF's prediction. Based on these results, we believe that the public database of protein-ligand interactions we are proposing in this work will be of high value to ML SFs and just as important to their accuracy as the repositories of molecular structures and experimental assay data.

5.5 Conclusion

We found that docking-specific ML models trained to explicitly predict RMSD values significantly outperform all conventional SFs in almost all testing scenarios. The estimated RMSD values of such models have a correlation coefficient of 0.825 on average with the true RMSD values. On the other hand, predicted binding affinities have a correlation of as low as -0.3 with the measured RMSD values. This difference in correlation explains the wide gap in docking performance between the top SFs of the two approaches. Our docking-specific SF based on boosted deep neural networks correctly identified the correct ligand binding poses for 95% of the protein targets in the core test set of PDBbind 2014. This is in comparison to 82% success rate obtained by the empirical SF GOLD::ChemPLP. On the 2007 release of PDBbind, the empirical SF GOLD::ASP, which is the best conventional model, achieved a success rate of 70% in identifying a pose that lies within 1 Å from the native pose of 195 different complexes. On other hand, our top RMSD-based SF, MARS::XARG, has a success rate of $\sim$80% on the same test set, which represents a signifi-

cant improvement in docking performance[1]. When using non-DDB descriptors for PDBbind 2007 complexes, the linear ML SF, MLR::XARG, and its nonlinear extension, MARS::XARG, showed relatively better accuracy than Random Forests and SVM when the target is a protein not present in the training dataset used to build the scoring model. Ensemble SFs fitted to the same complexes and descriptors, however, may prove more reliable when there is some similarity between training set proteins and the target protein. Our RMSD-based, ensemble deep learning and decision tree SFs, on the other hand, outperformed all other approaches in docking accuracy for all testing scenarios when fitted to complexes characterized by large number of multi-perspective descriptors generated using the proposed Descriptor Data Bank platform. We also observed steady gains in the performance of RMSD-based ML SFs as the training set size and number of features were increased by considering more descriptors and protein-ligand complexes and/or more computationally-generated ligand poses for each complex.

[1]Experiments on the 2007 version of PDBbind were conducted prior to the development of the proposed SFs based on deep neural networks. Therefore, they were not compared against the 16 conventional and 6 ML (RF, BRT, SVM, KNN, MARS, and MLR) scoring functions on PDBbind 2007.

CHAPTER 6

SCREENING-SPECIFIC SCORING FUNCTIONS FOR ACCURATE LIGAND BIOACTIVITY PREDICTION

One of the most important initial steps in drug discovery is the screening of very large databases of drug-like molecules to identify putative ligands and filter out the ones that are potentially inactive for the target protein. Predicting the binding affinity (BA) of all the database molecules accurately is not as critical in this phase as to reliably discriminate active molecules from the inactive ones. In addition, scoring functions have limited accuracy when predicting binding affinity of inactive ligands simply because they are only trained on active molecules. Therefore, binding affinity prediction and relative ranking of molecules against each other should take place after the screening campaign when the database is enriched with putative active compounds.

Almost all existing scoring functions in use today rely on binding affinity prediction to carry out database enrichment. These models are typically trained on datasets of proteins bound to one or more ligands that are known to be actives. High-throughput screening assays typically result in some active molecules and a large number of inactive compounds against the target protein. Inactive compounds end up being discarded during the scoring function training process due to the lack of their binding affinity data. In such cases, conventional BA-based scoring functions could fail in producing reliable binding affinity scores for inactive compounds since they are only exposed to active molecules. Therefore, using conventional scoring functions to rank active and inactive ligands housed in large databases in order to enrich them will be problematic as it will be demonstrated in our experiments in this chapter.

6.1 Proposed screening-specific scoring functions

To overcome the limitations of conventional approaches, we introduce screening-specific scoring functions that are trained on both active and inactive compounds for a diverse set of protein targets. Our proposed models are based on deep neural networks and other machine learning

approaches that are optimized to directly reduce active/inactive misclassification rate instead of minimizing BA mean-squared error. The proposed models take advantage of the larger training set size that is constructed by combining active molecules and inactives vis-à-vis only the binders as is the case for the vast majority of conventional SFs. Taking into account both active and inactive molecules yield more diverse training complexes which could in turn substantially improve the screening accuracy of scoring functions on diverse families of ligands. The screening-specific scoring functions are also resilient to the uncertainty associated with experimentally collected binding affinity data due to the usage of binary labels instead of the measured BA values themselves as in the case of traditional SFs. The common sources of uncertainty in the binding affinity labels include measurement noise, data collection errors, incomplete and missing data. In addition, there are more and much larger public databases with assay results of binary (active/inactive) outcomes than sources with binding affinity data. For example, the biochemical databases PubChem [71] and ChEMBL [70] host millions of bio-assay and bioactivity data whereas the number of complexes with BA in archives such as PDBbind [23] and BindingDB [74] do not exceed several thousand. As we will see in the following sections, the proposed methods for constructing screening-specific SFs will result in substantial improvement in screening accuracy compared to those achieved by conventional models.

We use the 2014 PDBbind core set *Cr* as a benchmark for assessing the screening accuracy of the conventional and proposed SFs. As explained in Section 2.1.1, the core set consists of 65 protein families, each is bound to three different ligands with known and diverse binding affinity values. These are the active compounds that an ideal scoring function would identify. We assume the inactive molecules for each protein target are the crystallized ligands of the remaining 64 receptors in the test set as shown in Figure 6.1. Although this bioactivity assumption requires further investigation, it appears to be reasonable since all the 65 proteins have a pair-wise sequence similarity of less than 90%. Furthermore, all the assumed inactive protein-ligand ($65 \times (64 \times 3) =$) 12480 pairs were also checked in the ChEMBL database for possible activity and cross binding [70]. Forty of these protein-ligand pairs were found to be actives and therefore the final test

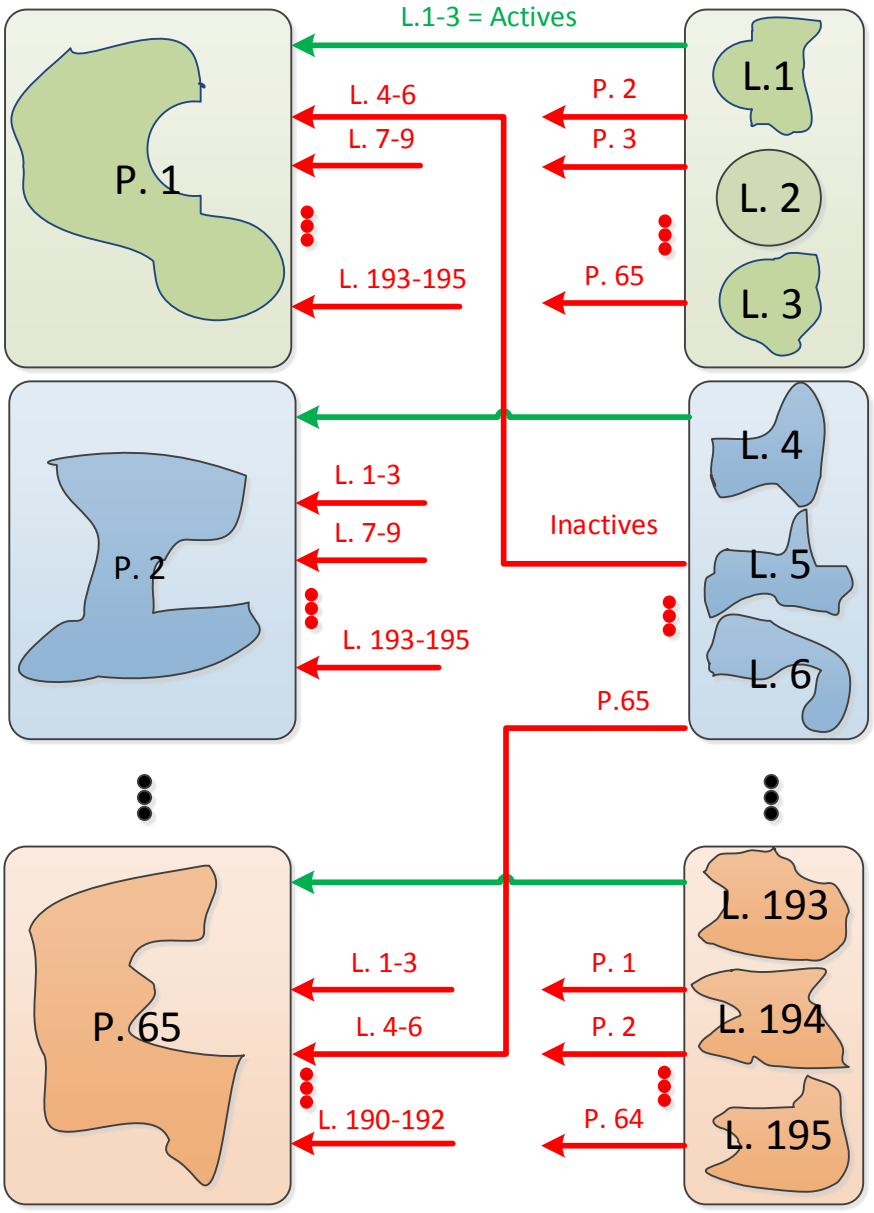


Figure 6.1: Constructing training and test data sets of active and inactive protein-ligand complexes using the core sets of PDBbind 2007 and 2014.

set consists of a total of $(195+40=)$ 235 binders and $(65\times 192-40=)$ 12440 inactive molecules. This translates to a total of $(235+12440=)$ 12675 test protein-ligand pairs and the actives to inactive ratio is approximately 1:64 for each target. A similar approach was also followed by Li et al. [25] to construct a test set for screening purposes. Their test set was also derived from the

2014 version of the PDBbind core set and it also consists of 65 proteins each binding to 3 different ligands. This set overlaps with the 2007 PDBbind core set in only 10 protein families. We use the remaining 55 protein families of the 2007 PDBbind test set including their active and (generated) inactive molecules as training set for our proposed screening-specific ML SFs. The overall size of this training set is 10726 protein-ligand pairs without any overlap with the screening test set.

6.2 Results and Discussion

6.2.1 Evaluation of the screening power of scoring functions

In this work, we predict the binding affinity or activity probability score for every protein-ligand complex in our test sets. Our test sets are composed of clusters of protein families. Each cluster represents a protein and several ligands and we are interested in classifying which of them are active. Therefore, we use the predicted binding affinity or activity probability score for the ligands to rank-order them in each cluster. Then the screening power of all SFs will be assessed based on their enrichment performance on the screening test set of protein clusters. The enrichment factor is a measure that accounts for the number of true binders among the top $x\%$ ranked molecules and is calculated as follows: $\mathrm{EF}_{x\%} = \mathrm{NTB}_{x\%}/(\mathrm{NTB}_{\mathrm{total}} \times x\%)$. Here $\mathrm{NTB}_{x\%}$ refers to the number of actives in the top-ranked $x\%$ of molecules, and $\mathrm{NTB}_{\mathrm{total}}$ is the total number of binders for the given target protein which is typically three. In our results, we report the average enrichment factor across protein clusters for the given test set.

6.2.2 Screening-specific ML SFs vs. conventional approaches on a diverse test set

We conduct two sets of experiments in this section to demonstrate the screening accuracy of the proposed SFs. In the first, we build 8 machine learning SFs trained on the 2007 protein-ligand complexes characterized with one or more feature sets of types X, A, R, and/or G. The ML SFs we construct are based on ensemble deep neural networks BsN-Score and BgN-Score, ensemble decision trees BRT and RF, SVM, kNN, MARS, and MLR. Two versions for each of these 8 models have been built. A scoring-specific version in which the training complexes (active only)

Table 6.1: Screening powers of proposed and conventional scoring functions on diverse complexes from the PDBbind 2014 core (*Cr*) test set. Training and test protein-ligand complexes are characterized using a combination of descriptors from at least one of the four types X, A, R, or G.

Scoring function	BA Modeling			Scoring function	Bio-activity Modeling		
	$EF_{1\%}$	$EF_{5\%}$	$EF_{10\%}$		$EF_{1\%}$	$EF_{5\%}$	$EF_{10\%}$
Glide::GlideScore-XP	19.54	6.27	4.14	BsN-Score::XAG	33.85	8.72	5.03
GOLD::ChemScore	18.91	6.83	4.08	SVM::XAG	30.26	8.41	5.08
Glide::GlideScore-SP	16.81	6.02	4.07	BgN-Score::XAG	28.72	8.51	4.92
DS::LUDI3	12.53	4.28	2.80	MLR::XARG	28.72	8.10	4.77
GOLD::ASP	12.36	6.23	3.79	MARS::XARG	28.72	7.90	4.82
MLR::XR	8.72	2.46	1.90	BRT::XAG	26.15	8.31	4.72
GOLD::GoldScore	7.95	4.52	3.16	RF::XA	24.10	7.80	4.97
SVM::A	7.69	3.69	2.78	DNN-Score::XAG	22.05	7.59	4.46
BsN-Score::XA	7.18	3.80	3.03	Glide::GlideScore-XP	19.54	6.27	4.14
BRT::XARG	7.18	3.60	2.51	GOLD::ChemScore	18.91	6.83	4.08
BgN-Score::XAG	7.18	2.97	2.36	Glide::GlideScore-SP	16.81	6.02	4.07
RF::XR	7.15	2.67	2.15	*k*NN::XA	14.85	7.69	4.56
DS::PLP1	6.92	4.28	3.04	DS::LUDI3	12.53	4.28	2.80
MARS::RG	6.15	1.85	1.80	GOLD::ASP	12.36	6.23	3.79
DS::Jain	5.90	2.51	1.80	GOLD::GoldScore	7.95	4.52	3.16
SYBYL::PMF-Score	5.38	2.21	1.90	DS::PLP1	6.92	4.28	3.04
SYBYL::ChemScore	5.26	2.38	2.18	DS::Jain	5.90	2.51	1.80
DS::PMF	4.87	2.87	2.63	SYBYL::PMF-Score	5.38	2.21	1.90
AffiScore	3.62	2.04	1.92	SYBYL::ChemScore	5.26	2.38	2.18
DNN-Score::XARG	3.08	1.54	1.44	DS::PMF	4.87	2.87	2.63
$DrugScore^{CSD}$	2.62	1.48	1.16	AffiScore	3.62	2.04	1.92
X-Score::HMScore	2.31	2.14	1.41	$DrugScore^{CSD}$	2.62	1.48	1.16
SYBYL::D-Score	2.31	1.79	1.46	X-Score::HMScore	2.31	2.14	1.41
*k*NN::XR	2.05	1.95	1.74	SYBYL::D-Score	2.31	1.79	1.46
SYBYL::G-Score	1.92	1.26	1.44	SYBYL::G-Score	1.92	1.26	1.44

are labeled using the measured binding affinity data ($Y \in \mathbb{R}$). For the proposed-specific version, we use binary values of active or inactive ($Y \in \{1,0\}$) to label the protein-ligand complexes. These two sets of 8 ML SFs are then compared against the conventional, BA-based SFs.

In Table 6.1 we list the screening powers of the 8 ML and 16 conventional SFs in terms of their ($x =$) 1, 5, and 10 per cent enrichment factors. In the left half of the table, we report screening performance when the ML SFs were trained to predict binding affinity instead of bioactivity

(whether the ligand is active or not). We notice that all of the BA-based ML SFs are outperformed by conventional approaches. The right half of the table shows the screening accuracy of conventional scoring functions and the proposed task-specific models. The ML screening-specific SFs that were trained as bioactivity classifiers are the top performers (except for *k*NN::XA). Our proposed SF that is based on ensemble neural networks has the highest screening performance of 33.85 in terms of $EF_{1\%}$. This is an order of magnitude difference in performance as compared to that of X-Score whose screening performance is very low. The BsN-Score::XAG is also substantially more accurate (by about 73%) than the best performing conventional SF GlideScore-XP whose $EF_{1\%}$ value is 19.54. In addition, the screening-specific version of BsN-Score achieved more than 370% improvement in enrichment accuracy over its generic BA counterpart BsN-Score::XA whose $EF_{1\%} = 7.15$. This boost in performances can be primarily attributed to the task-specific learning approach, as is apparent by comparing the performance of BsN-Score::XAG learned through classification to its best BsN-Score regression version BsN-Score::A (note that the regression version of BsN-Score::XAG has an even smaller $EF_{1\%}$ value of 6.88). We are confident that a substantial further improvement in screening accuracy is achievable when all the ideas proposed in Chapter 8 are fully developed and applied.

In the second experiment, we consider an updated version of ensemble deep-learning models to build scoring- and screening-specific SFs BsN-Score and BsN-Screen. We compare them to the conventional SFs RF-Score and X-Score which are based on Random Forests and linear regression. All four SFs are trained and tested on complexes from PDBbind 2007 and 2014. The core test set of PDBbind 2014 remains one of the main validation sets for the new SFs. We also use two additional test sets based on cross-validation (*CV*) and leave-clusters-out (*LCO*) as described in Section 2.1. The methodology we followed in the previous section (6.1) to construct active and inactive complexes are also followed here for the three test sets *Cr, CV*, and *LCO*. The training and test complexes for BsN-Score and BsN-Screen are characterized by the 2714 multi-persective descriptors generated using the the proposed Descriptor Data Bank platform. RF-Score and X-Score utilize their original descriptor sets of 36 geometrical and 6 physiochemical features, respectively.

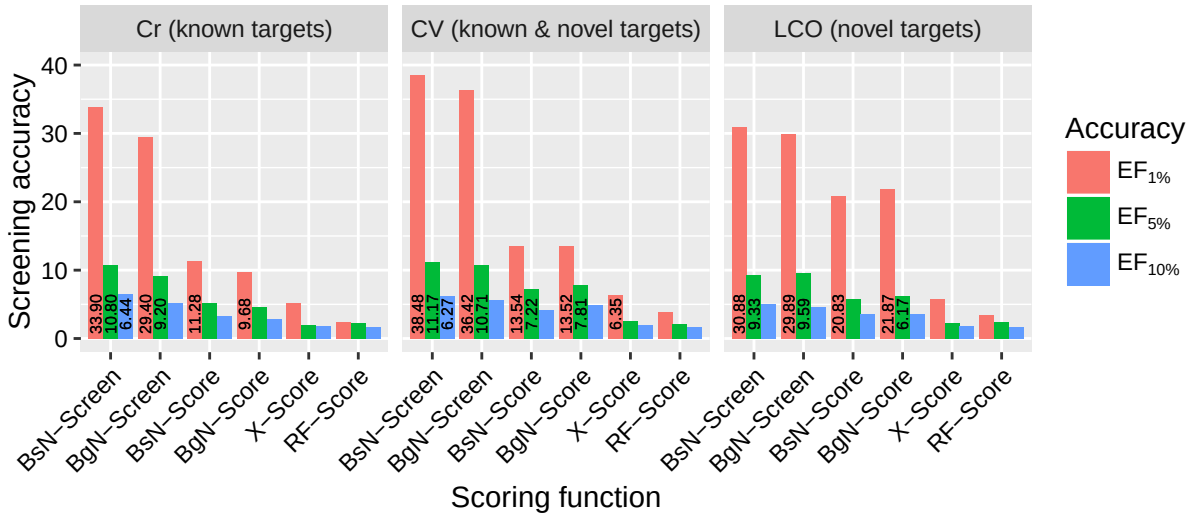


Figure 6.2: The screening accuracy of screening-specific (proposed) and binding-affinity-based (conventional) scoring functions when evaluated on test complexes with proteins that are either fully represented (*Cr*), partially represented (*CV*), or not represented (*LCO*) in the SFs' training data. The screening accuracy is expressed in terms of the 1%, 5%, and 10% enrichment factors of SFs in classifying ligands to actives and inactives against diverse set of proteins from the PDBbind 2014 benchmark.

The performance of the four SFs in terms of enrichment factors are shown in Figure 6.2 for known and novel test protein targets.

The plots clearly show that the screening-specific SF BsN-Screen is considerably more accurate than the generic, binding affinity-based methods regardless of the model's familiarity with the protein target. The gap in performance between BsN-Screen and BsN-Score is mainly attributed to the use of both active and inactive complexes to train BsN-Screen rather than just active compounds on which BT-Score is trained. BsN-Screen and BsN-Score are otherwise identical in terms of the training protein targets, descriptors, and the underlying ensemble deep-learning method they use to fit the data. BsN-Screen is also more accurate than the best performing conventional SFs as shown in Figure 6.3. The screening accuracy of GlideScore-SP, ChemScore, and GlideScore-XP has only been reported on known the core test set complexes by Li et al [25]. However, we expect their performance to be similar as we observed for the other BA-based SFs in Figure 6.2.

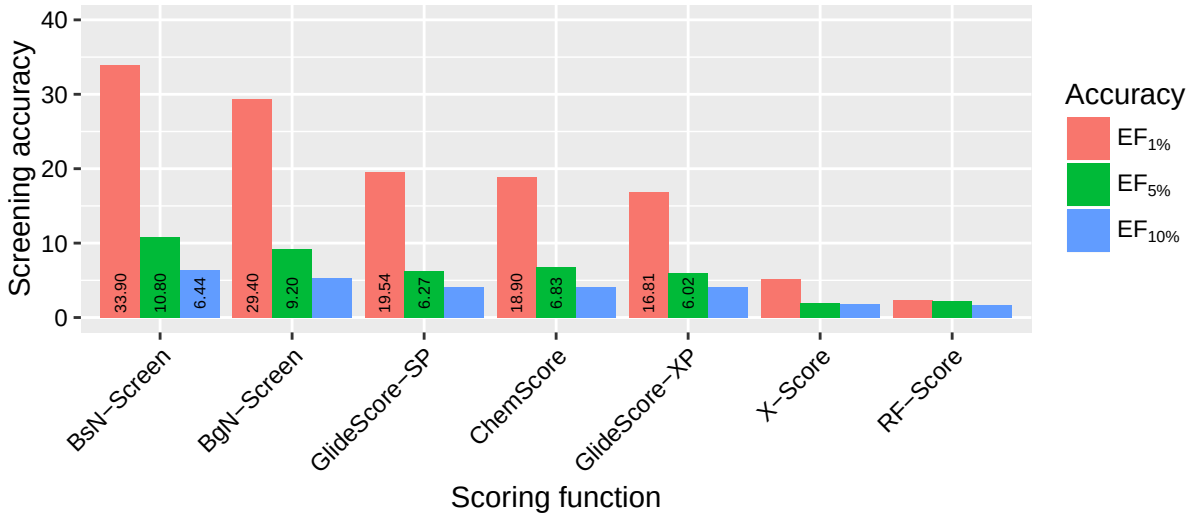


Figure 6.3: Screening accuracy of screening-specific and top-performing conventional scoring functions in terms of 1%, 5%, and 10% enrichment factors of protein targets sampled from the core test set of PDBbind 2014.

6.2.3 Impact of the number of training protein targets on the screening accuracies

Experimental information about 3D structures and bioassays of new protein-ligand complexes are regularly determined. This contributes to the growing size of public biochemical repositories and corporate compound databases. To assess the impact that a larger training set size would have on the predictive accuracy of generic and screening-specific SFs, we consider the 2007 and 2014 PDBbind datasets. We use the core (*Cr*) set of PDBbind 2014 as our main test set in this experiment. We use the independent core (*Cr*) set complexes of PDBbind 2007 and the primary (*Pr*) complexes of PDBbind 2014 to train generic and screening-specific SFs. Therefore, our training data consists of 3225 complexes in the primary (*Pr*) set as well as $(55 \times 3 =)$ 165 complexes in *Cr* which yields a total of 3390 complexes. We use these complexes to simulate the effect of increasing training set size on the screening accuracy of the models under investigation. More specifically, for a given number of training complexes x, where $x \in \{100, 680, 1260, 1840, 2420, 3000\}$, we select x complexes randomly (without replacement) from the 3390 PLCs to train the three the models. We then test them on the out-of-sample core set *Cr* comprising 195 complexes. This process is repeated 50 times to obtain robust average enrichment factors $EF_{1\%}$ and $EF_{5\%}$, which are plotted in

Figure 6.4: Dependence of screening accuracy of screening-specific and binding-affinity-based scoring models on training set size when training complexes are selected randomly (without replacement) from the refined set of PDBbind 2014 and the core set of PDBbind 2007. The screening accuracy is expressed in terms of 1%, 5%, and 10% enrichment factors of the PDBbind 2014 core test set.

Figure 6.4. The SFs we build here include the boosted decision tree SFs BT-Screen and BT-Score based on XGboost. BT-Screen is our screening-specific SF trained on active and inactive complexes characterized by our multi-perspective descriptors. BT-Score is only trained on active complexes. However, it also uses the same descriptors and underlying learning algorithm of boosted trees as BT-Screen. We also train our version of RF-Score and X-Score which are constructed by fitting Random Forests and Linear Regression models to the training complexes characterized by their the original RF-Score's 36 geometrical descriptors and X-Score's 6 physiochemical energy terms, respectively.

Figure 6.4 shows the screening accuracies of BT-Screen, BT-Score, and our versions of RF-Score and X-Score. For both enrichment factors ($EF_{1\%}$ and $EF_{5\%}$), we observe that BT-Screen and BT-Score improve as the size of the training dataset increases. The screening performance of RF-Score and X-Score are substantially lower than both ensemble SFs and their rate of im-

provement is almost zero beyond 680 training complexes. Similar rise in improvement was also achieved using Random Forest (whose accuracy plots are not shown) when it was fitted to our multi-perspective descriptors instead of the RF-Score's original 36 geometric descriptors. Another set of Random Forest models fitted to the 6 features used by X-Score were also found to be improving in screening accuracy as the number of their training complexes increases. This indicates that ensemble SFs have a potential for improvement as more training data becomes available. On the other hand, the linear model used in X-Score and other empirical SFs showed minimal rise in screening power as a response to increasing the training set size which is an indication of a very small potential for improvement in the future. This is due to the rigidity of linear models whose performance tends to saturate. Many of the conventional SFs considered in this study (Table 1.1) are empirical and thus they are also most likely to suffer from the same limitation. In fact, the best performing model of those 16 in terms of scoring power is X-Score; whose performance we study here. Therefore, one should consider better prediction approaches to derive accurate models from training data available on hand and from future updates. SF designers can conduct similar experiments to estimate accuracy enhancement when their proposed functions are re-calibrated on larger number of data instances.

6.2.4 Impact of the number of descriptor sets on the screening accuracies

To investigate the effect of increasing the number of descriptor sets on screening accuracy of SFs, we randomly select a maximum of 100 combinations of x descriptor sets from all possible (16 choose x) combinations of the 16 types listed in Table 3.1, where $x \in \{1,4,7,10,13,16\}$, and use them to characterize the training set complexes, which we then use to train the XGboost, Random Forest, and MLR models to build BT-Screen and BT-Score, RF-Score and X-Score analogs. These models are subsequently tested on the *Cr* dataset characterized by the same descriptors. For each number, x, of descriptor sets, the performance of the 100 SFs are averaged to obtain robust overall $EF_{1\%}$ and $EF_{5\%}$ statistics, which are plotted in Figure 6.5.

Similar to the case of increasing training data in terms of new PLCs, the performance of the en-

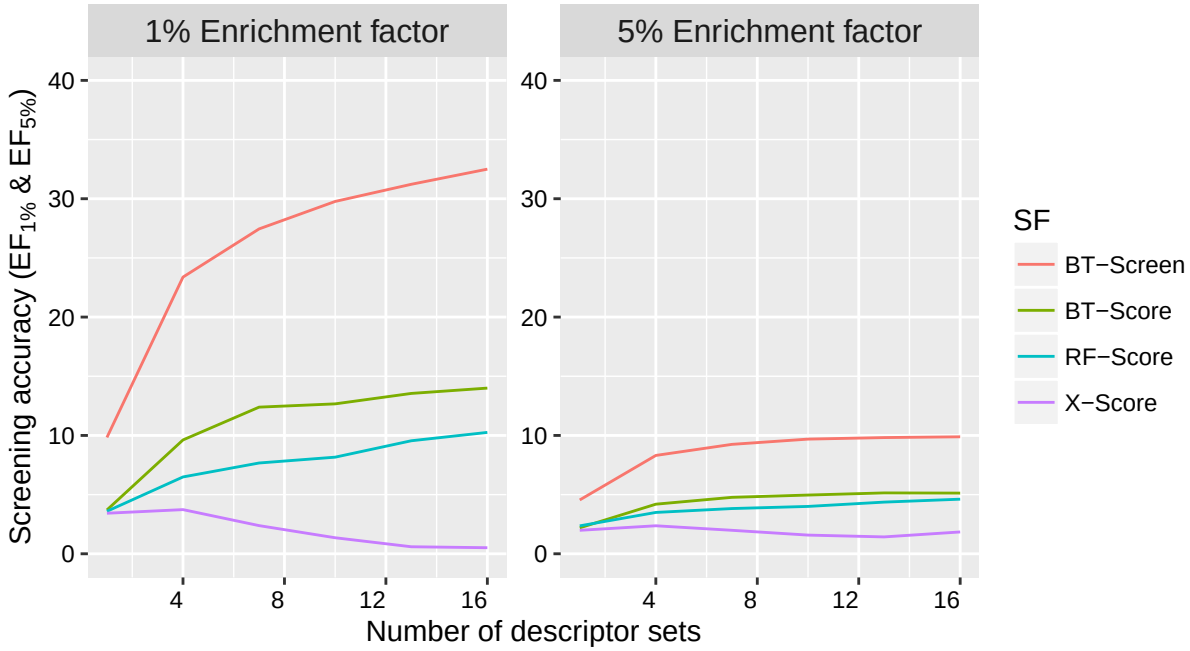


Figure 6.5: Dependence of screening accuracy of screening-specific (proposed) and binding-affinity-based (conventional) scoring models on the number of descriptor (feature) sets. The sets are drawn randomly (without replacement) from a pool of 16 feature types in Descriptor Data Bank (DDB) and used to characterize the training and out-of-sample core *Cr* test set complexes of PDBbind 2014. The docking accuracy is expressed in terms of 1%, 5%, and 10% enrichment factors of protein targets sampled from the core test set.

semble SFs are clearly benefiting from increasing the number of protein-ligand interaction models for both enrichment factor statistics. The number of descriptor sets (interaction perspectives) are as important as the number of training complexes in improving the quality of prediction. Based on these results, and similar improvement trends for the scoring and docking accuracies in the previous chapters, we believe that the public database of protein-ligand interactions we introduced in this work will be of high value to ML SFs and just as important to their screening accuracy as the resources of raw structural and experimental data.

6.3 Conclusion

In this work, we developed the screening-specific scoring functions BsN-Screen and BgN-Screen using novel ensemble models of boosted and bagged deep neural networks. The models are fitted to a large and diverse database of active and inactive complexes characterized by thousands

of multi-perspective descriptors. BsN-Screen and BgN-Screen are optimized to directly model the ligand activity as a classification problem to improve its accuracy in finding real active compounds from databases of new ligands not seen in their training set. We compare BsN-Screen and BgN-Screen with their generic counterparts BsN-Score and BgN-Score which are trained on (only) active complexes labeled with binding affinity data and characterized using our multi-perspective descriptors. The regression models BsN-Score and BgN-Score, as well as the vast majority of existing SFs, order test ligands based on their predicted binding affinity such that potential active compounds are ranked above the molecules that the models deem inactive. Our extensive experiments indicate that the screening-specific SFs are substantially more accurate in enriching ligand databases with more active compounds than their generic, BA-based counterparts. BsN-Screen and BgN-Screen have top 1% enrichment factors of more than 35 as opposed to 11 obtained by their scoring-specific analogs BsN-Score and BgN-Score. The best conventional SF GlideScore-SP, an empirical generic model implemented in GOLD, achieves an enrichment factor of 19.54 when tested on the same benchmark complexes [25]. Furthermore, the proposed screening-specific SFs also excelled when their improvement capacity was tested in response to more training data and descriptor sets.

CHAPTER 7

MULTI-TASK DEEP NEURAL NETWORKS FOR SIMULTANEOUS DOCKING, SCREENING, AND SCORING OF PROTEIN-LIGAND COMPLEXES

7.1 Introduction

Protein-ligand complexes fed to the proposed docking, scoring, and screening-specific SFs for prediction are typically characterized by the same descriptors and they are formed with the same protein targets. The training datasets of these task-specific scoring functions may only differ in their sizes and type of the output (independent) variable. When fitting a boosting tree or any other ML model to the dataset of one task, it does not take advantage of the datasets available for the other tasks to improve its performance. Rather, the three scoring functions are trained and applied in isolation despite the commonalities between the three problems they are modeling. In this chapter, we propose MT-Net, a novel multi-task deep neural network that can be effectively trained and applied to the three datasets simultaneously. MT-Net leverages information sharing across the three molecular modeling problems and uses abundant data for one problem to improve upon the performance of the tasks with limited data availability. In a sense, optimizing the network for multiple tasks simultaneously self-regularizes it against overfitting the tasks with limited data if they were to be learned separately.

Multi-task learning using deep neural networks has been successfully applied in recent studies related to ligand-based drug design [115, 116, 117]. Unlike the three tasks studied here in this work for screening, scoring, and docking tasks, each task in those studies is a predefined protein system for which compounds are classified as active or inactive molecules. As is the case for other QSAR models, the multi-target neural network must be re-trained from scratch in order to be used for targets different from the training proteins. On the other hand, the multi-task SFs we develop in this work are applicable to any protein system without the need for extra training. During their construction, our task-specific and multi-task scoring functions take into account information about

the ligand and protein explicitly instead of implicitly via multi-target learning as in some QSAR models [115, 116, 117]. Therefore, the proposed scoring functions in this work are applicable to any target as was shown by our experiments in the previous chapters (refer to SFs performance on known and novel targets).

7.2 Network architecture

The neural network we propose here consists of an input layer for the raw descriptors $\mathbf{x}_1^s$, L^s hidden layers shared (s) among the three tasks for extracting low-level representations for the molecular interactions, L^t task-specific hidden layers for each task $t \in \{\text{dock, screen, score}\}$ to generate high-level representations specific to each task, and Y^t corresponding output layers as depicted in Figure 7.1. During prediction, lower-level features universal for the three tasks are first extracted from the hidden shared layers as follows:

$$\mathbf{x}_{l+1}^s = H(\mathbf{W}_l^s \mathbf{x}_l^s + \mathbf{b}_l^s), \quad l = 1, \ldots, L^s$$

where $\mathbf{W}_l^s$ and $\mathbf{b}_l^s$ are respectively the weight matrix and the bias vector for the l-th shared hidden layer, and H is the activation function associated with it which is selected to be a rectified linear unit (ReLU) in this work. The output of the final shared hidden layer is then fed as an input ($\mathbf{x}_1^t = \mathbf{x}_{L^s+1}^s$) to the task-specific layers to extract higher-level representations specialized for each task using the following transformations:

$$\mathbf{x}_{l+1}^t = H(\mathbf{W}_l^t \mathbf{x}_l^t + \mathbf{b}_l^t), \quad l = 1, \ldots, L^t$$

Here, each task-specific layer is parameterized by a weight matrix $\mathbf{W}_l^t$ and a bias vector $\mathbf{b}_l^t$. The final outputs for the three tasks are simply the following transformations for the features $\mathbf{x}_{L^t+1}^t$ generated by the high-level hidden layers:

$$Y^t = O^t(\mathbf{W}_o^t \mathbf{x}_{L^t+1}^t + \mathbf{b}_o^t),$$

where O^t is the output function for the t-th task which is logistic for the screening problem and linear for the docking and scoring tasks. The number and sizes (number of hidden units) of the

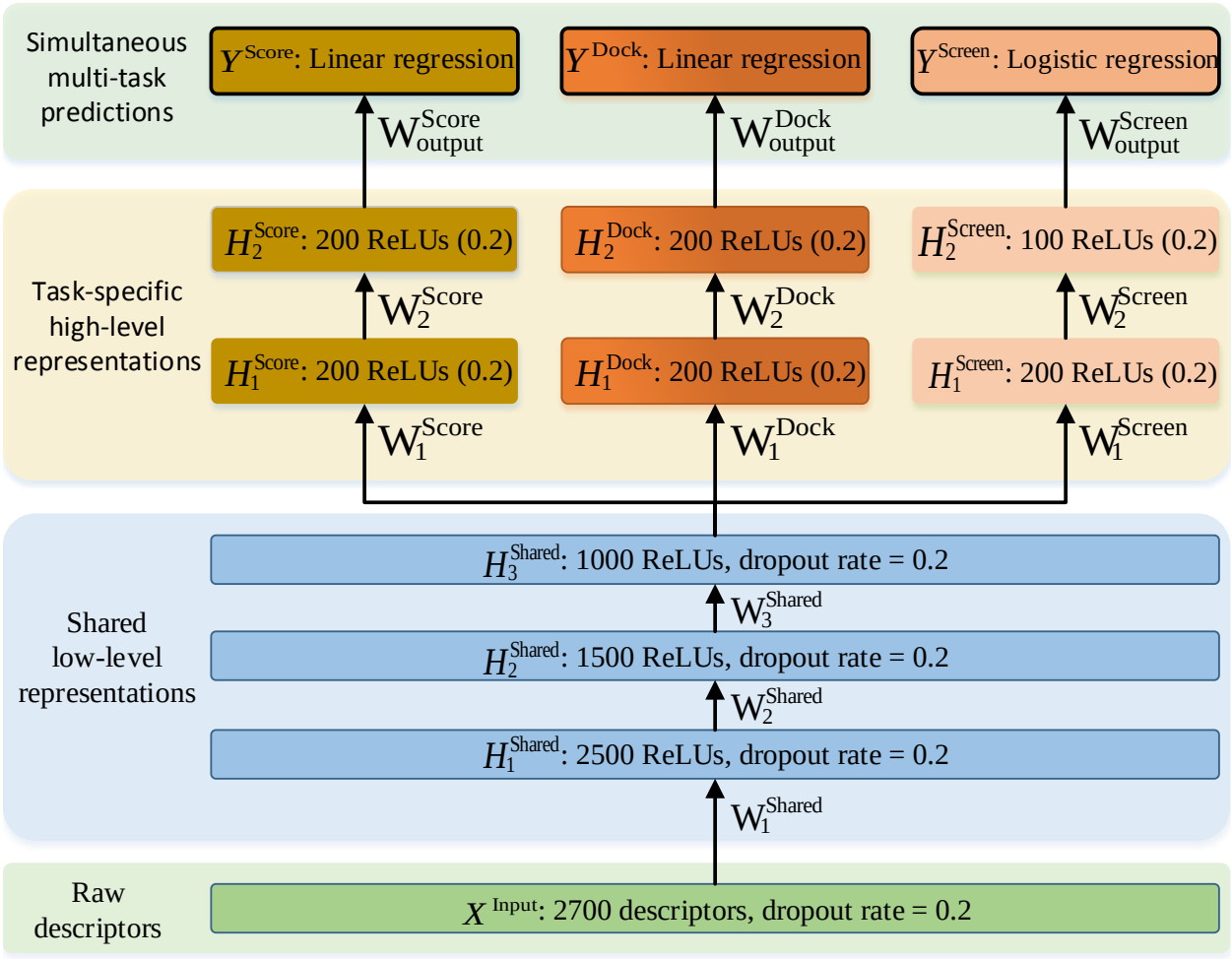


Figure 7.1: The architecture of the multi-task deep neural network SF MT-Net.

shared and task-specific hidden layers were tuned using cross-validation. We tested multiple pyramidal configurations for the network and found the configuration of three hidden layers with sizes $\{2500, 1500, 1000\}$ to be optimal for the shared layers, $\{200, 200\}$ for the docking and scoring-specific layers, and $\{200, 100\}$ for the screening-specific layers. Further details about the network's training procedure will be provided later in the following section.

7.3 Training the multi-task network

Nowadays, training wide and deep neural networks is simpler than ever before—thanks to recent advances in deep learning and computational resources such as Graphics Processing Units (GPU). However, even with such tools, building custom deep neural network scoring functions for

multiple tasks is not a trivial machine-learning problem. The following sections summarize the main challenges we faced during the model training process and the methods we followed to solve them.

7.3.1 Stochastic learning for imbalanced data

The three molecular modeling tasks benefit from training sets with different sizes. For the scoring task, we have an average of 3000 complexes with valid binding affinity labels. The screening scoring function is trained with more than 15000 PLCs with binary labels indicating whether the complexes are active or not. The docking task benefits from the largest training data which includes 300000 unique protein-ligand conformations. For the network to perform well across the three tasks, it should utilize all available training data while guarding its weights from overfitting the tasks with the larger training sizes. We achieved this by performing stochastic gradient descent with equally-sized minibatches sampled randomly from the training data of the three tasks. More specifically, we sample 10 PLCs from each task for a total minibatch size of 30 examples during every training (weight updating) step. We also used larger batch sizes of 100 complexes sampled proportionally from the three tasks according to their sizes. To avoid the bias of over-optimizing the weights on tasks with larger training data, we weighted the gradients up or down for each task depending on the number of examples in each batch for that task. This resulted in small improvement in the training speed. However, we resorted back to the former uniform sampling approach due to the increased implementation complexity of the weighted gradient approach.

7.3.2 Gradient computing for unavailable labels

Despite having common training protein targets across the three molecular modeling tasks, the majority of proteins form unique complexes across tasks. Complexes with the same protein differ amongst each other in the bound ligands and/or the conformations of those ligands. Complexes available for one task are typically not present for the other two tasks and consequently they do not share the same labels. The task-specific labels we use here are binding affinity, RMSD of a

pose from its respective native conformation, or active/inactive binary value. This poses a challenge during training while computing the gradient to update the network's weights. Gradients are calculated at the output layer of each task based on the difference between the actual values and labels predicted by those output layers of each task. We use gradients of zero for the tasks with unlabeled examples such that the task-specific weights associated with missing labels do not get updated while weights in shared layers are updated with gradients calculated from labeled examples only. It should be noted that each of the 30 training examples in any random minibatch has at least one available label and at most two missing labels.

7.3.3 Network regularization

Networks with wide and deep hidden layers and multiple output layers such as the one we build here have millions of parameters (weights and biases) which outnumber our training size multiple fold. Therefore, regularizing the network is necessary to avoid overfitting. MT-Net has a powerful regularization mechanism in place enabled by the simultaneous optimization of shared layers with different training data for each task which help prevent the shared weights from overfitting individual tasks. To further guard against overfitting, we use the dropout technique of hidden units by randomly selecting 20% of the network's activations and deliberately switching them off to zero for each protein-ligand complex in the current training minibatch. Random dropout was introduced by Hinton et al. to avoid overfitting by preventing co-adaptations among the hidden units of each layer [118]. A network trained with dropout can effectively be viewed as an average of an ensemble of multiple thinned networks. The cost of using dropout is longer training time since parameter updates are noisy due to the random replacement of activations with zeros.

7.3.4 Computational requirements

Training a wide and deep neural network with thousands of input descriptors and hidden units in several deep layers and three output layers results in millions of parameters. A network of such scale requires a powerful computer system to train and execute. We use an Amazon AWS instance

with NVIDIA Kippler GK210 GPU featuring 2496 parallel processing cores and 12 Gigabytes of graphics-dedicated memory. Training the network on this system takes five hours of wall clock time. Predicting a dataset of 1000 protein-ligand complexes can be achieved in 5 milliseconds on the same machine. We use the Python machine-learning libraries TensorFlow and Keras to build our multi-task neural network scoring functions [43, 119].

7.4 Prediction and accuracy

In this section, we investigate the benefits of multi-task learning to simultaneously predict the ligand binding pose, binding affinity, and whether it's active or not for any protein target. We build a scoring function based on the multi-task neural network architecture described in Section 7.2 and compare it with conventional approaches and its single-task counterpart for the scoring, screening, and docking tasks. The SFs based on the multi-task (MT-Net) and single-task neural networks are trained on the primary training set of PDBbind 2014 and tested on the out-of-sample core test set *Cr*. The performance results of these scoring functions and the best conventional approaches are listed in Table 7.1.

Table 7.1: The docking, screening, and scoring performance of multi-task and single-task SFs on PDBbind 2014 core test set (*Cr*).

SF type	Docking (S_1^1%)	Screening ($EF_{1\%}$)	Scoring (R_p)
Multi-task (MT-Net)	79.97 %	29.17	0.804
Single-task	80.51 %	22.87	0.803
Best conventional[1]	–	19.54 (ChemPLP)	0.627 (X-Score)

[1] We report the screening and scoring accuracy of the top performing conventional scoring functions GlideScore and X-Score on these tasks. We use the screening accuracy for ClideScore-XP from the recent comparative study by Li et al [25]. The scoring performance of X-Score is calculated by us since we have access to its software. We do not have access to the docking success rate statistic S_1^1 for the most accurate conventional SF ChemPLP. However, ChemPLP's S_2^1 is 82% while MT-Net and the single-task NN achieve 90% on this less stringent test.

The screening and scoring performance of MT-Net are substantially higher than those of GlideScore-XP and X-Score, which are the best performing conventional approaches on these two tasks. MT-

Net's binding pose prediction is also more accurate than the top performing conventional scoring function in the docking task, ChemPLP. MT-Net and ChemPLP correctly identified the correct binding modes of 90% and 82% of protein-ligand complexes in the core test set of PDBbind 2014, respectively. The binding mode is considered correctly identified if the top-scoring pose of the ligand was found to lie within 2 Å RMSD from the known native conformation. MT-Net achieves 80.51% success rate on the more stringent test where the top-scoring pose must lie within 1 Å RMSD (instead of 2 Å RMSD) from the known binding mode to be considered native or near-native. We denote this success rate by S_1^1 and the former less stringent test by S_2^1. We do not report the S_1^1 success rate for ChemPLP since it was not available to us in the original publication [25] and we do not have access to the scoring function's predictions for the test complexes to compute its S_1^1 value.

The screening performance of the multi-task SF is substantially higher than that obtained by the single-task neural network. The gap in performance between the two functions is much narrower for the docking and scoring tasks with a slight lead for the traditional single-task approach. The finding that docking performance is not benefiting from the multi-task approach might be attributed to the fact that the docking data set size is already large so that the additional screening and scoring data are relatively too small to make any difference. The reason behind the small scoring is not very clear and warrants further investigation. However, it might be due to the fact that the docking and screening data sets are generated from the scoring complexes and therefore the derived complexes may not contribute new information. We believe that the gains in performance will be much greater if scoring, screening, and docking data were more diverse and larger in size.

To our knowledge, such a novel architecture for scoring functions has not been attempted before and we believe it could be used as a blue-print for future scoring functions with the availability of more data.

7.5 Conclusion

We proposed a novel multi-task scoring function for simultaneously predicting ligand binding pose, its activity classification label (active/inactive), and its binding affinity with a target protein. The scoring function, MT-Net, is based on wide and deep multi-task neural network with shared hidden layers and three sets of task-specific and output layers for the docking, screening, and scoring tasks. When fitted to our current training sets, MT-Net performed equally well with the single-task neural network SF when tested on out-of-sample test sets carefully designed to evaluate screening, docking, and scoring accuracy. Due to their high-capacity, we anticipate the accuracy of scoring functions based on multi-task deep networks to improve beyond those based on single-task NNs when larger and more diverse datasets are used for training. Currently we are developing a pipeline for automated data retrieval from biochemical databases to continuously train the network on new protein-ligand complexes and screening assays as they become available. When coupled with the proposed descriptor databank (DDB) platform, we think the active multi-task learning pipeline will produce an evolving, powerful scoring function.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 Proposed approaches and key findings

The aim of this research is to develop next generation scoring functions for successful computer-aided drug discovery. The work presented in this dissertation helped to achieve this goal by the development of Descriptor Data Bank and novel task-specific scoring functions based on machine-learning approaches. In the following sections, we briefly describe the two components and summarize their effectiveness quantitatively.

8.1.1 Descriptor Data Bank (DDB) for multi-perspective characterization of protein-ligand interactions

In this work we presented Descriptor Data Bank (DDB), an online platform for facilitating multi-perspective modeling of PL interactions. The online platform is an open-access hub for depositing, hosting, executing, and sharing tools and data arising from a diverse set of interaction modeling hypotheses. In addition to the descriptor extraction tools, the platform also implements a machine-learning (ML) toolbox that drives the DDB's automatic descriptor filtering and evaluation utilities as well as scoring function fitting and prediction functionalities. To enable local access to many of DDB's utilities, command-line programs and a PyMOL plug-in have also been developed and can be freely downloaded for offline multi-perspective modeling. We seed DDB with 16 diverse descriptor extraction tools developed in-house and collected from the literature. The tools combined generate over 2700 descriptors that characterize (i) proteins, (ii) ligands, and (iii) protein-ligand complexes. The in-house descriptors we propose here, REPAST & RETEST, are target-specific and based on pair-wise sequence and structural alignment of proteins followed by target clustering and triangulation. We built and used the fit/predict service in DDB to fit ML SFs to the in-house descriptors and those collected from the literature. We then evaluated them on several

data sets that were constructed to reflect real-world drug screening scenarios. We found that multi-perspective SFs that were constructed using large and diverse number of DDB interaction models outperformed their single-perspective counterparts in all evaluation scenarios with an average improvement of 15%. We also found that our proposed target-specific descriptors improve upon the accuracy of SFs that were trained without them. In addition, DDB's filtering module was able to exclude noisy and irrelevant descriptors when artificial noise was included as new features. We also observed that the tree-based ensemble ML SFs implemented in DDB are robust even with the presence of noisy and decoy descriptors.

8.1.2 Boosted and bagged deep neural networks for task-specific scoring functions

The proposed descriptor data bank platform made it possible to efficiently capture thousands of important protein-ligand interactions in the form of molecular descriptors or features for entire databases of complexes. However, current scoring functions lack the flexibility and capacity to take advantage of such data fully. Moreover, conventional modeling strategies have been centered around building generic scoring functions based on simple linear regression models that attempt to predict ligand binding poses (docking task) and binary class labels of activity (screening task) indirectly via binding affinity prediction (scoring task). In the previous chapters, we showed the limitations of traditional scoring functions not only in the docking and screening tasks, but also the scoring task for which they are optimized to model. Therefore, we presented a novel set of task-specific scoring functions based on large ensemble of deep neural networks to model the three tasks. The boosted and bagged neural network scoring functions are BsN-Score and BgN-Score for the scoring task, BsN-Dock and BgN-Dock for the docking task, and BsN-Screen and BgN-Screen for the screening task.

We evaluated the docking, screening, scoring, and ranking accuracies of the proposed scoring functions, as well as several conventional approaches in the context of the 2007 and 2014 versions of the PDBbind benchmark that encompasses a diverse set of high-quality protein families and drug-like molecules. In terms of scoring accuracy, we found that the ensemble NN SFs, BgN-Score

and BsN-Score, have more than 35% better Pearson's correlation coefficient (0.844 and 0.840 vs. 0.627) between predicted and measured binding affinities compared to that achieved by a state-of-the-art conventional SF. We further found that ensemble NN models surpass SFs based on other state-of-the-art ML algorithms such as Boosted Regression Trees, Random Forests, and SVM. Similar results have been obtained for the ranking task. Within clusters of protein-ligand complexes with different ligands bound to the same target protein, we found that the best ensemble NN SF is able to rank the ligands correctly based on their experimentally-determined binding affinities 64.6% of the time and identify the top binding ligand 78.4% of the time. Given the challenging nature of the ranking problem and that SFs are used to screen millions of ligands, this represents a significant improvement over the the best conventional SF we studied, for which the corresponding ranking performance values are 57.8% and 73.4%. A substantial improvement in the docking task has also been achieved by our proposed docking-specific SFs. We found that the best performing ML SF has a success rate of 95% in identifying poses that are within 2 Å root-mean-square deviation from the native poses of 65 different protein families. This is in comparison to a success rate of only 82% achieved by the best conventional SF, ChemPLP, employed in the commercial docking software GOLD. As for the ability to distinguish active molecules from inactives, our screening-specific SFs showed excellent improvements over the conventional approaches. The proposed SF BsN-Screen achieved a screening enrichment factor of 33.90 as opposed to 19.54 obtained from the best conventional SF which is employed in GOLD. For all tasks, we observed that the proposed task-specific NN SFs benefit more than their conventional counterparts from increases in the number of features and the size of training dataset. They also perform better than the conventional SFs on novel proteins that they were never trained on before. The accuracies of ensemble NN SFs are even higher when they predict for protein-ligand complexes that are related to their training sets. The high predictive accuracy of ensemble SFs BsN-Score and BgN-Score is due to the following three factors: (i) the low bias error of the highly-nonlinear neural network base learners, (ii) the low variance error achieved by using bagging and boosting, and (iii) the employed multi-perspective set of diverse descriptors we extract for protein-ligand complexes using Descriptor Data Bank.

8.1.3 Multi-task deep neural networks for simultaneous binding pose, activity, and affinity prediction

In addition to the three task-specific models, we developed a novel multi-task scoring function to simultaneously predict ligand binding poses, their activity class labels (active/inactive), and binding affinities against the target protein. The scoring function, MT-Net, is based on wide and deep multi-task neural network with shared hidden layers and three sets of task-specific hidden and output layers for the docking, screening, and scoring tasks. When fitted to our current training sets, MT-Net outperformed conventional scoring functions and matched single-task neural network SFs when tested on out-of-sample test sets carefully designed to evaluate screening, docking, and scoring accuracy. Due to their high-capacity, we expect the accuracy of scoring functions based on multi-task deep networks to improve beyond those based on single-task NNs when larger and more diverse datasets are used for training. Currently we are developing a pipeline for automated data retrieval from biochemical databases to continuously train the network on new protein-ligand complexes and screening assays as they become available. When coupled with the proposed descriptor databank (DDB) platform, we think the active multi-task learning pipeline will produce an evolving, powerful scoring function.

8.2 Future work

In addition to the excellent predictions we have achieved in ligand scoring, ranking, docking, and screening, we are working on the following ideas that we believe will make the predictions even more accurate.

8.2.1 Enriching DDB with more molecular descriptors

Protein-ligand complexes have been characterized by 16 feature types in the experiments we presented in this work. Currently, these are the only descriptor types hosted in Descriptor Data Bank (DDB). After publishing the platform in the literature, it will gain exposure and we believe the wider research community will quickly enrich it with much larger number of descriptor sets and

feature extraction tools. The collected descriptors will then be used in conjunction with ensemble deep neural networks to build task-specific scoring functions. DDB has automated task-specific filters in-place to select the most relevant descriptors for the given task.

8.2.2 Pipeline for self-learning scoring functions

The proposed descriptor data bank and the task-specific scoring functions based on ensembles of wide and deep neural networks are two important components for building evolving scoring models. More specifically, we are planning to develop a pipeline with the following automated stages to create self-learning scoring functions: (i) retrieval of raw data of proteins and genes, ligands, protein-ligand complexes, and screening assays from public databases such as PDB [9], Zinc [68], PDB-Ligand [120], PDBbind [10], BindingDB [74], DUD [121], CSAR [82], ChEMBL [70], and PubChem [71]; (ii) data preprocessing, integration, and descriptor extraction using Descriptor Data Bank platform; and (iii) model training and updating of the boosted and multi-task deep neural networks introduced in this work. We believe such pipeline will result in a scoring function that will constantly improve upon newly released molecules and experimental data.

APPENDIX

APPENDIX

FEATURES USED TO CHARACTERIZE PROTEIN-LIGAND COMPLEXES

Below we list the descriptors we used to characterize protein-ligand complexes.

- **X-Score (X) features** [56][1]

    1. van der Waals interactions between the ligand and the protein.
    2. Hydrogen bonding between the ligand and the protein.
    3. Hydrophobic effect: accounted for by counting pairwise hydrophobic contacts between the protein and the ligand.
    4. Hydrophobic effect: accounted for by examining the fit of hydrophobic ligand atoms inside hydrophobic binding site.
    5. Hydrophobic effect: accounted for by calculating the hydrophobic surface area of the ligand buried upon binding.
    6. The deformation effect, expressed as the contribution of the number of rotors for the ligand.

- **AffiScore (A) features** [75, 57, 122]

    1. The number of non-hydrogen atoms in the ligand.
    2. The total sum of hydrophobicity values for all hydrophobic protein-ligand atom pairs.
    3. The total difference in hydrophobicity values between all hydrophobic protein-ligand atom pairs.

[1]The software packages to calculate X-Score and AffiScore features can be obtained from their respective authors via the links `http://sw16.im.med.umich.edu/software/xtool/` and `http://www.bch.msu.edu/~kuhn/software/slide/`. The packages were modified so that the features are written to a file since they are not written out by default.

4. The total difference in hydrophobicity values between all protein-ligand hydrophobic/hydrophilic mismatches.
5. The total sum of hydrophobicity values for all hydrophilic protein-ligand atom pairs.
6. The total difference in hydrophobicity values between all hydrophilic protein-ligand atom pairs.
7. The total number of protein-ligand salt bridges.
8. The total number of regular hydrogen bonds.
9. The total number of uncharged polar atoms at the protein-ligand interface which do not have a bonding partner.
10. The total number of charged atoms at the protein-ligand interface which do not have a bonding partner.
11. The fraction of ligand heavy atoms (non-hydrogen) which are at the protein-ligand interface (defined as atoms within 4.5 Å of opposite protein or ligand atoms).
12. The number of ligand atoms at the interface.
13. The number of hydrophilic ligand atoms which are not at the interface.
14. The number of rotatable single bonds in the ligand molecule.
15. The number of protein and ligand rotatable single bonds at the interface.
16. The sum of the average hydrophobicity values for all ligand atoms. The average hydrophobicity value for the ligand atoms was determined by calculating the average hydrophobicity value of all neighboring hydrophobic protein atoms.
17. The sum of hydrophobicity values for all hydrophobic protein-ligand atom pairs (as computed in the old AffiScore version).
18. The number of rotatable single bonds of the ligand at the interface.

19. The sum of the degree of similarity between protein-ligand hydrophobic atom contacts. A higher value indicates the protein and ligand atoms had more similar raw hydrophobicity values.
20. A normalized version of the degree of similarity between the ligand atoms and their hydrophobic protein neighbors. This is the version currently used in AffiScore.
21. Pairwise count of hydrophobic-hydrophobic protein-ligand contacts.
22. Pairwise count of hydrophilic-hydrophilic protein-ligand contacts.
23. Pairwise count of hydrophobic-hydrophilic protein-ligand contacts.
24. Total of protein hydrophobicity values for protein atoms involved in hydrophobic-hydrophobic contacts.
25. For hydrophobic/hydrophilic mismatch pairs, sum of the hydrophobic atom hydrophobicity values.
26. Total hydrophilicity of all protein interfacial atoms.
27. A distance normalized version of feature number 24.
28. Total of protein hydrophobicity values for protein atoms at the interface.
29. A distance normalized version of feature number 28.
30. Total of ligand hydrophilicity values for interfacial ligand atoms.

- **Gold (G) features** [52]

1. The number of ligand exposed hydrophobic atoms.
2. The number of ligand acceptor atoms.
3. The number of ligand donor atoms.
4. The number of ligand atoms that clash with protein atoms.
5. The number of hydrogen bonds.
6. The molecular weight of the ligand.

7. The number of occluded ligand acceptor atoms.
8. The number of occluded ligand donor atoms.
9. The number of occluded ligand polar atoms.
10. The number of occluded protein acceptor atoms.
11. The number of occluded protein donor atoms.
12. The number of occluded protein polar atoms.
13. The number of ligand rotatable bonds.
14. The buried surface area of the ligand upon docking.

- **RF-Score or geometrical (R) features** [27][2]

1. The number of protein-ligand carbon-carbon (C-C) contacts.
2. The number of protein-ligand nitrogen-carbon (N-C) contacts.

3-36. Similarly, the number of protein-ligand contacts for the pairs: O-C, S-C, C-N, N-N, O-N, S-N, C-O, N-O, O-O, S-O, C-F, N-F, O-F, S-F, C-P, N-P, O-P, S-P, C-S, N-S, O-S, S-S, C-Cl, N-Cl, O-Cl, S-Cl, C-Br, N-Br, O-Br, S-Br, C-I, N-I, O-I, and S-I.

[2]The features can be readily calculated and output by the open-source software package RF-Score that can be found in the supplementary data of [27] or directly downloaded from `http://bioinformatics.oxfordjournals.org/content/suppl/2010/03/18/btq112.DC1/bioinf-2010-0060-File007.zip`

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] L. Kavraki, "Rigid receptor-flexible ligand docking: Overview and examples," *Connexions*, 2006.

[2] X. Fradera and J. Mestress, "Guided docking approaches to structure-based design and screening," *Current Topics in Medicinal Chemistry*, vol. 4, pp. 687–700, 2004.

[3] R. A. Friesner, J. L. Banks, R. B. Murphy, T. A. Halgren, J. J. Klicic, D. T. Mainz, M. P. Repasky, E. H. Knoll, M. Shelley, J. K. Perry, D. E. Shaw, P. Francis, and P. S. Shenkin, "Glide: A new approach for rapid, accurate docking and scoring. 1. method and assessment of docking accuracy," *Journal of medicinal chemistry*, vol. 47, no. 7, pp. 1739–1749, 2004.

[4] G. Warren, C. Andrews, A.-M. Capelli, B. Clarke, J. LaLonde, M. Lambert, M. Lindavall, N. Nevins, S. Semus, S. Senger, G. Tedesco, I. Wall, J. Woolven, C. Peishoff, and M. Head, "A critical assessment of docking programs and scoring functions," *Journal of medicinal chemistry*, 2005.

[5] M. Cases and J. Mestres, "A chemogenomic approach to drug discovery: focus on cardiovascular diseases," *Drug discovery today*, vol. 14, no. 9-10, pp. 479–485, 2009.

[6] X. Xu, M. Kasembeli, X. Jiang, B. Tweardy, and D. Tweardy, "Chemical probes that competitively and selectively inhibit Stat3 activation," *PLoS One*, vol. 4, no. 3, 2009.

[7] K. Simons, R. Bonneau, I. Ruczinski, and D. Baker, "Ab initio protein structure prediction of CASP III targets using ROSETTA," *Proteins: Structure, Function, and Genetics*, vol. 37, no. S3, pp. 171–176, 1999.

[8] A. Favia, I. Nobeli, F. Glaser, and J. Thornton, "Molecular docking for substrate identification: The short-chain dehydrogenases/reductases," *Journal of Molecular Biology*, vol. 375, no. 3, pp. 855–874, 2008.

[9] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The protein data bank," *Nucleic Acids Research*, vol. 28, no. 1, pp. 235–242, 2000.

[10] R. Wang, X. Fang, Y. Lu, and S. Wang, "The PDBbind database: Collection of binding affinities for protein-ligand complexes with known three-dimensional structures," *Journal of Medicinal Chemistry*, vol. 47, no. 12, pp. 2977–2980, 2004, pMID: 15163179.

[11] F. Allen and O. Kennard, "Cambridge Structural Database (CSD)," *Chemical Design Automation News*, vol. 8, pp. 1–31, 1993.

[12] P. D. Lyne, "Structure-based virtual screening: An overview," *Drug Discovery Today*, vol. 7, no. 20, pp. 1047 – 1055, 2002.

[13] N. Singh, G. Chevé, D. Ferguson, and C. McCurdy, "A combined ligand-based and target-based drug design approach for g-protein coupled receptors: Application to salvinorin a, a selective kappa opioid receptor agonist," *Journal of Computer-Aided Molecular Design*, vol. 20, no. 7, pp. 471–493, 2006.

[14] T. Marrone, J. Briggs, and, and J. McCammon, "Structure-based drug design: Computational advances," *Annual Review of Pharmacology and Toxicology*, vol. 37, no. 1, pp. 71–90, 1997.

[15] N. I. of Health, "Structure-based drug design: from the computer to the clinic," available at: http://publications.nigms.nih.gov/structlife/chapter4.html.

[16] M. Mathieu and P. I. Corporation, *PARAXEL's Pharmaceutical R&D Statistical Sourcebook 2001*. PAREXEL International Corp., 2001.

[17] T. Cheng, X. Li, Y. Li, Z. Liu, and R. Wang, "Comparative assessment of scoring functions on a diverse test set," *Journal of Chemical Information and Modeling*, vol. 49, no. 4, pp. 1079–1093, 2009.

[18] T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani, *The elements of statistical learning*. Springer, 2009, vol. 2.

[19] W. Mooij and M. Verdonk, "General and targeted statistical potentials for protein-ligand interactions," *Proteins*, vol. 61, no. 2, p. 272, 2005.

[20] M. L. Verdonk, J. C. Cole, M. J. Hartshorn, C. W. Murray, and R. D. Taylor, "Improved protein–ligand docking using gold," *Proteins: Structure, Function, and Bioinformatics*, vol. 52, no. 4, pp. 609–623, 2003.

[21] Accelrys Software Inc., *The Discovery Studio Software*, San Diego, CA, 2001, version 2.0.

[22] Y. Li, Z. Liu, J. Li, L. Han, J. Liu, Z.-X. Zhao, and R. Wang, "Comparative assessment of scoring functions on an updated benchmark: I. compilation of the test set," *Journal of chemical information and modeling*, 2014.

[23] R. Wang, X. Fang, Y. Lu, C.-Y. Yang, and S. Wang, "The pdbbind database: Methodologies and updates," *Journal of Medicinal Chemistry*, vol. 48, no. 12, pp. 4111–4119, 2005, pMID: 15943484.

[24] H. M. Ashtawy and N. R. Mahapatra, "BgN-Score and BsN-Score: Bagging and boosting based ensemble neural networks scoring functions for accurate binding affinity prediction of protein-ligand complexes," *BMC bioinformatics*, vol. 16, no. Suppl 4, p. S8, 2015.

[25] Y. Li, L. Han, Z. Liu, and R. Wang, "Comparative assessment of scoring functions on an updated benchmark: 2. evaluation methods and general results," *Journal of chemical information and modeling*, 2014.

[26] H. M. Ashtawy and N. R. Mahapatra, "Machine-learning scoring functions for identifying native poses of ligands docked to known and novel proteins," *BMC bioinformatics*, vol. 16, no. Suppl 6, p. S3, 2015.

[27] P. Ballester and J. Mitchell, "A machine learning approach to predicting protein-ligand binding affinity with applications to molecular docking," *Bioinformatics*, vol. 26, no. 9, p. 1169, 2010.

[28] H. M. Ashtawy and N. R. Mahapatra, "A comparative assessment of predictive accuracies of conventional and machine learning scoring functions for protein-ligand binding affinity prediction," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[29] P. Ballester and J. Mitchell, "Comments on "leave-cluster-out cross-validation is appropriate for scoring functions derived from diverse protein data sets": Significance for the validation of scoring functions," *Journal of Chemical Information and Modeling*, vol. 51, no. 8, pp. 1739–1741, 2011.

[30] D. M. Hawkins, "The problem of overfitting," *Journal of chemical information and computer sciences*, vol. 44, no. 1, pp. 1–12, 2004.

[31] R. D. C. Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2010, ISBN 3-900051-07-0.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[33] J. H. Friedman, "Multivariate adaptive regression splines," *The Annals of Statistics*, vol. 19, no. 1, pp. pp. 1–67, 1991.

[34] T. H. Stephen Milborrow and R. Tibshirani, *earth: Multivariate Adaptive Regression Spline Models*, 2010, r package version 2.4-5.

[35] J. Rudy, "Py-earth: Github repository," 2016.

[36] K. Hechenbichler and K. Schliep, "Weighted k-nearest-neighbor techniques and ordinal classification," Citeseer, Tech. Rep., 2004.

[37] R. J. Samworth, "Optimal weighted nearest neighbour classifiers," *The Annals of Statistics*, vol. 40, no. 5, pp. 2733–2763, 2012.

[38] K. S. . K. Hechenbichler, *kknn: Weighted k-Nearest Neighbors*, 2010, r package version 1.0-8.

[39] V. Vapnik, *Statistical learning theory*. Wiley, New York, 1998.

[40] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[41] E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer, , and A. Weingessel, *e1071: Miscellaneous Functions of the Department of Statistics (e1071), TU Wien*, 2010, r package version 1.5-24.

[42] B. Ripley, "nnet: Feed-forward neural networks and multinomial log-linear models," *R package version*, vol. 7, no. 5, 2011.

[43] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous systems, 2015," *Software available from tensorflow. org*, vol. 1, 2015.

[44] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*, 2001.

[45] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[46] J. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of Statistics*, pp. 1189–1232, 2001.

[47] Y. Freund and R. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational learning theory*. Springer, 1995, pp. 23–37.

[48] G. Ridgeway, *gbm: Generalized Boosted Regression Models*, 2010, r package version 1.6-3.1.

[49] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," *CoRR*, vol. abs/1603.02754, 2016.

[50] J. D. Durrant and J. A. McCammon, "NNScore: A neural-network-based scoring function for the characterization of protein- ligand complexes," *Journal of Chemical Information and Modeling*, vol. 50, no. 10, pp. 1865–1871, 2010.

[51] Tripos Inc., *The SYBYL Software*, 1699 South Hanley Rd., St. Louis, Missouri, 63144, USA, 2006, version 7.2.

[52] G. Jones, P. Willett, R. Glen, A. Leach, and R. Taylor, "Development and validation of a genetic algorithm for flexible docking," *Journal of Molecular Biology*, vol. 267, no. 3, pp. 727–748, 1997.

[53] R. A. Friesner, J. L. Banks, R. B. Murphy, T. A. Halgren, J. J. Klicic, D. T. Mainz, M. P. Repasky, E. H. Knoll, M. Shelley, J. K. Perry, D. E. Shaw, P. Francis, and P. S. Shenkin, "Glide: A new approach for rapid, accurate docking and scoring. 1. Method and assessment of docking accuracy," *Journal of Medicinal Chemistry*, vol. 47, no. 7, pp. 1739–1749, 2004, pMID: 15027865.

[54] Schrödinger, LLC, *The Schrödinger Software*, New York, 2005, version 8.0.

[55] H. F. G. Velec, H. Gohlke, and G. Klebe, "DrugScore CSD - Knowledge-based scoring function derived from small molecule crystal data with superior recognition rate of near-native ligand poses and better affinity prediction," *Journal of Medicinal Chemistry*, vol. 48, no. 20, pp. 6296–6303, 2005.

[56] R. Wang, L. Lai, and S. Wang, "Further development and validation of empirical scoring functions for structure-based binding affinity prediction," *Journal of Computer-Aided Molecular Design*, vol. 16, pp. 11–26, 2002, 10.1023/A:1016357811882.

[57] V. Schnecke and L. A. Kuhn, "Virtual screening with solvation and ligand-induced complementarity," in *Virtual Screening: An Alternative or Complement to High Throughput Screening?*, G. Klebe, Ed. Springer Netherlands, 2002, pp. 171–190.

[58] A. N. Jain, "Scoring noncovalent protein-ligand interactions: A continuous differentiable function tuned to compute binding affinities," *Journal of Computer-Aided Molecular Design*, vol. 10, pp. 427–440, 1996.

[59] A. Krammer, P. D. Kirchhoff, X. Jiang, C. Venkatachalam, and M. Waldman, "LigScore: A novel scoring function for predicting binding affinities," *Journal of Molecular Graphics and Modelling*, vol. 23, no. 5, pp. 395 – 407, 2005.

[60] H. Bohm, "The development of a simple empirical scoring function to estimate the binding constant for a protein-ligand complex of known three-dimensional structure," *Journal of Computer-Aided Molecular Design*, vol. 8, no. 3, pp. 243–256, 1994.

[61] D. K. Gehlhaar, G. M. Verkhivker, P. A. Rejto, C. J. Sherman, D. R. Fogel, L. J. Fogel, and S. T. Freer, "Molecular recognition of the inhibitor ag-1343 by HIV-1 protease: Conformationally flexible docking by evolutionary programming," *Chemistry & Biology*, vol. 2, no. 5, pp. 317 – 324, 1995.

[62] I. Muegge, "Effect of ligand volume correction on PMF scoring," *Journal of Computational Chemistry*, vol. 22, no. 4, pp. 418–425, 2001.

[63] M. D. Eldridge, C. W. Murray, T. R. Auton, G. V. Paolini, and R. P. Mee, "Empirical scoring functions: I. The development of a fast empirical scoring function to estimate the binding affinity of ligands in receptor complexes," *Journal of Computer-Aided Molecular Design*, vol. 11, pp. 425–445, 1997, 10.1023/A:1007996124545.

[64] T. Ewing, S. Makino, A. Skillman, and I. Kuntz, "DOCK 4.0: Search strategies for automated molecular docking of flexible molecule databases," *Journal of Computer-Aided Molecular Design*, vol. 15, no. 5, pp. 411–428, 2001.

[65] H. M. Ashtawy and N. R. Mahapatra, "A comparative assessment of conventional and machine-learning-based scoring functions in predicting binding affinities of protein-ligand complexes," in *Bioinformatics and Biomedicine (BIBM), 2011 IEEE International Conference on*. IEEE, 2011, pp. 627–630.

[66] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[67] G. Hinton, D. Y. Li Deng, G. Dahl, A. rahman Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury, and T. Sainath, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, pp. 82–97, November 2012.

[68] T. Sterling and J. J. Irwin, "Zinc 15-ligand discovery for everyone," 2015.

[69] C. Knox, V. Law, T. Jewison, P. Liu, S. Ly, A. Frolkis, A. Pon, K. Banco, C. Mak, V. Neveu *et al.*, "Drugbank 3.0: a comprehensive resource for 'omics' research on drugs," *Nucleic acids research*, vol. 39, no. suppl 1, pp. D1035–D1041, 2011.

[70] A. Gaulton, L. Bellis, A. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani *et al.*, "ChEMBL: a large-scale bioactivity database for drug discovery," *Nucleic acids research*, vol. 40, no. D1, pp. D1100–D1107, 2012.

[71] Y. Wang, J. Xiao, T. Suzek, J. Zhang, J. Wang, Z. Zhou, L. Han, K. Karapetyan, S. Dracheva, B. Shoemaker *et al.*, "Pubchem's bioassay database," *Nucleic acids research*, vol. 40, no. D1, pp. D400–D412, 2012.

[72] K. L. Damm-Ganamet, R. D. Smith, J. B. Dunbar Jr, J. A. Stuckey, and H. A. Carlson, "Csar benchmark exercise 2011–2012: evaluation of results from docking and relative ranking of blinded congeneric series," *Journal of chemical information and modeling*, vol. 53, no. 8, pp. 1853–1870, 2013.

[73] M. L. Benson, R. D. Smith, N. A. Khazanov, B. Dimcheff, J. Beaver, P. Dresslar, J. Nerothin, and H. A. Carlson, "Binding MOAD, a high-quality protein-ligand database," *Nucleic Acids Research*, vol. 36, no. suppl 1, pp. D674–D678, 2008.

[74] T. Liu, Y. Lin, X. Wen, R. Jorissen, and M. Gilson, "BindingDB: a web-accessible database of experimentally determined protein–ligand binding affinities," *Nucleic acids research*, vol. 35, no. suppl 1, pp. D198–D201, 2007.

[75] M. I. Zavodszky, P. C. Sanschagrin, L. A. Kuhn, and R. S. Korde, "Distilling the essential features of a protein surface for improving protein-ligand docking, scoring, and virtual screening," *Journal of Computer-Aided Molecular Design*, vol. 16, pp. 883–902, 2002.

[76] O. Korb, T. Stutzle, and T. Exner, "Empirical scoring functions for advanced protein- ligand docking with plants," *Journal of chemical information and modeling*, vol. 49, no. 1, pp. 84–96, 2009.

[77] M. A. Johnson and G. M. Maggiora, "Concepts and applications of molecular similarity," 1990.

[78] A. R. Leach and V. J. Gillet, *An introduction to chemoinformatics*. Springer Science & Business Media, 2007.

[79] T. Madden, "The BLAST sequence analysis tool," *The NCBI Handbook. National Library of Medicine (US), National Center for Biotechnology Information, Bethesda, MD*, 2002.

[80] Y. Zhang and J. Skolnick, "Tm-align: a protein structure alignment algorithm based on the tm-score," *Nucleic acids research*, vol. 33, no. 7, pp. 2302–2309, 2005.

[81] H. Berman, K. Henrick, and H. Nakamura, "Announcing the worldwide protein data bank," *Nature Structural & Molecular Biology*, vol. 10, no. 12, pp. 980–980, 2003.

[82] J. Dunbar Jr, R. Smith, C. Yang, P. Ung, K. Lexa, N. Khazanov, J. Stuckey, S. Wang, and H. Carlson, "CSAR benchmark exercise of 2010: Selection of the protein–ligand complexes," *Journal of chemical information and modeling*, vol. 51, no. 9, pp. 2036–2046, 2011.

[83] O. Trott and A. J. Olson, "Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading," *Journal of computational chemistry*, vol. 31, no. 2, pp. 455–461, 2010.

[84] Y. Cao and L. Li, "Improved protein–ligand binding affinity prediction by using a curvature-dependent surface-area model," *Bioinformatics*, p. btu104, 2014.

[85] C. W. Yap, "PaDEL-descriptor: An open source software to calculate molecular descriptors and fingerprints," *Journal of computational chemistry*, vol. 32, no. 7, pp. 1466–1474, 2011.

[86] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *Journal of chemical information and modeling*, vol. 50, no. 5, pp. 742–754, 2010.

[87] M. McGann, "Fred pose prediction and virtual screening accuracy," *Journal of chemical information and modeling*, vol. 51, no. 3, pp. 578–596, 2011.

[88] P. Schmidtke, V. Le Guilloux, J. Maupetit, and P. Tufféry, "Fpocket: online tools for protein ensemble pocket detection and tracking," *Nucleic acids research*, vol. 38, no. suppl 2, pp. W582–W589, 2010.

[89] G. Neudert and G. Klebe, "DSX: a knowledge-based scoring function for the assessment of protein–ligand complexes," *Journal of chemical information and modeling*, vol. 51, no. 10, pp. 2731–2745, 2011.

[90] H. M. Ashtawy and N. R. Mahapatra, "Descriptor Data Bank (DDB): A platform for multi-perspective modeling of protein-ligand interactions," *To be published, check www.descriptordb.com*, 2016.

[91] D. R. Koes, M. P. Baumgartner, and C. J. Camacho, "Lessons learned in empirical scoring with smina from the csar 2011 benchmarking exercise," *Journal of chemical information and modeling*, vol. 53, no. 8, pp. 1893–1904, 2013.

[92] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367 – 378, 2002.

[93] O. Soufan, D. Kleftogiannis, P. Kalnis, and V. B. Bajic, "Dwfs: a wrapper feature selection tool based on a parallel genetic algorithm," *PloS one*, vol. 10, no. 2, p. e0117988, 2015.

[94] B. Sahu and D. Mishra, "A novel feature selection algorithm using particle swarm optimization for cancer microarray data," *Procedia Engineering*, vol. 38, pp. 27–31, 2012.

[95] Y. Cheng and W. Prusoff, "Relationship between the inhibition constant (ki) and the concentration of inhibitor which causes 50% inhibition (ic50) of an enzymatic reaction," *Biochem. Pharmacol*, vol. 22, no. 23, pp. 3099–3108, 1973.

[96] R. A. Copeland, D. Lombardo, J. Giannaras, and C. P. Decicco, "Estimating KI values for tight binding inhibitors from dose-response plots," *Bioorganic & Medicinal Chemistry Letters*, vol. 5, no. 17, pp. 1947 – 1952, 1995.

[97] H. Gohlke, M. Hendlich, and G. Klebe, "Knowledge-based scoring function to predict protein-ligand interactions," *Journal of Molecular Biology*, vol. 295, no. 2, pp. 337 – 356, 2000.

[98] R. Wang, Y. Lu, X. Fang, and S. Wang, "An extensive test of 14 scoring functions using the PDBbind refined set of 800 protein-ligand complexes," *Journal of Chemical Information and Computer Sciences*, vol. 44, no. 6, pp. 2114–2125, 2004, pMID: 15554682.

[99] G. Schneider and P. Wrede, "Artificial neural networks for computer-based molecular design," *Progress in Biophysics and Molecular Biology*, vol. 70, no. 3, pp. 175 – 222, 1998.

[100] L. Douali, D. Villemin, A. Zyad, and D. Cherqaoui, "Artificial neural networks: Non-linear QSAR studies of HEPT derivatives as HIV-1 reverse transcriptase inhibitors," *Molecular Diversity*, vol. 8, no. 1, pp. 1–8, 2004.

[101] D. Winkler, "Neural networks as robust tools in drug lead discovery and development," *Molecular Biotechnology*, vol. 27, pp. 139–167, 2004, 10.1385/MB:27:2:139.

[102] R. D. Head, M. L. Smythe, T. I. Oprea, C. L. Waller, S. M. Green, and G. R. Marshall, "Validate: A new method for the receptor-based prediction of binding affinities of novel ligands," *Journal of the American Chemical Society*, vol. 118, no. 16, pp. 3959–3969, 1996.

[103] S. So and M. Karplus, "A comparative study of ligand-receptor complex binding affinity prediction methods based on glycogen phosphorylase inhibitors," *Journal of computer-aided molecular design*, vol. 13, no. 3, pp. 243–258, 1999.

[104] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[105] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[106] M. Stinchcombe and H. White, "Approximating and learning unknown mappings using multilayer feedforward networks with bounded weights," in *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*. IEEE, 1990, pp. 7–16.

[107] D. Steinberg and P. Colla, "CART: classification and regression trees," *The Top Ten Algorithms in Data Mining*, vol. 9, p. 179, 2009.

[108] V. Schnecke and L. A. Kuhn, "Virtual screening with solvation and ligand-induced complementarity," *Perspectives in Drug Discovery and Design*, vol. 20, no. 1, pp. 171–190, 2000.

[109] D.-S. Cao, Q.-S. Xu, Y.-Z. Liang, L.-X. Zhang, and H.-D. Li, "The boosting: A new idea of building models," *Chemometrics and Intelligent Laboratory Systems*, vol. 100, no. 1, pp. 1–11, 2010.

[110] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[111] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks." in *Aistats*, vol. 9, 2010, pp. 249–256.

[112] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[113] J. Overington, B. Al-Lazikani, and A. Hopkins, "How many drug targets are there?" *Nature Reviews Drug Discovery*, vol. 5, no. 12, pp. 993–996, 2006.

[114] BioSolveIT., *LeadIT*, St. Augustin, Germany, 2012, version 2.1.

[115] B. Ramsundar, S. Kearnes, P. Riley, D. Webster, D. Konerding, and V. Pande, "Massively multitask networks for drug discovery," *arXiv preprint arXiv:1502.02072*, 2015.

[116] G. E. Dahl, N. Jaitly, and R. Salakhutdinov, "Multi-task neural networks for qsar predictions," *arXiv preprint arXiv:1406.1231*, 2014.

[117] T. Unterthiner, A. Mayr, G. Klambauer, M. Steijaert, J. K. Wegner, H. Ceulemans, and S. Hochreiter, "Deep learning as an opportunity in virtual screening," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[118] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[119] F. Chollet, "Keras," 2015.

[120] J. Shin and D. Cho, "Pdb-ligand: a ligand database based on pdb for the automated and customized classification of ligand-binding structures," *Nucleic acids research*, vol. 33, no. suppl 1, pp. D238–D241, 2005.

[121] J. Irwin, "Community benchmarks for virtual screening," *Journal of Computer-Aided Molecular Design*, vol. 22, pp. 193–199, 2008, 10.1007/s10822-008-9189-4.

[122] M. I. Zavodszky and L. A. Kuhn, "Side-chain flexibility in protein-ligand binding: The minimal rotation hypothesis," *Protein Science*, vol. 14, no. 4, pp. 1104–1114, 2005.